Deep forests with tree-embeddings and label imputation for weak-label learning

Pedro Ilidio*, Ricardo Cerri†, Celine Vens‡§, Felipe Kenji Nakano‡§
*Instituto de Física de São Carlos, Universidade de São Paulo, São Carlos, São Paulo, Brazil
† Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, SP, Brazil
‡ KU Leuven, Campus KULAK, Dept. of Public Health and Primary Care, Kortrijk, Belgium
§ Itec, imec research group at KU Leuven, Kortrijk, Belgium
Email: ilidio@alumni.usp.br, cerri@icmc.usp.br, celine.vens@kuleuven.be and felipekenji.nakano@kuleuven.be

Abstract—**Due to recent technological advances, a massive amount of data is generated on a daily basis. Unfortunately, this is not always beneficial as such data may present weak-supervision, meaning that the output space can be incomplete, inexact, and inaccurate. This kind of problems is investigated in weakly-supervised learning. In this work, we explore weak-label learning, a structured output prediction task for weakly-supervised problems where positive annotations are reliable, whereas negatives are missing. For the first time in this class of problems, we investigate deep forest algorithms based on tree-embeddings, a recently proposed feature representation strategy leveraging the structure of decision trees. Furthermore, we propose two new procedures for label-imputation in each layer, named Strict Label Complement (SLC), which provides fixed conservative estimates for the number of missing labels and employs them to restrict imputations, and Fluid Label Addition (FLA), which performs such estimations on every layer and uses them to adjust the imputer's predicted probabilities without any restrictions. We combine the new approaches with deep forest architectures to produce four new algorithms: SLCForest and FLAForest, using output space feature augmentation, and also the cascade forest embedders CaFE-SLC and CaFE-FLA, employing both tree-embeddings and the output space. Our results reveal that our methods provide superior or competitive performance to the state-of-the-art. Furthermore, we also noticed that our methods are associated with better results even in cases without weak-supervision.**

1. Introduction

In recent years, we have seen the rise of deep learning as the predominant solution for problems with unstructured data. That is, problems where data is presented in a raw format, such as images, videos and audio. However, when it comes to tabular (structured) data, variants of tree-ensemble methods are the current state-of-the-art [1].

In this regard, deep forests have recently been proposed as a powerful, efficient, and light-weighted solution [2]. Similarly to their neural network counterpart, deep forests are composed by several layers where the input features are sequentially augmented using the output of previous layers. In this case, however, layers are composed of tree-ensembles.

Originally, deep forests employed the output space features (predicted probabilities) as augmented features for the next layers, in a similar fashion to classifier chains [3]. Despite achieving remarkable performance, recent work [4] has shown that replacing output space features by tree-embeddings, a feature transformation based on the structure of the trees, is more informative, especially regarding structured output prediction.

As defined in [5], [6], structured output prediction is a field where multiple outputs, which are correlated to each other, must be predicted. In this domain, the veracity of the data is an important challenge, since datasets present imperfect data in their output space [5], which requires specific methods. This type of problems is investigated in the field of weakly-supervised learning [7].

As its name suggests, weakly-supervised learning is an umbrella term for problems with weak-supervision. Such weak-supervision can be presented in three forms and combinations thereof: i) incomplete supervision, where only a subset of the annotations is given; ii) inexact, where annotations are given on a coarse-grained level; and iii) inaccurate supervision, where annotations are not completely reliable.

In this work, we investigate weak-label learning, a weakly supervised learning task where datasets are presented with missing annotations. More precisely, positive annotations are reliable, whereas negative ones may either be real negatives or missing positives. Thus, weak-label learning methods can impute missing positive annotations (change negative annotations into positives), generating a new imputed dataset. This imputation can either be performed as a pre-processing step and be given as input to a multi-label method [8], or occur during the construction of the predictive model [9].

The current state-of-the-art in tabular weak-label learning, LCForest [9], employs output space features for feature augmentation and an internal cross-validation procedure to impute missing positive annotations, which relies on a fixed classification threshold to impute. It also employs a label frequency estimation procedure adapted from positive unlabeled learning [10], which often overestimates the number of missing positive annotations to be imputed. Furthermore, LCForest allows the estimated bounds for the imputation to be surpassed, resulting in excessive imputation. Lastly, the proposed early-stopping mechanism is often ineffective, since it can be overly strict.

In order to address these drawbacks, we propose a novel deep-forest method for weak-label learning that combines tree-embeddings (TEs) and output space features (OS) for feature augmentation, with two novel label imputation procedures, strict label complement (SLC) and fluid label addition (FLA). More specifically, SLC implements new more restrictive imputation upper bounds, while FLA controls imputation by adjusting its imputation threshold according to estimates of the amount of missing data. Altogether, we propose four new variants of our method. SLCForest and FLAForest implement the new imputation strategies and a recently-proposed post-pruning scheme [4], which we further refine with out-of-bag internal performance estimates. CaFE-SLC and CaFE-FLA additionally employ tree-embedding feature augmentation approaches introduced in recent studies [4].

