Weinberg and Last *J Big Data* (2019) 6:23
https://doi.org/10.1186/s40537-019-0186-3

Journal of Big Data

RESEARCH Open Access

Selecting a representative decision tree from an ensemble of decision-tree models for fast big data classification

Abraham Itzhak Weinberg* and Mark Last

*Correspondence:
WeinberA@post.bgu.ac.il
Department of Software and Information Systems Engineering, Ben-Gurion University of the Negev, BeerSheba, Israel

Abstract

The goal of this paper is to reduce the classification (inference) complexity of tree ensembles by choosing a single representative model out of ensemble of multiple decision-tree models. We compute the similarity between different models in the ensemble and choose the model, which is most similar to others as the best representative of the entire dataset. The similarity-based approach is implemented with three different similarity metrics: a syntactic, a semantic, and a linear combination of the two. We compare this tree selection methodology to a popular ensemble algorithm (majority voting) and to the baseline of randomly choosing one of the local models. In addition, we evaluate two alternative tree selection strategies: choosing the tree having the highest validation accuracy and reducing the original ensemble to five most representative trees. The comparative evaluation experiments are performed on six big datasets using two popular decision-tree algorithms (J48 and CART) and splitting each dataset horizontally into six different amounts of equal-size slices (from 32 to 1024). In most experiments, the syntactic similarity approach, named SySM—Syntactic Similarity Method, provides a significantly higher testing accuracy than the semantic and the combined ones. The mean accuracy of SySM over all datasets is 0.835 ± 0.065 for CART and 0.769 ± 0.066 for J48. On the other hand, we find no statistically significant difference between the testing accuracy of the trees selected by SySM and the trees having the highest validation accuracy. Comparing to ensemble algorithms, the representative models selected by the proposed methods provide a higher speed for big data classification along with being more compact and interpretable.

Keywords: Big data, Ensemble learning, Lazy ensemble evaluation, Decision trees, Editing distance, Tree similarity

Introduction

With the vast growth of information volume and variety in the recent years, many organizations focus on big data platforms and technologies [6]. In order to train machine learning algorithms on big data there is a need for a distributed framework such as MAPREDUCE, which can induce in parallel multiple models out of small subsets of massive-scale training data, which cannot fit into the memory of a single machine. Here, we limit our discussion to the model combining phase of distributed data processing, known as REDUCE. More specifically, we focus on induction of decision tree models.

Due to their simplicity and classification effectiveness [7], decision trees are among the most popular models for data mining analysts and users. In addition, they provide human-readable classification rules, which is considered important in data analytics. The goal of our methodology is to reduce the classification (inference) complexity of tree ensembles by choosing a single representative model out of multiple decision trees induced by ensemble methods. This is an extreme case of a *lazy ensemble evaluation* scheme [5] that uses a minimal subset of ensemble members to make a fast and accurate prediction.

In [38], we have introduced the SySM (Syntactic Similarity Method) approach, which chooses the model, which is syntactically most similar to other decision-tree models, as the best representative of the entire big dataset. In this paper, we compare it to the semantic similarity approach as well as to a combination of the semantic and the syntactic similarity metrics. In addition, we compare SySM results to a randomly chosen decision tree and to the most accurate tree on a validation set. This paper also evaluates the accuracy of a reduced ensemble of the top five most representative decision trees. The similarity-based approach to selecting a single representative decision tree is more computationally efficient than algorithms like [26, 33] that use time consuming activities in the REDUCE phase such as sorting of attribute values or [31] that calculates covariance between the induced models. One of the advantages of using a single representative decision tree is a higher inference (classification) speed. In this way, we save the computational effort needed for traversing multiple trees in the ensemble approach or going back to the raw data for additional computation like in [4]. Such evaluation efficiency is an important issue when dealing with big data environments and massive data streams. The popular ensemble methods yield a relatively high accuracy, yet their model is not necessarily interpretable [15]. An additional advantage of the similarity approach is related to data privacy preservation, since there is no need to expose the local training data (e.g., belonging to different hospitals) after the distributed induction phase. Only the representative model is retained for classifying new records. Moreover, there is no need to reveal the source of the selected model, which provides an additional privacy level.

Panda et al. [30] present PLANET, a scalable distributed framework for merging induced decision trees into one decision tree. PLANET involves a significant communication overhead for data transfer between computing nodes. Magana-Mora and Bajic [25] offer OmniGA, a framework for the optimization of omnivariate decision trees based on a parallel genetic algorithm, coupled with deep learning structure and ensemble learning methods. Since the framework involves deep learning, it has a limited interpretability. The decision tree merging approach of [4] needs extensive computational resources in the model combining phase due to manipulation of decision trees and the original dataset. In contrast, the similarity-based approach introduced by us in [38] needs a smaller amount of computational resources, since it only chooses one of the trees induced in the distributed training phase.

Another method proposed for dealing with multiple decision trees and fitting them all together to the entire dataset is presented in [22]. The proposed approach transforms induced decision trees into Fourier spectra and merges them by vector addition in the dual space. This approach is complete and technically sound; however, it is difficult to

extend to non-binary trees. In addition, it is unclear how such extension will affect the algorithm performance, which is an important metric in the big data framework. An additional method, CMM, introduced in [13] is aimed at improving the model stability. This is achieved by providing the base learner with a new training set, composed of a large number of examples generated and classified according to the ensemble, plus the original examples, which would be infeasible in the big data framework. The described approach consumes extensive computational resources, especially in the combining phase and in the phase of running all models.

Our approach does not affect the distributed processing phase and hence does not change the amount of data traffic between computation nodes.

According to [7, 8] the challenge of large-scale learning is to cope with the increasing ratio between the amount of available data to the processing time constraints. Our algorithm decreases the computational effort of the REDUCE phase by making only one pass over the induced decision trees with no need to reprocess the original data. We propose a novel tree selection approach, which does not follow a centralized tree-growing schema and does not need any significant computation resources for tree merging. According to [7], the classical merging methods for building one tree from induced trees re-sort all numerical attributes of the original dataset for node splitting decisions. This becomes costly in terms of running time and memory size, especially when decision trees are induced from large datasets.

