Interval-based Interpretable Decision Tree for Time Series Classification

Malte Schmidt, Volker Lohweg

inIT – Institute Industrial IT, Technische Hochschule Ostwestfalen-Lippe
Campusallee 6, 32657 Lemgo, Germany
E-Mail: {malte.schmidt, volker.lohweg}@th-owl.de

Abstract

In this paper we present the first iteration of a novel time series classification algorithm which is *globally* and *inherently* interpretable. The need for model interpretability or explainability is commonly agreed upon in industry [1]. Model interpretability is an important characteristic of a classifier to build trust in the decisions of the classifier and makes it possible to iteratively improve a model with domain knowledge.

The proposed algorithm first performs an unsupervised clustering of random segments of random length of a time series to find the most discriminating patterns. After finding segments with discriminating patterns, a decision tree is trained using the cluster labels as features. Therefore, the decision tree is restricted to learn a mapping from discriminating clusters to given class labels.

The performance of our algorithm is compared to state-of-the-art algorithms with a computational feasible subset of the University of California, Riverside, time series archive [2]. The first iteration of our algorithm is computationally expensive and does not achieve state-of-the-art accuracy. We point out shortcomings of the current iteration and discuss planned improvements to our algorithm to tackle these shortcomings. We find that our algorithm creates shallow decision trees which boosts interpretability. In contrast, not all state-of-the-art approaches provide interpretable models.

DOI: 10.58895/ksp/1000138532-7 erschienen in:
Proceedings - 31. Workshop Computational Intelligence : Berlin, 25. - 26. November 2021
DOI: 10.58895/ksp/1000138532 | https://www.ksp.kit.edu/site/books/m/10.58895/ksp/1000138532/

1 Introduction

During the last decades research on time series classification (TSC) has made considerable progress and the University of California, Riverside, time series archive (UCR TSA) [2] is often used to benchmark novel TSC algorithms on one dimensional time series. Often the term time series refers to any ordered series and is not limited to value-index pairs ordered by time. For example, the UCR TSA also includes series generated by spectrographs and object outlines mapped to one-dimensional series.

In industrial and medical applications interpretability of a model is regarded as an important characteristic for a wide adoption of machine learning techniques in these fields [1, 3]. Furthermore, the type of interpretability a model provides is of interest. Here, we differentiate between types of interpretability regarding two different viewpoints.

First, it is important to know how an explanation of a decision is produced. We adopt the differentiation from Rudin [4] and differentiate between the following types:

- *Post hoc explanation of models*. A model is explained post hoc by a second model. An example of a post hoc explanation method often applied to neural networks is LIME [5].
- *Inherently interpretable models*. The model itself provides a faithful explanation of its decisions. An example for an inherently interpretable model is a (small) decision tree with interpretable features.

Second, we are interested in what type of explanation is provided by the model. Here, we adopt the differentiation from Hong [1]:

- *Locally interpretable models*. The model explanation is given on a per instance basis. An example for this type of explanations are saliency maps.
- *Globally interpretable models*. The logical structure of the model itself explains how it works globally. An example for globally interpretable models are, once again, decision trees with interpretable features.

In this paper we propose an algorithm which is globally and inherently interpretable. The features the algorithm utilises are regions of interest in the time series based on their visual appearance (shape). These regions of interest or intervals are phase-dependent which makes our algorithm appropriate for applications which require phase-dependency.

The rest of the paper is organised as follows. In Section 2 we give an overview of state-of-the-art TSC algorithms which are related to our work. Next, we present the design of our algorithm in Section 3. In Section 4 we evaluate the performance of our algorithms and discuss advantages and shortcomings of it before finishing the paper with a conclusion and outlook in Section 6.

2 Related Work

One of the most basic approaches to TSC is a k-nearest-neighbours classifier using an elastic distance metric as similarity measure. An often used elastic distance metric is dynamic time warping (DTW) [6] or variations of it [7, 8]. While this approach is not competitive to current state of the art in terms of accuracy, it still provides a reasonable baseline.

As in other fields, there exist a growing number of approaches to TSC which use neural networks [9]. Neural networks, especially neural networks including convolutional layers, are found to be competitive to other state-of-the-art approaches in terms of accuracy. Some of the recent approaches rely on fully convolutional networks [10] or are inspired by successful architectures in computer vision like the Inception architecture [11]. Wang et al. [10] and Fawaz et al. [9] also explored explaining models with CAM, a post hoc explanation method for CNN-based models [12].