Our results showcase superior or competitive performance when compared to state-of-the-art approaches. While metrics based on the binary outputs greatly favor the tree-embedding-based CaFE-SLC and CaFE-FLA, SLCForest is confidently shown to outperform its predecessor LCForest [9] when label ranking is evaluated in multiple ways, being the overall best-placed model for probability-based metrics under moderate amounts of missing positive annotations.

The remainder of this paper is organized as follows. Section 2 provides an overview of current strategies involving deep forest models and weak-label scenarios. Section 3 provides a detailed presentation of our newly developed methods. The datasets we utilize, evaluation procedure, and hyperparameter configurations are described in Section 4. The discussion around our obtained results are organized in Section 5, while Section 6 presents our final remarks and concludes our work.

2. Background and Related work

In this section, we present the background and related work. Initially, we present deep forests and their main components, followed by recent deep forest-based methods. Further, we present weak-label learning, deep forest methods for weak-label learning, and the current state-of-the-art of the field.

2.1. Deep forests

Similarly to deep neural networks, deep-forests are models composed of several layers, where input features are sequentially augmented using the output of previous layers until convergence is reached. In this case, layers consist of tree ensembles. The resulting method is a *cascade* structure, where the internal models act as consecutive transformations of the data and a final layer wraps the augmented dataset into final predictions. Deep forests are composed of three main components:

1) **Layer structure:** Tree-ensembles built for prediction in each layer. For instance, the original deep-forest (gcForest) [2] employed 2 random forests (RFs) [11] and 2 randomized trees (ETs) per layer [12];
2) **Feature augmentation:** the feature transformation procedure used in each layer. The majority of the literature relies on variants of prediction probabilities, similar to classifier chains [3]. Alternatively, recent work on structured output prediction has shown that tree-embeddings [4], features based on the structure of the trees, can be more representative. Furthermore, overfitting prevention methods are often employed, such as using separate models to generate augmented features [4], [9] or *crossing-over* where the augmented features generated using one ensemble are used as input to others in the next layer, e.g, assuming a RF and an ET per layer, the augmented features generated using the RF would be used as input for the ET in the next layer;
3) **Length control:** A criterion to define the optimal number of layers in the method.

More specifically, in each layer, feature augmentation is performed using the original features or the augmented features from the previous layer. Thus, new features are generated and concatenated with the original features, resulting into new augmented features. The newly generated augmented features are then used by the tree-ensembles in the layer structure to build predictive models, and are propagated forward to the next layer. This procedure is repeated until the length control procedure is activated.

Table 1 presents recent work on deep-forest methods. As can be seen, most of the studies employ the same components as the original deep-forest [2]. That is, two RFs and two ETs for the layer structure, output space (OS) for feature augmentation, and first drop in score for length control. Furthermore, it is noticeable that most of the studies focus on single-output problems, such as binary and multi-class classification. Lastly, the work of Nakano et al. [4] is the only one that employed tree-embeddings.

2.2. Weak-label learning

Weak-label learning is defined as follows. Given an input space $\mathbb{X} = \mathbb{R}^d$ of d dimensions and an output space $\mathbb{Y} = \{0,1\}^p$ with p dimensions, a weak-label dataset of m instances consists of $\{(\mathbf{x}_i, \mathbf{y}_i | 1 \leq i \leq m)\}$, also organized into matrices $X = (x_{ij})$ and $Y = (y_{ij})$. $\mathbf{x}_i \in \mathbb{X}$ is a d dimensional vector representing the features $\begin{bmatrix} x_{i1} & x_{i2} & \cdots & x_{id} \end{bmatrix}$ and $\mathbf{y}_i \in \mathbb{Y}$ is the associated vector of labels $\begin{bmatrix} y_{i1} & y_{i2} & \cdots & y_{ip} \end{bmatrix}$ where $\mathbf{y}_i$ is partially provided in the sense that $y_{ij} = 1$ corresponds to the ground truth, nonetheless $y_{ij} = 0$ can represent both a real negative or a missing positive annotation. The objective is to find a function $f : \mathbb{X} \to \mathbb{Y}$.

