A first approach towards a fuzzy decision tree for multilabel classification

Ronaldo C. Prati*, Francisco Charte† and Francisco Herrera†
*Center of Mathematics, Computer Science and Cognition
Federal University of ABC (UFABC)
Santo André, SP, Brazil
†Department of Computer Science and Artificial Intelligence
University of Granada
18071 Granada, Spain

***Abstract*—This paper proposes a multilabel fuzzy decision tree classifier named FuzzDT$_{\mathrm{ML}}$. The algorithm uses generalized fuzzy entropy, aggregated over all labels, to choose the best attribute for growing the tree. The proposed algorithm also can generate leaves predicting partial label sets, which can incorporate to some degree the dependence among labels, as well as produce more interpretable models. An empirical analysis shows that, although the algorithm does not yet incorporate pruning nor fuzzy interval adjustment phases, it is competitive with other tree based approaches for multilabel classification, with better performance in data sets having numerical features that can be fuzzified.**

I. Introduction

Multilabel classification [1] is a data mining task in which more than a single label, from a set of labels $\mathcal{L}$, can be assigned to a given instance at the same time. The labels are generally grouped into a binary vector Y of size $|\mathcal{L}|$, where a 1/0 value at position y_i indicates the relevance/irrelevance of the label l_i to an a given instance X, from a set of instances $\mathcal{X}$. A Multilabel Classifier (MLC) $\mathcal{F}$ is a mapping function defined as $\mathcal{F} : \mathcal{X} \to \mathcal{Y}$, which aims to predict the label vector Y for a given instance X.

Algorithms for inducing MLCs can be divided into two groups: (a) *problem transformation methods*, where the MLC problem is organized as a set of single-label classification tasks, so that traditional classifier learning algorithms can be applied; and (b) *algorithm adaptation methods*, where a specific learning algorithm is extended in order to cope to MLC problems directly. The former has the advantage that, once the problem was transformed, any single label classifier can be applied. However, depending on how the labels are transformed, they should be considered independently, loosing inter-label information; or joined as a single class token, loosing label diversity as only combinations that appear in the training set can be predicted. Furthermore, a single-label model needs to be induced for each transformed problem, increasing computational costs. The later can be used to build an overall, single model. The drawback is that the algorithm adaption is not trivial, and each algorithm requires specific adaptations.

In this paper, we propose FuzzDT$_{\mathrm{ML}}$, a MLC fuzzy decision tree algorithm. The reasons for developing a fuzzy decision tree MLC are two-fold: firstly, we can use the inherent interpretability of fuzzy based systems to give some intuition or explanation about a classification. This is a very important feature in some data mining and knowledge discovery tasks where not only a "black-box" classification is necessary, but also some interpretation of the classification. Second, MLCs has often some degree of vagueness among the labels boundaries, which cannot be properly caught by standard crisp (non-fuzzy) classifiers.

The main characteristics of FuzzDT$_{\mathrm{ML}}$ are:

- the algorithm induces an overall, single MLC model, facilitating its interpretation;
- it can generate leaves with partial label sets, which can incorporate in the model some aspects of label dependency;
- the performance is comparable with other tree-based MLC, but the algorithm shows some advantage with data sets with numerical attributes.

This paper is organized as follows: Section II presents related work. Section III describes the proposed algorithm. Section IV reports the carried out empirical evaluation to analyze the performance of the proposed method, and Section V presents some concluding remarks and future research directions.

II. Related Work

Decision tree is a widely-used classification technique due to its easily understandable tree-like representation [2]. The main idea consists in growing the branches of a tree where each split corresponds to a test through an attribute's value. The choice of the splitting attribute is performed heuristically, and the process is recursively repeated for each branch. Each split is called a node and the first split is called the root of the tree. When some stopping criterion is met, the splitting process is terminated, and a leaf node containing a prediction is created. Post-processing steps, such as pruning, is often carried out for avoiding overfitting and improve interpretability.

There are some proposals for adapting decision tree algorithms for multilabel classification. A straightforward approach is to use some data transformation method, and then apply a decision tree algorithm as the base classifier. Two common

978-1-5090-6034-4/17/$31.00

approaches are the Binary Relevance transformation, which transform a MLC problem into a set of binary problems, one for each label; and the Label Power Set transformation, which transform the original MLC problem into a multiclass one, where classes correspond to each possible label combination from the label power set. Binary relevance generates a tree for each label, whereas the label power set generates a single tree model. An adaptation of the C4.5 decision tree algorithm was proposed in [3]. This adaptation computes the sum of labels' entropies for choosing the best attribute to grow the tree, and leaves predict a vector of labels.

