LESS IS MORE: SELECTING THE RIGHT BENCHMARKING SET OF DATA FOR TIME SERIES CLASSIFICATION

Anonymous authors
Paper under double-blind review

ABSTRACT

In this paper, we have proposed a new pipeline for landscape analysis of machine learning datasets that enables us to better understand a benchmarking problem landscape, allows us to select a diverse benchmark datasets portfolio, and reduce the presence of performance assessment bias via bootstrapping evaluation. Combining a large multi-domain representation corpus of time-series specific features and the results of a large empirical study of time-series classification (TSC) benchmark, we showcase the capability of the pipeline to point out issues with non-redundancy and representativeness in the benchmark. By observing discrepancy between the empirical results of the bootstrap evaluation and recently adapted practices in TSC literature when introducing novel methods we warn on the potentially harmful effects of tuning the methods on certain parts of the landscape (unless this is an explicit and desired goal of the study). Finally, we propose a set of datasets uniformly distributed across the landscape space one should consider when benchmarking novel TSC methods.

1 INTRODUCTION

Reliable and unbiased algorithm performance evaluation is paramount in machine learning (ML) research, due to its indispensable role in the identification of the strengths and weaknesses of existing algorithms, tracking the improvements made by newly introduced algorithms, and determining directions for future research efforts (Cawley & Talbot, 2010). Such an evaluation is conditioned on the existence of high-quality benchmark datasets, which are typically subjected to the requirements of high representativeness, non-redundancy, scalability, reusability, experimental verification, and inclusion of both positive and negative cases (Sarkar et al., 2020; Schaafsma & Vihinen, 2018; Bartz-Beielstein et al., 2020). The role of representativeness is especially highlighted by the fact that low diversity of instances in benchmark libraries can lead to developing algorithms that are tuned to achieve good results on the benchmark instances, without taking into account their generalization to diverse or previously unseen instances, and making sure the performance is invariant to certain instance properties (Smith-miles et al., 2014).

The necessity of generating a time series benchmark has been claimed for a long period. Eamon et al. (Keogh & Kasetty, 2003) argue that the low quality of the empirical evaluation is manifesting itself in different tasks like clustering indexing, classification, and segmentation. Furthermore, the authors identify data bias, defined as the conscious or unconscious use of a particular testing dataset to confirm the desired finding, to be one of the main obstacles towards fair evaluation and comparison of time series algorithms. They further show that many of the advances claimed in the literature have little generalizability to other problems and introduce the need for assembling large, heterogeneous benchmark datasets, which provide good coverage of the entire spectrum of time series properties, such as stationarity, smoothness, symmetry, among other relevant time series properties. Since then, efforts have been dedicated to mitigating the data bias issue in the evaluation of time series algorithms by increasing the quantity of the available benchmark datasets. However, the part that is often overlooked is the evaluation of the extent to which the benchmarks satisfy the previously introduced quality requirements and which parts from the problem landscape they cover.

The University of California, Riverside (UCR) repository of time series classification and clustering datasets (Dau et al., 2018), has been one of the major contributors to the improvement of the evalua-

tion of algorithms tackling the multi-class classification task of time series data. Over the years, the repository has gradually been expanded, and currently, it contains 128 datasets. In a recent study, the authors of (Bagnall et al., 2016) reported one thousand published papers that made use of at least one of the datasets in the repository (Dau et al., 2018). Because of its frequent use in evaluating existing and newly introduced algorithms, the affirmation of its quality is of critical importance.

According to a survey done for the latest expansion of the UCR repository in 2018, the most active researchers in the time series data mining community pointed out the following directions for further improvement of the repository: the inclusion of longer time series, datasets of variable length, multi-variable datasets, highly unbalanced datasets, and data sets with very small training sets, suitable for benchmarking data augmentation techniques. This is an indicator that time series practitioners were aware of the lack of representativeness in the benchmark, and proposed an expansion of the work to address these concerns and other criticisms of the repository (Hu et al., 2015). We believe that further investigation of the coverage of the time series properties provided by the benchmark is required, to reveal potentially unknown areas of the feature space (i.e., the problem landscape) that are over, or under-represented, i.e., to identify issues with non-redundancy and representativeness.

In this study, we introduce a general benchmark evaluation pipeline for landscape analysis of ML datasets and we showcase its applicability on the UCR time series classification (TSC) datasets. By generating a shared representation of the instances belonging to the different datasets within the benchmark, we generate a problem vector space which enables us to perform a complementary analysis of the benchmark, i.e. to determine the degree to which the instances from all datasets complement each other to produce a better coverage of the problem benchmark landscape. Sampling this problem space allows the generation of subsets of problem instances with reduced redundancy and increased representativeness, which we use to identify potential discrepancies in the evaluation of existing TSC algorithms and warn on the potentially harmful effects of tuning the methods on certain parts of the landscape. The contributions of the paper can be summarized in the following:

- Introduce a general pipeline for analysis of problem space coverage by benchmark datasets.
- Evaluate the extent to which the UCR benchmark satisfies the quality requirements of non-redundancy and representativeness.
- Provide directions for fairer evaluation of time series algorithms achieved by a careful selection of the benchmark dataset portfolio.
- To select an algorithm that performs well across the entire landscape of dataset distributions, we should uniformly select a set of benchmark datasets; sample the same number of datasets from all distributions, and then select the algorithm that performs the best.
- To select an algorithm that performs well for a specific application, first based on the shared representations of the new dataset we should select the cluster that consists of datasets with similar distributions and then select the best performing algorithm (and not look in all dataset distributions).

2 Related work

Instance space analysis. The most closely related methodology with the pipeline introduced herein, resides with the Instance Space Analysis (ISA) methodology (Smith-Miles et al., 2014). ISA is a visualization methodology for benchmark evaluation that employs analysis of the distribution of feature-vector representations of dataset instances to understand how instance characteristics affect algorithm performance, analyzing the diversity of benchmark instances, identifying areas of the instance space where certain algorithms perform better than others to support automated algorithm selection. It further extends to generating instances that maximize the performance difference between algorithms to highlight their strengths and weaknesses. ISA has been already tested for different ML learning tasks including classification (Muñoz et al., 2017), regression (Muñoz et al., 2021), and clustering (dos Santos Fernandes et al., 2021). Its application is also shown in different problems such as car sequencing (Sun et al., 2020), rotating workforce scheduling (Kletzander et al., 2020), and outlier detection (Kandanaarachchi et al., 2019). Several recent studies also addressed the coverage of the single- and multi-objective optimization problems and investigate their distribution over the problem space to create more unbiased benchmark sets (Škvorc et al., 2020; Yap et al., 2020; Lang & Engelbrecht, 2021). The main difference of the newly proposed pipeline with the

ISA is that it performs analysis only in the feature space (i.e., the problem landscape), focusing only on the distribution of the dataset over the problem space and not investigating the relations with the performance space. This means that the analysis is not affected by the ML algorithms involved in the analysis, and it helps understand the problem landscape no matter which ML algorithms it will be further tested on.

