AutoClust: A Framework for Automated Clustering based on Cluster Validity Indices

Yannis Poulakis
Dept. of Digital Systems
University of Piraeus
Piraeus, Greece
gpoul@unipi.gr

Christos Doulkeridis
Dept. of Digital Systems
University of Piraeus
Piraeus, Greece
cdoulk@unipi.gr

Dimosthenis Kyriazis
Dept. of Digital Systems
University of Piraeus
Piraeus, Greece
dimos@unipi.gr

***Abstract*—Automated machine learning (AutoML) aims to minimize human intervention during a machine learning task, for example by means of automatic algorithm selection and its configuration for the data set at hand. Although this research direction has attracted much interest lately, both in academia and industry, existing systems and tools mainly target the domain of supervised learning. However, unsupervised learning, in particular clustering, also calls for AutoML solutions, especially due to the ambiguity involved when evaluating clustering results. Motivated by this shortcoming, in this paper, we introduce a framework for automated clustering that encompasses two main modules: algorithm selection and hyperparameter tuning. Our approach to algorithm selection relies on meta-learning, based on novel meta-features extracted from data sets that attempt to capture similarities in the clustering structure. This approach is coupled with a method for hyperparameter tuning based on Bayesian optimization, where the main novelty is the proposal of an optimization goal that combines different cluster validity indices. We demonstrate the merits of our approach by empirical evaluation on 24 real-life data sets, which shows promising results when compared to existing methods.**

***Index Terms*—automatic clustering, hyperparameter tuning, meta-learning**

I. Introduction

Automated machine learning (AutoML) is the discipline of fully automating the modelling process of machine learning pipelines. Recently, the topic of AutoML has attracted attention with many systems, such as Auto-Weka [25], Auto-Sklearn [9] and Google's Vizier [10]. Unfortunately, even though several such works target AutoML in the context of supervised learning, there is much less work in the area of unsupervised learning, in particular AutoML for clustering. Partly, this is due to the inherent difficulty in evaluating clustering results. There is a certain amount of subjectivity when evaluating the quality of clustering due to the lack of ground truth or a commonly accepted evaluation metric. Instead, data mining experts typically resolve to visualization techniques for evaluation, a method that is not always viable, especially in cases of high-dimensional or complex data.

Motivated by this shortcoming, in this paper, we address the Combined Algorithm Selection and Hyperparameter optimization problem (CASH), as defined in [25], but in the context of unsupervised learning. Given a clustering task, an input data set, and a set of available clustering algorithms, the problem is to select the best clustering algorithm along with appropriate values for its input parameter(s) according to an optimization criterion. Essentially, the problem can be cast as a search problem, where the search space consists of the available clustering algorithms and the domain of values for all potential input parameters, and the objective is to identify the best performing configuration. We tackle this problem by applying a combination of meta-learning (learning from already clustered data sets) and Bayesian optimization techniques.

In brief, our contributions can be summarized as follows:

- We present AutoClust, an end-to-end framework for automatic clustering algorithm selection based on meta-learning and cluster validity indices (Sect. III).
- We introduce a novel set of meta-features for clustering that outperform previous approaches for algorithm selection [8] based on statistical information and distance distribution (Sect. III-C).
- We propose a method for hyperparameter tuning of clustering algorithms, which capitalizes on a new optimization criterion, namely regression of cluster validity indices (Sect. III-D).
- We show an empirical evaluation of our approach using various real-life data sets that demonstrates its advantages against state-of-the-art methods (Sect. IV).

Also, we review related work in Sect. II, and we conclude the paper in Sect. V.

II. Related Work

Our work relates to *meta-learning* and *automated machine learning*, focusing on unsupervised learning, in particular clustering. The survey paper [23] reviews in detail how meta-learning can be applied for algorithm selection. For more recent surveys on meta-learning, we refer to [15], [26]. Also, we refer to [3] for an overview of the application of meta-learning to data mining problems.

Automated machine learning. Regarding the Combined Algorithm Selection and Hyperparameter optimization problem (CASH), state-of-the-art solutions such as Auto-Sklearn [9], Auto-Weka [25], TPOT [21], etc., are based on optimization techniques, such as genetic and Bayesian optimization. Auto-Net [18] is considered as an extension to Auto-

Sklearn with sole focus on parameter exploration for modern neural networks. Google's Vizier [10] is a state-of-the-art research, with focus on black-box optimization. SmartML [17] is a system for automated selection of classification algorithms and for hyperparameter tuning that uses both meta-learning and Bayesian optimization. Additionally, Nural et al. [20] explore how meta-learning can be applied to regression problems via the open source ScaLation framework. We also refer to recent advances in Automated Data Science [2].