Many fuzzy decision tree induction algorithms have been proposed in the literature [4]. Fuzzy decision tree algorithms have been successfully applied to problems in many areas such as decision making, data mining, knowledge engineering and industrial applications [5]. They can be considered as a generalization of crisp decision trees. A fuzzy decision tree allows the transverse of multiple branches of a node with different satisfaction degrees within the range of $[0, 1]$. The most commonly used fuzzy decision tree algorithms is the Fuzzy ID3 algorithm [6]. The main idea of fuzzy ID3 is similar to classical ID3. The key difference between two algorithms is the use of fuzzy entropy to choose the best attribute to grow the tree. Other approaches include Min-Ambiguity algorithm [7], which selects the attribute with the minimum uncertainty as an extended attribute based on possibility theory, and the selection based on the Gini index [8]. A recent approach uses generalized information entropy [9], that can be applied to data sets having numerical and categorical features. Post-processing steps in fuzzy decision tree also includes the adjustment of fuzzy membership functions to fine-tune fuzziness evaluation [10].

There are a few attempts to use fuzzy sets in MLC. [11] proposes a nearest neighbor fuzzy MLC using the approximate reasoning framework of veristic variables, which is competitive to non-fuzzy approaches. [12] also proposes a fuzzy nearest neighbor approach based on fuzzy sets for text classification. They propose a modified fuzzy similarity measure developed for restricting the search space. The authors report that the method performs better than other methods in terms of precision and execution time. [13] proposes a multilabel fuzzy classifier for MLC. A fuzzy relevance measure is adopted to transform high-dimensional documents to low-dimensional fuzzy relevance vectors to avoid the curse of dimensionality. The approach speed-ups classification, as well as produce competitive results with other multilabel approaches. [14] uses fuzzy hyper graph regularization for multilabel subcellular location prediction. They report superior results due to the benefit of exploiting both feature correlations and label correlations. [15] analyses the behavior of FURIA [16], a rule based classifier, associated to problem transformation methods. FURIA achieved good classification performance compared with non fuzzy rule-based systems. To the best of our knowledge, there are no studies involving fuzzy decision tree for multilabel classification.

III. FuzzDT$_{\mathrm{ML}}$

Recent fuzzy decision tree systems generally include four components:

Fuzzy partitioning where linguistic variables are created by fuzzifying numerical attributes. This phase is usually defined either by means of expert knowledge or homogeneously over the input space;

Attribute selection for tree growth where the fuzzified features are evaluated in order to choose the best feature for branching the tree;

Tree pruning which heuristically remove some possible unnecessary tree branches;

Fuzzy partitioning tuning where membership functions are adjusted in the post-processing phase to improve efficiency.

The main objective of this paper is the development of an attribute selection strategy to grow the tree in MLC context. This section describes the proposed FuzzDT$_{\mathrm{ML}}$ algorithm. The algorithm pseudo-code is presented in Algorithm 1. The algorithm takes as input a set of instances $\mathcal{X}$ together to their corresponding label vectors $\mathcal{Y}$, the set of labels $\mathcal{L}$, the fuzzy membership degree of the current node D (which in the beginning of the execution is 1 for all instances), and a pointer reference to the current node (the tree root in the first call). The tree is grown recursively.

Algorithm 1 FuzzDT$_{\mathrm{ML}}$

```
function FUZZDT_ML(𝒳,𝒴,ℒ,D,Node)
    ℒ' = {l_i ∈ ℒ | l_i can be a leaf}
    Node.addNewLeaf(ℒ')
    ℒ'' ← ℒ \ ℒ'
    if ℒ' ≠ ∅ then
        A ← best splitting attribute
        for all A_j ∈ D(A) do
            D' = M(P ∩ A_j)
            Child ← Node.addNewChild(A_j)
            FuzzDT_ML (𝒳,𝒴,ℒ”,D',Child)
        end for
    end if
    return Node
end function
```

An interesting feature of FuzzDT$_{\mathrm{ML}}$ is that leaves predicting subsets of labels can be generated, and the induction will continue with the remaining labels. From the current label set $\mathcal{L}$, the algorithm first verifies which labels pass the leaf creation criteria. If there is a non-empty subset $\mathcal{L}'$ of labels that can generate a leaf, a new leaf is created predicting the (partial list of) labels $\mathcal{L}'$ which pass these criteria. The leaf generation criteria are based on two parameters:

- Let $f_i^j = M(D \cap j)/M(D), j \in \{0, 1\}$ be the membership state for the relevance($j = 1$)/irrelevance($j = 0$) of label l_i for the current fuzzy partition D. The label l_i is included in $\mathcal{L}'$ if $f_i^j \geq \delta, \forall j \in \{0, 1\}$, with value $l_i = j$

and weight f_i^j, where δ is a parameter defined by the user.
- Let $n = |M(D)| > 0$. The label l_i is included in $\mathcal{L}'$ if $n \leq n_0$, with value $l_i = \max_j(f_i^j)$ and weight f_i^j, where n_0 is a parameter defined by the user.