TSC studies. We did an overview of 8 recent papers on TSC published over the last year (2020-2021) (Anthony Bagnall & Keogh, 2021), where novel methods were introduced. We selected these papers since we want to check what are the current most commonly-used benchmarking practices. From the analysis conducted on these papers, we concluded that even though all of these papers were published shortly after the latest update of the UCR archive (in 2018), the authors benchmark the algorithms on the previous version, i.e., on the 85 datasets from the 2015 version of the archive, which are all fixed-length univariate time series. From the 8 papers that we reviewed, four of them (Cabello et al., 2020; Fawaz et al., 2020; Dempster et al., 2020; Shifaz et al., 2020) use the 2015 version of the UCR archive, one of which (Fawaz et al., 2020) uses an additional synthetic univariate TSC dataset, intending to control the length of the time series data as well as the number of classes and their distribution in time. Only one of these four studies (Dempster et al., 2020) performs an additional evaluation of their proposed methodology on the newly added 43 datasets to the UCR archive 2018 version. An important thing to note here is that the benchmark analyses on the UCR 2018 version archive are done on 112 out of the 128 datasets, removing any datasets that are unequal length or contain missing values, a summary of these 112 datasets is given in (Matthew Middlehurst, 2021) [1]. An additional change that is common (Middlehurst et al., 2021; 2020a;b) is removing the dataset Fungi as it only provides a single train case for each class. From the performed analysis, we concluded that the most commonly-used benchmarking approach is to involve all datasets that are available in some version of the UCR repository and then to perform statistical analysis of algorithms' performance achieved on them.

3 LANDSCAPE ANALYSIS PIPELINE

Figure 1 presents the proposed pipeline for landscape analysis of ML datasets. It consists of three parts: *i) defining a shared representation* - where data instances characteristics (i.e., features) are calculated (i.e., feature extraction) to define a shared representation across different datasets; *ii) complementary analysis* - where the data instances are clustered to find similar instances (either from the same or different datasets) that cover the same problem space. This analysis helps us understand how instances from different datasets complement each other to produce a good benchmark; and *iii) bootstrapping statistical evaluation* - where the landscape analysis results are used to select a portfolio of benchmark datasets to be involved in hypothesis testing of ML algorithms' performance.

To demonstrate how the proposed pipeline is executed, we consider a use case in univariate time-series analysis. The big boxes presented in the figure are the steps that are performed for an arbitrary ML problem. The details inside the boxes are specific for the ML problem, i.e time-series classification demonstrated herein. In the remainder of the section, all parts from the pipeline are discussed in more detail specifically for time-series data. In general, the applicability of the methodology is related to the existence of shared representation (i.e., meta-features) for the ML problem being solved, which is completely another goal and is not the focus of this paper. However, nowadays, with the huge progress done in representation learning, finding a representation for different ML learning tasks should not be a problem. The key idea behind the representation or finding a set of shared meta-features across different datasets is projecting them to the same embedded space.

3.1 DEFINING A SHARED REPRESENTATION / FEATURE EXTRACTION

The landscape analysis methodology requires a shared representation of the instances of the different datasets. The datasets originate from various application domains where different characteristics of the time series can be relevant. For example, the presence or absence of a certain shape in the time series (e.g., absence of the T-interval in supraventricular fibrillation) is relevant when distinguishing the specifics of the different classes in classifying ECG signals (Kaplan Berkaya et al., 2018). In

[1] https://sites.google.com/view/icdm-cif/home

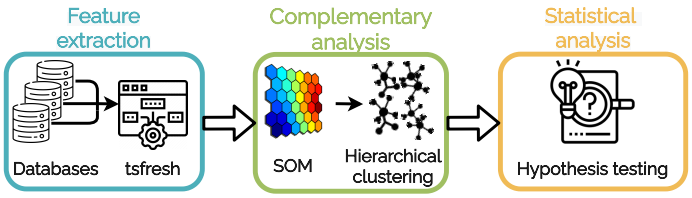

Figure 1: Landscape analysis pipeline for ML datasets. (The icons in the flowchart are taken from (S.L, 2021))

other datasets describing the problem of movement detection, for example, the features representing the burstiness of the signal (e.g. *increased temporal stationarity*) are more important.

Owning to the diverse problems as part of the UCR TSC benchmark, we considered a rich set of multi-domain time-series feature extractors collected in *tsfresh* (Christ et al., 2018). *Tsfresh* is a library for the extraction of global and local features from time series. The collected features originate from diverse scientific areas, yielding diverse complementary views over the individual time series instances. In general, we considered 8 groups of features, namely: statistical, information-theoretic, model-based, stationarity, fractal, frequency-based (Fourier and wavelet transform), symbolic and domain-specific features. In this work, we use 794 features for each dataset.

3.2 Complementary analysis

To assess the structural differences between the dataset instances we performed complementary analysis over the instances. The complementary analysis is performed in two steps: learn a self-organizing map (SOM) representation and cluster the learned SOM representations to find similar data instances.

Self-Organizing Maps representation. We used SOM as a pre-step for the clustering since the reduction of dimensionality and grid clustering is a natural fit when contrasting the quantifiable and topographical structures of the instance representation space (Yang et al., 2012). Furthermore, it fosters the interpretation of the similarities between the data instances (Kaski & Lagus, 1996). The quality of the generated SOMs is evaluated in terms of the quantization error and the topographic error. When training a SOM model, both measures are subject to minimization. Since the evaluation is treated as a multi-objective optimization (i.e., two objectives) problem, the evaluation result is a Pareto front, consisting of solutions that are pairs of quantization and topographic errors. To select the best SOM, we adopt a decision-making strategy based on maximizing the amount of explained variance as a quality measure.

Hierarchical clustering. The result of SOM is a map that consists of cells composed of the different time-series instances that fall within. Each of the cells is represented by its unique prototypical codebook. The number of SOM's cells can be large, leading to over-specification of the shared properties among the different instances. To foster the practical usage of the obtained mappings (e.g. when accessing the performance of the different algorithms), we further clustered the SOM's codebooks. The codebooks were clustered using agglomerative hierarchical clustering. The cluster that is obtained for each codebook is further assigned to all time-series data instances that belong to it. Therefore, a higher level of specification of the instances is achieved. Ultimately, this improves the interpretability and comprehension of the landscape analysis.

3.3 Statistical analysis

The output of the landscape analysis is a distribution of the datasets (i.e., clusters of datasets) across the problem landscape. We used the clusters to select a benchmark datasets portfolio that will be uniformly distributed over the whole problem landscape. Further, the selected portfolio will be used to perform a statistical analysis of algorithms' performance. Such kind of benchmark datasets portfolio selection is required to provide a fair and robust evaluation, not biased by over or under representativeness in the problem space that can be in favour only for some algorithms tuned on those parts of the problem landscape.

