Autoassociative Random Forest Clustering

Robbe D'hondt*[,1,2][0000−0001−7843−2178], Felipe Kenji Nakano[1,2][0000−0002−4884−9420], and Celine Vens[1,2][0000−0003−0983−256X]

[1] KU Leuven, Dept. Public Health and Primary Care, Kortrijk, Belgium
{robbe.dhondt,felipekenji.nakano,celine.vens}@kuleuven.be
[2] itec, imec research group at KU Leuven, Kortrijk, Belgium

Abstract. Random forests are state-of-the-art supervised machine learning models for tabular data. The decision paths in a trained random forest can be used to generate an affinity measure between instances, which can in turn be used in any similarity matrix-based clustering algorithm. However, in the unsupervised setting, labels are not available to train the random forest in the first place. A popular extension of the algorithm for unsupervised learning involves the introduction of a synthetic "negative class", transforming the problem into binary classification. In this paper, we instead propose AutoAssociative Random Forest Clustering (AARFC), transforming the problem into multi-target regression by training the random forest on the variance in the input features. AARFC more directly optimizes towards the end goal of clustering the instances and does not increase the number of instances in the original input data. We have extensively benchmarked AARFC against the current state-of-the-art using a collection of 223 benchmark datasets with reference labelings. The results show that AARFC is computationally superior, while being competitive. We also propose a hybrid approach that adaptively chooses the best method based on several internal validation criteria. Despite the fact that this hybrid approach outperforms both the current approach used in the literature as well as AARFC, k-means still outperforms the random forest clustering method in the majority of benchmark datasets.

Keywords: random forest · clustering · autoassociation · multi-target

1 Introduction

In machine learning, clustering is the unsupervised task of discovering groups of similar objects in a dataset, where objects in different groups should be as dissimilar as possible. There is a wide variety of existing clustering algorithms, all with different properties and underlying assumptions. In this study, we focus on random forest based clustering algorithms.

Random forests, originally introduced by Breiman [9], are popular supervised models obtained by training an ensemble of decision trees on independent bootstrap samples using only a random subset of all the available features per split. The random forest algorithm is a flexible algorithm that can generate complex decision boundaries relatively fast, is robust to noise and outliers, and can easily

handle skewed features and mixed feature types. Because of these nice properties, it is an attractive algorithm to use for the clustering setting, and hence several unsupervised adaptations of the algorithm have been proposed.

A popular approach to make random forests work in the unsupervised setting, originally introduced by Breiman and Cutler [10], is to transform the problem into a binary classification task by introducing a synthetic "negative class" (whereby the original instances form the "positive class"). Instances of this negative class are for example sampled uniformly from the feature space [24], or (more popular) sampled from the empirical marginal distributions of the features [30]. After training a random forest on this binary classification problem, an affinity measure between the instances can be defined, based on the path they take through the decision trees of the forest. This affinity can then be used in any distance-based clustering algorithm, such as agglomerative or spectral clustering.

Related research since the introduction of random forest clustering has mostly focused on the properties of the resulting affinity measure [30], investigating different affinity normalizations [36], and more explainable clusterings generated by pruned single decision trees [7, 24]. The research area is still under active development, with several recently proposed adaptations [4–6, 23, 26, 34]. However, most of these adaptations are still based on the introduction of a synthetic negative class. This effectively doubles the number of instances in the original dataset and only indirectly optimizes towards the end goal of clustering the data.

Therefore, here we propose instead to train an autoassociative random forest, which uses a multi-target splitting heuristic on the input features themselves. This results in a more compact and representative random forest. This proposed approach extends the ideas of unsupervised predictive clustering trees that generate a hierarchical clustering from a single tree [7], autoassociative regression trees [31], and autoreplicative random forests [1]. Specifically, we build more powerful ensembles of multi-target trees, and process the decision paths of this ensemble into an affinity matrix which can be used for clustering. Additionally, we also introduce a hybrid approach that automatically selects the best-performing approach based on internal clustering validation criteria.

The contributions of this manuscript are threefold: (1) we propose a novel, more computationally efficient way of similarity-based clustering with random forests, by training on the variance of the original input features, (2) this novel approach is extensively compared to the current approach (i.e., by introducing a synthetic negative class) through a comprehensive benchmarking suite of 223 datasets, and (3) we propose a hybrid approach that adaptively chooses the best clustering between the two approaches, which we then compare to a set of traditional clustering algorithms on this same benchmarking suite.