The parameter δ controls the "purity" of the label in the leaf, whereas the parameter n_0 controls the quantity of instances to continue growing the tree. When the level of purity for the relevance/irrelevance of a label surpasses a threshold (δ), or when the number of instances with non zero membership degree is below a minimum (n_0), a leaf node is created.

The possibility to create leaves with partial labels can naturally incorporate in the model some aspects of label dependency [17]. An example of a decision tree of a toy multilabel data set with 10 features and 5 labels, generated using the synthetic data set generator for multilabel learning[1] and available in the `utiml` R package [2] is shown in Figure 1. It can be seen from figure that the branch $Att_1 = low$ has a leaf with partial labels $[y_1, y_2, y_3, y_5]$, while the value of label y_4 depends on the sibling branch on Att_2. A similar situation occurs in the branch $Att_1 = high$, where a partial leaf with labels $[y_1, y_3, y_4, y_5]$ exists, and the value of label y_2 depends on the sibling branch on Att_9.

If the set difference $\mathcal{L}''$ between the label set $\mathcal{L}$ and the set of labels which became leaves in the current execution $\mathcal{L}'$ is not empty, the algorithm continues the tree growth by choosing the best attribute to split. The choice of the best attribute is based on an adaption of the generalized fuzzy information [9] for MLC. The fuzzy entropy of condition attribute A_i, with domain $A_{i1}, \ldots, A_{ik}$, is defined as:

$$FE_{l_i}(D, A_i) = \sum_{j=1}^{j=k} \frac{\overline{m_{ij}}}{m_i} E(D, A_{ij}) \tag{1}$$

$$E_{l_i}(D, A_{ij}) = - \sum_{l \in \{0,1\}} \frac{m_{ijl}}{\overline{m_{ij}}} \log_2 \frac{m_{ijl}}{\overline{m_{ij}}} \tag{2}$$

where $m_{ij} = M(D \cap A_{ij})$, $m_i = \sum_{j=1}^{k} m_{ij}$, $m_{ijl} = M(D \cap A_{ij} \cap l_l)$, $\overline{m_{ij}} = \sum_{l \in \{0,1\}} m_{ijl}$. $M(\cdot)$ is the membership function of a fuzzy partition.

As in [3], the extension of FE to the MLC case is defined as the sum of $FE_{l_i}, \forall l_i \in \mathcal{L}$, as shown in Equation 3. The attribute with minimum FE is shown to grown the tree. Observe that if a leaf with partial labels has been created as an ancestor of the current node, they are not taken into account for computing FE.

$$FE = \sum_{l_i \in \mathcal{L}} FE_{l_i} \tag{3}$$

One of the characteristics of the generalized fuzzy entropy as proposed by [9] is that it can be applied to data sets with numerical (fuzzifyied) and categorical attributes.

[1] http://sites.labic.icmc.usp.br/mldatagen/

[2] https://CRAN.R-project.org/package=utiml

Finally, the induced tree is converted to a rule base, in order to classify new instances. This allow the use of different fuzzy reasoning configurations. Tree pruning and fuzzy partition adjustment were not implemented yet.

IV. Experimental Evaluation

To gain some insights in the performance of our proposed algorithm, we used 9 multilabel data sets available from the `mldr` repository [18] for evaluation. The main data set characteristics are shown in Table I. For each data set, the table shows the number of instances; number of input features (the number of categorical/numeric features are shown in brackets); number of labels; number of label sets; number of single label sets; cardinality (average number of relevant labels per instance); density (average proportion of relevant labels per instance); mean label imbalance ratio; SCUMBLE (concurrence among frequent and rare labels) [19]; and theoretical complexity score [20]. The first four data sets contain numerical features only, and following four contains categorical features only. The ninth data set contains both numerical and categorical features.

We compare the behavior of FuzzDT$_{\text{ML}}$ with three baselines:

BR(J48) binary relevance problem transformation method, using J48 as base classifier;

LPS(J48) label power set problem transformation method, using J48 as base classifier;

MLC45 multilabel extension of C4.5, as proposed in [3].