Coverage matrix. The distribution of the datasets across the problem landscape is given as a coverage matrix. In the coverage matrix, the rows correspond to the different datasets and the columns to the different clusters. As such, it provides information about the percentage of each dataset that belongs to each cluster. Since the same dataset can be distributed across different clusters (i.e., have some percentage of instances that belong to them), the datasets between clusters can overlap. To assess the degree to which the benchmark can provide fair and robust evaluation, we need to ensure uniform coverage over the problem landscape. To avoid redundancy of dataset selection across different clusters, the representative datasets for each cluster has to be identified. Therefore, we adopt a thresholding technique that specifies the percentage of dataset instances that should be preserved in the cluster, so the dataset to be considered as representative of the cluster itself.

Hypothesis testing. To test the outcome of the selected portfolio of benchmark datasets, 14 multi-class classification algorithms were compared using accuracy as a performance measure. We would like to point out that we are not training the multi-class classification time-series algorithms, but using the algorithms' performance results which are publicly available at (Ruiz et al., 2021). The comparison was performed using a bootstrapping approach that involved hypothesis testing of the algorithms' performance (i.e., comparing the algorithms several times using different selected portfolios of benchmark datasets). Comparing the algorithms several times on different samplings of the representatives from each cluster, we tested the consistency and robustness of the statistical outcome (i.e., whether there is statistical significance between the algorithms' performance). The bootstrapping approach allows us to explore if the results are robust if we repeat the hypothesis testing using different benchmark datasets which uniformly cover the space of all possible dataset distributions.

4 EVALUATION

In this section, we provide details about the data and the conducted experiments. We conducted the experiments using the UCR repository, containing 128 datasets (Dau et al., 2018).[2]

4.1 FEATURE EXTRACTION

To represent the instances into a shared representation space, we used 63 feature methods from the *tsfresh* library. We parameterized the feature methods with their defaults, a procedure resulting in 794 features for each of the 128 datasets. Due to the scarcity of the datasets with unequal lengths of the time series per class, as a potential artifact in the benchmark, 15 datasets were marginalized out in the further experiments. In total, the benchmark was presented with 113 datasets (the list of the datasets included in this study is available at our GitHub repository). Some of the features were not computable for specific datasets and were removed. To account for the numerical vector space, all the non-numeric features were removed. Finally, this resulted in a total of 324 features used in the analysis. We did not perform any additional feature selection, because the instances are coming from different time-series nature In addition, we tested the expressiveness of all features introduced for representing time-series instances.

4.2 COMPLEMENTARY ANALYSIS

SOM's representation. We used the *kohonen* R package (Wehrens et al., 2007; Wehrens & Kruisselbrink, 2018) to generate SOMs with a hexagonal topology and a radius in the range from 2.65 to -2.65. The learning rate was initially set to 0.05 and it declined to 0.01, over a set of 100 updates, and sum of squares was used as a distance function. These are the default values for the parameters in the 3.0.10 version of the *kohonen* library. To find the best SOM for the data, we used a grid search of square maps starting from 25×25 to 50×50. The range was chosen by apriori empirical experimentation, using Kohonen's report formula as a starting formula for estimating the SOM's grid. To select the best SOM, the evaluation result is a Pareto optimal front (i.e., an approximation set), with two non-dominated solutions (SOM configurations: 35×35 and 48×48). In the end, we chose to work with 48×48 SOM, since it has a higher percentage of explained variance (i.e., 89.49%).

[2]For reproducibility purposes, we open-source the code, data, and all results supplementing the paper at `https://anonymous.4open.science/r/stamp-C6F2/README.md`. The experiments were conducted on a system with CPU Intel i7 9750H and 16GB of memory running Ubuntu 18.04.

With the selected configurations, we ended up with 2,304 cells represented with their codebooks. The results for 35×35 SOM has a percentage of explained variance of 85.65%.

Hierarchical clustering. The SOM's cells represented with their codebooks were further clustered together with agglomerate hierarchical clustering (Python package *scikit-learn* version 0.24.2 (Pedregosa et al., 2011)) to form fewer clusters. SOM clusters were merged based on average linkage and the cosine distance between the codebook vectors. Cosine distance is often used when clustering high-dimensional data due to being dependant only on the direction of vectors and not their length. Compared to other commonly used distance measures (Euclidean, Manhattan, etc.), cosine distance is not as sensitive to outliers (Shirkhorshidi et al., 2015) and was therefore selected as the most appropriate distance measure. Due to the presence of outliers, using outlier-sensitive metrics like euclidean distance produced one cluster with a majority of elements and few small clusters. When performing hierarchical clustering decision has to be made on the number of clusters we want as a result. Based on the maximization of the cosine distance silhouette scores (Rousseeuw, 1987), a metric that determines the clustering quality based on mean intra-cluster distance and the mean nearest-cluster distance, one local optimal number of clusters was determined to be 6. Using a larger number of clusters can produce clusters that have higher silhouette scores, where datasets are fragmented/distributed over multiple clusters. However, to determine the number of clusters, we also took into account the cluster purity and minimized the number instances from the same dataset belonging to different clusters.

4.3 STATISTICAL EVALUATION

Coverage matrix. Figure 2 depicts the distribution of the datasets within the clusters, i.e. the percentage of dataset instances that belong to each cluster. As can be seen from Figure 2, 46 of the datasets belong only to one cluster. Using only one of these datasets will not cover the entire problem landscape, and may lead to biased evaluation. We can also observe that one of the clusters contains instances from a single dataset, which are 0.02% of the instances from the *UWaveGestureLibraryZ* dataset. This indicates that there are parts of the landscape that are not thoroughly covered by the benchmark, and introduces a direction for further extension with new instances. Clusters 1-6 contain instances from 65, 53, 7, 68, 55, and 1 unique dataset, respectively. On average, each cluster contains instances from 41.5 different datasets, meaning that even though the benchmark contains a large number of datasets, the instances in these datasets cover similar areas of the problem instance space, i.e. the benchmark does not satisfy the non-redundancy quality requirement.

We can also observe that the instances from some datasets of similar nature are distributed in similar clusters. For instance, all instances from datasets *SemgHandGenderCh2*, *SemgHandMovementCh2*, *SemgHandSubjectCh2* are assigned to cluster five, all instances from datasets *ProximalPhalanxOutlineAgeGroup*, *ProximalPhalanxOutlineCorrect*, *ProximalPhalanxTW* belong to cluster four, all instances from datasets *NonInvasiveFetalECGThorax1* and *NonInvasiveFetalECGThorax2* are in cluster one, etc. Even in the cases when not all instances from one dataset are distributed in a single cluster, the instances from different datasets that are of the similar nature, are distributed in the same clusters. Such examples are the datasets *CricketX*, *CricketY*, *CricketX*, and the datasets *FreezerRegularTrain* and *FreezerSmallTrain*. This suggests that the SOMs and the subsequent clustering can capture related datasets and map them close in the vector space.