2 Proposed Approach

In this section, we introduce the proposed random forest clustering approach. In Section 2.1, we define the autoassociative approach and show how it is different from the literature. Then, in Section 2.2, the hybrid approach is introduced.

2.1 Autoassociative Random Forest Clustering

Let $X \in \mathbb{R}^{n \times p}$ be the input data matrix with n instances and p features. For the approach used in the literature [10, 24, 30, 36], all instances in X are assigned label $\oplus$. Then, a synthetic negative class $\ominus$ is created and assigned to all instances in $\tilde{X} \in \mathbb{R}^{n \times p}$, a copy of X where in each column independently the values have been randomly permuted. A random forest is then trained on the augmentation of X with $\tilde{X}$ and their corresponding binary class labels. We call this approach Synthetic Negative Random Forest Clustering (= SNRFC, see Eq. (1)). In this paper, we propose to train the random forest on instances in X alone, by instead considering the summed variance reduction of the original features. This is achieved by considering the z-scores $z_{ij} = \frac{x_{ij} - \hat{\mu}(x_{:j})}{\hat{\sigma}(x_{:j})}$ of the original dataset as outputs (normalization ensures that each target is weighted equally) and training a multi-target regression[3] random forest [22]. We call this approach AutoAssociative Random Forest Clustering (= AARFC, see Eq. (1)).

$$\text{SNRFC: (existing)} \left[\begin{array}{ccc|c} x_{11} & \cdots & x_{1p} & \oplus \\ \vdots & \ddots & \vdots & \vdots \\ x_{n1} & \cdots & x_{np} & \oplus \\ \hline \tilde{x}_{11} & \cdots & \tilde{x}_{1p} & \ominus \\ \vdots & \ddots & \vdots & \vdots \\ \tilde{x}_{n1} & \cdots & \tilde{x}_{np} & \ominus \end{array}\right] \quad \text{vs.} \quad \text{AARFC: (proposed)} \left[\begin{array}{ccc|ccc} x_{11} & \cdots & x_{1p} & z_{11} & \cdots & z_{1p} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{np} & z_{n1} & \cdots & z_{np} \end{array}\right] \tag{1}$$

Once the random forest has been trained, we collect the *decision path* of each instance $x_{i:}$ in the input dataset X. The decision path is defined as the binary vector indicating for each node in the random forest whether an instance passed through that node (= 1) or not (= 0). Note that a decision path vector has as many entries as there are nodes in the random forest (let's call this ν). The sparse matrix of decision paths $D \in \{0,1\}^{n \times \nu}$ is then used to define an affinity matrix amongst the instances: the more nodes in the random forest they have in common, the more similar the instances. The affinity or similarity s for instances $x_{i:}$ and $x_{j:}$ with decision path vectors $d_{i:}$ and $d_{j:}$ can be defined as

$$s(x_{i:}, x_{j:}) = \frac{2 d_{i:}^T d_{j:}}{\sum_{k=1}^{\nu} d_{ik} + \sum_{k=1}^{\nu} d_{jk}}. \tag{2}$$

The resulting similarity matrix $S \in \mathbb{R}^{n \times n}$ is then used in a distance-based clustering algorithm to obtain a final clustering $\hat{c} \in \{1, 2, \ldots, k\}^n$. Here, we opt for spectral clustering. In short, spectral clustering uses k-means on an embedded space spanned by a subset of the eigenvectors of the normalized Laplacian of the similarity matrix. It is a useful algorithm for finding non-convex clusters and is especially suitable when the data is in the form of a similarity matrix (as we obtain from the random forest here).

[3] In future work, the approach can be extended to also handle categorical features (and even missing data).

