

USING DECISION TREES FOR KNOWLEDGE-ASSISTED TOPOLOGICALLY STRUCTURED DATA ANALYSIS

C. Simon, J. Meessen, D. Tzovaras, and C. De Vleeschouwer

Communication and Remote Sensing Lab, UCL, Louvain-La-Neuve, Belgium

Informatics and Telematics Institute, Thermi, Greece

Multitel, Mons, Belgium

ABSTRACT

Supervised learning of an ensemble of randomized trees is considered to recognize classes of events in topologically structured data (e.g. images or time series). We are primarily interested in classification problems that are characterized by severe scarcity of the training samples. The main idea of our paper consists in favoring the selection of attributes that are known to efficiently discriminate the minority class in those nodes of the tree that are close to the leaves and where classes are represented by a small number of training examples. In practice, the knowledge about the ability of an attribute to discriminate the classes represented in a particular node is either provided by an expert or inferred based on a pre-analysis of the entire initial training set. The experimental validation of our approach considers sign language and human behavior recognition. It reveals that the proposed knowledge-assisted tree induction mechanism efficiently compensates for the shortage of the training samples, and significantly improves the tree classifier accuracy in such scenarios.

I. OVERVIEW

Due to the proliferation of sensor and memory technologies in a wide variety of domains - from medical imaging to robotics and multimedia -, the acquisition and storage of topologically structured data have significantly grown over the past years. By definition, topologically structured data are composed of measures of similar nature related to each other by a spatio-temporal neighborhood relation. Examples are images or time series. In this context, our work considers supervised learning to build an ensemble of semi-randomized decision trees to recognize potentially rare classes among some data set. The use of classification trees for shape recognition or temporal data analysis has been shown to achieve competitive classification or detection accuracy in many fields [1][2][3][4]. Section II briefly reviews the reference tree induction mechanism considered in our study.

Our work differentiates from earlier research by the fact that it focuses on classification problems that are characterized by a severe scarcity of the learning samples. This is for example the case in a videosurveillance context, where only a few occurrences of the events of interests (intrusion in a forbidden area, aggression, theft, etc.) are available to learn the classifier. The lack of training samples is known to dramatically impair the performance of the classifier [5]. In our work, we propose to inject some external knowledge into the tree growing process to compensate for the shortage of training examples. For this purpose, we introduce two original and complementary mechanisms.

- On the one hand, the set of attributes envisioned to split a tree-node is augmented, so as to define elaborated, preferably semantically meaningful, concepts. This is done in two steps. First, pre-processing of the input data samples is considered to define augmented features. For example, when the classification problem deals with objects displacements, we promote the use of velocity and acceleration features, next to the initial object position feature. Second, the attributes are defined to characterize the distribution of the augmented features in a window sliding over the data according to their underlying topology. Section III

describes how elaborated attributes can be defined automatically for topologically structured data.

- On the other hand, the attributes that are *a priori* known to be relevant to the classification or detection problem envisioned in a particular node get more chance to be selected to split this node than other attributes. During the tree growing process, the knowledge is injected in nodes where only very few examples of a given class are available. The *a priori* knowledge basically informs the learning system about the relative efficiency of the attributes in discriminating the minority class from the other classes. Section IV explains how this knowledge is formally represented, and how it may be provided by an expert or inferred based on a pre-analysis of the entire initial training set.

The claim supported by our paper is that the complementarity between these two mechanisms allows injecting *a priori* knowledge in the tree growing process, so as to reduce bias and improve classification accuracy. Section V supports our argument based on experiments that are run on data samples composed of a temporal sequence of spatially organized elementary pieces. In the first set of experiments, the data samples correspond to a sequence of hands positions and locations, and the purpose is to recognize the signs from the Australian Sign Language. The second set of simulations considers a video surveillance context. The input data reflect the spatio-temporal displacements of a group of humans, and the objective is to trigger an alarm upon detection of some particular behavior of interest (intrusion in a forbidden area, fight, meeting, etc.). Section VI concludes our paper.