Assuming an optimistic scenario, where very few positive annotations are missing, weak-label learning could

Algorithm	Layer structure	Feature augmentation	Length control	Task	Application
gcForest [2]	2 RFs - 2ETs	OS	First score drop	Binary and multi-class classification	-
MLDF [13]	1 RF - 1 ET	OS	Training score tolerance 3 iterations	Multi-label classification	-
IMDF	Adaboost	OS	First score drop	(Imbalanced) Binary classification	-
Deep-Resp-Forest [14]	2 RFs - 2ETs	OS	First score drop	Binary classification	Drug response
CSDF [15]	2 RFs - 2ETs	OS	Maximum number of iterations	Multi-class classification	Price prediction
DFH [16]	2 RFs - 2ETs	OS	First score drop	Hashing learning	-
SDF [17]	2 RFs - 2ETs	OS	Maximum number of iterations	Multi-class classification	-
BCDForest [18]	2 RFs - 2ETs	OS	First score drop	Multi-class classification	Cancer subtype
Deep tree ensembles [4]	1 RF - 1 ET	TE	Best training score	Multi-label classification and multi-target regression	-

TABLE 1. Recent literature on deep forest methods.

be addressed using traditional multi-label classification, nonetheless, these methods often underperform in realistic scenarios, as the noise is overlooked.

To overcome this challenge, initial works on weak-label learning followed a transductive approach where missing positive annotations are imputed using low-rank matrix composition methods [8], [19]. More recent methods, however, have adopted a predictive approach [20].

Weak-label learning is closely related to semi-supervised multi-label classification, where instances may present an entire empty label set [21]. Further, positive unlabeled learning [22] is also similar since negative annotations are also unreliable. The main difference relies on the number of labels since positive unlabeled learning focuses on single output problems. Lastly, partial (multi)-label learning addresses problems where positive annotations are unreliable [23].

2.3. Deep-forest methods for weak-label learning

Besides the three components mentioned in Section 2.1, layer structure, feature augmentation, and length control, deep forests for weak-label learning also employ a fourth component, namely label imputation.

2.3.1. Label imputation. As its name suggests, the component label imputation is responsible for imputing missing annotations at each layer of the deep forest. Formally, we describe this component in Equation 1 where Y^{l+1} is the output matrix used for the next layer, Y^0 is the original output matrix and $\hat{Y}^l$ is the output imputed matrix of a particular layer l. $\vee$ denotes the element-wise OR operation.

$$Y^{l+1} = Y^0 \vee \hat{Y}^l \tag{1}$$

Notice that the modified label matrix Y^l used as input for the forests at level l is directly dependent only on the original annotations Y^0 and the last layer's imputation. That is, imputed annotations are not cumulative throughout the levels.

2.3.2. LCForest. The current state-of-the-art in weak-label learning, LCForest [9], employs the following components: one RF and one ET for layer structure and output space features for feature augmentation. As for length control, the authors propose a label-wise upper bound u_j for the number of imputed labels, which is measured using the positive unlabeled method proposed in [10]. If u_j is reached for all possible labels, then the training is stopped. Lastly, LCForest employs label complement (LC) as the label imputation procedure.

LC imputes missing positive annotations using a procedure based on nested cross-validation in such a way that each predicted probability is returned by the model that did not use the corresponding instance for training. Further, the output of the ensemble is only positive if the classification threshold is satisfied by all forests, which requires unanimity instead of the usual majority voting. According to the authors, this procedure avoids overconfident estimates and consequently wrongly imputed annotations. Given an ensemble of forests in a layer l, Equation 2 presents the procedure to generate $\hat{Y}^l$ (imputed output matrix), where $\tilde{Y}^{l,t}$ is the predicted probabilities of forest t and θ is the classification threshold.

$$\hat{Y}^l = \prod_t (\tilde{Y}^{l,t} > \theta) \tag{2}$$

To additionally prevent excessive imputation, imputation is stopped for a particular label j if u_j is surpassed, and the imputed entries for j are kept fixed for the downstream cascade levels [9].

Despite its proven effectiveness, we identify three key limitations of the original LCForest algorithm:

1) **Overshooting:** the number of imputed labels is allowed to exceed u_j before imputation is stopped. The imputer annotates all predicted labels that cross the classification threshold, tests if u_j has been surpassed and stops the imputation if so, keeping the excessive imputations;
2) u_j **overestimation:** the original estimation technique is biased towards larger values of u_j, especially with the default parameters, as their authors report [10];
3) **Insufficient stopping criterion**: the stopping criterion is rarely met and cascades tend to overgrow. This is partially a consequence of item 2.

3. Proposed methods

In this Section, we present the feature augmentation and label imputation procedures employed by our method.

3.1. Feature augmentation

We explore two ways of performing feature augmentation: output space features and tree-embeddings.

3.1.1. Output space features. Output space features (OS) correspond to employing predicted probabilities as extra features, similarly to classifier chains [3].

3.1.2. Tree-embeddings. Given a tree-ensemble method, tree-embeddings are vectors z in which each position z_n corresponds to a tree node indexed by n, for all nodes in all the decision trees of an ensemble. For a given instance x that traverses the tree, Equation 3 defines how to obtain z. In this case, s_n corresponds to the number of training instances that reach node n.

$$z_n = \begin{cases} \log(s_n + 1)^{-1} & \text{if } x \text{ traverses } n \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

Note that using all tree nodes in a forest could easily become infeasible to deal with for larger datasets. To mitigate this issue, nodes are filtered according to s_n and further given as input to a dimensionality reduction method, such as PCA.