BR and LPS are implemented in `mulan` [30], and use the J48 `weka` [31] implementation of C4.5 decision tree algorithm. MLC45 is implemented in `clus`[3]. FuzzDT$_{\text{ML}}$ was implemented in Java, using the fuzzy decision tree toolkit implementation[4] as base. For classifying new instances, the `jfuzzylite` [32] was used as the inference system. The minimum t-norm was used form rule conjunction, while the maximum s-norm for inference disjunction. Numerical attributes were fuzzified using triangular membership functions, with tree fuzzy partitions for each attribute. Experiments were run using 10-fold cross validation. The parameter δ in FuzzDT$_{\text{ML}}$ was set to 0.8, as suggested by [9]. The parameter n_0 was set 5, as the default parameter in J48. The evaluation was performed using three different performance measures: Hamming Loss, Ranking Loss, and Micro-averaged AUC.

BR(J48) induces an independent decision tree for each label. On the other hand, LPS(J48), MLC45 and FuzzDT$_{\text{ML}}$ induces an overall, single model for all labels. Thus, LPS(J48), MLC45 and FuzzDT$_{\text{ML}}$ can be considered as more interpretable models than BR(J48).

Hamming Loss (Equation 4) is the average Hamming distance between the actual true label vector (Y) and predicted label vector (Z). The Hamming distance is the symmetrical (xor) difference between the two vectors, normalized by the vector size. As this is a loss function, the lower its value the better, and the lower bound is zero.

[3] http://clus.sourceforge.net

[4] https://github.com/mhjabreel/FDTKit

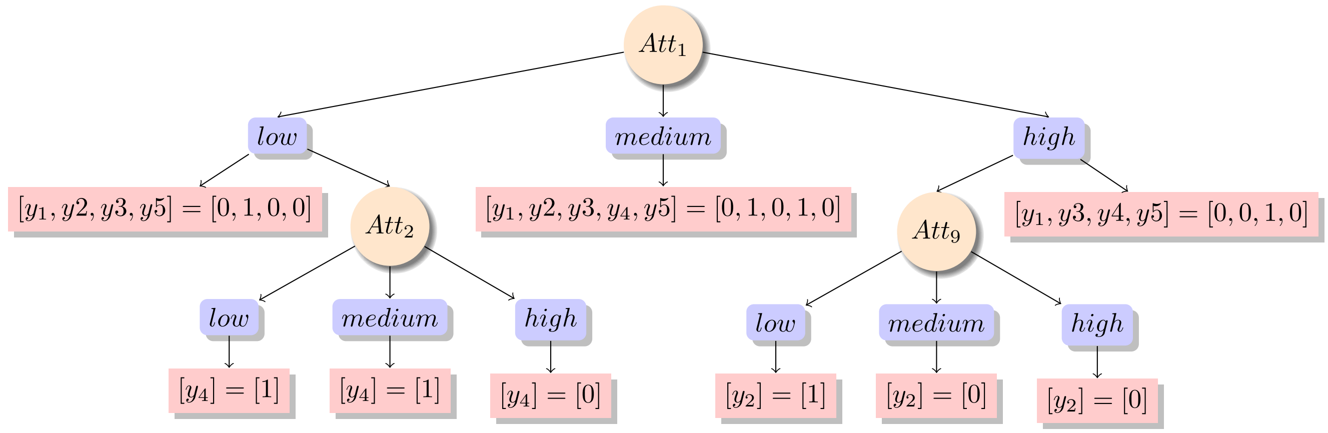

Fig. 1. An example of a MLC fuzzy decision tree induced by FuzzDT$_{\text{ML}}$

TABLE I
MULTILABEL DATA SET CHARACTERISTICS

dataset	num. instances	num. features	num. labels	num. labelsets	num. single labelsets	cardinality	density	meanIR	SCUMBLE	TCS
cal500 [21]	502	68 (0/68)	174	502	502	26.044	0.150	20.578	0.372	15.597
emotions [22]	593	72 (0/72)	6	27	4	1.868	0.311	1.478	1.265	9.364
scene [23]	2407	294 (0/294)	6	15	3	1.074	0.179	1.254	4.251	10.183
yeast [24]	2417	103 (0/103)	14	198	77	4.237	0.303	7.197	1.064	12.562
genbase [25]	662	1186 (1186/0)	27	32	10	1.252	0.046	37.315	3.614	13.840
medical [26]	978	1449 (1449/0)	45	94	33	1.245	0.028	89.501	3.043	15.629
slashdot [27]	3782	1079 (1079/0)	22	156	56	1.181	0.054	17.693	4.396	15.125
tmc2007 [28]	28596	500 (500/0)	22	1172	408	2.220	0.101	17.134	0.967	16.372
flags [29]	194	19 (9/10)	7	54	24	3.392	0.485	2.255	1.103	8.879

$$HLoss = \frac{1}{|\mathcal{X}|} \sum \frac{Y \Delta Z}{|\mathcal{L}|} \quad (4)$$