II. ENSEMBLE OF RANDOMIZED TREES

Our system uses the classical top-down procedure to build unpruned decision trees [6] from the learning samples. Each node of the tree can be seen as a weak classifier that organizes the input samples into two branches, based on an attribute and a cut-point parameter. Given a set of labelled samples in a tree node, the growing process selects the attribute and the cut-point in a vast database so as to provide a significant entropy gain, i.e. so as to reduce the impurity of the output variable within the local learning subset [7]. Once the entire tree has been grown, a sample to classify simply falls from the root to one of the leaves, and receives the label of the dominating class among the training samples that reached that leaf during the learning phase.

Similar to numerous previous works, randomization methods are considered to improve the tree decision accuracy. These methods explicitly randomize the tree growing algorithm, and perform multiple runs of the growing process, so as to produce an ensemble of more or less diversified tree models, whose predictions are aggregated by a simple majority vote. In our case, we follow the ExtraTree methodology [7], meaning that we select both the attributes and the cut-points partially at random, but exploit the entire training set to grow every individual tree. This contrasts with bagging approaches, which introduce randomization based on subsampling of the input training data. In our case, we do not want to grow the tree models based on a subset of the training samples because we are interested in scenarios for which the training set is already quite small.

Further subsampling would thus dramatically impact the model accuracy. In the ExtraTree paradigm, attributes and cut-points are then selected at random until finding a split that achieves an entropy gain higher than a pre-defined threshold.

III. DEFINITION AND SELECTION OF ATTRIBUTES FOR THE CLASSIFICATION OF TOPOLOGICALLY STRUCTURED DATA

In this section, we propose to pre-process the raw input data to generate so-called *augmented attributes* that are suited to express semantically meaningful concepts about topologically structured data. We then explain how randomization is implemented efficiently in that particular case.

Previous works have already considered decision trees to classify topologically structured data. In particular, the segment and combine approach presented in [1] recommends to (i) randomly segment the structured data into pieces, (ii) learn a model relating pieces to data classes, and (iii) infer object classification based on the prediction made for its pieces. The approach is elegant and generic. However, inferring accurate models from random segments of raw training data implies sufficiently deep tree models, which in turns requires a large amount of training samples. In our case, as we are interested in scenarios for which only a small number of training examples are available (at least for some of the classes), we recommend to pre-process the raw input data samples so as to create attributes that more directly discriminate the relevant substructures hidden in the data.

In general, each topologically structured data sample corresponds to a set of spatio-temporally organized elements, each element being a potentially multi-agent and multi-dimensional vector. By multi-agent, we mean that a data element might be defined by multiple feature vectors, corresponding to distinct agents or entities involved in the event to recognize. By multi-dimensional, we mean that each agent might be characterized by several features, e.g. its size, position, colour, etc.

To generate the augmented attributes associated to such data, we first derive a set of *augmented features* for each of the components of the multi-dimensional input vector. In our experiments, the augmented features include each of the input vector components, together with the first and second derivative of these components (along the time). When a data element is defined by several agents, the difference between the components of each pair of agents is computed, component by component, and added to the set of augmented features, along with its derivatives.

For a given augmented feature f , the augmented attributes are then defined based on a window sliding over the topologically structure data. Two kinds of attributes are then considered in our work. The first kind checks whether some pre-defined property is valid for all possible positions of the sliding window. The second checks whether a property is valid for at least one window position. Equivalently, when multiple agents are involved in the problem, the attribute defines whether the property is valid for all or at least one of the (pairs of) agents. The set of properties to consider could a priori depend on some knowledge one has about the classification problem at hand. In our experiments, we however limited the choice to three simple and generic properties that characterize the histogram of the feature f over the sliding window w . They check whether either the number of values contained in a given range, or the average, or the variance of f over w lies below some threshold.