3.2. Label imputation

In this subsection, we present our label imputation procedures and their strategies used to estimate the label frequency.

3.2.1. Label frequency estimation from out-of-bag sets. We estimate the number of positive annotations to be imputed to determine how aggressive our imputers should be. This number of missing positive annotations for each label j is encoded as the probability in Equation 4, named c_j.

$$P(y_j = 1 \mid y_j^* = 1) = c_j \tag{4}$$

This is the target quantity we set to approximate. y_j^* denotes the true label j of an instance, considering a hypothetical completely-supervised scenario, while y_j is the observed label in our weakly-supervised dataset.

We then use probabilities estimated by our forests to approximate c_j. Specifically, we first collect the out-of-bag probabilities of the forests as reasonable estimates for $P(y_j = 1)$. To assure the condition $y_j^* = 1$ in Equation 4, we then only use the positive instances of the out-of-bag validation sets. Each predicted probability of a positive instance in the validation sets is an estimate $\tilde{c}_j$. We choose the final c_j as a large percentile p of the distribution of $\tilde{c}_j$ (Equation 5).

$$P(\tilde{c}_j \leq c_j) = p \tag{5}$$

c_j being constant results in Equation 6 [22], [24].

$$P(y_j^* = 1) = \frac{P(y_j = 1)}{c_j} \tag{6}$$

As a result, upper bounds for the number of imputed labels can be obtained as in Equation 7.

$$u_j = \left\lfloor \frac{\sum_i Y_{ij}^0}{c_j} \right\rfloor \tag{7}$$

The percentile approach is performed to ensure conservative imputation, favoring lower u_j.

3.2.2. The strict label complement (SLC) mechanism. We address each of the limitations of LCForest presented in Section 2.3.2 as follows:

1) **Overshooting:** we strictly disallow the u_j upper bounds to be crossed. We select the best-ranked predictions to enforce the number of imputed labels in output j to always be u_j at most;
2) u_j **overestimation:** we use more conservative upper bound estimates for the number of missing labels. The technique is described in Section 3.2.1 and is designed to favor lower values of u_j;
3) **Insufficient stopping criterion:** we select the level with the best out-of-bag scores as a post-pruning strategy.

We also release the requirement for unanimity in the imputer ensemble and instead use a conventional average probability between the n_F forests (substituting Equation 2 with Equation 8).

$$\hat{Y}^l = \left(\frac{1}{n_F} \sum_t \tilde{Y}^{l,t} \right) < \theta \tag{8}$$

We argue that the parameter θ is sufficient for controlling the imputer's tolerance. Finally, we use OOB probabilities as $\tilde{Y}^{l,t}$ rather than the original k-fold procedure.

3.2.3. Fluid label addition (FLA). Our second imputation strategy, Fluid Label Addition (FLA), is more aggressive when more annotations are expected to be missing, but more conservative otherwise. We achieve this by dividing the probabilities outputted by the imputer model by the estimated c_j (Section 3.2.1) and only then applying the classification threshold. After the division, the predicted probabilities then represent an approximation for $P(y_j^* = 1)$ rather than for $P(y_j = 1)$ (Equation 6). We can equivalently multiply the θ classification threshold by c_j^l instead of dividing the probabilities, and the method is then defined by Equation 9.

$$\hat{y}_j^l = \tilde{y}_j^l < \theta \cdot c_j^l \tag{9}$$

We estimate a c_j^l in every layer for this strategy, and no upper limit is used for the number of imputed labels. We denote by $\tilde{Y}^l$ the average predicted probabilities across all forest imputers in level l. Again, we use out-of-bag probability estimates as $\tilde{Y}^l$ to mitigate overfitting.

4. Experimental settings

In this section, we present our datasets, competitor methods, parameters employed, and evaluation measures.

4.1. Datasets

Dataset	Samples	Features	Labels	Density
VirusGO	207	749	3	0.33
VirusPseAAC	207	440	3	0.33
flags	194	19	6	0.54
GrampositivePseAAC	519	440	3	0.32
CHD_49	555	49	5	0.51
emotions	593	72	6	0.31
birds	645	260	9	0.086
Gram_negative	1392	1717	6	0.17
PlantGO	978	3091	9	0.11
medical	978	1449	12	0.086
scene	2407	294	6	0.18
yeast	2417	103	14	0.30
enron	1702	1001	25	0.13

TABLE 2. DATASETS OVERVIEW.

We utilized a total of 13 publicly available multi-label datasets[1][2] encompassing diverse domains such as text, biological data, sound, and imagery. We discarded all label columns with less than 30 positive occurrences. The resulting dimensions for each dataset are presented by Table 2.