In 2016 Bagnall et al. [13] published an extensive review of the current state of the art in TSC. The best performing algorithm was an ensemble of classifiers, named COTE [14]. COTE combines state-of-the-art classifiers which work in different transformation domains. It was later extended and called HIVE-COTE [15]. This ensemble still achieves state-of-the-art accuracy on the UCR

TSA benchmark due to continuous updating of the ensemble with the latest developments in TSC [16].

A class of features which provides inherent explanations when combined with suitable classifiers are shapelets. Shapelets are phase-independent discriminative time series sub-sequences [17]. Classification is done based on the presence or absence or the count of these discriminative subsequences. One successful approach transforms the time series with a shapelet transformation and then a standard classifiers is trained on the transformed time series [18]. Learning of the k best shapelets through a heuristic gradient descent with a k-means clustering as shapelet initialization is presented by Grabocka et al. [19]. Brunello et al. [20] use a decision tree to build a classifier after finding phase-independent shapelets with evolutionary algorithms.

Decision trees or forests are common classifiers for TSC problems due to their fast training time and interpretability. Deng et al. propose a time series forest (TSF) which uses statistics calculated from random interval as features [21]. They also propose a post hoc explanation through importance curves. More recently multiple ensembles of decision trees for TSC, which achieve state-of-the-art accuracy, are proposed [22, 23].

Another algorithm which achieves state-of-the-art accuracy and is inherently interpretable is the algorithm proposed by Nguyen et al. [24]. A symbolic representation of time series is combined with a sequence learner originally developed for biological sequence classification to search for the most discriminating sub-sequences in the symbolic representations. This approach provides an inherently and locally interpretable model through saliency maps. Nguyen et al. recently compared the explanations provided by CAM, LIME, and the inherent explanations of their sequence learner [25].

Although many decision tree approaches for TSC exist, we think there is still room for further exploration of this approach. We focus on designing an algorithm which creates inherently and globally interpretable models by relying on (shape-based) clustering results of intervals as features. Our hypothesis is that this gives us a distinctive separation of phase-dependent shapes of the times series which improves interpretability. A high level of interpretability enables verification and improvement of the model by an expert.

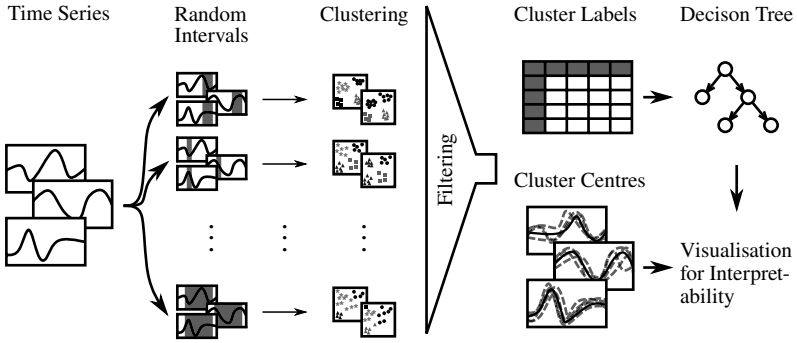

Figure 1: Concept of the proposed classification algorithm.

3 Algorithm Design

The overall concept of our algorithm can be seen in Fig. 1. First, intervals with random start index and random length are chosen. Next, for each interval multiple k-means clusterings with different configurations are computed. After a preliminary filtering of the clusters by the silhouette score [26], the remaining clustering results provide the features for a decision tree. Finally, after training the decision tree, the cluster centres of the selected clusterings visualise the decision process of the decision tree.

In the following required definitions and notations are introduced.

Time Series. A time series is a sequence $\mathbf{t} = (t_1, \ldots, t_L)$ of L values (observations) ordered by some criterium (e.g. time, frequency or wavelength). The length of time series $\mathbf{t}$ is L.

Discrete Interval. A discrete interval $\mathscr{I} = [s..e]$ is a set of integers, i.e. $\{s, s+1, \ldots, e\}$. We express the indices of a sub-sequence of a time series with an interval. For example, $\mathbf{t}(\mathscr{I}) = (t_s, \ldots, t_e)$ is the sub-sequence of time series $\mathbf{t}$ over the interval $\mathscr{I}$. The length of interval $\mathscr{I}$ is given by its cardinality $|\mathscr{I}|$. We assume all intervals are valid, i.e. $1 \leq s \leq e \leq L$ holds.