Hypothesis testing results. To select the representative datasets for each cluster that can further be involved in the comparison, we set a threshold of 90% (see the coverage matrix, Figure 2). It means that a dataset can be representative of a cluster, if the cluster contains at least 90% of the dataset instances. The threshold was selected for illustration purposes. Using this criterion, we selected 23, 9, 5, 25, 16 datasets for the first, second, third, fourth, and fifth cluster, respectively. The sixth cluster was omitted from the sampling, since it did not have a representative dataset, and it contained only 0.02% of the instances in the single dataset present. Different thresholds (i.e., 50%, 80%, and 100%) were also tested and the results are available at our GitHub repository. We excluded five datasets from the analysis, since not all algorithms had been tested on them: *Fungi*, *NonInvasiveFetalECGThorax1*, *HandOutlines*, *NonInvasiveFetalECGThorax2*, and *FordB*.

Each comparison was performed using the Friedman test (R package *scmamp* version 0.2.55 (Calvo & Santafe, 2015)) and the Nemenyi posthoc test (R package *PMCMR* version 4.3 (Pohlert, 2014)) was utilized to find if there is statistical significance between the performance of all pairwise com-

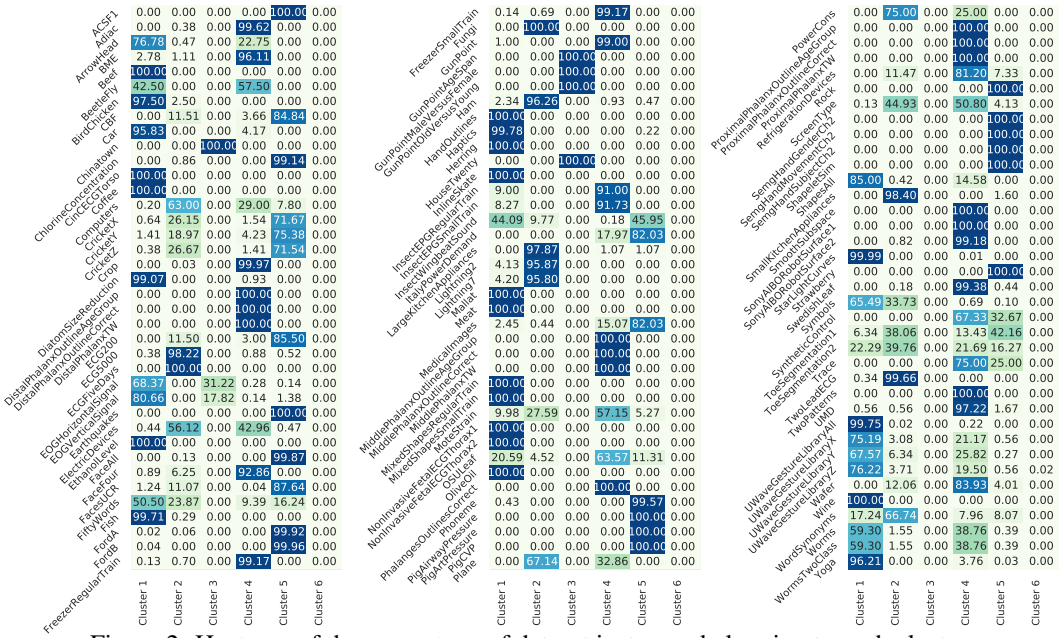

Figure 2: Heatmap of the percentage of dataset instances belonging to each cluster.

parisons (i.e., multiple pairwise comparison p-values corrections). The significance level was set to 0.05. The Nemenyi test provides a p-value for each pair of algorithms tested on the selected datasets. If the p-values are greater or equal than the significance level, the null hypothesis is not rejected (i.e., we translate this to 1), otherwise there is a statistical significance between the performance of the compared pair of algorithms (i.e., we translate this to 0).

Table 1 presents the statistical outcome of the pair-wise comparison of algorithms when all datasets are involved in the comparison (i.e., in our case 108, after removing 5 datasets for which some of the algorithms are not tested). This is the most commonly used benchmarking practice across recently published papers. Looking at the results, it seems that there are a lot of pairs of algorithms where statistical significance is found. However, we need to be careful when involving all datasets, since the datasets are not uniformly distributed over the landscape that is also visible from the coverage matrix. This means that including more datasets from the same part of the landscape is in favour of some of the algorithms and the statistical outcome is questionable.

Table 1: Statistical results obtained for each pair of algorithms where all 108 datasets are used in the comparison. Each cell presents 1 or 0, where 1 indicates that there is no statistical significance between the performance of the pair of algorithms and 0 otherwise.

	BOSS	Catch22	HIVE-COTEv1_0	InceptionTime	ProximityForest	RISE	ROCKET	ResNet	S-BOSS	STC	TS-CHIEF	TSF	WEASEL
Catch22	0.00												
HIVE-COTEv1_0	0.00	0.00											
InceptionTime	0.00	0.00	1.00										
ProximityForest	1.00	0.00	0.00	0.00									
RISE	1.00	1.00	0.00	0.00	0.00								
ROCKET	0.00	0.00	1.00	1.00	0.00	0.00							
ResNet	1.00	0.00	0.00	0.00	1.00	0.00	0.00						
S-BOSS	1.00	0.00	0.00	0.00	1.00	0.00	0.00	1.00					
STC	1.00	0.00	0.00	1.00	1.00	0.00	0.00	1.00	1.00				
TS-CHIEF	0.00	0.00	1.00	1.00	0.00	0.00	1.00	0.00	0.00	0.00			
TSF	1.00	1.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00		
WEASEL	1.00	0.00	0.00	0.00	1.00	0.00	0.00	1.00	1.00	1.00	0.00	0.00	
cBOSS	1.00	0.00	0.00	0.00	1.00	1.00	0.00	1.00	1.00	1.00	0.00	1.00	1.00

Since different algorithms may perform differently to different datasets distribution (i.e., clusters of datasets), we performed the hypothesis testing comparing the algorithms on the representative datasets from each cluster separately. For this purpose, we did this analysis for the first, fourth, and fifth clusters when the threshold for selecting the representatives was set to 90%. The second, third, and sixth clusters were omitted since the number of representative datasets is lower than 10 and not enough to talk about statistical significance (i.e., see details about the Friedman test (Eftimov & Korošec, 2020)). The obtained results are presented in Appendix A.1 (see Table 3). Using the results presented in the table, we can see that there exist algorithms that have different performances using different clusters of datasets. For example, using the representatives from the first cluster

there is no statistical significance between the algorithms *HIVE-COTEv1_0* and *S-BOSS*. The same results are also true when using the representative datasets from the fifth cluster, while there is a statistical significance between them if the representatives from the fourth datasets are involved in the comparison. Such results indirectly indicate that the statistical results depend on the number of datasets that will be included from each cluster. If the number of datasets is not uniformly distributed across all clusters, then the statistical results will be biased to the cluster that is represented with the highest number of datasets (i.e., the selected test statistic is affected).