During the learning process, each node randomly selects a feature f and one of the above attributes. The information gain provided on the local learning subset by the selected attribute is then estimated, for a set of randomly chosen attribute parameters. The operation is repeated until the split reach enough information gain, or until a pre-defined number of iterations is performed. The

process is similar to the random subspace algorithm introduced in [8]. It is detailed by Algorithm 1. The selection of the attribute a defines both the kind of attribute (\forall, \exists), and the investigated property (average, variance, histogram). The attribute parameters $\{\theta_i\}$ define both the decision cut-point, and the intrinsic parameters of the attribute (e.g. the histogram range). Together, the feature f , the attribute a and the parameters $\{\theta_i\}$ define a possible split in a node. N_1, N_2 and N_3 define the maximum number of loops at different step of the learning process. They are chosen heuristically, to trade-off randomization and computation time. The information gain IG is defined as in [7]. It ranges between 0 and 1, and the level of randomization carried by the generated tree model is proportional to the threshold S_{th} .

Algorithm 1 Training a node

Input: a training set S

Output: a node labelled by a specific test, the split of the training set into two parts.

```

1:  $n_1 = n_2 = n_3 = Score = MaxScore = 0$ 
2: while  $Score < S_{th} \ \& \ n_1 < N_1$  do
3:   Select the feature  $f$ , and the attribute  $a$  randomly
4:   while  $Score < S_{th} \ \& \ n_2 < N_2$  do
5:     Choose the window size  $w$  randomly;
6:     while  $Score < S_{th} \ \& \ n_3 < N_3$  do
7:       Choose the attribute parameters  $\{\theta_i\}$  randomly;
8:        $Score \leftarrow IG(f, a, w, \{\theta_i\})$ ;
9:       if  $Score > MaxScore$  then
10:         $MaxScore = Score$ ;
11:         $s \leftarrow [f, a, w, \{\theta_i\}]$ ;
12:       end if
13:        $n_3 \leftarrow n_3 + 1$ 
14:     end while
15:      $n_2 \leftarrow n_2 + 1$ 
16:   end while
17:    $n_1 \leftarrow n_1 + 1$ 
18: end while
19: Split the node according to  $s$ 

```

IV. KNOWLEDGE REPRESENTATION, ACQUISITION, AND EXPLOITATION

Classification models are generally difficult to infer when few training samples are available compared to the problem dimension. To circumvent the problem, this section proposes to guide the tree induction mechanism by encouraging the selection of split decisions that are a priori known to be particularly relevant regarding the classification task. To implement our idea, the set of possible splits is divided into non-overlapping subsets, based on their involved feature and property. A probability distribution vector (PDV) is then defined over the partition to skew the random selection performed by step 2 of Algorithm 1. The feature and properties that are *a priori* known to be more relevant have more chance to be selected. The main question is then: how can we learn and exploit appropriate PDVs ?

As the PDV basically tells how to look at the data to classify in a node, it should depend on the data at hand, i.e. on the learning samples that have reached the node of interest. For this reason, we propose to compute the PDV based on the number of samples of each class in the node. The method relies on *generative PDVs*, which define how to skew the random selection of feature and attribute when a single class is significantly less represented than one or multiple other classes in a node.

The generative PDVs associated to class i are selected automatically within a pre-defined set \mathcal{P} of candidate PDVs, based on a X-Fold cross-validation procedure. In practice, the set \mathcal{P}

is defined either randomly or based on the user feedback (see Section V-A). It typically contains the reference uniform PDV, together with some additional vectors that hopefully favor the selection of attributes that efficiently discriminate class i .

For example, imagine we are interested in the generative PDV \bar{p}_i to use in a node where few elements of class i are compared to a significant number of elements of all other classes. Each step of the cross-validation can then be described as follows:

1. Divide the learning set (LS) into a Learning-LS (LLS) and a Test-LS (TLS), so that LLS contains significantly less elements of class i than elements from other classes.
2. For each \bar{p} in \mathcal{P} :
 - 2.1. Build an ensemble of trees from LLS, using \bar{p} to randomly select the feature and attribute in a node.
 - 2.2. Compute the classification error for the ensemble of trees over TLS.
 - 2.3. Add this classification error to the sum of errors computed during previous cross-validation steps associated to \bar{p} .