The average Hamming Loss for each data set is shown in Table II. For calculating the predicted labels, the fuzzy output of FuzzDT$_{\text{ML}}$ was binarized considering and threshold of 0.5. The best (lowest) result for each data set is highlighted in bold. BR(J48) achieved the best results in 5 data sets, while FuzzDT$_{\text{ML}}$ and MLC45 achieved the best results is two data sets each. Label power set did not achieve the best result in any data set.

TABLE II
AVERAGE HAMMING LOSS ↓

dataset	BR(J48)	LPS(J48)	MLC45	FuzzDT$_{\text{ML}}$
cal500	0.1610	0.2014	0.1371	**0.1367**
emotions	0.2497	0.2734	**0.2421**	0.2490
scene	**0.1311**	0.1494	0.1341	0.1573
yeast	0.2467	0.2778	0.2250	**0.2244**
genbase	0.0484	0.0660	**0.0090**	0.0463
medical	**0.0104**	0.0131	0.0229	0.0216
slashdot	**0.0422**	0.0548	0.0497	0.0525
tmc2007	**0.0550**	0.0706	0.0721	0.0827
flags	**0.2577**	0.2861	0.2661	0.3076

A statistical comparison of these algorithms can be visualized in Figure 2. This figure plots the average (Friedman) rank diagram for each algorithm. Although FuzzDT$_{\text{ML}}$ only appears in the third position, according to the aligned rank with Holm p-value correction multiple comparison procedure [33], with 95% confidence level, no statistical difference exists among the two first ranked algorithms and FuzzDT$_{\text{ML}}$ (indicated by a line joining the three algorithms). However, it is interesting to note that FuzzDT$_{\text{ML}}$ performs quite well in the data sets with numerical attributes (the two best performances occur within these data sets). These data sets are the ones which most benefice from the fuzzification process. The good performance of BR(J48) in terms of Hamming Loss in general, and with data sets with categorical attributes in particular, can be explained by the creation of individual models for each label.

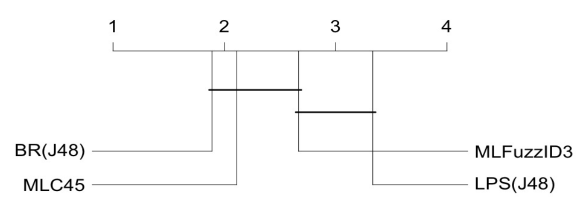

Fig. 2. Average ranks diagram for Hamming Loss

Although Hamming loss is one of the most used measures for evaluating MLC, it ignores the scoring information provided by the algorithms, as only the crisp classification values are taken into account. To overcome this limitation, we also evaluated two measures which uses the scores provided by the algorithms, analyzing the ranking of relevant/irrelevant labels that can be derived from these scores.

The second measure used to evaluate the algorithms is Ranking Loss (Equation 5). This measure ranks all labels according to the likelihood of being relevant, and then takes into account all possible combinations of relevant and irrelevant labels. The measure count how many times an irrelevant label ($y_{ir} \in \overline{Y_i}$) has a higher rank than a relevant label ($y_r \in Y_i$). The measure is normalized by the product of the number of relevant and irrelevant labels. The lower the ranking loss, the better the performance, according to this measure.

$$RLoss = \frac{1}{|\mathcal{X}|} \sum \frac{1}{|Y_i| \cdot |\overline{Y_i}|} |y_r, y_{ir} : r(x_i, y_r) < r(x_i, y_{ir})| \quad (5)$$

The average ranking loss is shown in Table III. For FuzzDT$_{\mathrm{ML}}$, the fuzzy output was used as the scoring function to rank the labels, whereas for the other algorithms the scores of relevance of the labels. MLC45 achieved the best (lowest) result in five data sets, followed by BR(J48) and FuzzDT$_{\mathrm{ML}}$, with the best results in two data sets each. Furthermore, FuzzDT$_{\mathrm{ML}}$ again achieved good results in the data sets which have numerical attributes, where the two best performance was obtained. Label power set did not achieve the best result in any data set.