To perform the bootstrapping hypothesis testing for comparing the performance of 14 multi-class classification algorithms, we selected datasets that are representatives of each cluster. The comparison process was done three times concerning the number of representative datasets sampled from each cluster. The sample sizes used are 2, 3, and 4, which means that the algorithms were compared using 10, 15, and 20 datasets, respectively. In Appendix A.2 we provide empirical guidelines on how many datasets should be sampled per cluster that depends on the coverage matrix distribution. For each sample size, the comparison was performed 30 times (30 times selecting 10, 15, and 20 datasets), in order to check the robustness of the statistical results. All selected portfolios of benchmark datasets involved in the comparisons are available at our GitHub repository. By repeating this 30 times for each sample size separately, and by performing a summation for each pair of algorithms, we tested the robustness of the statistical results. The counting approach is only an indicator if the same statistical results (i.e., multi-hypothesis correction scenario) are robust if we repeat it using different benchmark datasets that will uniformly cover the space of all possible dataset distributions. More information about the statistical comparison design can be found in Appendix A.3.

Tables 2a and 2b present the bootstrapping statistical results obtained for each pair of algorithms where the sample size of the representative datasets was set to 2 and 3, respectively. The results obtained when the samples size is set to 4 are presented in Appendix A.4. Each cell presents the number of comparisons where there is no statistical significance between the performance of the pair of algorithms, with the maximal number of such comparisons being 30. We can conclude that if the number of such comparisons is greater than 15 then there is no statistical significance between the performances of the algorithms and vice-versa. Looking across the tables, we can conclude that the statistical outcome between the pairs of algorithms is almost consistent since they provide robust statistical results. For example, regardless of the sample size of the representative datasets from each cluster (i.e., 2, 3, or 4), there is no statistical significance of the performances between the (*Catch22* and *BOSS*, all sample sizes return 30 out of 30). There are also a small number of inconsistencies when the sample size of the representative datasets changes (i.e., only in 3 out of 88 pairs). For example, there is no statistical significance between the performance of the algorithms (*TSF* and *TS-CHIEF*) when we sample 2 representatives from each cluster (20 out of 30), however, there is a statistical significance between their performances when we sample 3 or 4 representatives from each cluster (10 out of 30 and 2 out of 30, respectively). There is no statistical significance between the performances of the (*ROCKET* and *TSF*) when the sample sizes are 2 or 3 (25 out of 30 and 16 out of 30, respectively), however, there is a statistical significance when the sample size is set to 4 (7 out of 30 comparisons). Same results are also obtained for the pair (*RISE* and *TS-CHIEF*). This indicates that some of the algorithms can also have different performances in the same part of the landscape space, which further opens directions to analyze their behaviour more extensively, focusing on instance analysis and different data instance transformations that are presented there. The bootstrapping approach provides robust and reproducible statistical outcomes, which is not the case when different datasets are involved in the most commonly-used practice. This means that no matter how many datasets are included in the comparison (i.e., in our case 10, 15, or 20), the statistical results will be the same if they uniformly covered the problem landscape.

By performing a sensitivity analysis of selecting different thresholds for choosing the representatives (i.e., 50%, 80%, 90%, and 100%), similar statistical outcomes are obtained. Having a lower threshold (in our case 50%) indicates the presence of more statistical inconsistencies between pairs of algorithms concerning different sample sizes for sampling. In such a case, this is expected since a dataset representative covers greater or equal than half of the dataset and the other part of it could be distributed across the other clusters. The remaining part of the dataset landscape distributed in other cluster/s can also be in favour of some algorithms related to that/those landscape parts. Having a greater threshold value also means more homogeneous datasets being selected as representatives, where most of their instances (i.e., the landscape they covered) belong to the same cluster.

In an ideal scenario, we should select one dataset that is uniformly distributed across all clusters. However, such a dataset that is uniformly distributed across the problem landscape does not exist. Selecting the threshold for the representative datasets is related to the number of clusters. To select it, the distribution of the percentage of coverage from the coverage matrix should be investigated. Higher threshold values are more welcome since they guarantee the purity of the representatives. One possible way of determining the number of clusters is also taking into account cluster purity i.e. minimizing instances from the same dataset belonging to different clusters.

Table 2: Bootstrapping statistical results obtained for each pair of algorithms where the sample size of the representative datasets is selected with threshold of 90%. Each cell presents the number of comparisons (out of 30) where there is no statistical significance observed between the pairs.

	BOSS	Catch22	HIVE-COTEv1_0	InceptionTime	ProximityForest	RISE	ROCKET	ResNet	S-BOSS	STC	TS-CHIEF	TSF	WEASEL
Catch22	30.00												
HIVE-COTEv1_0	30.00	2.00											
InceptionTime	30.00	13.00	30.00										
ProximityForest	30.00	28.00	28.00	30.00									
RISE	30.00	30.00	23.00	29.00	30.00								
ROCKET	30.00	7.00	30.00	30.00	29.00	26.00							
ResNet	30.00	24.00	30.00	30.00	30.00	30.00	30.00						
S-BOSS	30.00	30.00	30.00	30.00	30.00	30.00	30.00	30.00					
STC	30.00	30.00	30.00	30.00	30.00	30.00	30.00	30.00	30.00				
TS-CHIEF	30.00	6.00	30.00	30.00	29.00	25.00	30.00	30.00	30.00	30.00			
TSF	30.00	30.00	16.00	23.00	30.00	30.00	25.00	30.00	30.00	30.00	20.00		
WEASEL	30.00	29.00	29.00	30.00	30.00	30.00	29.00	30.00	30.00	30.00	30.00	30.00	
cBOSS	30.00	29.00	30.00	30.00	30.00	30.00	30.00	30.00	30.00	30.00	30.00	30.00	30.00

(a) Sample size 2

	BOSS	Catch22	HIVE-COTEv1_0	InceptionTime	ProximityForest	RISE	ROCKET	ResNet	S-BOSS	STC	TS-CHIEF	TSF	WEASEL
Catch22	30.00												
HIVE-COTEv1_0	20.00	0.00											
InceptionTime	30.00	14.00	30.00										
ProximityForest	30.00	29.00	24.00	30.00									
RISE	30.00	30.00	7.00	30.00	30.00								
ROCKET	27.00	3.00	30.00	30.00	28.00	18.00							
ResNet	30.00	26.00	28.00	30.00	30.00	30.00	29.00						
S-BOSS	30.00	30.00	27.00	30.00	30.00	30.00	29.00	30.00					
STC	30.00	26.00	30.00	30.00	30.00	30.00	30.00	30.00	30.00				
TS-CHIEF	25.00	0.00	30.00	30.00	28.00	12.00	30.00	27.00	28.00	28.00			
TSF	30.00	30.00	6.00	26.00	30.00	30.00	16.00	30.00	30.00	30.00	10.00		
WEASEL	30.00	30.00	25.00	30.00	30.00	30.00	29.00	30.00	30.00	30.00	23.00	30.00	
cBOSS	30.00	29.00	26.00	30.00	30.00	30.00	29.00	30.00	30.00	30.00	26.00	30.00	30.00