2.2 Hybrid Approach

To improve the clustering performance, we also propose a hybrid approach. For this, we first use both approaches (AARFC and SNRFC) to obtain two separate clusterings. Then, for both clusterings we compute several *internal* clustering validation criteria (i.e., evaluation criteria that do not use a reference labeling), which can be seen as proxies of clustering performance. Specifically, we consider here the Davies-Bouldin index [3, Def. 5], Caliński-Harabasz index [12, Eq. (3)], the Silhouette index [29, Sec. 2] (both the overall average silhouette score and the cluster-weighted average), and the Ball-Hall index [2, $\overline{\text{AD}}$]. For each of these indices, a 'win' is assigned either to AARFC or SNRFC, based on which algorithm produces the best score. The algorithm with the highest number of wins for a given dataset is then chosen for the final clustering.

3 Evaluation Setup

The primary focus of our evaluation is to compare SNRFC with AARFC (see Eq. (1)). Therefore, the steps explained in Section 2.1 (computing the similarity score and running spectral clustering) are used as-is for both algorithms and not tuned further. Note that only the spectral clustering step depends on the number of clusters k that we want to split our dataset into. Hence, when computing a clustering on the same dataset for several values of k, only the spectral clustering step needs to be rerun.

Both approaches were subjected to the comprehensive clustering benchmark suite presented by Gagolewski et al. [17][4]. This suite consists of 9 distinct *batteries* (= collections) of clustering datasets, each with 1 or more reference labelings. The suite has a wide variety of different datasets in terms of number of instances, features, and clusters (see Table 1 for a summary of the dataset characteristics). Of the 223 available datasets, 6 were too big to be able to store the $n \times n$ affinity matrix. More specifically, these are 4 of the 20 datasets in `sipu` (`birch1`, `birch2`, `worms2`, `worms64`) and the 2 datasets in the `mnist` battery. In these cases, the approach was fitted and evaluated only on a balanced subset of 10% of the original instances instead (in Table 1 the original size is shown).

The clustering pipeline was implemented[5] in Python 3.11.6 using scikit-learn 1.3.2. All algorithms (spectral clustering, random forest classifier and random forest regressor) were used with default parameters. The SNRFC and AARFC approaches are compared in terms of their runtime and maximum adjusted rand index (ARI) across all reference labelings for a particular dataset (as we assume the algorithm 'discovered' that reference labeling). The ARI of a clustering $\hat{c} \in \{1, 2, \ldots, k\}^n$ as compared to a reference clustering $c \in \{1, 2, \ldots, k\}^n$ with k

[4] See `https://clustering-benchmarks.gagolewski.com` (version 1.1.2).
[5] The code is available at `https://github.com/robbedhondt/AARFC`.

Table 1. Summary of benchmark battery characteristics. For each battery, the number of datasets # is shown, along with the average number of instances $\bar{n}$, number of dimensions $\bar{p}$, and number of clusters $\bar{k}$.

Battery	#	$\bar{n}$	$\bar{p}$	$\bar{k}$	Synthetic?
`graves` [19]	10	636.0	2.0	3.2	Yes
`fcps` [32]	9	1054.9	2.4	3.1	Yes
`other` [15, 18, 20, 27]	8	4695.5	2.5	5.4	Some
`wut`	22	2560.9	2.1	3.9	Yes
`sipu` [16]	20	22948.1	5.1	19.4	Yes
`uci` [21]	8	706.2	22.2	5.0	No
`g2mg` [17]	72	2048.0	31.9	2.0	Yes
`h2mg` [17]	72	2048.0	31.9	2.0	Yes
`mnist` [8, 13]	2	70000.0	751.5	10.0	No
all	223	4525.9	29.1	4.6	

clusters[6] is defined as (where $\mathbf{1}_A$ represents the indicator function):

$$\mathrm{ARI}(c,\hat{c}) = \frac{\mathrm{RI}(c,\hat{c}) - \mathbb{E}_{\hat{c}}[\mathrm{RI}(c,\hat{c})]}{\max_{\hat{c}}(\mathrm{RI}(c,\hat{c})) - \mathbb{E}_{\hat{c}}[\mathrm{RI}(c,\hat{c})]}$$
$$\mathrm{RI}(c,\hat{c}) = \frac{\sum_{i=1}^{n}\sum_{j=i+1}^{n}\mathbf{1}_{[(c_i=c_j)\wedge(\hat{c}_i=\hat{c}_j)]\vee[(c_i\neq c_j)\wedge(\hat{c}_i\neq\hat{c}_j)]}}{n(n-1)/2}.$$

The secondary focus of our evaluation is the performance of the hybrid approach. First, we compare the actually best clustering (AARFC or SNRFC) to the clustering proposed by the hybrid approach. Then, the performance (on ARI) of the hybrid approach is compared to a set of clustering algorithms covering the major types of clustering methodologies (centroid-based, distribution-based, and hierarchical). Specifically, we compare to k-means [25], Gaussian mixture modeling [14], BIRCH [35], spectral clustering [28], and agglomerative clustering [33].