The PDV that minimizes the sum of classification errors when all steps of cross-validation are exhausted is selected to be the generative PDV for class i . One could obviously generalize the procedure to select a PDV $\bar{p}_{i,j}$ for nodes where class i faces elements of class j .

Given the generative PDVs for all (pairs of) classes, we now explain how to compute the PDV \bar{p} to consider in a node characterized by N_i samples of class i . We first observe that when numerous learning samples are available for all classes in the node, generic randomization of the tree induction process works fine. For this reason, \bar{p} has to be defined to ensure that the split decision in a node is only skewed when the number of learning samples of at least one class is small in the node.¹ More specifically, the knowledge carried by the generative PDVs should only support classification in nodes where few learning samples are available for some classes, and the bias introduced by \bar{p} should guide the classifier in discriminating these rarest classes. For this reason, we propose to compute \bar{p} based on the generative PDV of the rarest class in the node. Letting k denote the index of the rarest class, we have:

$$\bar{p} = \frac{N_k - 1}{N_k} \cdot \bar{u} + \frac{1}{N_k} \cdot \bar{p}_k \quad (1)$$

where \bar{u} denotes the uniform PDV. Similar definitions can be imagined when the generative PDVs for a class i are defined with respect to a specific class j . In that case, the generative PDV $\bar{p}_{i,j}$ guiding the discrimination between the rarest class i and the dominating class j of the node is selected.

The method followed to define the PDV in a node results from intuitive arguments and rough heuristics. The procedure is definitely open to discussion and most probably to significant optimization. It needs to be further investigated in future research, but has the merit to support the proof-of-concept experiments presented in Section V.

V. PRELIMINARY RESULTS

We tested our system on two applications, which both involve two agents :

- Sign language interpretation. Auslan is the language of the deaf australian community. The signs were recorded using instrumented gloves, providing 22 measurements corresponding to hand and finger positions and orientations, sampled at 100Hz. A total of 2565 signs divided in 95 classes were collected[3].
- Events recognition in a videosurveillance context. The *no*

¹These nodes are often close to the leaves of the tree, as the number of samples exponentially decrease with the depth of the node in the tree.

Table I. Accuracy with 3 training samples for each class

	W^{out} knowledge	W^{th} knowledge
error	$16 \pm 5 \%$	$11 \pm 3 \%$
% error related to sign <i>boy</i>	$68 \pm 5 \%$	$20 \pm 5 \%$

event, *fighting* and *pocket-picking* classes are recognized based on the trajectories followed by two people on the ground plane.

Our experiments primarily investigate the benefit obtained from knowledge-assisted induction of ensemble tree models. In all experiments, 100 trees have been generated. We also set up the score threshold S_{th} to 0.1, and the N_1 , N_2 and N_3 parameters were set to 20, 30 and 50, respectively.

As a preliminary observation, our experiments have revealed that a class with less training samples also results in less dedicated leaves in each tree. The bias increases when the S_{th} parameter decreases, i.e. when the randomization increases. To compensate for the classification bias caused by the shortage of representative leaves for the those classes, we consider to adapt the decision process by increasing the weight of the votes associated to the trees that classify a sample in a minority class. This method is called "voting privilege", and is further discussed below.

V-A. Australian sign language recognition

In this section, we consider the recognition of the first 10 signs of the Australian sign language, as indexed in [3]. The classification accuracy is estimated based on a ten-fold cross-validation in all experiments.

In a first experiment, we have implemented conventional randomized tree induction mechanisms, with all candidate features and attribute properties having the same probability to be selected. The learning set is composed of 14 signs of each class. The system achieved a 99.3% accuracy, which competes with the best results refered in previous literature [3], [9]. It demonstrates the efficiency of ensemble of randomized trees with our proposed set of augmented features for topologically structured data analysis.