According to [4], distributed learning approaches should also consider the practical limitations imposed by the computing environment, including constraints on the memory of individual nodes, size of exchanged messages, and the communication patterns influencing scalability. This can be done by reducing memory and data transfer requirements at the cost of accuracy. The similarity-based algorithms add only a minor computational effort to the REDUCE phase and, hence, add only a slight overhead to the processing time. Thus in case of the syntactic approach (SySM), the only additional step is transformation of the trees into the tree bracket format and comparing them to each other. As a result, the tree selection phase (REDUCE) takes less time than the tree merging methods such as [4].

To sum-up, the original contributions of our approach are the following:

- We present a computationally-efficient solution for the challenge of selecting representative decision tree models induced in a distributed framework, including secured environments, where confidentiality of local data is required.
- The representative tree is chosen out of an ensemble of induced trees rather than built from the global dataset.
- The representative tree is found by computing pairwise syntactic and \or semantic similarity of induced trees.
- The proposed methods are evaluated in terms of their testing accuracy and classification (inference) time on six benchmark datasets.

Part of this work was presented at the 6th International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications, KDD 2017, August 14, Halifax, Nova Scotia, Canada.

The rest of this paper consists of four sections. "Background and related work" section discusses the possible ways of inducing global decision tree models from big data. "Methodology" section presents the tree selection algorithms based on syntactic and semantic similarity measures. "Empirical evaluation" section discusses the experimental results obtained on six benchmark datasets using two popular decision-tree algorithms. Finally, "Conclusions" section summarizes the presented work and the future research directions.

Background and related work

The challenge of the REDUCE phase in big data processing is to build a global model that will be both accurate and interpretable and that can be induced and applied to new records with minimal computing resources.

Usually, there is a trade-off between all these characteristics and there is no perfect solution that excels in all. We can categorize the approaches dealing with decision tree induction from big data as follows: building one big tree [2, 4, 11, 14, 17, 28–30, 32, 35, 41], transferring all decision trees into one rule base and back into a decision tree, ensemble approaches [9, 18, 23, 24], and other approaches like [22] that do not build any tree and use a combination of tree results. According to [7], another way to categorize the different types of algorithms for handling large datasets is to divide them into the following two groups: pre-sorting of the data and using approximate representations of the data. Under the first category we can mention SLIQ [26], its newer version SPRINT [33] and ScalParC [21]. The second group includes algorithms that approximate representations of the data by sampling and histograms construction. This group includes the following algorithms: BOAT [16], CLOUDS [1], and SPIES [20]. Usually, pre-sorting techniques are more accurate but they are computationally intensive when running on big data sets.

As shown in Fig. 1, distributed decision tree construction algorithms can be grouped by different approaches to parallelism: task parallelism, data parallelism and hybrid parallelism [3, 36]. Task parallelism distributes the decision tree nodes among the processors in a dynamic way. Data parallelism distributes the training set among the processors in a way that each processor is responsible for a distinct part of the data. This category may be divided into two sub categories: horizontal partitioning and vertical partitioning. The parallel strategy based on vertical data distribution [3, 12] splits the data by letting each processor test different attributes whereas horizontal parallelism partitions the data

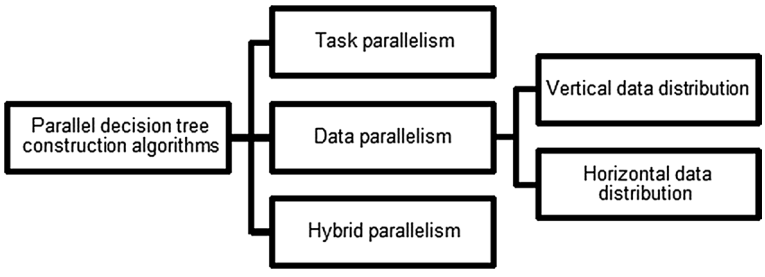


Fig. 1 Taxonomy of parallel decision tree construction algorithms

so that different processors see different rows. The hybrid parallelism uses data parallelism as a combination between horizontal and vertical approaches. Its decision whether to use horizontal or vertical parallelism is a function of the processing capability of each computing node and the constraints of communication volume between them. The tree selection algorithms presented in this paper deal with decision trees constructed by the horizontal parallelism approach.

Building a new decision tree from several induced decision trees is a well-known approach in big data. This approach usually excels in accuracy but needs significant computing resources [7]. The computing resources are needed for controlling the distributed stage and for dividing the database in a specific way [30] as well as for merging subtrees in the post-processing phase [4, 29, 30, 41]. The need for extensive computational resources and the long processing time are considered as major disadvantages in cases where fast results are needed for decision making.

A framework for comparing between induced decision trees is proposed in [27]. They use two types of similarity: semantic similarity and dataset similarity. Semantic similarity is computed in terms of the agreement of class predictions the decision trees return over the attribute space. The dataset similarity is based on the attribute space probability distribution, attribute-class joint probability distribution, and attribute conditional class probability distribution. The presented framework can be used to decide when to update the global decision tree nodes and values as a result of a change in data. In addition, there is a need to go back to the raw data in order to compute probabilities.

Most of the methods mentioned above do not treat the induced decision trees as final entities. They use the induced decision trees as a basis for additional computations that go back and forth to the original dataset. From a computational point of view, this approach causes redundant work since it requires going back to the decision tree induction phase that was already completed.

Miglio and Soffritti [27] present a tree similarity algorithm is based on semantic approach. Their approach is based on [34]. The RTED algorithm [32] used in our methodology is more robust than the approach [34] and it works well for different decision tree structures. Moreover, the tree selection phase of the semantic approach requires additional running time for applying each induced decision tree to the validation instances.

Methodology

Our goal is to select the best single tree that can represent the entire massive dataset after it was partitioned horizontally into multiple slices. The similarity-based approach to tree selection assumes that the most representative tree should be most similar to all other induced trees.

The tree similarity can be calculated by syntactic and semantic metrics or their combination. Following [40], we calculate the syntactic similarity of two trees by a simple and fast editing distance algorithm called RTED (Robust Tree Edit Distance).

The RTED algorithm [32] counts the node edit operations that transform one tree into another.

We measure the syntactic similarity of internal nodes in the two compared decision trees using the following two parameters: the tested attribute name and the node

postorder position in the tree. These two parameters practically determine the tree structure. The node's position determines its interrelation to other nodes in the tree and hence its influence on the model outcome. Since we assume the two trees to be induced from random subsets of the same dataset, we also expect the derived split values of the same continuous attribute to be close to each other.