4 Results

In this section, we discuss the results of our study. In Section 4.1, we go over the full benchmarking comparison between AARFC and SNRFC. To further analyze these results, we present a case study in Section 4.2, comparing both methods in more detail. We follow up in Section 4.3 with the timing results of the benchmarking effort. Finally, in Section 4.4 the results of the hybrid approach are discussed.

[6] The ARI score is also defined when the number of clusters in both clusterings is different. However, this is never the case here, as the parameter determining the number of clusters is always set to the number of clusters in the reference clustering.

4.1 Benchmarking Results

In Fig. 1, SNRFC and AARFC are compared on the full benchmark suite. Depending on the battery, the difference between both methods can be rather large. We see that our proposed AARFC approach definitely works better on the `fcps` and `h2mg` batteries, and the current SNRFC approach works better on `g2mg`. However, with a mean and median ARI difference of 0.01 and 0.002 respectively, both approaches perform very similar on average and we cannot appoint one of them as the definite winner (a Wilcoxon rank sum test fails to assess any overall statistically significant difference, $p = 0.135$). Instead, these results justify the hybridization of both approaches.

For further insight in these results, the performance for every separate dataset was compared in Fig. 2. Interestingly, we see that in many datasets from `h2mg`, SNRFC performs no better than randomly assigning clusters, while the results for the other batteries are a bit more nuanced. The large difference in results between `g2mg` and `h2mg` is rather intriguing. Indeed, these batteries are quite similar in nature, as they both represent two Gaussian-shaped clusters in various dimensionalities (from 1 to 128) with various distribution variances. While the points in `g2mg` datasets are sampled from a Gaussian distribution themselves, the points in `h2mg` datasets are sampled from a sphere of which the radius follows a Gaussian distribution. Upon closer inspection, after some discordant results in the relatively 'easier' datasets ($p \leq 8$), there is a clear trend of AARFC winning in `h2mg` and SNRFC winning in `g2mg`. A principal component analysis (see Fig. 3) reveals that the instances in the synthetic negative class have more overlap with the existing data for `h2mg` than for `g2mg`, resulting in a worse division of the 2 groups. This could explain the bad performance of SNRFC on a big chunk of these datasets.

4.2 AARFC Versus SNRFC: A Case Study

We argue that our AARFC approach is more intuitive than the SNRFC approach being used in the literature. The SNRFC approach uses a random forest that is optimized to disambiguate instances in the original data manifold from instances outside (i.e., instances where each feature value is sampled at random from one of the original instances). Rather than *splitting* the original set of instances into clusters, this random forest tries rather to group them *together* (to split them from the synthetic negative class). In contrast, our AARFC approach will directly try to make clusters of similar instances that are dissimilar from the other instances, as our random forest is optimized to reduce the variance in the original features. Despite this, SNRFC seems to work rather well in practice, explaining also the popularity of this approach in the literature (see Section 1).

To exemplify this counterintuitivity, let us make a case study of the `fuzzyx` dataset $\in \mathbb{R}^{1000\times 2}$ in the `graves` battery. This dataset has reference clusterings with 2, 4, and 5 clusters. Without loss of generality, we focus on the $k = 4$ case here, as the conclusions of the case study remain the same for $k = 2$ and $k = 5$. As can be seen in Fig. 4, SNRFC fails to find a relevant clustering, while

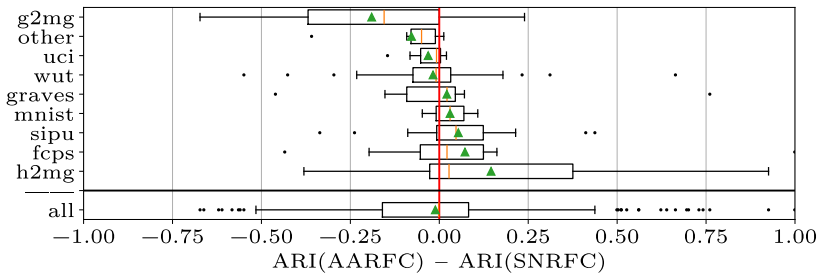

Fig. 1. Per-battery improvement in adjusted rand index (ARI) of the proposed autoassociative approach (AARFC) over the current approach with synthetic negative class (SNRFC). On the boxplots, orange vertical lines indicate the median, green triangles indicate the mean.