3.1 Interval Selection

Instead of choosing all possible intervals, we limit the number of intervals for a time series with length L to $O(L)$ to reduce the time complexity of our algorithm. For the selection of the intervals, we follow the approach of Deng et al. [21]:

1. Select $\sqrt{L}$ window lengths from the set of possible window lengths $W_p = \{1,\ldots,L\}$ by random sampling without replacement.
2. For each window length w, select $\sqrt{L-w+1}$ start indices from the set of possible start indices $\mathscr{S} = \{1,\ldots,L-w+1\}$ by random sampling without replacement.

Each pair of selected window length w and start index i forms an interval $\mathscr{I} = [i..i+w-1]$. We use these intervals to extract sub-sequences from the time series for further processing.

By selecting sub-sequences from time series, we follow an interval-based approach for our algorithm and introduce phase-dependency. The idea is to select regions of interest which possibly contain distinctive shapes. Ideally, they should have a causal relation to the class labels.

3.2 Clustering

If we are interested in inherently and globally interpretable models, we require meaningful features for our model. In TSC one type of meaningful features are distinctive shapes. After choosing interval candidates which represent possible regions of interest containing such shapes, one way to find meaningful features is to cluster the sub-sequences resulting from the intervals.

We follow this rationale and apply k-means clustering with DTW as dissimilarity measure to find clusters of sub-sequences which intuitively match in shape. Each cluster is then represented by an average series calculated with DTW barycentre averaging (DBA) [27]. We used the k-means clustering implementation `TimeSeriesKMeans` from the `tslearn` [28] library. k-means++ [29] is applied as cluster initialization method.

We are only interested in cluster results which give a good separation between clusters. Therefore, before using the clustering results as training input for a decision tree, we pre-filter the results to exclude clustering results with high overlap between different clusters. For this we calculate the mean silhouette score [26] of all samples for each clustering result with DTW as the dissimilarity measure.

The (mean) silhouette score can take values between -1 and 1. A value below zero indicates overlapping of clusters while a value above 0 indicates a non-overlapping separation of clusters. We accept all cluster results with an overall silhouette score greater than zero for further processing.

3.3 Decision Tree Induction

In the last step, a decision tree is trained on the cluster labels of the remaining clustering results. As the decision tree induction algorithm we use an implementation of the ID3-algorithm by Quinlan [30] with two domain specific modifications:

- Induce bias towards a preferred interval length.
- Restrict the allowed overlap for intervals in the same tree branch.

For attribute selection gain ratio [30] is applied. In early experiments we noticed that intervals with a high overlap often have the same (maximal) gain ratio. To break the tie, we introduce a weighting function which weights the gain ratio depending on the interval length. With this weighting function, we induce bias towards a preferred interval length. We prefer a shorter interval length over a longer one because the shape present in shorter intervals is usually less complex and easier to interpret. However, if the interval length gets too small (a single value in the extreme case), the shape may not be meaningful and dominated by noise.

We propose to use a parametrisable unimodal weighting function

$$f(x) : [0,1] \rightarrow [0,1]$$

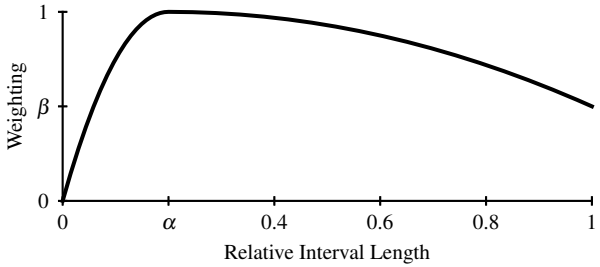

Figure 2: Proposed weighting function.

which maps the relative interval length to a weight for the gain ratio. The parameters are α, the preferred interval length relative to the length of the time series, and β, the weighting value for the whole series length. We set $f(\alpha) = 1$ and $f(1) = \beta$. The weighting function is given by

$$f(x) = \begin{cases} -\frac{1}{\alpha^2}(x-\alpha)^2 + 1 & \text{for } x \leq \alpha, \\ \frac{\beta-1}{(1-\alpha)^2}(x-\alpha)^2 + 1 & \text{for } x > \alpha \end{cases} \tag{1}$$

with $0 < \alpha < 1$ and $0 \leq \beta \leq 1$. Fig. 2 shows an example for $f(x)$ with $\alpha = 0.2$ and $\beta = 0.5$.

A quadratic function is chosen for the weighting function because of its simplicity (in terms of parameters) while still having a modest slope around its maximum in contrast to e.g. a triangle function. However, other unimodal function types are also valid candidates.

It can also happen that clustering results for the same interval but with a different number of cluster centres have the same (maximal) gain ratio. In this case we select the clustering result with the highest silhouette score because we want the features to be as interpretable as possible. However, other preferences, i.e. selecting the result with the fewest cluster centres, are also valid options.