An alternative tree similarity measure is the semantic similarity. We use it instead of the syntactic similarity in Algorithm 2, since decision trees induced from different slices of the same dataset may still have a different structure. This is because decision tree algorithms are known to be instable, i.e. small variations in the data may result in significant changes in the tree structure. This problem can be mitigated by ensemble or semantic similarity approaches.

When choosing a single representative model out of multiple decision trees, we are not interested to go back to the original training data or to modify the induced decision trees.

Algorithm 1 Syntactic Similarity algorithm (SySM) [38]

INPUT: Group T of n induced decision trees, where n = number of dataset slices
OUTPUT: Chosen Decision Tree T_{cm} where $cm \in [1..n]$

if T is null **then**
  return $failure$
end if
for all $i = 1$ to n **do**
  Set NM = list of node attribute names of $\{Ti\}$ (the nodes are in DFS order)
  Set $TBF = \{\}$, TBF is Tree Bracket Format
  Call UpdateNodes($NM(1)$), $NM(1)$ is root of the tree
  Set $TBF_i = TBF$
end for
for all i=1 to n **do**
  for all j=1 to n **do**
    Set $DM_{ij} = RTED\ (T_i, T_j)$ ▷ // DM_{ij} is Distance Matrix, $RTED$ is Robust Tree Edit Distance
  end for
end for
$ChoosingTree$ Procedure:
for all i=1 to n **do**
  Set $AED_i = \sum_{j=1}^{n} DM_{ij}\ /\ n$ ▷ // Compute the average edit distance per matrix row $tree\ AED$ = Average Edit Distance
end for
$cm = argmin_i AED_i$ ▷ // Choose the representative decision tree model (RM) by minimal average edit distance tree
Return T_{cm}
End Procdure

Algorithm 2 Semantic similarity algorithm

INPUT: Group T of n induced decision trees, where n = number of dataset slices, $DSTest$ = validation dataset with r records
OUTPUT: Selected Decision Tree T_{cm} where $cm \in [1..n]$

if T is null **then**
  return $failure$
end if
for all i=1 to n **do**
  for all j=1 to r **do**
    Set MV_{ij} = Classification of T_i decision tree over $DSTest_j$ record
  end for
end for
$SemanticSim_{1..n,1..r} = 0$
for all i=1 to n **do**
  for all i=j to r **do**
    $SemanticSim_{ij}$ = mean (number of times where $MV_{.,i} == MV_{.,j}$) ▷ // $SemanticSim_{ij}$ represents the classification agreement between the decision tree models i and j
  end for
end for
$ChoosingTree$ Procedure ▷ // Similar to described in SySM Algorithm
for all i=1 to n **do**
  Set $ASS_i = \sum_{j=1}^{n} SemanticSim_{ij}\ /\ n$ ▷ // Compute the average model agreement per matrix row $tree\ ASS$ = Average Semantic Similarity
end for
$cm = argmax_i ASS_i$
Return T_{cm}

Algorithm 3 Combined distance algorithm

INPUT: Group T of n induced decision trees, where n = number of dataset slices, $DSTest$ = validation dataset with r records, $WEIGHT$ = syntactic distance weight
OUTPUT: Chosen Decision Tree T_{cm} where $cm \in [1..n]$

```
if T is null then
    return failure
end if
Normalize(DM) syntactic distance matrix as described in SySM Algorithm
for all i=1 to n do
    for all j=1 to n do
        Set SemanticDist_ij = (1-SemanticSim_ij)
    end for
end for
Normalize(SemanticDist) semantic agreement matrix as described in Semantic similarity Algorithm ▷ // Normalizing Matrix divides each row by its maximal value
for all i=1 to n do
    for all j=1 to n do
        Set CM_ij = Weight * DM_i,j + (1-Weight) * SemanticDist_i,j
    end for
end for
ChoosingTree Procedure                                   ▷ // Similar to described in SySM Algorithm
```

The SySM algorithm (Algorithm 1) transforms each of the induced decision trees into the Bracket Tree Format (BTF). This is done by scanning the tree nodes from top down. The BTF is a way of representing the tree as one sequence of node labels, where the nodes are separated from each other using bracket symbol. Different levels of the nodes can be distinguished by the bracket symbol as well. Calculating the difference between the number of open brackets to the close ones determines the level of the specific node in the tree. For example, for the following BTF: $\{a\{b\}\{c\{d\}\{e\}\}\}$ we can deduce that b and c are at the same level since the difference between open brackets to close brackets before b is 2 and before c is 2 as well. We can also infer from the bracket locations of the BTF that both d and e are siblings and children of c. In BTF representation of a tree, each node is labeled by the name of the corresponding tested attribute, is more compact than the visual and the textual decision tree representations. Using the SySM algorithm, we refer only to the structure of the induced trees and hence there is no need to make any change to the tree structure or to perform additional computations on the raw data. The RTED [32] algorithm, which is the core of our similarity calculation procedure, computes the distance between each pair of induced trees by finding the mismatches of each node label in the equivalent positions of the compared trees. In case of such mismatch there are three editing operations: deleting an existing node, inserting a new node or changing the label of an existing node. The amount of relevant changes is accumulated. The total edit distance between the trees represents the similarity metric. Practically, one induced tree is transformed into another tree. The RTED [32] is symmetric with respect to the order of the compared trees. For multiple comparisons between one decision tree to the rest of the induced decision trees, we add an overall distance matrix where each cell represents the edit distance between the tree represented by the row number to each of the trees represented by the columns. For each row, the average of the edit distances represents the similarity of the tree to the rest of decision trees. The representative model chosen by our algorithm is the model with the minimal average edit distance to other models.

As an alternative for SySM we propose two more approaches for calculating the tree similarity: semantic similarity and combined distance. The semantic similarity implemented by Algorithm 2 is calculated as the classification agreement ratio between

each pair of induced decision trees over a development (validation) set. The combined approach shown in Algorithm 3 calculates the distance results using both approaches, normalizes them, and takes their average as the distance between trees.