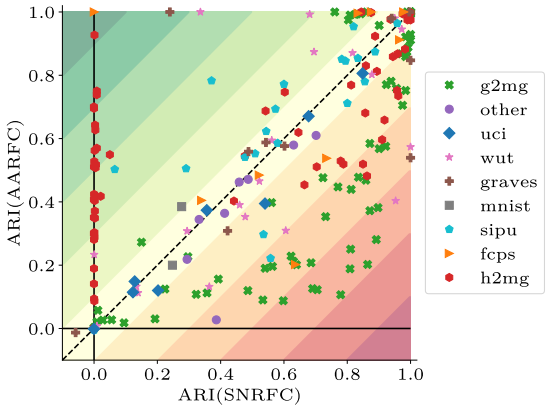

Fig. 2. Per-dataset ARI comparison between SNRFC and AARFC. Each point in this plot represents one dataset, its marker links it to its battery.

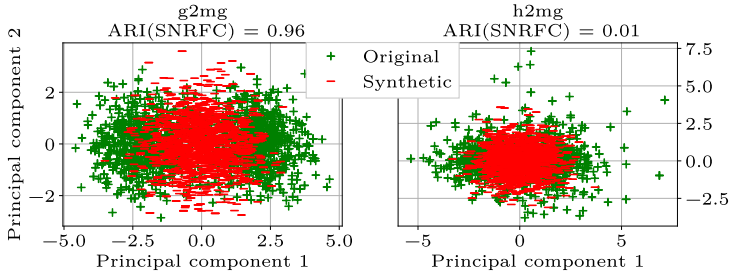

Fig. 3. Principal component analysis for a typical dataset where SNRFC performs well on `g2mg` but not on `h2mg` ($p = 16$). In both projections, there are two clusters: one centered in the left side of the graph and one in the right of it (corresponding to the sign of the first principal component). Unlike `h2mg`, for `g2mg` the synthetic dataset can clearly be separated from the two clusters in the original data.

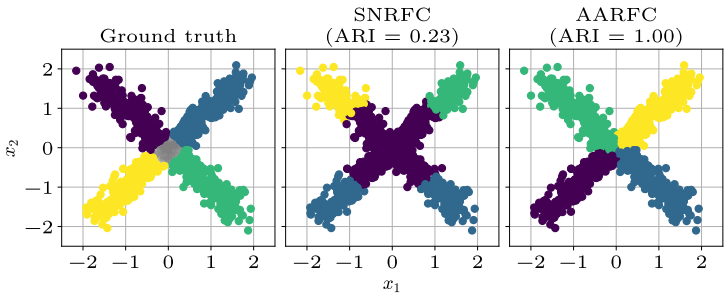

Fig. 4. Case study of `fuzzyx` in the `graves` battery for $k = 4$. In the first figure, the reference labeling is shown, with noise points (not used for performance evaluation) indicated in gray. The second and third figure show the labelings obtained by SNRFC and AARFC respectively.

AARFC does manage to find back the original clusters (see also [30, Figure A.2]). This can be explained as follows: note that the distribution of x_1 and x_2 values are relatively uniform over their respective axes. The synthetic negative class hence consists of instances approximately sampled uniformly at random from $\approx [-2, 2]^2$. The resulting binary classification is hard, as the 2 classes are mixed well. Heuristically, the best thing to do for the random forest is to first 'split out' the middle part of the cross (as this is a region of high $\oplus$ density), which is why this always turns up as a separate cluster (also for $k = 2$ and $k = 5$).

4.3 Timing Comparison

For the computational complexity of the methods, there are two factors at play: the number of instances n and the number of features p. Asymptotically, the time complexity for training a single-target (balanced) decision tree is $\mathcal{O}(n \log(n) p)$, as we need to test p features for each of the n examples at each of the $\log(n)$ depth levels of the tree. Doubling the number of instances thus at least doubles the complexity.