(b) Sample size 3

5 CONCLUSIONS

In this paper, we have proposed a new pipeline for landscape analysis of ML datasets that helps us understand the problem landscape, allows us to select a diverse benchmark datasets portfolio, and reduce the performance assessment bias using a bootstrapping evaluation. The results conducted using time-series data for multi-class classification have shown that more robust and reproducible benchmarking statistical results are obtained compared to the results obtained by the most commonly used benchmarking practices. In addition, the newly proposed pipeline does not provide an opportunity to manually selecting datasets that leads to the desired outcome of the study. In addition, a sensitivity analysis of the hyperparameters and techniques used in each step of the pipeline will be investigated in more detail. Therefore, different shared time-series representation such as *T-Loss* (Franceschi et al., 2019), *DTCR* (Ma et al., 2019), and *TNC* (Tonekaboni et al., 2020) will be tested instead of the *tsfresh* representation to check if they can lead to reproducible statistical outcomes. Further, different dimensionality reduction techniques such as PCA (Wold et al., 1987) will be also tested instead of SOM, since in many ML tasks there are small number of datasets available for benchmarking. Finally, such selection of a benchmark datasets portfolio will be also investigated to link the problem landscape to the performance achieved by the ML algorithms more in the direction of meta-learning studies (Vanschoren, 2018).

With this study, we do not discourage the use of any existing TSC datasets, all of them should be included in the existing repository. Practitioners can still use all of the datasets in order to develop new algorithms or solve a specific application scenario. The study showed that we should take great care when deciding which datasets will be involved in the statistical analysis of a newly introduced method. Even more, for a newly introduced TSC dataset in the future, the proposed pipeline can be used as a criterion to decide if the dataset should be included in a benchmark datasets portfolio or not. This can be done by observing the problem landscape that is covered by it. If the same problem landscape is also covered by other dataset/s, there is no reason to include the dataset in the benchmark datasets portfolio, and vice-versa.

REFERENCES

William Vickers Anthony Bagnall, Jason Lines and Eamonn Keogh. The uea & ucr time series classification repository (selected recent tsc papers), 2021. URL www.timeseriesclassification.com.

A. Bagnall, Jason Lines, Aaron Bostrom, J. Large, and Eamonn J. Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, 31:606 – 660, 2016.

Thomas Bartz-Beielstein, Carola Doerr, Jakob Bossek, Sowmya Chandrasekaran, Tome Eftimov, Andreas Fischbach, Pascal Kerschke, Manuel López-Ibáñez, Katherine Mary Malan, Jason H. Moore, Boris Naujoks, Patryk Orzechowski, Vanessa Volz, Markus Wagner, and Thomas Weise. Benchmarking in optimization: Best practice and open issues. *CoRR*, abs/2007.03488, 2020. URL https://arxiv.org/abs/2007.03488.

Nestor Cabello, Elham Naghizade, Jianzhong Qi, and Lars Kulik. Fast and accurate time series classification through supervised interval search. In *2020 IEEE International Conference on Data Mining (ICDM)*, pp. 948–953. IEEE, 2020.

Borja Calvo and Guzman Santafe. scmamp: Statistical comparison of multiple algorithms in multiple problems. *The R Journal*, Accepted for publication, 2015.

Gavin C. Cawley and Nicola L. C. Talbot. On over-fitting in model selection and subsequent selection bias in performance evaluation. *Journal of Machine Learning Research*, 11(70):2079–2107, 2010. URL http://jmlr.org/papers/v11/cawley10a.html.

Maximilian Christ, Nils Braun, Julius Neuffer, and Andreas W. Kempa-Liehr. Time series feature extraction on basis of scalable hypothesis tests (tsfresh – a python package). *Neurocomputing*, 307:72–77, 2018.

Hoang Anh Dau, Anthony J. Bagnall, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, and Eamonn J. Keogh. The UCR time series archive. *CoRR*, abs/1810.07758, 2018. URL http://arxiv.org/abs/1810.07758.

Angus Dempster, François Petitjean, and Geoffrey I Webb. Rocket: exceptionally fast and accurate time series classification using random convolutional kernels. *Data Mining and Knowledge Discovery*, 34(5):1454–1495, 2020.

Luiz Henrique dos Santos Fernandes, Ana Carolina Lorena, and Kate Smith-Miles. Towards understanding clustering problems and algorithms: An instance space analysis. *Algorithms*, 14(3):95, March 2021. doi: 10.3390/a14030095. URL https://doi.org/10.3390/a14030095.

Tome Eftimov and Peter Korošec. Statistical analyses for meta-heuristic stochastic optimization algorithms: Gecco 2020 tutorial. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, pp. 724–746, 2020.

Hassan Ismail Fawaz, Benjamin Lucas, Germain Forestier, Charlotte Pelletier, Daniel F Schmidt, Jonathan Weber, Geoffrey I Webb, Lhassane Idoumghar, Pierre-Alain Muller, and François Petitjean. Inceptiontime: Finding alexnet for time series classification. *Data Mining and Knowledge Discovery*, 34(6):1936–1962, 2020.

Jean-Yves Franceschi, Aymeric Dieuleveut, and Martin Jaggi. Unsupervised scalable representation learning for multivariate time series. In *Thirty-third Conference on Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Bing Hu, Yanping Chen, and Eamonn Keogh. Classification of streaming time series under more realistic assumptions. *Data Mining and Knowledge Discovery*, 30(2):403–437, June 2015. doi: 10.1007/s10618-015-0415-0. URL https://doi.org/10.1007/s10618-015-0415-0.

Sevvandi Kandanaarachchi, Mario A. Munoz, and Kate Smith-Miles. Instance space analysis for unsupervised outlier detection. In *Proceedings of the 1st Workshop on Evaluation and Experimental Design in Data Mining and Machine Learning co-located with SIAM International Conference on Data Mining (SDM 2019), Calgary, Alberta, Canada, May 4th, 2019.*, pp. 32–41, 2019. URL http://ceur-ws.org/Vol-2436/article_4.pdf.

Selcan Kaplan Berkaya, Alper Kursat Uysal, Efnan Sora Gunal, Semih Ergin, Serkan Gunal, and M. Bilginer Gulmezoglu. A survey on ecg analysis. *Biomedical Signal Processing and Control*, 43:216–235, 2018.

Samuel Kaski and Krista Lagus. Comparing self-organizing maps. In *Artificial Neural Networks — ICANN 96*, pp. 809–814. Springer Berlin Heidelberg, 1996. doi: 10.1007/3-540-61510-5_136. URL https://doi.org/10.1007/3-540-61510-5_136.

Eamonn Keogh and Shruti Kasetty. *Data Mining and Knowledge Discovery*, 7(4):349–371, 2003. doi: 10.1023/a:1024988512476. URL https://doi.org/10.1023/a:1024988512476.