The tree selection procedure can be deployed on a single machine, since it deals with the REDUCE phase of processing all models, which have been induced in the distributed (MAP) phase on multiple local machines. Therefore, the proposed approach does not affect in any way the amount of data traffic required by the distributed phase. From the computational complexity perspective, the REDUCE phase of the similarity-based algorithms has a training complexity of $O(n^2)$ where n is the number of induced trees/computation nodes, since the algorithm calculates the average distance/similarity of each induced tree to the rest of the trees. RTED has the complexity of $O(m^2)$ where m is the number of tree nodes which implies that SySM has the complexity of $O(m^2n^2)$. When using the semantic approach or choosing the most accurate tree, unlike SySM, we apply each decision tree to every record in the development (validation) set. Hence, the tree selection complexity of the semantic and the most accurate tree approaches is $O(vn^2)$, where v is the number of validation records. For finding the most accurate decision tree, instead of $O(n^2)$ for SySM and the semantic approach, the complexity is $O(nlog(n))$ as it requires sorting the accuracies of n trees. The evaluation (classification) complexity of using a single representative tree is $O(log(t))$ per one testing instance, where t is the number of terminal nodes.

In contrast, the evaluation complexity of the ensemble approach is $O(nlog(t))$ per instance, since it requires traversing n trees for classifying each new instance.

In addition, from the practical perspective, if classification is done on parallel machines, we still have to add the network transportation time from each machine to the central server, which then requires the majority voting time.

As for the combined approach, we can run the SySM and the semantic approaches in parallel, because they do not depend on each other. In that case, the running time will be the maximal between the both.

Empirical evaluation

In this section, we perform experiments to evaluate the performance of the proposed tree selection algorithms.

Design of experiments

Our experiments are aimed at comparing the following approaches to classifying each new instance:

1. Ensemble (baseline): majority voting over an ensemble of all induced trees.
2. SySM (Syntactic Similarity Method): using a single tree, which is most similar syntactically to other induced trees.
3. Semantic similarity: using a single tree, which is most similar semantically to other induced trees over a development (validation) set.

4. Combined similarity: using a single tree, which is chosen based on a linear combination of the syntactic and the semantic similarity measures. An equal weight of 0.5 is assigned to each similarity measure.
5. Most accurate: using a single tree, which is most accurate on a development (validation) set.

We evaluate the above approaches in six different experiments by the following performance metrics: accuracy, running time, and interpretability. We also evaluate a reduced ensemble of top five trees chosen by each methodology.

In the first experiment, we compare the accuracies of the evaluated approaches. We expect the ensemble to yield higher accuracy levels since its classification is based on all dataset shards while the similarity-based approaches consider one representative shard only.

In the second experiment, we evaluate the running time of the algorithms. The evaluated methodology has three phases: inducing the local models from the training data, choosing a representative model, and applying the selected model to new records. We expect the similarity-based approaches to be computationally cheaper in the third (testing) stage, since they classify the new data by only one model, whereas an ensemble needs to classify each incoming record by multiple models.

The third experiment evaluates the accuracy of a selected tree compared to a randomly chosen decision tree. We find the percentile of the selected model accuracy in the cumulative distribution of all models. The proposed model selection methodology can be considered useful if the percentile of the selected model accuracy exceeds 50% (the median) of all models.

In the fourth experiment, we evaluate the correlation between the syntactic and semantic distance metrics. We expect a positive correlation, since the greater the structural difference between the trees there should be less agreement between their predictions.

We train the algorithm over 80% of each dataset (the training set), which is split it into chunks. For calculating the semantic similarity and choosing the most accurate decision tree, we run the decision trees over the development set that is 10% of the dataset. We estimate the testing accuracy of all algorithms by running the selected decision tree over the testing set that is the rest 10% of the dataset.

In the same way, we choose a small ensemble of the top five decision trees based on each similarity-based methodology and use majority voting for classifying new records. We expect such a small ensemble to have better results than a single selected tree for datasets that have a higher variance between the local data chunks.

We evaluate our methodology over six big multivariate datasets from the UCI repository, which are used as benchmarks in other papers on big data such as [37] and which are shown in Tables 1, 2.

The representative trees are selected and evaluated on a computer having the following characteristics:

Processors: i7-4710MQ, Cores: 8 per processor (16 threads), Clock speed: 2.50 GHz, Cache: 256 MB, Hard drive: 240 GB, RAM: 64 GB

Table 1 Big datasets used for empirical evaluation

ID	UCI dataset name	Samples	Attributes	Classes
DS1	Poker Hand—consisting of five playing cards	1,025,010	11	9
DS2	SUSY —Monte Carlo simulations of kinematic properties measured by the particle detectors	5,000,000	18	2
DS3	Record Linkage Comparison Patterns—decide from a comparison pattern whether the underlying records belong to one person	5,749,132	9	2
DS4	KDD Cup 1999—build a network intrusion detector	4,898,431	42	23
DS5	Individual household electric power consumption	2,075,259	9	Continuous
DS6	HIGGS—distinguish between a signal process, which produces Higgs bosons, and a background process	11,000,000	28	2

Table 2 Decision tree parameters for empirical evaluation

ID	UCI dataset name	J48 M	CART minbucket
DS1	Poker Hand	30	250
DS2	SUSY	40	500
DS3	Record Linkage Comparison Patterns	2	500
DS4	KDD Cup 1999	40	7
DS5	Individual household electric power consumption	2500	3500
DS6	HIGGS	5500	3000

We run the algorithms in experiments one to five over each dataset in six variations of horizontal partitioning: 32-folds, 64-folds, 128-folds, 256-folds, 512-folds, and 1024-folds. The folds are equal-size slices of the training dataset assigned to different computation nodes. J48 and CART decision trees are induced from each fold of every dataset using relevant packages in R.

Our algorithm can be implemented either over distributed parallel framework or on single machine as described in Ye et al. [39].

In order to meet the RAM capacity we searched for a parameter that yields decision trees with the number of leaves and the tree size that fits in computer memory. Therefore, we searched for the minimum value of the minimum number of instances per leaf (M parameter in J48 and *minbucket* parameter in CART) that allows the induced trees to fit in computer memory. J48 tree is induced from POKER dataset by setting the M parameter to 30. For SUSY dataset as well as KDDCUP, we used $M = 40$. For RLCP dataset, J48 is induced using the default value of ($M = 2$).