However, in SNRFC we train a single-target random forest, while in AARFC a multi-target random forest with p targets must be trained. This effectively multiplies the complexity by p, as each split has to compute the variance on p targets instead of just 1. Despite this, we argue that the number of targets p will never be *too* high in practical applications (especially compared to n). Clustering methodologies suffer greatly from the curse of dimensionality and it is usually advised to first somewhat reduce the dimensionality in preprocessing (e.g. with a principal component analysis). Furthermore, the method could easily be made more scalable by considering random output projections [11], where for each decision tree in the random forest only a random subset of the targets is chosen to compute the variance reduction.

Without any adaptations, the timing results for fitting the random forest and obtaining the affinity matrix in all benchmark datasets are shown in Fig. 5.

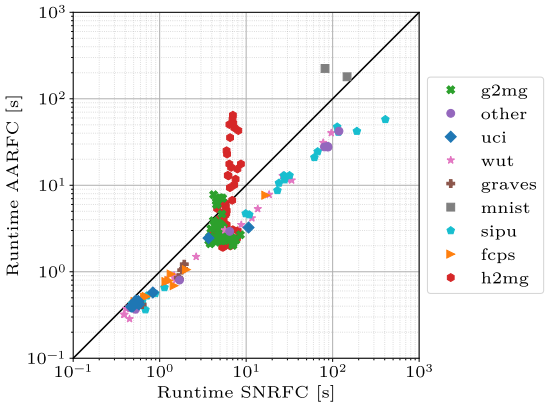

Fig. 5. Comparison of SNRFC and AARFC in terms of their wall-clock runtime (excluding the spectral clustering step).

The spectral clustering step is omitted, as it is of the same complexity for both methods and it only dilutes the results shown in this figure. For 85% of all datasets (191 out of 223), the runtime of AARFC is strictly smaller than the one of SNRFC (a Wilcoxon rank sum test indicates that it is significantly smaller, $p < 10^{-14}$). Even for the `mnist` battery, with $n = 7000$ and $p \in \{719, 784\}$, our method still performs quite well compared to SNRFC.

4.4 Hybrid Approach and Comparison to State-of-the-Art

As a final experiment, we investigate the possibility of integrating both approaches into one by adaptively choosing the best one. With the decision rule of using either AARFC or SNRFC based on which one scores better than the other on the most internal validation criteria, we find that the best clustering (in terms of ARI) is chosen for 177 datasets. This simple decision rule is thus already optimal for 79% of the datasets in the benchmark battery (while picking an approach at random is only optimal in 50% of datasets).

This hybrid approach was then compared to a set of traditional clustering algorithms, as shown in Fig. 6. In the majority of datasets, the hybrid approach performs better than BIRCH, spectral clustering, hierarchical clustering, AARFC and SNRFC. Surprisingly, the hybrid approach performs worse than k-means and Gaussian mixture modeling. This can be attributed to the nature of the Gaussian-shaped clusters in the `h2mg` and `g2mg` batteries, which are especially suitable for the k-means objective function. Indeed, k-means is doing at least as good as all other methods on at least 95% of these datasets.

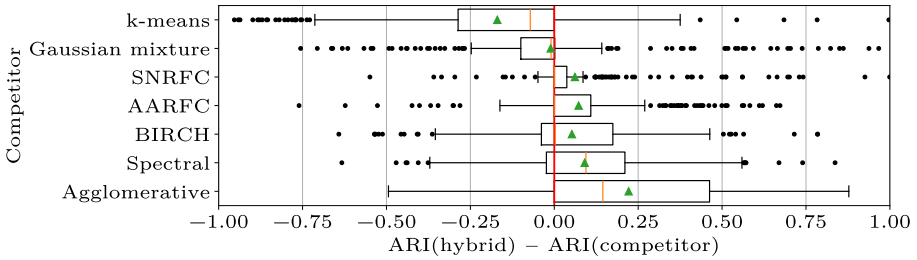

Fig. 6. ARI improvement of hybrid approach over traditional clustering algorithms.