In a second experiment we decreased the number of training samples of every class to 3. At first, all features and attributes had equal chance of being picked in each node. Table I reveals that the classification accuracy severely degrades due to the shortage of learning samples. Moreover, we observe that the sign *boy* is involved in 68% percent of the misclassified samples. A deeper analysis of the errors shows that confusions mainly appear between the sign *boy* and one of the following classes: *alive*, *all*, *change mind*, and *come*. To validate our intuition that introducing a bias in the selection of the node attributes may help in discriminating classes, we decided to define PDVs manually, based on visual comparison of examples of the class *boy* with one of the other classes. The purpose is to identify which features and which attribute property are relevant to discriminate the sign *boy* from each of the other classes. For instance, the sign *boy* and the sign *come* appear to be quite similar, but sign *boy* is moving mainly horizontally whilst the sign *come* moves vertically. It could thus be beneficial to advantage the x and y position of the hand when selecting the feature in the second line of Algorithm 1. In our experiment, we have defined the PDV empirically, by giving a 10 times higher probability to the features that were considered to be relevant based on visual comparison of the different pairs of classes. Afterwards, knowledge-assisted tree models were build as described by Equation (1) for the case where the generative PDVs $\bar{p}_{i,j}$ consider nodes where class i faces elements of class j . Results are provided in Table I.

In a third experiment, the system is trained based on 3 training samples for the sign *boy* and 14 samples for all other classes. The same PDV than the one used in the second experiment is considered to push a priori knowledge in the tree growing

Table II. 3 learning samples for the sign *boy*, and 14 samples for all other classes.

	Overall accuracy	Accuracy for sign <i>boy</i>
W^{out} K & W^{out} VP	90 %	5 %
W^{out} K & W^{th} VP	95.5 %	65 %
W^{th} K & W^{out} VP	94 %	55 %
W^{th} K & W^{th} VP	97.5 %	85 %

process. We compared the accuracy of the classifiers constructed with and without knowledge (K), i.e. with skewed and uniform PDVs. We also implemented the voting privilege (VP) principle, by empirically giving a 3 times higher weight to trees that are voting for the sign *boy*. Results are provided in Table II. We observe that the voting privilege significantly improves the classifier performance. We also observe that a non-uniform PDV is capable to improve the classifier accuracy, which is a novel and quite promising result. It reveals an unexpected asset of tree-based classifiers when dealing with problems that are characterized by a small number of training examples. In particular, it suggests that the accuracy of such classifier can be significantly improved by driving the selection of the node attributes based on some external knowledge. In turns, these results also raise a number of appealing questions for future research, such as the questions related to the selection of appropriate and optimal PDVs in each node. The question is partially explored in the next section.

V-B. Event recognition based on human trajectories

Most of the events occurring in a videosurveillance context involve a relation between a human and an object, or between humans together. Human trajectories then constitute the main information that enable to distinguish the events. In order to get these trajectories, a segmentation and a tracking process can be implemented. In our case, to generate a large number of samples for cross-validation purposes, a matlab program has been written to emulate people trajectories for the three events of interest mentioned above. The code together with a brief explanation of the assumptions underlying the random process generating people trajectories is available in [10].