For GAS Sensors and Household Electric Power the $M = 2500$ and $M = 3500$ whereas the *minbucket* $= 3500$ and *minbucket* $= 3000$, respectively. In addition, for HIGGS dataset we used randomly chosen 4 million records out of 11 million. The M values are determined for each dataset using sensitivity analysis. The best value of M is chosen by analysing two random slices of 32-folds. For each M value, we found the model accuracy over training and validation datasets, the number of the tree leaves, and the total size of the induced tree.

Results

As mentioned above, one of the core steps in the proposed methodology is comparison between the induced decision trees. The semantic comparison focuses on the proportion of identical classifications of validation instances, whereas the syntactic approach builds upon the tree structure. We expect the single-tree approach to yield shorter classification times than the ensemble due to the fact that there is no need to run all decision tree models over the testing data. In addition, we compare the accuracy of the models chosen by the different approaches.

In the first experiment, as can be seen in Fig. 2, we compare the testing accuracy of the tree chosen in semantic and syntactic mode and their combined results to ensemble majority voting and most accurate tree. The figure presents the results for DS3, which has the largest number of records out of the six benchmark datasets. In this dataset, the syntactic approach outperforms the semantic one for all J48 runs and 50% of CART runs. The ensemble approach, which employs multiple models in the classification stage, yields a slightly higher accuracy than the similarity algorithms based on a single model. We can also see that the most accurate tree has a very similar accuracy to SySM, whereas the combined approach has lower results in comparison to other approaches. As indicated below, similar results are observed in most experiments with the other datasets.

We compare the accuracy of ensemble vs. SySM and SySM vs. the semantic approach for every dataset and each decision tree algorithm (J48 and CART) using the following formula:

$$\frac{\mathrm{SySM}_{\mathrm{Accuracy}} - \mathrm{Ens}_{\mathrm{Accuracy}}}{\sqrt{F(\mathrm{SySM}_{\mathrm{Accuracy}}) + \mathrm{F}(\mathrm{Ens}_{\mathrm{Accuracy}})}}$$

where

$$F(x) = \frac{x(1-x)}{\text{number of testing rec}}$$

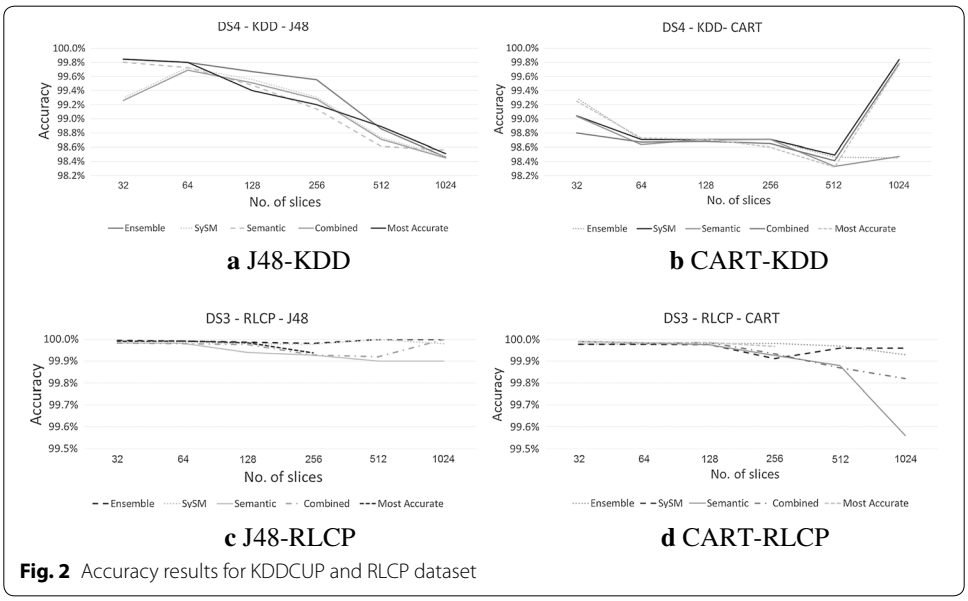


Fig. 2 Accuracy results for KDDCUP and RLCP dataset

We run 24 experiments for every dataset (for each fold: J48, CART). For POKER dataset, in 83.3% of cases the syntactic approach obtains a significantly higher accuracy than the semantic one (p-value = 0.05). The ensemble accuracy is significantly higher in comparison to SySM in 83.3% of cases. For SUSY, SySM has the highest accuracy in 50% of cases. The syntactic approach yields significantly higher accuracy results than semantic in 75% of cases. In RLCP, there is no significant difference between ensemble accuracy and SySM, whereas the syntactic outperforms significantly the semantic in 58.3% of cases. In KDDCUP, ensemble has a significantly higher accuracy than SySM in 41.7% of cases. In addition, SySM is significantly more accurate than semantic in 50% of cases.

For HOUSEHOLD ELEC dataset, SySM has a higher accuracy than ensemble in 83.3% of cases. In HIGGS dataset, in 87.5% of cases, SySM approach obtains a significantly higher accuracy than the ensemble approach.

The high accuracy of the ensemble of decision trees is expected, since each induced decision tree that sees a local slice of the trained dataset takes part in voting for classifying each tuple in the testing dataset. In this way, although each induced decision tree sees only part of the trained dataset the voting combines their predictions over the testing dataset. However, the results of the ensemble algorithm are not easily interpretable, since they are not based on a single decision tree model. In addition, the classification phase of ensemble includes two computation steps: running the induced trees over the testing instances and computing the majority voting result. The number of runs of the induced trees over the testing instances is equal to the number of dataset slices. In a distributed system, the induction of decision trees is done over each training set in parallel. However, for ensemble-based evaluation, there is a need to apply each decision-tree model to the same testing instances. This adds computational and memory complexity to ensemble algorithms.

In the second experiment, as can be seen in Fig. 3, the total running time over the testing dataset is indeed higher for ensemble than for SySM at the order of the number of shards.

In the third experiment, we find that the average percentile of the chosen trees accuracy in the syntactic mode (Algorithm 1) are 0.645 and 0.524 for J48 and CART, respectively. This means that the syntactic mode performs better than choosing randomly one of the induced decision trees, especially when the trees are built by J48.