Similarly to [9], we have randomly masked (turning 1s into 0s) $\{0\%, 30\%, 50\%, 70\%\}$ of the occurrences of each label in the training set prior to providing it to the model, resulting into 3 corrupted versions of each dataset. The frequencies are also denoted incomplete label rates (ILR) [9]. We use experiments without dropping any labels to validate the hypothesis that multi-label datasets are inherently weakly-supervised, and also to use them as references for comparison.

4.2. Comparative analysis

We compare our four proposed weakly-supervised cascade forests to four baseline algorithms from the literature and the current state-of-the-art LCForest (Section 2.3.2) [9], as summarized by Table 3. RF+ET corresponds to the simple combination of a random forest [11] and an ensemble of randomized trees [12], which outputs the average of both models' predicted label probabilities. RF+ET is equivalent, in our case, to the final estimator of a deep forest without any feature augmentation.

gcForest refers to the original cascade forest proposal by Zhou [2] adding predicted probabilities in each layer as augmented features. We assume tabular format for the input features and thus do not employ the multi-grained scanning procedure the authors proposed for structured inputs.

Since no unified denomination seems to exist in the literature, we suggest the acronym CaFE (Cascade Forest Embedders) to encompass the broad category of deep forests using tree-embedding techniques. We then refer to the two algorithms from [4] as CaFE and CaFE-OS, employing solely tree-embeddings and tree-embeddings in conjunction with output space features, respectively.

1. http://www.uco.es/kdis/mllresources/
2. https://itec.kuleuven-kulak.be/pattern-recognition-2021/

Baseline weak-label comparison methods, such as [8], [19], were not analyzed since they were designed for transductive learning, whereas we focus on predictive learning. Additionally, traditional off-the-shelf multi-label methods, such as MlKNN [25], Rakel [26] and DBPNN [27], have already been included as comparison methods in previous works [4], [9]. Their results were consistently worse with statistical significance, thus we have opted to not include them.

Our four proposals correspond to the combinations of our two imputation methods FLA (Section 3.2.3) and SLC (Section 3.2.2), with two feature augmentation methods: output space features alone, or output space in conjunction with tree-embeddings (Table 3).

4.3. Model parameters

Similarly to previous related work [2], [4], [9], we use a random forest [11] in conjunction with an ensemble of randomized trees [12] as the base estimator for all label imputers and feature augmentation procedures in the deep forests. Each of these forests was composed of 150 trees, with a minimum leaf size of 5 samples, and selecting, at each node, $\sqrt{n_f}$ from the total n_f features. A maximum of 10 levels is set for all cascades. We build each tree with half the initial number of samples received by the cascade, for all forests. This enables us to calculate validation scores using the out-of-bag sets that will be used by some of the models. This also works as a measure to prevent overfitting when generating OS features and tree-embeddings, analogously to cross-validation in [2], [4].

We set the percentile p for label frequency estimation (Section 3.2.1) to 95%, and we use $\theta = 0.5$ for the base classification threshold of FLA (Section 3.2.3) and SLC (Section 3.2.2).

4.4. Evaluation

For each dataset, we report results using the average performance on 5-fold cross-validation with iterative stratification [28], [29]. In the experiments where labels were masked, we use the masked training set to train the models but evaluate their performance on the unmasked test sets. We also use the masked training sets to calculate the validation scores used for length control.

We use the micro average of five well-established metrics to evaluate our models: the area under the receiver-operating characteristic curve (AUROC), average precision (AP) score, Hamming distance, label ranking loss, and Matthews correlation coefficient (MCC) [5], [6].

5. Results and discussion

The results of the comparisons involving all models are presented in the graphs of Figure 1.

For $ILR = 30\%$ and $ILR = 0\%$ (30% or 0% masking), SLCForest was the best-ranked model in all evaluation metrics that do not require establishing a classification threshold

Algorithm	Feature augmentation	Crossing-over	Length control	Label imputation	Reference
RF+ET	-	-	Single layer	-	[11], [12]
gcForest	OS	No	First score drop	-	[2]
LCForest	OS	No	Imputation limit	LC	[9]
CaFE	TE	Yes	Best training score	-	[4]
CaFE-OS	TE and OS	Yes	Best training score	-	[4]
SLCForest	OS	Yes	Best OOB score	SLC	Proposed
FLAForest	OS	Yes	Best OOB score	FLA	Proposed
CaFE-SLC	TE and OS	Yes	Best OOB score	SLC	Proposed
CaFE-FLA	TE and OS	Yes	Best OOB score	FLA	Proposed

TABLE 3. SUMMARY OF THE ALGORITHMS BEING COMPARED IN THIS WORK. OS REFERS TO OUTPUT SPACE FEATURES, TE TO TREE-EMBEDDINGS. OOB REFERS TO OUT-OF-BAG ESTIMATES, WHILE LC, SLC AND FLA RESPECTIVELY MEAN LABEL COMPLEMENT (SECTION 2.3.2), STRICT LABEL COMPLEMENT (SECTION 3.2.2) AND FLUID LABEL ADDITION (SECTION 3.2.2).