The consecutive selection of multiple highly overlapping intervals in one tree branch may lead to overfitting. Suppose one distinctive sub-sequence is covered by overlapping intervals multiple times. Then this sub-sequence is im-

plicitly selected as classification criterium multiple times. To prevent this, we restrict the allowed overlap for the intervals used consecutively in a tree branch.

Let $\mathscr{I}_1, \ldots, \mathscr{I}_{N-1}$ be the intervals used consecutively in one tree branch and let $\mathscr{I}_N$ be the interval which we want to use for splitting at the next node. Then the maximum relative overlap o_{max} for any of these intervals is given by

$$o_{\max}\left(\mathscr{I}_1, \ldots, \mathscr{I}_N\right) = \max_x \frac{\left| \mathscr{I}_x \cap \bigcup_{i=1, i \neq x}^{N} \mathscr{I}_i \right|}{|\mathscr{I}_x|} \,. \tag{2}$$

o_{max} can have values between 0 (no overlap) and 1 (at least one interval fully overlaps with others). At each new node in a tree branch o_{max} is calculated including the new interval we want to use. The new interval is only accepted if o_{max} does not exceed a threshold θ.

4 Evaluation

4.1 UCR TSA Subset Selection

The current iteration of our algorithm is computationally expensive due to the clustering and silhouette score computation. Therefore, for this early evaluation of the algorithm, a subset of the UCR TSA is selected. To be as objective as possible, we define a computational complexity score with which we rank the datasets and pick the first 25 datasets for our evaluation. We limit our selection to datasets of the 2015 version of the UCR TSA [31] because accuracies for the provided train-test-splits are available for these datasets on the UCR TSA website [32].

We define the complexity score S of a dataset as

$$S = L \cdot \left(k^2 \cdot N \cdot L^2 + k \cdot I_{\mathrm{K}} \cdot N \cdot L^2 + I_{\mathrm{K}} \cdot I_{\mathrm{B}} \cdot N \cdot L^2\right) \tag{3}$$

Table 1: The UCR TSA dataset subset selected for evaluation.

No.	Dataset	No.	Dataset
1	ItalyPowerDemand	14	MiddlePhalanxTW
2	SonyAIBORobotSurface1	15	ProximalPhalanxTW
3	SonyAIBORobotSurface2	16	DistalPhalanxTW
4	MoteStrain	17	MiddlePhalanx-
5	TwoLeadECG		OutlineCorrect
6	ECGFiveDays	18	DistalPhalanx-
7	CBF		OutlineCorrect
8	SyntheticControl	19	ProximalPhalanx-
9	ECG200		OutlineCorrect
10	GunPoint	20	Plane
11	ProximalPhalanx-	21	ArrowHead
	OutlineAgeGroup	22	MedicalImages
12	MiddlePhalanx-	23	Coffee
	OutlineAgeGroup	24	Wine
13	DistalPhalanx-	25	ToeSegmentation1
	OutlineAgeGroup		

with the number of classes n_c in the dataset, the maximal number of clusters $k = \max\{10,\ 2 \cdot n_c\}$ to compute, the number of samples N, the time series length L, the number of iterations of the k-means algorithm I_K, and the number of iterations for barycentre calculation I_B of this dataset. The score is composed of the time complexity of the k-means++ [29] cluster centre initialization (first summand), the time complexity of the distance calculation to the cluster centres across all iterations (second summand), and the time complexity of the DBA across all iterations [27] (third summand).

The 25 datasets with the lowest score S are listed in Tab. 1. It is important to note that selecting datasets by complexity ranking necessarily introduces a bias towards datasets with shorter time series and fewer training samples. However, for an evaluation of this early iteration of our algorithm, the selected datasets are sufficient to draw preliminary conclusions and point out future research directions.

Table 2: Important parameter settings of the `TimeSeriesKMeans` algorithm.

Parameter	Values	Explanation
n_clusters	$\{2, \ldots, \max\{10, 2 \cdot n_c\}\}$	Number of clusters. n_c: Number of classes.
max_iter	50	Iterations for k-means.
metric	dtw	Metric to be used.
max_iter_barycenter	10	Iterations for DBA.
init	k-means++	Cluster initialization method.

4.2 Experiment Setup

Important parameters of the `TimeSeriesKMeans` algorithm are listed in Tab. 2 algonside the values we used. The limits for `n_clusters`, `max_iter`, and `max_iter_barycenter` were chosen to limit the computation time required by the algorithm. The parameters of the interval weighting function are set to $\alpha = 0.2$ and $\beta = 0.6$. The threshold for the maximal allowed overlap $o_{\max}$ is set to $\theta = 0.4$ and a minimal gain ratio of 0.05 is required for a node split to be considered. For the current evaluation no hyperparameter optimization is considered and the allowed warping path for the DTW calculation has no additional restrictions.