In the fourth experiment, we compare the semantic vs. syntactic distance values in the tree distance matrix for 32-folds (shards) of the Poker dataset. The semantic distance is calculated as (1 – semantic agreement percentage) as described in detail in Algorithm 3. The hypothesis that there is a significant correlation between the two is rejected as indicated by the low R^2 values in Table 3 and the raw data shown in Fig. 4.

Figure 5 compares the testing accuracy of different approaches for 128 shards. As can be seen the differences in accuracy between SySM and ensemble are minor for both J48 and CART in all datasets.

Table 4 compares between the average accuracy of each algorithm per dataset. As can be seen in Table 4, the number of times that SySM has the best accuracy is about two times higher than choosing the most accurate tree (8 times out of 12). The averages of all results are almost the same (with a slight advantage for SySM): 0.764 vs. 0.762. When examining the results for majority voting over choosing the five best trees we can see

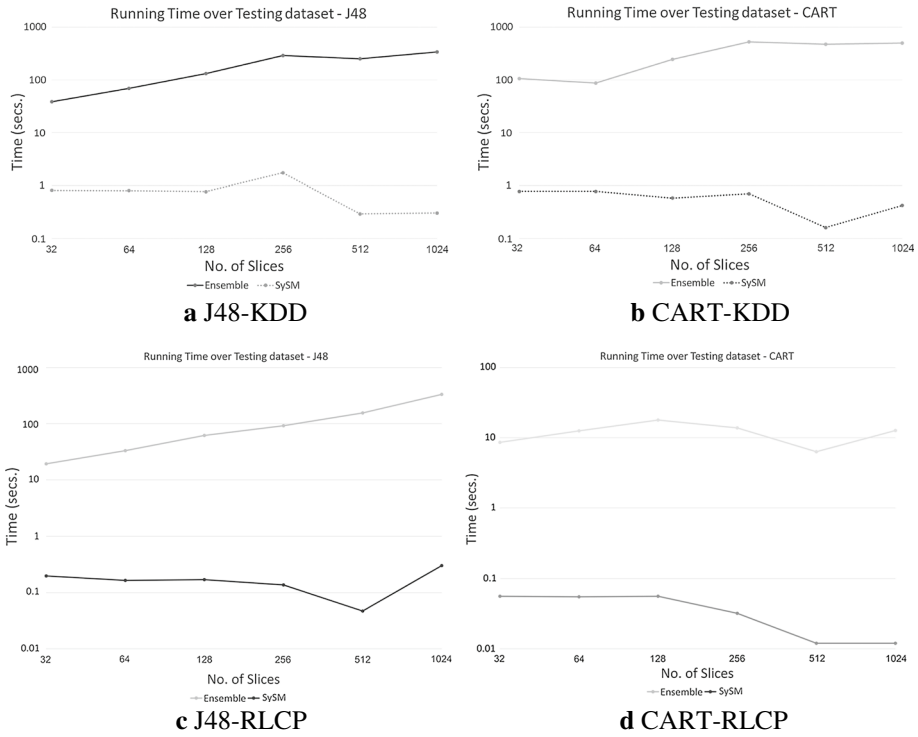


Fig. 3 Classification time over KDDCUP and RLCP testing dataset

Table 3 32 shards: correlation between semantic and syntactic distance metrics

ID	UCI dataset name	R^2 (J48)	R^2 (CART)
DS1	Poker Hand	0.0243	0.2039
DS2	SUSY	0.0594	0.0015
DS3	RLCP	0.2318	0.7709
DS4	KDD Cup	0.1555	0.5613
DS5	Household Electric	0.1023	0.4741
DS6	HIGGS	0.0938	0.3617

in Table 5 that choosing the most accurate trees has an accuracy advantage over SySM in 6 cases. The averages of all methods are almost the same (with a slight advantage for most accurate trees) 0.75 versus 0.753. When comparing between the most accurate tree to SySM, we use the Hodges Jr and Lehmann [19] test. We implement the test both for CART and J48. We deduce from the test that we can reject our initial hypothesis that there is a significant difference between the accuracies of results of the twoapproaches with p-value of 5%.

Discussion

In the previous section, we have evaluated several alternative classification approaches based on decision-tree ensembles. Our empirical evaluation has focused on the following two performance measures: classification accuracy and testing (inference) time. As

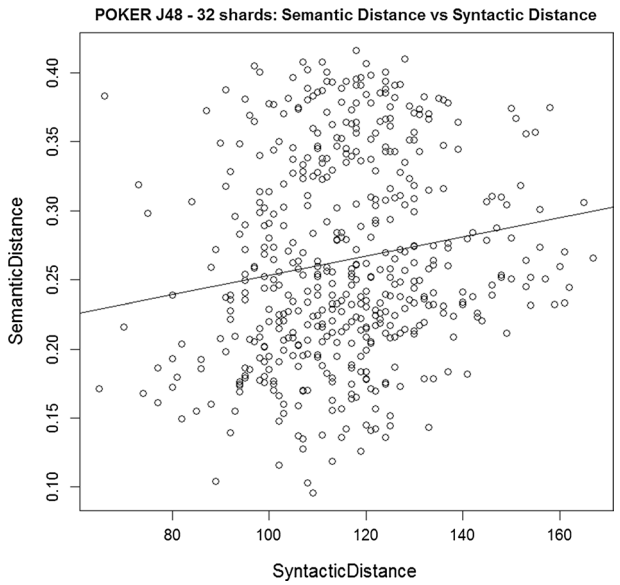
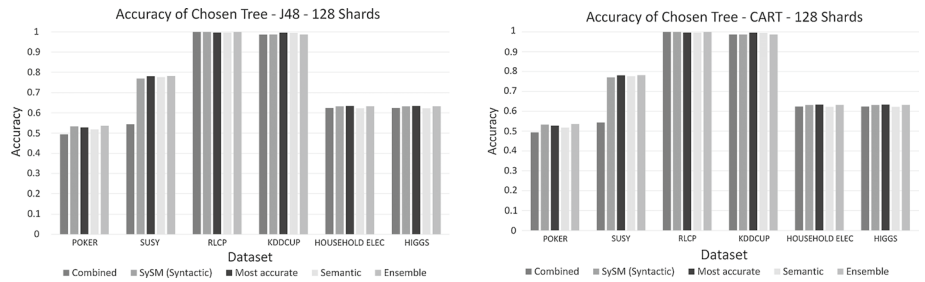