(ranking error, AUROC, AP). The same occurred for ranking error and $ILR = 30\%$. Meanwhile, the original LCForest was either the worst or the second worst-performing model for these same three metrics in all label masking configurations. This indicates that SLC is a significant methodology improvement over the original LC.

We also notice the simple RF+ET surpassed a considerable number of models on the threshold-independent metrics under higher masked-label fractions (50% and 70%). In fact, no model significantly outperformed RF+ET in these conditions. Nevertheless, RF+ET displays the highest inconsistency for these metrics, which is observable in the form of larger interquartile ranges that result in fewer statistically significant comparisons. Hence, despite providing remarkable results, the performance of RF+ET largely depends on the dataset.

LCForest, CaFE-SLC, and CaFE-FLA are the consistent and clear winners for the threshold-dependent metrics MCC and Hamming Loss under masked labels scenarios (30%, 50%, and 70%). LCForest is significantly surpassed by the other two models in terms of MCC for $ILR = 50\%$ and $ILR = 70\%$ while significantly outperforming both for the Hamming loss under $ILR = 70\%$. Therefore, our results suggest that CaFE-SLC and CaFE-FLA are preferable choices over LCForest. They show prominent superiority for threshold-independent metrics and highly competitive results for MCC and the Hamming loss.

The supervised models (RF+ET, CaFE, CaFE-OS, gcForest), where no imputation is performed, are the four worst-performing ones in all threshold-dependent scenarios with masked labels. Therefore, label imputation seems to be important for the selection of binary outputs. A possible explanation would be that imputing labels makes the label density of training sets closer to the density of the test set. This would be especially important in our scenarios, where these densities clearly differ.

Overall, the tree-embeddings seem to be more beneficial in more challenging scenarios where fewer annotations are available, while OS features seem to perform better when the data is more reliable.

SLCForest seems to be the best option up until intermediate fractions of missing labels. This model resulted in the best performance even when no label masking was performed. Still, for $ILR = 0\%$, FLAForest was the second-best model for AUROC, Hamming loss, and MCC. These results demonstrate that label imputation is also effective for the original multi-label tasks, without considering weak-labels. This suggests that weak-labels could be present in the majority of benchmark multi-label problems. As such, future studies could benefit from integrating imputation as an important step for multi-label tasks in general.

6. Conclusion

We have presented four variations of weakly-supervised deep forests. To the best of our knowledge, our results demonstrate for the first time the effectiveness of tree-embeddings-based cascades in conjunction with label imputation for weak-label learning problems. Our results indicate that tree-embeddings combined with label imputation are especially beneficial when more label annotations are missing. When more annotations are available, output space features without tree-embeddings seem to be the better option. Tree-embeddings were also superior for metrics that require a classification threshold, the MCC, and Hamming loss. When the threshold is not needed (AUROC, AP, ranking loss), SLCForest was the best model overall.

We also show that label imputation is beneficial even if the original multi-label datasets, without masking positive annotations, were used. This suggests that future studies on multi-label problems should consider imputation even if the problems were expected to be supervised.

In future work, we would also like to investigate the removal of wrong positive annotations, in a similar fashion to partial multi-label learning [23]. Lastly, we would also like to adapt our method to hierarchical multi-label classification, where the correlation among outputs is pre-established by an underlying hierarchy, since these datasets are inherently weakly-supervised [30].

Acknowledgments

The authors would like to thank FWO, 1235924N, and FAPESP, grant number 2022/02981-8. This study was financed in part by the Coordenação de Aperfeiçoamento de