4.3 Results

First, the performance of our interval-based decision tree (IBIT) and a 1-nearest-neighbour classifier with DTW as distance metric (1NN-DTW) is compared in Fig. 3a. Each point represents accuracies for one dataset. Points above the dashed line indicate a better performance of IBIT. We expect an IBIT model to perform better than a simple 1NN-DTW model because it is based on the same underlying distance metric while having a more sophisticated decision process. However, for 16 out of 25 datasets the 1NN-DTW performance is better.

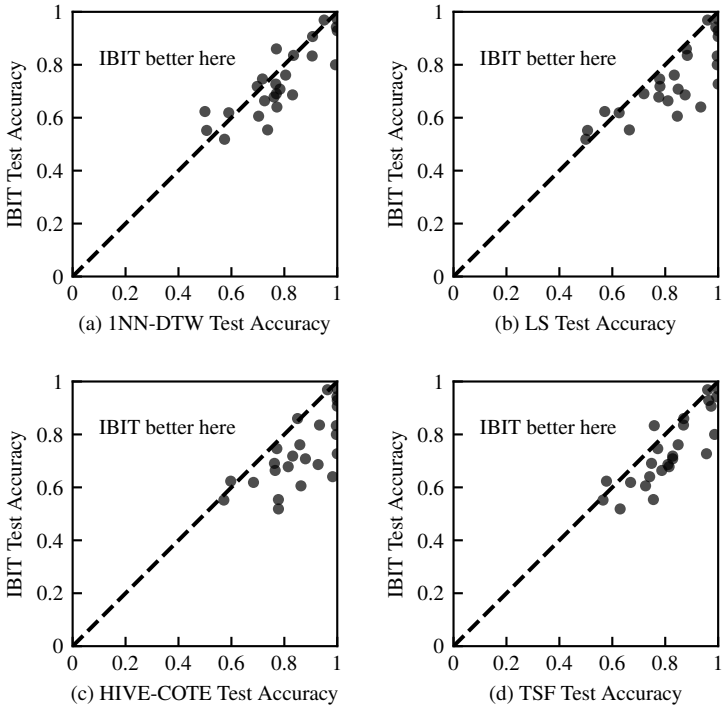

Figure 3: Test accuracies of IBIT model compared to test accuracies of selected algorithms as reported on the UCR TSA website [32] on 25 UCR TSA datasets.

This observation needs further investigating in the future. One possible reason for the lower performance of IBIT is the unsupervised clustering using k-means. For example, k-means clustering does not cope well with points which would be best clustered together but which are spread across a line in the feature space.

In addition, by looking at the cluster results, we observe that the hard cut-off of the time series at the interval limits may lead to a clustering dominated by shapes close to the interval limits. These shapes may be present inside the interval or outside of it depending on the stretch of the time series. Fig. 4 shows an example of this phenomenon for the interval $\mathscr{I} = [33..92]$ for the `ECG200` dataset. All time series posses steep slopes near the interval limits. However,

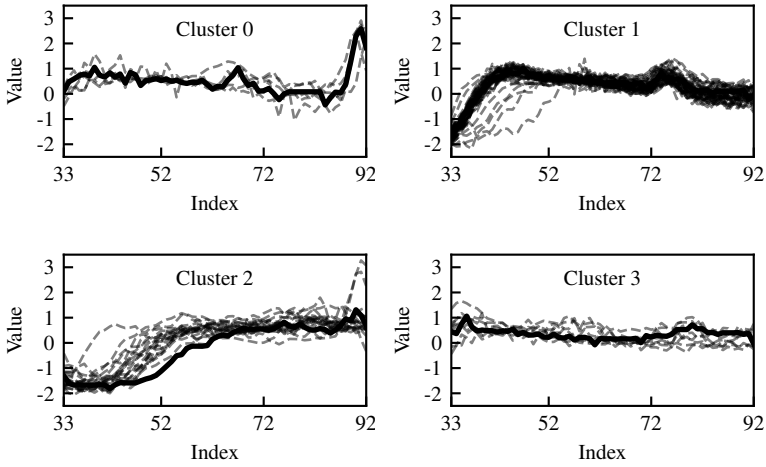

Figure 4: An example of a bad clustering due to the hard cut-off at the interval limits. The steep slopes at the limits of the interval do not always lie inside the interval. The presence or absence of these slopes inside the interval dominates the clustering result. Barycentres are displayed as solid lines.

these sections of steep slope are not always captured inside interval $\mathscr{I}$ because some time series are more stretched than others. The presence or absence of these slopes inside the interval dominates the clustering result and leads to a limited meaningfulness of the clustering result. One possible solution to limit the influence of the values close to the interval limits is to apply a weighted DTW penalizing these values. This should be investigated in the future.