Fig. 4 POKER-J48: semantic distance vs. syntactic distance



a J48 **b** CART

Fig. 5 128 shards: accuracy of a single selected tree across similarity types

Table 4 Comparison between accuracies of single best trees

Algorithm name	UCI dataset name	Best SySM tree train 80%	Most accurate tree train 80%	Best SySM tree train 90%
CART	Poker Hand	0.517	0.528	*0.530*
CART	SUSY	0.777	*0.781*	0.777
CART	RLCP	0.997	0.997	*0.998*
CART	KDD Cup	0.995	0.996	*0.997*
CART	Household Elec.	0.678	0.691	*0.713*
CART	HIGGS	0.995	0.996	*0.997*
J48	Poker Hand	0.550	*0.574*	0.531
J48	SUSY	0.673	0.736	*0.783*
J48	RLCP	0.999	*1.000*	0.999
J48	KDD Cup	0.998	*0.999*	0.998
J48	Household Elec.	0.627	0.638	*0.646*
J48	HIGGS	0.647	0.653	*0.654*

In italics the most accurate per line

Table 5 Comparison between accuracies of five best trees

Algorithm name	UCI dataset name	Five best SySM train 80%	Five most accurate trees train 80%	Five best SySM train 90%
CART	Poker Hand	0.529	*0.536*	0.519
CART	SUSY	0.782	*0.786*	*0.786*
CART	RLCP	0.997	*0.998*	*0.998*
CART	KDD Cup	0.998	0.997	*0.999*
CART	Household Elec.	0.998	0.997	*0.999*
CART	HIGGS	0.998	0.997	*0.999*
J48	Poker Hand	0.557	*0.589*	0.531
J48	SUSY	0.693	0.737	*0.783*
J48	RLCP	0.999	0.999	0.999
J48	KDD Cup	*1.000*	*1.000*	0.998
J48	Household Elec.	*0.557*	0.556	0.555
J48	HIGGS	*0.652*	0.649	0.648

In italics the most accurate per line

expected, the ensemble approach is, in most cases, the most accurate one, but it requires more computational resources for classifying each new instance than any single decision-tree classifier or a reduced ensemble of five decision trees.

Out of the three evaluated methods for choosing the best single decision tree model, the algorithm based on syntactic similarity of decision trees (SySM) is significantly more accurate than the semantic distance algorithms and in some datasets, it even outperforms the ensemble accuracy. However, when comparing SySM to the accuracy-based approach, we see that there is no significant advantage of either method in terms of testing accuracy. Hence, both the syntactic and the accuracy-based approaches are expected to be a good choice for choosing a single representative model.

One way to improve the classification accuracy at a small computational cost is to use a small ensemble of five best trees instead of one. The accuracy-based ensembles of five representative trees have provided more accurate classification results, in most cases, than the SySM-based ensembles. Thus, when choosing an ensemble of five models, one should prefer the five most accurate decision trees.

Conclusions

In this paper, we propose and evaluate several methods for selecting one representative model out of multiple decision trees induced from different slices of the same massive dataset. These methods can be very useful for big data and secured environments due to having a higher inference (classification) speed than ensemble methods. First, we suggest a syntactic approach, named SySM, that is based solely on the induced decision trees structure. In addition we present the semantic similarity and the combined distance algorithms. All use the same similarity-based approach but calculate the tree similarity differently.

The accuracy of the similarity-based methods is compared to other well-known algorithms on six big benchmark datasets. Each dataset was divided to number of slices from 32 to 1024.

When comparing the accuracy of SySM to semantic similarity and combined distance, we see that SySM has a higher accuracy in most cases. This is in addition to the fact the model selection phase is faster than semantic similarity oriented approaches. When comparing SySM to choosing the most accurate model there is no difference in the number of times where one approach is more accurate than the other and the mean accuracies of all results are almost the same. When choosing an ensemble of five models, there is a slight advantage for the accuracy-based approach.

In the future work, one may utilize the fast computation characteristics of the SySM algorithm for distributed data streaming environment. In this environment, SySM may save expensive I/O disk operations, since all induced decision trees can fit in the memory of a single computer without the need to re-process the raw data. The SySM approach can be combined with the most accurate tree approach in different variations. An additional future approach may use alternative data structures such as DAG (Directed Acyclic Graph) and additional similarity metrics between decision-tree models. The algorithm may also be extended for dealing with ensembles of decision trees generated by random subspace selection methods such as random forests [10]. One may also try to optimize the weights of the combined method.

Abbreviations
CART: Classification and Regression Tree; SySM: Syntactic Similarity Method; DAG: Directed Acyclic Graph; RTED: Robust Tree Edit Distance.

Authors' contributions
AlW was the primary author who initiated the idea through developing the algorithms, running the experiments and writing the paper. ML refined the concepts discussed and the design of experiments. Both authors read and approved the final manuscript.

Acknowledgements
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Availability of data and materials
The data is available in UCI repository.

Consent for publication
All authors have consented for publication of this paper.

Ethics approval and consent to participate
All authors give ethics approval and consent to participate in submission and review process.

Funding
Not applicable.

Publisher's Note

Received: 17 November 2018 Accepted: 14 February 2019
Published online: 28 February 2019

References

1. AlSabti K, Ranka S, Singh V. Clouds: classification for large or out-of-core datasets. In: Conference on knowledge discovery and data mining. 1998
2. Amado N, Gama J, Silva F. Parallel implementation of decision tree learning algorithms. In: Progress in artificial intelligence. Berlin: Springer; 2001. p. 6–13.
3. Amado N, Gama J, Silva F. Exploiting parallelism in decision tree induction. In: Proceedings from the ECML/PKDD workshop on parallel and distributed computing for machine learning. 2003. p. 13–22.