TABLE III
AVERAGE RANKING LOSS ↓

dataset	BR(J48)	LPS(J48)	MLC45	FuzzDT$_{\mathrm{ML}}$
cal500	0.2968	0.6550	**0.1807**	0.1811
emotions	0.2977	0.3330	0.2624	**0.2087**
scene	0.2362	0.2199	**0.1862**	0.2409
yeast	0.3130	0.4015	0.2033	**0.1952**
genbase	0.6040	0.6039	**0.0062**	0.3797
medical	**0.0663**	0.1364	0.1119	0.1122
slashdot	**0.1389**	0.2586	0.1930	0.1876
tmc2007	0.1099	0.3230	**0.0954**	0.1401
flags	0.2463	0.4910	**0.1998**	0.2517

A statistical comparison using the ranking loss is shown in Figure 2, which also plots the average (Friedman) rank diagram for each algorithm. According to the aligned rank with Holm p-value correction multiple comparison procedure [33], the algorithms can be grouped within two groups of no statistical differences: MLC45, FuzzDT$_{\mathrm{ML}}$ and BR(J48); and FuzzDT$_{\mathrm{ML}}$, BR(J48), and LPS(J48). FuzzDT$_{\mathrm{ML}}$ was ranked second, with no statistical differences between the first and third ranked algorithms. BR(J48), which achieved the best mean rank score in terms of Hamming Loss, is ranked third in terms of Ranking Loss. A possible reason to this fact is that the binary relevance transformation considers the labels in isolation, contrary to the other methods.

The third measure used for comparing algorithms is Micro-averaged AUC[5] (microAUC - Equation 6). MicroAUC also use the ranking information provided the scores, but differently from ranking loss, which compares ranks of the labels for each instance, microAUC computes the fraction of pairs of relevant labels ranked over irrelevant ones, no matter which instances they belong to. The higher the value of microAUC, the better the algorithm, according to this measure.

[5]Area under the ROC curve

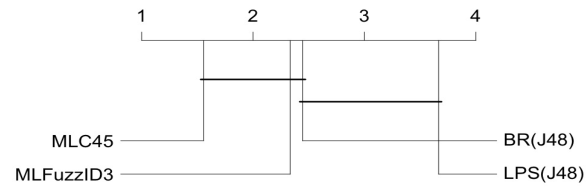

Fig. 3. Average ranks diagram for Ranking Loss

$$microAUC = \frac{|x_r, y_r, x_{ir}, y_{ir} : r(x_r, y_r) \geq r(x_{ir}, y_{ir})|}{|Y_r| \cdot |Y_{ir}|} \quad (6)$$

Table IV shows the average microAUC values for each data set for the four algorithms. MLC45 achieved the best value in 6 data sets, BR(J48) in two, and FuzzDT$_{\mathrm{ML}}$ in one data set. LPS(J48) did not achieve the best microAUC in any data set.

TABLE IV
AVERAGE MICROAUC ↑

dataset	BR(J48)	LPS(J48)	MLC45	FuzzDT$_{\mathrm{ML}}$
cal500	0.7019	0.4312	**0.8157**	0.7654
emotions	0.7038	0.7140	0.7693	**0.7906**
scene	0.7553	0.7601	**0.8341**	0.7079
yeast	0.6863	0.6710	**0.7948**	0.7753
genbase	0.5265	0.5806	**0.9927**	0.6387
medical	**0.9283**	0.8871	0.9015	0.9031
slashdot	**0.8538**	0.7379	0.8085	0.8140
tmc2007	0.8783	0.7881	**0.9020**	0.8519
flags	0.7641	0.6489	**0.8121**	0.7258

Figure 4 shows the statistical comparison among the four algorithms by plotting the average (Friedman) ranks of each algorithm. According to the aligned rank with Holm p-value correction multiple comparison procedure [33], the algorithms can be grouped within two groups of no statistical differences: MLC45, FuzzDT$_{\mathrm{ML}}$, and BR(J48); FuzzDT$_{\mathrm{ML}}$, BR(J48), and LPS(J48). Although FuzzDT$_{\mathrm{ML}}$ only achieves the best microAUC in one data set, it is ranked second in terms of microAUC, and no statistical differences was detected when compared to the first and third ranked algorithms.