Fig. 3b and 3c compare the IBIT performance to a shapelet-based approach (LS) [19] and an ensemble of classifiers including shapelet-based classifiers (HIVE-COTE). For 7 datasets LS and HIVE-COTE both achieve a classification accuracy close to 100% and the IBIT performance is not competitive for at least 3 of these datasets (`TwoLeadECG`, `ECGFiveDays`, `SyntheticControl`). All these datasets include approximately phase-aligned samples. Therefore, the low performance of IBIT on these datasets contradicts our expectations. A possible reason for the low performance of IBIT on these datasets is a selection of intervals which misses the important regions of interest in these time series.

This highlights the importance of interval selection. A further detailed analysis is required to come to a conclusive result in this case.

A performance comparison to TSF, a random forest with simple statistics of intervals as features, is shown in Fig. 3d. Although TSF only uses simple features (mean, standard deviation, slope) it outperforms the IBIT accuracy on most datasets. TSF has two main differences to our decision tree classifier. First, at each node in a decision tree of the TSF ensemble a new selection of intervals is considered. Therefore, the algorithm evaluates more intervals than IBIT does. Second, an ensemble of decision trees is used increasing the evaluation of different features further. While considering more than $O(L)$ intervals can be a suitable improvement to our algorithm, using an ensemble of decision tress cannot. This would lead to loss of interpretability of our models.

Although the unmodified IBIT models do not achieve state-of-the-art performance, they have the advantage of being interpretable. This does not only mean that the models can be verified but it also means that the IBIT models can be improved iteratively. An expert can investigate the intervals and cluster results an IBIT model uses for its decision process and iteratively refine the intervals or can add new clustering results with modified configurations. For instance, an expert can identify the inappropriate clustering results shown in Fig. 4 and make suitable adjustments to the intervals.

An evaluation of the tree complexities of 250 IBIT models trained on 10 different subsets of the training data shows that most decision trees are not overly complex and can easily be interpreted and modified by an expert. Each model is trained on 80% of the available training data. Fig. 5 shows the tree depths of all 250 models investigated. Interestingly, none of these 250 models has a tree depth greater than six. Possibly this is due to the fact that we put a restriction on the maximum overlap of intervals and at some deeper nodes of the tree no new interval candidates are available. For this evaluation, we did not prune the decision trees and the same underlying intervals were selected. Variation of intervals and evaluation of pruning techniques is planned in the future.

Fig. 6 shows the number of decision tree leafs across all 250 models. 50% of models have fewer than 11 leaves and 90% of models have fewer than 40

Figure 5: Decision tree depth counts of 250 trees from a 10-fold cross-validation for each of the 25 datasets.

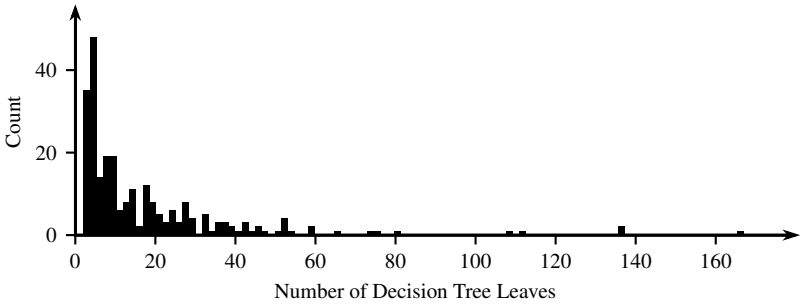

Figure 6: Decision tree leaf counts of 250 trees from a 10-fold cross-validation for each of the 25 datasets.

leaves. Only 10% of models have more than 40 leaves making them hard or at least tedious to interpret. This supports the hypothesis that IBIT models can be iteratively improved by an expert. This also shows that the learned decision trees are shallow but wide decision trees.

5 Conclusion and Outlook

In this paper we presented an algorithm to train interval-based interpretable decision trees. The algorithm is designed to create easy to interpret models

which can be iteratively improved by an expert. Modifications to improve the models can be identified and applied by an expert because of the inherent and global interpretability of the models. The simplicity of the resulting decision trees and the intuitive features help achieve this goal.