5 Conclusion

In this paper, we have proposed an alternative way to train a random forest for clustering. While a popular technique is to introduce a synthetic negative class (SNRFC), doubling the number of instances in the original dataset, our proposed approach instead trains an autoassociative random forest on the original input features (AARFC). The resulting random forest is optimized more directly for the clustering objective. AARFC is also computationally superior over SNRFC, and could in future work easily be improved in the higher-dimensional regime by using random output projections per tree. Therefore, future work in the field of could benefit from using the AARFC approach over SNRFC.

In terms of clustering performance, AARFC and SNRFC are competitive. Since either of the two approaches can be superior to the other on a dataset-per-dataset bases, we have also defined a hybrid approach that intelligently selects one of the two approaches at runtime. This selection is currently based on a simple decision rule using several internal cluster validation criteria, but could in future work also be based on a pretrained model that incorporates a meta-learning heuristic. Through extensive benchmarking, we have shown that the hybrid approach outperforms AARFC and SNRFC but, as a limitation, it underperforms traditional clustering approaches such as k-means. By extension, the current approach with synthetic negative class, in light of this elaborate benchmarking suite, is also outperformed by k-means.

Code Availability The benchmarking code is available at `https://github.com/robbedhondt/AARFC`.

Acknowledgements This work was funded by Research Fund Flanders (FWO fellowships 1S38023N and 1235924N) and supported by the Flemish government (through the AI Research Program).

References

[1] E. Antonenko and J. Read. *Chains of Autoreplicative Random Forests for Missing Value Imputation in High-Dimensional Datasets*. 2023. arXiv: 2301.00595. Pre-published.

[2] G. H. Ball and D. J. Hall. *ISODATA, a Novel Method of Data Analysis and Pattern Classification*. Vol. 4. Stanford, Menlo Park, CA, 1965.

[3] J. Bezdek and N. Pal. "Some New Indexes of Cluster Validity". In: *IEEE Trans. Syst. Man. Cybern.* 28.3 (1998), pp. 301–315.

[4] M. Bicego. "DisRFC: A Dissimilarity-Based Random Forest Clustering Approach". In: *Pattern Recognit.* 133 (2023), p. 109036.

[5] M. Bicego, F. Cicalese, and A. Mensi. "RatioRF: A Novel Measure for Random Forest Clustering Based on the Tversky's Ratio Model". In: *IEEE Trans. Knowl. Data Eng.* (2021), pp. 1–1.

[6] M. Bicego and F. Escolano. "On Learning Random Forests for Random Forest-clustering". In: *25th International Conference on Pattern Recognition (ICPR)*. 2021, pp. 3451–3458.

[7] H. Blockeel, L. De Raedt, and J. Ramon. "Top-Down Induction of Clustering Trees". In: *Proceedings of the Fifteenth International Conference on Machine Learning*. ICML '98. San Francisco, CA, USA, 1998, pp. 55–63.

[8] L. Bottou et al. "Comparison of Classifier Methods: A Case Study in Handwritten Digit Recognition". In: *12th IAPR International Conference on Pattern Recognition, Vol. 3 - Conference C*. Vol. 2. 1994, 77–82 vol.2.

[9] L. Breiman. "Random Forests". In: *Machine Learning* 45.1 (2001), pp. 5–32.

[10] L. Breiman and A. Cutler. *Manual–Setting Up, Using, And Understanding Random Forests V4.0*. UC Berkeley, 2003.

[11] M. Breskvar, D. Kocev, and S. Džeroski. "Ensembles for Multi-Target Regression with Random Output Selections". In: *Mach Learn* 107.11 (2018), pp. 1673–1709.

[12] T. Caliński and J. Harabasz. "A Dendrite Method for Cluster Analysis". In: *Commun. Stat.* 3.1 (1974), pp. 1–27.

[13] G. Cohen, S. Afshar, J. Tapson, and A. van Schaik. *EMNIST: An Extension of MNIST to Handwritten Letters*. Version 1. 2017. DOI: 10.48550/arXiv.1702.05373. arXiv: 1702.05373 [cs]. Pre-published.

[14] A. P. Dempster, N. M. Laird, and D. B. Rubin. "Maximum Likelihood from Incomplete Data via the EM Algorithm". In: *J R Stat Soc Series B Stat Methodol* 39.1 (1977), pp. 1–38.