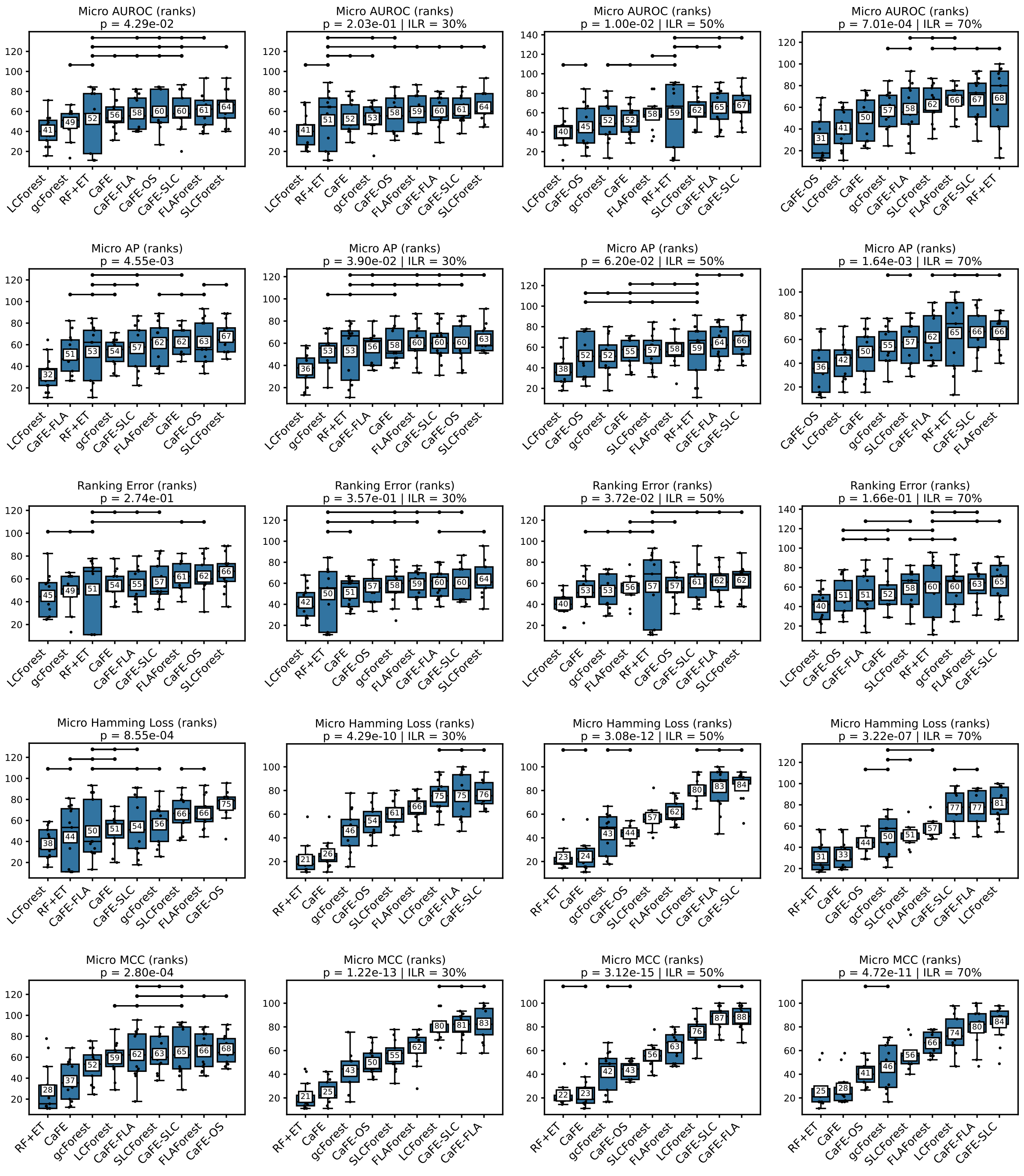

Figure 1. Comparison of percentile ranks of the analyzed estimators under several performance metrics and incomplete label ratios (ILR). The scores were ranked across estimators within each of the 5 cross-validation folds of each dataset. The ranks were then averaged across folds for each dataset and the resulting average percentile ranks are shown as boxplots, with each point referring to a dataset. Each point thus represents, for a particular dataset, the percentage of other estimators outperformed by the selected estimator. The average percentage across datasets is then plotted in white annotated boxes. The omnibus p-value resulting from a Friedman test is displayed above each plot, while crossbars connecting dots over each box indicate groups of estimators that could not be statistically distinguished ($p > 0.05$) under a Wilcoxon Signed Rank test followed by Benjamini-Hochberg procedure. The multiple-tests correction procedure included all the significant comparisons omitted in each plot.

Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

Data and code availability

The datasets used in this work and the necessary code to reproduce our results are both made available at https://itec.kuleuven-kulak.be/supporting-material/ and https://github.com/pedroilidio/ijcnn2024.

References

[1] L. Grinsztajn, E. Oyallon, and G. Varoquaux, "Why do tree-based models still outperform deep learning on typical tabular data?" *Advances in Neural Information Processing Systems*, vol. 35, pp. 507–520, 2022.

[2] Z.-H. Zhou and J. Feng, "Deep forest," *National Science Review*, vol. 6, no. 1, pp. 74–86, 2019.

[3] J. Read, B. Pfahringer, G. Holmes, and E. Frank, "Classifier chains: A review and perspectives," *Journal of Artificial Intelligence Research*, vol. 70, pp. 683–718, 2021.

[4] F. K. Nakano, K. Pliakos, and C. Vens, "Deep tree-ensembles for multi-output prediction," *Pattern Recognition*, vol. 121, p. 108211, 2022.