Although the algorithm does not achieve state of the art in terms of accuracy, it is important to note that accuracy is not the single most important criterium in all circumstances. Interpretable models can be analysed and verified by experts easily and spurious correlations in the data learned by the model can be identified and prohibited. Interpretable model are easy to improve iteratively to achieve certain goals and optimisation is not restricted to a single metric, e.g. the accuracy score. To investigate this hypothesis, cases studies where IBIT models are improved iteratively are a future field of research.

The evaluation presented in this paper shows preliminary results and a more comprehensive study is planned in the future. Once a more comprehensive study is done, we also plan to publish the code of our algorithm to make the results as reproducible as possible for the research community.

Further improvements to the algorithm we plan to investigated are

- extending the features by interpretable shapelet-based features to include phase-independent features,
- improving the scalability of the clustering through pruning strategies [33] or using a clustering strategy based on autocorrelation [34],
- applying pruning strategies to the decision trees,
- using weighted DTW to penalize values near interval limits,
- and optimising hyperparameters of the algorithm.

Acknowledgments

We would like to thank all UCR TSA data contributors and Bagnall et al. [32] for providing the datasets and accuracy results. Furthermore, we would like

to thank all contributors of open source software we used to implement our algorithm with, especially all contributors of the `tslearn` [28] library.

This work was partly funded by the German Federal Ministry of Education and Research (BMBF) within the project ITS.ML (grant no. 01IS18041D) and the Ministry of Economic Affairs, Innovation, Digitalisation and Energy of the State of North Rhine-Westphalia (MWIDE) within the project ML4Pro^2 (grant no. 005-1807-0090).

References

[1] S. R. Hong, J. Hullman and E. Bertini. “Human factors in model interpretability: Industry practices, challenges, and needs”. In: *Proceedings of the ACM on Human-Computer Interaction 4.CSCW1*, pp. 1–26. 2020.

[2] H. A. Dau, A. Bagnall, K. Kamgar, C.-C. M. Yeh, Y. Zhu, S. Gharghabi, C. A. Ratanamahatana and E. Keogh. “The UCR time series archive”. In: *IEEE/CAA Journal of Automatica Sinica 6.6*, pp. 1293–1305. 2019.

[3] A. Vellido. “The importance of interpretability and visualization in machine learning for applications in medicine and health care”. In: *Neural Computing and Applications 32.24*, pp. 18069–18083. 2019.

[4] C. Rudin. “Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead”. In: *Nature Machine Intelligence 1.5*, pp. 206–215. 2019.

[5] M. T. Ribeiro, S. Singh and C. Guestrin. “"Why should i trust you?" Explaining the predictions of any classifier”. In: phProceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1135–1144. 2016.

[6] H. Sakoe and S. Chiba. “Dynamic programming algorithm optimization for spoken word recognition”. In: *IEEE Transactions on Acoustics, Speech, and Signal Processing 26.1*, pp. 43–49. 1978.

[7] Y.-S. Jeong, M. K. Jeong and O. A. Omitaomu. “Weighted dynamic time warping for time series classification”. In: *Pattern Recognition 44.9*, pp. 2231–2240. 2011.

[8] T. Górecki and M. Łuczak. “Non-isometric transforms in time series classification using DTW”. In: *Knowledge-Based Systems 61*, pp. 98–108. 2014.

[9] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar and P.-A. Muller. “Deep learning for time series classification: A review”. In: *Data Mining and Knowledge Discovery 33.4*, pp. 917–963. 2019.

[10] Z. Wang, W. Yan and T. Oates. “Time series classification from scratch with deep neural networks: A strong baseline”. In: ph2017 International Joint Conference on Neural Networks (IJCNN), pp. 1578-1585. 2017.

[11] H. I. Fawaz, B. Lucas, G. Forestier, C. Pelletier, D. F. Schmidt, J. Weber, G. I. Webb, L. Idoumghar, P.-A. Muller and F. Petitjean. “InceptionTime: Finding AlexNet for time series classification”. In: *Data Mining and Knowledge Discovery 34.6*, pp. 1936–1962. 2020.

[12] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh and D. Batra. “Grad-CAM: Visual explanations from deep networks via gradient-based localization”. In: *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 618–626. 2017.

[13] A. Bagnall, J. Lines, A. Bostrom, J. Large and E. Keogh. “The great time series classification bake off: A review and experimental evaluation of recent algorithmic advances”. In: *Data Mining and Knowledge Discovery* 31.3, pp. 606–660. 2016.