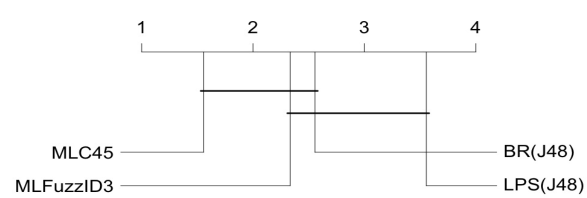

Fig. 4. Average ranks diagram for microAUC

By analyzing the three performance measures, we can see a common pattern. FuzzDT$_{\mathrm{ML}}$ is statistically comparable with MLC45 and BR(J48), and achieved good performance with numerical attributes. These attributes are benefited by the fuzzification process, showing the suitability of our proposed method for these data sets. This results can be considered very satisfactory, as FuzzDT$_{\mathrm{ML}}$ does not yet have pruning mechanisms, and fuzzy partition adjustment could be used to improve performance [10].

V. Conclusion

This paper presents FuzzDT$_{\mathrm{ML}}$, a fuzzy decision tree MLC. To the best of our knowledge this is the first multilabel fuzzy decision tree algorithm proposed in the literature. FuzzDT$_{\mathrm{ML}}$ produces a single model. Furthermore, leaves with partial labels can be induced. These factors contribute to model interpretability. Although FuzzDT$_{\mathrm{ML}}$ can be applied to data sets with categorical or mixed type attributes, it achieved good performance in data sets with numerical attributes. This fact shows the suitability of the fuzzification process in MLC decision tree induction.

Future research directions include the research of better ways to cope with data sets with categorical and mixed type features. Other open-ended issues include the development of pruning techniques and fuzzy partition adjustment, as well as developing enhancing the dealing with rare labels.

Acknowledgments

This work was carried out while the first author was visiting University of Granada. This work have been partially supported by the São Paulo State (Brazil) research council FAPESP under project 2015/20606-6, and by the Spanish National Research Project TIN2014- 57251-P, and the Andalusian Research Project P11-TIC-7765.

References

[1] F. Herrera, F. Charte, A. J. Rivera, and M. J. del Jesus, *Multilabel Classification*. Springer, 2016.

[2] L. Rokach and O. Maimon, *Data mining with decision trees: theory and applications*. World scientific, 2014.

[3] A. Clare and R. D. King, "Knowledge discovery in multi-label phenotype data," in *Proceedings of the 5th European Conference on PKDD*, 2001, pp. 42–53.

[4] I. Chiang and J. Hsu, "Fuzzy classification trees for data analysis," *Fuzzy Sets Systems*, vol. 130, no. 1, p. 8799, 2002.

[5] A. Altay and D. Cinar, "Fuzzy decision trees," in *Fuzzy Statistical Decision-Making*, ser. Studies in Fuzziness and Soft Computing, C. Kahraman and O. Kabak, Eds. Springer, 2016, vol. 343.

[6] M. Umanol, H. Okamoto, I. Hatono, H. Tamura, F. Kawachi, S. Umedzu, and J. Kinoshita, "Fuzzy decision trees by fuzzy id3 algorithm and its application to diagnosis systems," in *Fuzzy Systems, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the Third IEEE Conference on*. IEEE, 1994, pp. 2113–2118.

[7] Y. Yuan and M. J. Shaw, "Induction of fuzzy decision trees," *Fuzzy Sets and systems*, vol. 69, no. 2, pp. 125–139, 1995.

[8] B. Chandra and P. P. Varghese, "Fuzzifying gini index based decision trees," *Expert Systems with Applications*, vol. 36, no. 4, pp. 8549–8559, 2009.

[9] C. Jin, F. Li, and Y. Li, "A generalized fuzzy id3 algorithm using generalized information entropy," *Knowledge-Based Systems*, vol. 64, pp. 13–21, 2014.

[10] J. Sanz, H. Bustince, A. Fernández, and F. Herrera, "Iivfdt: Ignorance functions based interval-valued fuzzy decision tree with genetic tuning," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 20, no. supp02, pp. 1–30, 2012.

[11] Z. Younes, F. Abdallah, and T. Denoeux, "Fuzzy multi-label learning under veristic variables," in *IEEE International Conference on Fuzzy Systems (IEEE-FUZZ 2010)*. IEEE, 2010, pp. 1–8.

[12] J. Jiang, S. Tsai, and S. Lee, "FSKNN: multi-label text categorization based on fuzzy similarity and k nearest neighbors," *Expert Syst. Appl.*, vol. 39, no. 3, pp. 2813–2821, 2012.

[13] S. Lee and J. Jiang, "Multilabel text categorization based on fuzzy relevance clustering," *IEEE T. Fuzzy Systems*, vol. 22, no. 6, pp. 1457–1471, 2014.