4. Andrzejak A, Langner F, Zabala S. Interpretable models from distributed data via merging of decision trees. In: 2013 IEEE symposium on computational intelligence and data mining (CIDM). New York: IEEE; 2013. p. 1–9.
5. Basilico JD, Munson MA, Kolda TG, Dixon KR, Kegelmeyer WP. Comet: a recipe for learning and using large ensembles on massive data. In: 2011 IEEE 11th international conference on data mining (ICDM). New York: IEEE; 2011. p. 41–50.
6. Bekkerman R, Bilenko M, Langford J. Scaling up machine learning: parallel and distributed approaches. Cambridge: Cambridge University Press; 2011.
7. Ben-Haim Y, Tom-Tov E. A streaming parallel decision tree algorithm. J Mach Learn Res. 2010;11:849–72.
8. Bousquet O, Bottou L. The tradeoffs of large scale learning. In: Advances in neural information processing systems. 2008. p. 161–8.
9. Breiman L. Pasting small votes for classification in large databases and on-line. Mach Learn. 1999;36(1–2):85–103.
10. Breiman L. Random forests. Mach Learn. 2001;45(1):5–32.
11. Dai W, Ji W. A mapreduce implementation of c4. 5 decision tree algorithm. Int J Database Theory Appl. 2014;7(1):49–60.
12. DeWitt DJ, Naughton JF, Schneider D, et al. Parallel sorting on a shared-nothing architecture using probabilistic splitting. In: Proceedings of the first international conference on parallel and distributed information systems, 1991. New York: IEEE; 1991. p. 280–91.
13. Domingos P. Knowledge discovery via multiple models. Intell Data Anal. 1998;2(3):187–202.
14. Domingos P, Hulten G. Mining high-speed data streams. In: Proceedings of the sixth ACM SIGKDD international conference on knowledge discovery and data mining. New York City: ACM; 2000. p. 71–80.
15. Friedman JH, Popescu BE. Predictive learning via rule ensembles. Ann Appl Stat. 2008;2:916–54.
16. Gehrke J, Ganti V, Ramakrishnan R, Loh W-Y. Boat optimistic decision tree construction. In: ACM SIGMOD record, vol. 28. New York City: ACM, 1999. p. 169–80.
17. Goil S, Choudhary A. Parsimony: an infrastructure for parallel multidimensional analysis and data mining. J Parallel Distrib Comput. 2001;61(3):285–321.
18. Hansen LK, Salamon P. Neural network ensembles. IEEE Trans Pattern Anal Mach Intell. 1990;10:993–1001.
19. Hodges J Jr, Lehmann EL. Comparison of the normal scores and Wilcoxon tests. In: Proc. fourth Berkeley symp. math. statist. prob, vol. 1. 1961. p. 307–17.
20. Jin R, Agrawal G. Communication and memory efficient parallel decision tree construction. Philadelphia: SDM, SIAM; 2003. p. 119–29.
21. Joshi MV, Karypis G, Kumar V. Scalparc: a new scalable and efficient parallel classification algorithm for mining large datasets. In: Parallel processing symposium, 1998. IPPS/SPDP 1998. Proceedings of the first merged international... and symposium on parallel and distributed processing 1998. New York: IEEE; 1998. p. 573–9.
22. Kargupta H, Park B-H. A Fourier spectrum-based approach to represent decision trees for mining data streams in mobile environments. In: IEEE transactions on knowledge and data engineering, vol. 16, no. 2. 2004. p. 216–29.
23. Krogh A, Vedelsby J, et al. Neural network ensembles, cross validation, and active learning. Adv Neural Inf Process Syst. 1995;7:231–8.
24. Louppe G, Geurts, P. Ensembles on random patches. In: Machine learning and knowledge discovery in databases. Berlin: Springer; 2012. p. 346–61.
25. Magana-Mora A, Bajic VB. Omniga: optimized omnivariate decision trees for generalizable classification models. Sci Rep. 2017;7(1):3898.
26. Mehta M, Agrawal R, Rissanen J. Sliq: a fast scalable classifier for data mining. In: Advances in database technology EDBT'96. Berlin: Springer; 1996. p. 18–32.
27. Miglio R, Soffritti G. The comparison between classification trees through proximity measures. Comput Stat Data Anal. 2004;45(3):577–93.
28. Narlikar GJ. A parallel, multithreaded decision tree builder. DTIC Document: Technical report; 1998.
29. Ntoutsi I, Kalousis A, Theodoridis Y. A general framework for estimating similarity of datasets and decision trees: exploring semantic similarity of decision trees. Philadelphia: SDM, SIAM; 2008. p. 810–21.
30. Panda B, Herbach JS, Basu S, Bayardo RJ. Planet: massively parallel learning of tree ensembles with mapreduce. Proc VLDB Endow. 2009;2(2):1426–37.
31. Parisi F, Strino F, Nadler B, Kluger Y. Ranking and combining multiple predictors without labeled data. Proc Natl Acad Sci. 2014;111(4):1253–8.
32. Pawlik M, Augsten N. Rted: a robust algorithm for the tree edit distance. Proc VLDB Endow. 2011;5(4):334–45.
33. Shafer J, Agrawal R, Mehta M. Sprint: a scalable parallel classifier for data mining. In: Proc. 1996 int. conf. very large databases, Citeseer, 1996. p. 544–55.
34. Shannon WD, Banks D. Combining classification trees using MLE. Stat Med. 1999;18(6):727–40.
35. Sreenivas MK, AlSabti K, Ranka S. Parallel out-of-core decision tree classifiers. In: Kargupta H, Chan P, editors. Advances in distributed and parallel knowledge discovery. Menlo Park: AAAI; 2000. p. 317–36.
36. Srivastava A, Han E-H, Kumar V, Singh V. Parallel formulations of decision-tree classification algorithms. Berlin: Springer; 2002.
37. Triguero I, Peralta D, Bacardit J, Garcí S, Herrera F. MRPR: a mapreduce solution for prototype reduction in big data classification. Neurocomputing. 2015;150:331–45.
38. Weinberg AI, Last M. Interpretable decision-tree induction in a big data parallel framework. Int J Appl Math Comput Sci. 2017;27(4):737–48.
39. Ye T, Zhou H, Zou WY, Gao B. Zhang R. Rapidscorer: fast tree ensemble evaluation by maximizing compactness in data level parallelization. In: Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining. New York City: ACM; 2018. p. 941–50.
40. Zhang K, Shasha D. Simple fast algorithms for the editing distance between trees and related problems. SIAM J Comput. 1989;18(6):1245–62.
41. Zhang X, Jiang S. A splitting criteria based on similarity in decision tree learning. J Softw. 2012;7(8):1775–82.