[14] A. Bagnall, J. Lines, J. Hills and A. Bostrom. “Time-series classification with COTE: The collective of transformation-based ensembles”. In: *IEEE Transactions on Knowledge and Data Engineering 27.9*, pp. 2522–2535. 2015.

[15] J. Lines, S. Taylor and A. Bagnall. “Time series classification with HIVE-COTE”. In: *ACM Transactions on Knowledge Discovery from Data 12.5*, pp. 1–35. 2018.

[16] M. Middlehurst, J. Large, M. Flynn, J. Lines, A. Bostrom and A. Bagnall. "Hive-cote 2.0: a new meta ensemble for time series classification". In: *arXiv preprint arXiv:2104.07551*. 2021.

[17] L. Ye and E. Keogh. "Time series shapelets: : A new primitive for data mining". In: *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 947–956. 2009.

[18] J. Hills, J. Lines, E. Baranauskas, J. Mapp and A. Bagnall. "Classification of time series by shapelet transformation". In: *Data Mining and Knowledge Discovery 28.4*, pp. 851–881. 2013.

[19] J. Grabocka, N. Schilling and L. Schmidt-Thieme. "Learning time-series shapelets". In: *Proceedings of the 20th ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 392–401. 2014.

[20] A. Brunello, E. Marzano, A. Montanari and G. Sciavicco. "J48SS: A novel decision tree approach for the handling of sequential and time series data". In: *Computers 8.1*, pp. 21. 2019.

[21] H. Deng, G. Runger, E. Tuv and M. Vladimir. "A time series forest for classification and feature extraction". In: *Information Sciences 239*, pp. 142–153. 2013.

[22] M. Middlehurst, J. Large and A. Bagnall. "The canonical interval forest (CIF) classifier for time series classification". In: *2020 IEEE International Conference on Big Data (Big Data)*, pp. 188–195. 2020.

[23] A. Shifaz, C. Pelletier, F. Petitjean and G. I. Webb. "TS-CHIEF: A scalable and accurate forest algorithm for time series classification". In: *Data Mining and Knowledge Discovery 34.3*, pp. 742–775. 2020.

[24] T. L. Nguyen, S. Gsponer, I. Ilie, M. O'Reilly and G. Ifrim. "Interpretable time series classification using linear models and multi-resolution multi-domain symbolic representations". In: *Data Mining and Knowledge Discovery 33.4*, pp. 1183–1222. 2019.

[25] T. T. Nguyen, T. L. Nguyen and G. Ifrim. “A model-agnostic approach to quantifying the informativeness of explanation methods for time series classification”. In: *International Workshop on Advanced Analytics and Learning on Temporal Data*, pp. 77–94. 2020.

[26] P. J. Rousseeuw. “Silhouettes: A graphical aid to the interpretation and validation of cluster analysis”. In: *Journal of Computational and Applied Mathematics 20*, pp. 53–65. 1987.

[27] F. Petitjean, A. Ketterlin and P. Gançarski. “A global averaging method for dynamic time warping, with applications to clustering”. In: *Pattern Recognition 44.3*, pp. 678–693. 2001.

[28] R. Tavenard, J. Faouzi, G. Vandewiele, F. Divo, G. Androz, C. Holtz, M. Payne, R. Yurchak, M. Rußwurm, K. Kolar and E. Woods. “Tslearn, a machine learning toolkit for time series data”. In: *Journal of Machine Learning Research 21.118*, pp. 1–6. 2020.

[29] D. Arthur and S. Vassilvitskii. “K-means++: The advantages of careful seeding”. In: *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pp. 1027–1035. 2007.

[30] J. R. Quinlan. “Induction of decision trees”. In: *Machine Learning 1.1*, pp. 81–106. 1986.

[31] Y. Chen, E. Keogh, B. Hu, N. Begum, A. Bagnall, A. Mueen and G. Batista. “The UCR time series classification archive”. `www.cs.ucr.edu/~eamonn/time_series_data/` (Online, 23.09.2021). 2015.

[32] A. Bagnall, E. Keogh, J. Lines, A. Bostrom, J. Large and M. Middlehurst. “UEA/UCR time series classification repository”. `www.timeseriesclassification.com` (Online, 23.09.2021).

[33] N. Begum, L. Ulanova, J. Wang and E. Keogh. “Accelerating dynamic time warping clustering with a novel admissible pruning strategy”. In: *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.*, pp. 49–58. 2015.

[34] J. Paparrizos and L. Gravano. “k-shape: Efficient and accurate clustering of time series”. In: *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data.*, pp. 1855–1870. 2015.