[5] D. Xu, Y. Shi, I. W. Tsang, Y.-S. Ong, C. Gong, and X. Shen, "Survey on multi-output learning," *IEEE transactions on neural networks and learning systems*, vol. 31, no. 7, pp. 2409–2429, 2019.

[6] W. Waegeman, K. Dembczyński, and E. Hüllermeier, "Multi-target prediction: a unifying view on problems and methods," *Data Mining and Knowledge Discovery*, vol. 33, pp. 293–324, 2019.

[7] Z.-H. Zhou, "A brief introduction to weakly supervised learning," *National science review*, vol. 5, no. 1, pp. 44–53, 2018.

[8] Y.-Y. Sun, Y. Zhang, and Z.-H. Zhou, "Multi-label learning with weak label," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 24, no. 1, 2010, pp. 593–598.

[9] Q.-W. Wang, L. Yang, and Y.-F. Li, "Learning from Weak-Label Data: A Deep Forest Expedition," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, pp. 6251–6258, 2020.

[10] J. Bekker and J. Davis, "Estimating the Class Prior in Positive and Unlabeled Data Through Decision Tree Induction," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, no. 1, 2018.

[11] L. Breiman, "Random forests," *Machine learning*, vol. 45, pp. 5–32, 2001.

[12] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Machine learning*, vol. 63, pp. 3–42, 2006.

[13] L. Yang, X.-Z. Wu, Y. Jiang, and Z.-H. Zhou, "Multi-label learning with deep forest," *arXiv preprint arXiv:1911.06557*, 2019.

[14] R. Su, X. Liu, L. Wei, and Q. Zou, "Deep-resp-forest: a deep forest model to predict anti-cancer drug response," *Methods*, vol. 166, pp. 91–102, 2019.

[15] C. Ma, Z. Liu, Z. Cao, W. Song, J. Zhang, and W. Zeng, "Cost-sensitive deep forest for price prediction," *Pattern Recognition*, vol. 107, p. 107499, 2020.

[16] M. Zhou, X. Zeng, and A. Chen, "Deep forest hashing for image retrieval," *Pattern Recognition*, vol. 95, pp. 114–127, 2019.

[17] L. V. Utkin and M. A. Ryabinin, "A siamese deep forest," *Knowledge-Based Systems*, vol. 139, pp. 13–22, 2018.

[18] Y. Guo, S. Liu, Z. Li, and X. Shang, "Bcdforest: a boosting cascade deep forest model towards the classification of cancer subtypes based on gene expression data," *BMC bioinformatics*, vol. 19, no. 5, pp. 1–13, 2018.

[19] M. Xu, R. Jin, and Z.-H. Zhou, "Speedup matrix completion with side information: Application to multi-label learning," *Advances in neural information processing systems*, vol. 26, 2013.

[20] L. Wang, Y. Liu, C. Qin, G. Sun, and Y. Fu, "Dual relation semi-supervised multi-label learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, 2020, pp. 6227–6234.

[21] W. Zhan and M.-L. Zhang, "Inductive semi-supervised multi-label learning with co-training," in *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, 2017, pp. 1305–1314.

[22] J. Bekker and J. Davis, "Learning from positive and unlabeled data: A survey," *Machine Learning*, vol. 109, no. 4, pp. 719–760, 2020.

[23] M.-K. Xie and S.-J. Huang, "Partial multi-label learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.

[24] C. Elkan and K. Noto, "Learning classifiers from only positive and unlabeled data," in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '08. Association for Computing Machinery, 2008, pp. 213–220. [Online]. Available: https://doi.org/10.1145/1401890.1401920

[25] M.-L. Zhang and Z.-H. Zhou, "Ml-knn: A lazy learning approach to multi-label learning," *Pattern recognition*, vol. 40, no. 7, pp. 2038–2048, 2007.

[26] G. Tsoumakas and I. Vlahavas, "Random k-labelsets: An ensemble method for multilabel classification," in *European conference on machine learning*. Springer, 2007, pp. 406–417.

[27] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *science*, vol. 313, no. 5786, pp. 504–507, 2006.

[28] K. Sechidis, G. Tsoumakas, and I. Vlahavas, "On the Stratification of Multi-label Data," in *Machine Learning and Knowledge Discovery in Databases*, D. Gunopulos, T. Hofmann, D. Malerba, and M. Vazirgiannis, Eds. Springer Berlin Heidelberg, 2011, vol. 6913, pp. 145–158.

[29] P. Szymański and T. Kajdanowicz, "A Network Perspective on Stratification of Multi-Label Data," in *Proceedings of the First International Workshop on Learning with Imbalanced Domains: Theory and Applications*. PMLR, 2017, pp. 22–35.

[30] F. K. Nakano, M. Lietaert, and C. Vens, "Machine learning for discovering missing or wrong protein function annotations: A comparison using updated benchmark datasets," *BMC bioinformatics*, vol. 20, pp. 1–32, 2019.