Lucas Kletzander, Nysret Musliu, and Kate Smith-Miles. Instance space analysis for a personnel scheduling problem. *Annals of Mathematics and Artificial Intelligence*, 89(7):617–637, April 2020. doi: 10.1007/s10472-020-09695-2. URL https://doi.org/10.1007/s10472-020-09695-2.

Ryan Dieter Lang and Andries Petrus Engelbrecht. An exploratory landscape analysis-based benchmark suite. *Algorithms*, 14(3):78, 2021.

Qianli Ma, Jiawei Zheng, Sen Li, and Gary W Cottrell. Learning representations for time series clustering. *Advances in neural information processing systems*, 32:3781–3791, 2019.

Anthony Bagnall Matthew Middlehurst, James Large. Supporting information and data for the paper "the canonical interval forest (cif) classifier for time series classification", 2021. URL https://sites.google.com/view/icdm-cif.

Matthew Middlehurst, James Large, and Anthony Bagnall. The canonical interval forest (cif) classifier for time series classification. In *2020 IEEE International Conference on Big Data (Big Data)*, pp. 188–195. IEEE, 2020a.

Matthew Middlehurst, James Large, Gavin Cawley, and Anthony Bagnall. The temporal dictionary ensemble (tde) classifier for time series classification. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pp. 660–676. Springer, 2020b.

Matthew Middlehurst, James Large, Michael Flynn, Jason Lines, Aaron Bostrom, and Anthony Bagnall. Hive-cote 2.0: a new meta ensemble for time series classification. *arXiv preprint arXiv:2104.07551*, 2021.

Mario A. Muñoz, Laura Villanova, Davaatseren Baatar, and Kate Smith-Miles. Instance spaces for machine learning classification. *Machine Learning*, 107(1):109–147, December 2017. doi: 10.1007/s10994-017-5629-5. URL https://doi.org/10.1007/s10994-017-5629-5.

Mario Andrés Muñoz, Tao Yan, Matheus R. Leal, Kate Smith-Miles, Ana Carolina Lorena, Gisele L. Pappa, and Rômulo Madureira Rodrigues. An instance space analysis of regression problems. *ACM Transactions on Knowledge Discovery from Data*, 15(2):1–25, April 2021. doi: 10.1145/3436893. URL https://doi.org/10.1145/3436893.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Thorsten Pohlert. *The Pairwise Multiple Comparison of Mean Ranks Package (PMCMR)*, 2014. URL https://CRAN.R-project.org/package=PMCMR. R package.

Peter J Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of computational and applied mathematics*, 20:53–65, 1987.

Alejandro Pasos Ruiz, Michael Flynn, James Large, Matthew Middlehurst, and Anthony Bagnall. The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, 35(2):401–449, 2021.

Anasua Sarkar, Yang Yang, and Mauno Vihinen. Variation benchmark datasets: update, criteria, quality and applications. *Database*, 2020, 02 2020. ISSN 1758-0463. doi: 10.1093/database/baz117. URL https://doi.org/10.1093/database/baz117. baz117.

Gerard C. P. Schaafsma and Mauno Vihinen. Representativeness of variation benchmark datasets. *BMC Bioinformatics*, 19(1), November 2018. doi: 10.1186/s12859-018-2478-6. URL https://doi.org/10.1186/s12859-018-2478-6.

Ahmed Shifaz, Charlotte Pelletier, François Petitjean, and Geoffrey I Webb. Ts-chief: a scalable and accurate forest algorithm for time series classification. *Data Mining and Knowledge Discovery*, 34(3):742–775, 2020.

Ali Seyed Shirkhorshidi, Saeed Aghabozorgi, and Teh Ying Wah. A comparison study on similarity and dissimilarity measures in clustering continuous data. *PloS one*, 10(12):e0144059, 2015.

Urban Škvorc, Tome Eftimov, and Peter Korošec. Understanding the problem space in single-objective numerical optimization using exploratory landscape analysis. *Applied Soft Computing*, 90:106138, May 2020. doi: 10.1016/j.asoc.2020.106138. URL https://doi.org/10.1016/j.asoc.2020.106138.

Freepik Company S.L. Free vector icons and stickers - png, svg, eps, psd, css, 2021. URL https://www.flaticon.com/.

Kate Smith-miles, Davaatseren Baatar, Brendan Wreford, and Rhyd Lewis. Towards objective measures of algorithm performance across instance space. *Computers & Operations Research*, pp. 12–24, 2014.

Kate Smith-Miles, Davaatseren Baatar, Brendan Wreford, and Rhyd Lewis. Towards objective measures of algorithm performance across instance space. *Computers & Operations Research*, 45: 12–24, 2014.

Yuan Sun, Samuel Esler, Dhananjay Thiruvady, Andreas T. Ernst, Xiaodong Li, and Kerri Morgan. Instance space analysis for the car sequencing problem, 2020.

Sana Tonekaboni, Danny Eytan, and Anna Goldenberg. Unsupervised representation learning for time series with temporal neighborhood coding. In *International Conference on Learning Representations*, 2020.

Joaquin Vanschoren. Meta-learning: A survey. *arXiv preprint arXiv:1810.03548*, 2018.

Ron Wehrens and Johannes Kruisselbrink. Flexible self-organizing maps in kohonen 3.0. *Journal of Statistical Software*, 87(1):1–18, 2018.

Ron Wehrens, Lutgarde MC Buydens, et al. Self-and super-organizing maps in r: the kohonen package. *Journal of Statistical Software*, 21(5):1–19, 2007.

Svante Wold, Kim Esbensen, and Paul Geladi. Principal component analysis. *Chemometrics and intelligent laboratory systems*, 2(1-3):37–52, 1987.

Le Yang, Zhongbin Ouyang, and Yong Shi. A modified clustering method based on self-organizing maps and its applications. *Procedia Computer Science*, 9:1371–1379, 2012. doi: 10.1016/j.procs.2012.04.151. URL https://doi.org/10.1016/j.procs.2012.04.151.

Estefania Yap, Mario A. Munoz, Kate Smith-Miles, and Arnaud Liefooghe. Instance space analysis of combinatorial multi-objective optimization problems. In *2020 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, July 2020. doi: 10.1109/cec48606.2020.9185664. URL https://doi.org/10.1109/cec48606.2020.9185664.

Table 3: Statistical results obtained for each pair of algorithms when all representative datasets from each cluster are involved in the comparison separately (selected with threshold 90%). Each cell presents 1 or 0, where 1 indicates that there is no statistical significance between the performance of the pair of algorithms and 0 otherwise.