[14] J. Chen, Y. Y. Tang, C. Chen, B. Fang, Y. Lin, and Z. Shang, "Multi-label learning with fuzzy hypergraph regularization for protein subcellular location prediction," *IEEE Trans. on NanoBioscience*, vol. 13, no. 4, pp. 438–447, 2014.

[15] R. C. Prati, "Fuzzy rule classifiers for multi-label classification," in *Fuzzy Systems (FUZZ-IEEE), 2015 IEEE International Conference on*. IEEE, 2015, pp. 1–8.

[16] J. C. Hühn and E. Hüllermeier, "FURIA: an algorithm for unordered fuzzy rule induction," *Data Min. Knowl. Discov.*, vol. 19, no. 3, pp. 293–319, 2009.

[17] K. Dembczyński, W. Waegeman, W. Cheng, and E. Hüllermeier, "On label dependence and loss minimization in multi-label classification," *Machine Learning*, vol. 88, no. 1-2, pp. 5–45, 2012.

[18] F. Charte, D. Charte, A. Rivera, M. J. del Jesus, and F. Herrera, "R ultimate multilabel dataset repository," in *International Conference on Hybrid Artificial Intelligence Systems*. Springer, 2016, pp. 487–499.

[19] F. Charte, A. J. Rivera, M. J. del Jesus, and F. Herrera, "Concurrence among imbalanced labels and its influence on multilabel resampling algorithms," in *Proc. of the 9th International Conference on Hybrid Artificial Intelligent Systems (HAIS'2014)*, ser. LNAI, vol. 8480. Springer, 2014.

[20] F. Charte, A. Rivera, M. J. del Jesus, and F. Herrera, "On the impact of dataset complexity and sampling strategy in multilabel classifiers performance," in *International Conference on Hybrid Artificial Intelligence Systems*. Springer, 2016, pp. 500–511.

[21] D. Turnbull, L. Barrington, D. Torres, and G. Lanckriet, "Semantic annotation and retrieval of music and sound effects," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 16, no. 2, pp. 467–476, 2008.

[22] A. Wieczorkowska, P. Synak, and Z. Ra's, "Multi-label classification of emotions in music," in *Intelligent Information Processing and Web Mining*, 2006, vol. 35, ch. 30, pp. 307–315.

[23] M. Boutell, J. Luo, X. Shen, and C. Brown, "Learning multi-label scene classification," *Pattern Recognition*, vol. 37, no. 9, pp. 1757–1771, 2004.

[24] A. Elisseeff and J. Weston, "A kernel method for multi-labelled classification," in *Advances in Neural Information Processing Systems*, vol. 14, 2001, pp. 681–687.

[25] S. Diplaris, G. Tsoumakas, P. Mitkas, and I. Vlahavas, "Protein classification with multiple algorithms," in *Proc. 10th Panhellenic Conference on Informatics, Volos, Greece, (PCI'2005)*, 2005, pp. 448–456.

[26] K. Crammer, M. Dredze, K. Ganchev, P. P. Talukdar, and S. Carroll, "Automatic code assignment to medical text," in *Proc. Workshop on Biological, Translational, and Clinical Language Processing, Prague, Czech Republic, (BioNLP'2007)*, 2007, pp. 129–136.

[27] J. Read, B. Pfahringer, G. Holmes, and E. Frank, "Classifier chains for multi-label classification," *Machine Learning*, vol. 85, pp. 333–359, 2011.

[28] A. N. Srivastava and B. Zane-Ulman, "Discovering recurring anomalies in text reports regarding complex space systems," in *Aerospace Conference*, 2005, pp. 3853–3862.

[29] E. C. Goncalves, A. Plastino, and A. A. Freitas, "A genetic algorithm for optimizing the label ordering in multi-label classifier chains," in *25th IEEE International Conference on Tools with Artificial Intelligence (ICTAI'2013),*, 2013, pp. 469–476.

[30] G. Tsoumakas, E. Spyromitros-Xioufis, J. Vilcek, and I. Vlahavas, "Mulan: A java library for multi-label learning," *Journal of Machine Learning Research*, vol. 12, pp. 2411–2414, 2011.

[31] E. Frank, M. A. Hall, , and I. H. Witten, *The WEKA Workbench. Online Appendix for "Data Mining: Practical Machine Learning Tools and Techniques"*, 2016.

[32] J. Rada-Vilela, "fuzzylite: a fuzzy logic control library," 2014. [Online]. Available: http://www.fuzzylite.com

[33] S. García, A. Fernández, J. Luengo, and F. Herrera, "Advanced non-parametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power," *Information Sciences*, vol. 180, no. 10, pp. 2044–2064, 2010.