The variables used to describe the events are the X and Y absolute position, the magnitude of the first and second derivative (speed and acceleration), the angle with the X-axis and its first derivative, the distance and the speed difference between the pair of humans. We first obtained 96% accuracy by training the system with 100 learning samples and testing it on 1000 samples. In a second experiment, we decreased the number of samples to 6 for the event *fight*, and tried to compensate for the reduced number of training examples based on the use of a PDV p_{fight} in nodes where the class *fight* is poorly represented. Voting privilege (VP) has also been considered, by giving a 3 times higher weight to trees that are voting for the class *fight*. To define the PDV p_{fight} we have defined a set of candidate PDVs, and followed the procedure described in Algorithm IV to select the one that is expected to best cope with the lack of *fight* training examples. In particular, in each step of the cross-validation process defined by Algorithm IV, the LLS set has been chosen to include five examples of *fight* and all available examples of other classes. The classification error of step 2.2 is here computed for the single remaining element of the class *fight*. In practice, the error has been defined to measure the percentage of trees in the ensemble that do not classify the remaining element within the *fight* class. We observe that the candidate PDV that achieves the smallest error, and thus defines p_{fight} , is non-uniform and causes less than 32% of errors (instead of 50% with the uniform PDV). More interestingly, Table III provides the classification accuracy obtained when learning tree models based on the uniform and p_{fight} PDV, respectively. We observe that both the overall and

Table III. Accuracy with 3 training samples for event *fight*

	Overall accuracy	Accuracy for event <i>fight</i>
W^{out} K & W^{out} VP	80.5 %	41.5 %
W^{th} K & W^{out} VP	84 %	52 %
W^{out} K & W^{th} VP	91 %	73 %
W^{th} K & W^{th} VP	94 %	82 %

fight accuracy significantly increase when the knowledge carried by the selected p_{fight} PDV is exploited during the tree growing process. We conclude that appropriate PDVs can be selected automatically, which dramatically increases the practical use of our knowledge-assisted tree induction process.

VI. CONCLUSION AND PERSPECTIVE

Our work considers supervised learning of tree classifiers for topologically structured data, and demonstrates that it is possible to improve the classification performance of an ensemble of trees based on knowledge-assisted induction methods.

In a first scenario, the user has the capacity to (dis)advantage the split decision rules that (s)he considers (ir)relevant, and our simulations demonstrate that such intrusion of an expert in the tree growing process potentially increases the accuracy of the system when few training examples are available, or equivalently permits to reduce the number of training samples needed to reach a given level of accuracy.

In a second scenario, we have demonstrated that the knowledge about the attribute relevance can be estimated automatically based on a pre-analysis of the initial training set, without the need for an external expert. This result is important because it opens a novel and original path of research by raising unsolved questions related to the optimality of the procedure considered to acquire and inject of knowledge in the tree growing process.

VII. REFERENCES

- [1] R. Maree, P. Geurts, J. Piater, and L. Wehenkel, "Random subwindows for robust image classification," in *IEEE Int. Conf. on Computer Vision and Pattern Recognition*, Washington, DC, USA, June 2005.
- [2] Y. Amit and D. Geman, "Shape quantization and recognition with randomized trees," *Neural Comput.*, vol. 9, no. 7, 1997.
- [3] M. W. Kadous, *Temporal classification: extending the classification paradigm to multivariate time series*, Ph.D. thesis, 2002.
- [4] J. J. Rodriguez and C. J. Alonso, "Interval and dynamic time warping-based decision trees," in *ACM Symposium on Applied Computing*, Nicosia, Cyprus, 2004, pp. 548–552.
- [5] T. Hertz, A. Hillel, and D. Weinshall, "Learning a kernel function for classification with small training samples," in *ACM Int. Conf. on Machine learning*, Pittsburgh, Pennsylvania, 2006, pp. 401–408.
- [6] L. Breiman, J. Friedman, R. Olsen, and C. Stone, *Classification and Regression Trees*, CRC Press, 1984, ISBN 0412048418.
- [7] P. Geurts, D. Ernst, and L. Wehenkel, "Extremely randomized trees," *Machine Learning*, vol. 63, no. 1, pp. 3–42, April 2006.
- [8] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 8, pp. 832–844, August 1998.
- [9] A. Naftel and S. Khalid, "Motion trajectory learning in the DFT-coefficient feature space," in *IEEE Int. Conf. on Computer Vision Systems*, NY, USA, 2006, pp. 229–232.
- [10] <http://www.tele.ucl.ac.be/view-people.php?id=179>, "