[15] R. A. Fisher. "The Use of Multiple Measurements in Taxonomic Problems". In: *Annals of Eugenics* 7.2 (1936), pp. 179–188.

[16] P. Fränti and S. Sieranoja. "K-Means Properties on Six Clustering Benchmark Datasets". In: *Appl Intell* 48.12 (2018), pp. 4743–4759.

[17] M. Gagolewski. "A Framework for Benchmarking Clustering Algorithms". In: *SoftwareX* 20 (2022).

[18] M. Gagolewski, M. Bartoszuk, and A. Cena. "Genie: A New, Fast, and Outlier-Resistant Hierarchical Clustering Algorithm". In: *Inf. Sci.* 363 (2016), pp. 8–23.

[19] D. Graves and W. Pedrycz. "Kernel-Based Fuzzy Clustering and Fuzzy Clustering: A Comparative Experimental Study". In: *Fuzzy Sets Syst.* Theme: Forecasting, Classification, and Learning 161.4 (2010), pp. 522–543.
[20] G. Karypis, E. Han, and V. Kumar. "A Hierarchical Clustering Algorithm Using Dynamic Modeling". In: (1999).
[21] M. Kelly, R. Longjohn, and K. Nottingham. *The UCI Machine Learning Repository.* URL: `https://archive.ics.uci.edu`.
[22] D. Kocev, C. Vens, J. Struyf, and S. Džeroski. "Tree ensembles for predicting structured outputs". In: *Pattern Recognit.* 46.3 (2013), pp. 817–833.
[23] E. Laber, L. Murtinho, and F. Oliveira. "Shallow Decision Trees for Explainable k-Means Clustering". 2021. URL: `http://arxiv.org/abs/2112.14718` (visited on 2022-05-04).
[24] B. Liu, Y. Xia, and P. Yu. "Clustering Via Decision Tree Construction". In: *Foundations and Advances in Data Mining.* Studies in Fuzziness and Soft Computing. Berlin, Heidelberg: Springer, 2005, pp. 97–124.
[25] S. Lloyd. "Least Squares Quantization in PCM". In: *IEEE Transactions on Information Theory* 28.2 (1982), pp. 129–137.
[26] A. Mantero and H. Ishwaran. "Unsupervised Random Forests". In: *Stat Anal Data Min* 14.2 (2021), pp. 144–167.
[27] L. McInnes, J. Healy, and S. Astels. "Hdbscan: Hierarchical Density Based Clustering." In: *J. Open Source Softw.* 2.11 (2017), p. 205.
[28] A. Ng, M. Jordan, and Y. Weiss. "On Spectral Clustering: Analysis and an Algorithm". In: *Adv Neural Inf Process Syst* 14 (2001).
[29] P. J. Rousseeuw. "Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis". In: *J. Comput. Appl. Math.* 20 (1987), pp. 53–65.
[30] T. Shi and S. Horvath. "Unsupervised Learning With Random Forest Predictors". In: *J. Comput. Graph. Stat.* 15.1 (2006), pp. 118–138.
[31] C. Smyth, D. Coomans, Y. Everingham, and T. Hancock. "Auto-Associative Multivariate Regression Trees for Cluster Analysis". In: *Chemom. Intell. Lab. Syst.* 80.1 (2006), pp. 120–129.
[32] A. Ultsch. "Clustering Wih SOM: U*C". In: *Proc. Workshop on Self-Organizing Maps* (2005).
[33] J. H. Ward. "Hierarchical Grouping to Optimize an Objective Function". In: *J. Am. Stat. Assoc.* 58.301 (1963), pp. 236–244.
[34] Y. Yi, D. Sun, P. Li, T.-K. Kim, T. Xu, and Y. Pei. "Unsupervised Random Forest for Affinity Estimation". In: *Comp. Visual Media* 8.2 (2022), pp. 257–272.
[35] T. Zhang, R. Ramakrishnan, and M. Livny. "BIRCH: A New Data Clustering Algorithm and Its Applications". In: *Data Min. Knowl. Discov.* 1 (1997), pp. 141–182.
[36] X. Zhu, C. C. Loy, and S. Gong. "Constructing Robust Affinity Graphs for Spectral Clustering". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition.* 2014, pp. 1450–1457.