	BOSS	Catch22	HIVE-COTEv1_0	InceptionTime	ProximityForest	RISE	ROCKET	ResNet	S-BOSS	STC	TS-CHIEF	TSF	WEASEL
Catch22	1.00												
HIVE-COTEv1_0	0.00	0.00											
InceptionTime	1.00	0.00	1.00										
ProximityForest	1.00	0.00	1.00	1.00									
RISE	1.00	1.00	0.00	1.00	1.00								
ROCKET	0.00	0.00	1.00	1.00	1.00	0.00							
ResNet	1.00	0.00	1.00	1.00	1.00	1.00	1.00						
S-BOSS	1.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00					
STC	1.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00				
TS-CHIEF	0.00	0.00	1.00	1.00	1.00	0.00	1.00	1.00	1.00	1.00			
TSF	1.00	1.00	0.00	1.00	1.00	1.00	0.00	1.00	1.00	1.00	0.00		
WEASEL	1.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	
cBOSS	1.00	1.00	0.00	1.00	1.00	1.00	0.00	1.00	1.00	1.00	0.00	1.00	1.00

(a) First cluster.

	BOSS	Catch22	HIVE-COTEv1_0	InceptionTime	ProximityForest	RISE	ROCKET	ResNet	S-BOSS	STC	TS-CHIEF	TSF	WEASEL
Catch22	1.00												
HIVE-COTEv1_0	0.00	0.00											
InceptionTime	1.00	0.00	1.00										
ProximityForest	1.00	1.00	1.00	1.00									
RISE	1.00	1.00	0.00	1.00	1.00								
ROCKET	0.00	0.00	1.00	1.00	1.00	0.00							
ResNet	1.00	1.00	1.00	1.00	1.00	1.00	1.00						
S-BOSS	1.00	1.00	0.00	1.00	1.00	1.00	0.00	1.00					
STC	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00				
TS-CHIEF	0.00	0.00	1.00	1.00	1.00	0.00	1.00	1.00	0.00	1.00			
TSF	1.00	1.00	0.00	1.00	1.00	1.00	0.00	1.00	1.00	1.00	0.00		
WEASEL	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	
cBOSS	1.00	1.00	0.00	1.00	1.00	1.00	0.00	1.00	1.00	1.00	0.00	1.00	1.00

(b) Fourth cluster.

	BOSS	Catch22	HIVE-COTEv1_0	InceptionTime	ProximityForest	RISE	ROCKET	ResNet	S-BOSS	STC	TS-CHIEF	TSF	WEASEL
Catch22	1.00												
HIVE-COTEv1_0	1.00	0.00											
InceptionTime	1.00	1.00	1.00										
ProximityForest	1.00	1.00	0.00	1.00									
RISE	1.00	1.00	0.00	1.00	1.00								
ROCKET	1.00	1.00	1.00	1.00	1.00	1.00							
ResNet	1.00	1.00	0.00	1.00	1.00	1.00	1.00						
S-BOSS	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00					
STC	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00				
TS-CHIEF	1.00	0.00	1.00	1.00	1.00	1.00	1.00	0.00	1.00	1.00			
TSF	1.00	1.00	0.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.00		
WEASEL	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	
cBOSS	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00

(c) Fifth cluster.

A APPENDIX

A.1 STATISTICAL RESULTS OBTAINED USING THE DATASETS FROM EACH CLUSTER SEPARATELY.

A.2 EMPIRICAL GUIDELINES ON HOW MANY DATASETS SHOULD BE SAMPLED PER CLUSTER

In our experiments, the datasets have been distributed across 6 clusters, or actually in 5 clusters, where only a small percentage of one dataset belongs to the 6th cluster. We started sampling with 2 datasets from each cluster and ended up with 10 datasets. This was done since the required conditions for the safe use of the parametric test are not satisfied and we have paired samples, so we should continue with the non-parametric Friedman test (Eftimov & Korošec, 2020). To use the Friedman test, we should have at least 10 datasets in order to meet the condition that the Friedman statistic will follow the Chi-square distribution. Further, we continue to select 3 and 4 datasets per cluster separately, ending up with 15 and 20 datasets. The results showed that the statistical results obtained with 10, 15, or 20 datasets are robust (i.e., almost the same). The minimum number of datasets that should be included depends on the omnibus statistical test that will be utilized (i.e., in most cases the Friedman test is an appropriate one here), so we need to have at least 10 datasets that will be uniformly distributed across all clusters. The maximum number of datasets that can be selected should be the number of representatives from the smallest cluster.

A.3 HYPOTHESIS TESTING DESIGN

Each bootstrap experiment is an independent event that involves a multi-hypothesis correction method. Within each one, the set of 14 multi-class classification time-series algorithms are compared using a set of 10, 15, or 20 benchmark datasets. The comparison involves a paired samples scenario and since the required conditions of the safe use of the parametric tests are not satisfied, the appropriate omnibus statistical test is the non-parametric Friedman. To see between which pairs of algorithms the statistical significance exists we further utilized the Nemenyi test that is developed for all vs. all pairwise comparisons. Nemenyi test involves a multi-hypothesis correction method, where the p-values are corrected using the Bonferroni correction. So each comparison is a separate event of comparing the algorithms using a set of benchmark datasets. To test if the same statistical outcome will be reproduced if we follow the selection approach presented in this paper, we repeated each comparison 30 times using different sets of benchmark datasets. The common thing we are taking care of is that the selection of the datasets is always uniformly across all datasets distributions. The counting approach is only an indicator if the same statistical results (multi-hypothesis correction scenario) are robust if we repeat it using different benchmark datasets that will uniformly cover the space of all possible dataset distributions.

A.4 BOOTSTRAPPING STATISTICAL RESULTS WHEN THE NUMBER OF SAMPLED REPRESENTATIVES PER CLUSTER IS SET TO 4

Table 4: Bootstrapping statistical results obtained for each pair of algorithms where the sample size of the representative datasets (selected with threshold 90%). Each cell presents the number of comparisons out of 30, where there is no statistical significance between the performance of the pair of algorithms.

	BOSS	Catch22	HIVE-COTEv1_0	InceptionTime	ProximityForest	RISE	ROCKET	ResNet	S-BOSS	STC	TS-CHIEF	TSF	WEASEL
Catch22	30.00												
HIVE-COTEv1_0	16.00	0.00											
InceptionTime	30.00	4.00	30.00										
ProximityForest	30.00	23.00	22.00	30.00									
RISE	30.00	30.00	2.00	25.00	30.00								
ROCKET	22.00	0.00	30.00	30.00	27.00	4.00							
ResNet	30.00	20.00	25.00	30.00	30.00	30.00	30.00						
S-BOSS	30.00	29.00	17.00	30.00	30.00	30.00	26.00	30.00					
STC	30.00	19.00	28.00	30.00	30.00	29.00	29.00	30.00	30.00				
TS-CHIEF	20.00	0.00	30.00	30.00	27.00	3.00	30.00	29.00	25.00	28.00			
TSF	30.00	30.00	2.00	23.00	30.00	30.00	7.00	29.00	30.00	30.00	2.00		
WEASEL	30.00	26.00	17.00	30.00	30.00	30.00	27.00	30.00	30.00	30.00	22.00	30.00	
cBOSS	30.00	28.00	16.00	29.00	30.00	30.00	22.00	30.00	30.00	30.00	23.00	30.00	30.00

(a) Sample size 4