Although these frameworks provide solutions in the context of supervised learning (classification and regression), the CASH problem in unsupervised learning remains open. This is generally due to the lack of information to be used for validation purposes, such as the true clusters present in a data set, which hinders the selection of an appropriate objective function. Although many cluster validity indices have been proposed for the evaluation of clustering results, there still exists no universally applicable cluster validity index. The reason is that each index evaluates different criteria, e.g., the compactness of clusters, the separation between clusters, the density of clusters, etc. Consequently, it is not straightforward to define an optimization goal based on a specific cluster validity index, in contrast to the case of supervised learning where an obvious optimization goal is accuracy.

Automated Clustering. Only few works address the problem of automated clustering, and mostly tackle solely the problem of algorithm selection. To the best of our knowledge, Souto et al. [6] are the first to address the problem of algorithm selection in the context of clustering. They suggest a meta-learning approach in order to extract knowledge from similar, previously tackled, problem instances.

Following this approach, Ferrari et al. [8] propose a new set of novel distance-based meta-features for clustering algorithm selection along with a ranking method for algorithm performance. Also, Muravyov et al. [19] use meta-learning to predict the cluster validation index that is better used according to the data set characteristics as an additional step before selecting an algorithm. In the recent study [24], the authors propose a meta-learning approach with the use of internal clustering validation criteria to separate uniform from non-uniform data sets and predict cluster cardinality.

III. The AutoClustFramework

The goal of AutoClust, as an AutoML system, is to provide a machine learning algorithm along with a set of values for its input parameters that are expected to perform well for a new problem instance. The proposed architecture consists of two separate phases: (a) the *learning phase* (Sect. III-B), which is an offline process, and (b) the *online phase* where algorithm selection (Sect. III-C) and hyperparameter tuning (Sect. III-D) take place. First, we state the problem in a formal way (Sect. III-A).

A. Problem Formulation

Let $\mathcal{D}$ denote a set of n data sets $\mathcal{D} = \{D_1, D_2, \ldots, D_n\}$. Also, let $F(D) = \{f_1, \ldots, f_k\}$ denote a set of k meta-features extracted from data set D. Let $\mathcal{A} = \{A_1, \ldots, A_m\}$ be a set of clustering algorithms, and let Λ_i denote the domain of hyper-parameters of algorithm A_i. Finally, let $L(A_i(\lambda), D)$ denote the loss of A_i with hyperparameters $\lambda \in \Lambda_i$ on D. Then, the *Combined Algorithm Selection and Hyperparameter optimization (CASH)* problem is to find the joint algorithm and hyperparameter setting that minimizes this loss:

$$A^*, \lambda^* = \underset{A_i \in \mathcal{A}, \lambda \in \Lambda_i}{\arg\min} \; L(A_i(\lambda), D)$$

B. Offline Learning Phase

Following the paradigm of meta-learning, AutoClust learns from already processed data sets. This is achieved by (a) extracting a set of meta-features $F(D)$ for each data set D, and (b) applying different clustering algorithms on known data sets using various configurations, and recording their performance using cluster validity indices [11].

In more detail, for each clustering algorithm $A_i \in \mathcal{A}$ ($i \in [1, m]$), we have a domain Λ_i^j of discrete values for each hyperparameter λ_j of algorithm A_i. In case the input domain of a hyperparameter is continuous, we discretize it by selecting a reasonable set of discrete values. We explore a subset of the domain by applying a combination of *Grid Search* and *Random Search*, when the parametric space includes continuous values. The details on the explored search space are provided in Section IV-A.

In the learning phase, for each data set D, we execute each algorithm A_i several times (using $|\Lambda_i|$ to represent this number) for different values of hyperparameters, as indicated by the above search methods. Thus, we invoke $\sum_{i=1}^{m} |\Lambda_i|$ executions of clustering algorithms for each data set, and we evaluate the result quality. We employ a variety of widely-used cluster validity indices including ones that rely on labeled data. This allows for the framework to transfer knowledge gathered from evaluating labeled data to an unsupervised environment. The information about the execution of each algorithm on a data set, along with its hyperparameter values and cluster validity indexes is stored in a meta-knowledge repository, as shown in Figure 1. At any point, if n data sets have been processed by AutoClust, the meta-knowledge repository stores $\sum_{j=1}^{n} \sum_{i=1}^{m} |\Lambda_i|$ records, and each record consists of the measured cluster validity indices $\{v_1, v_2, \ldots\}$. Intuitively, the repository is constantly updated with new data sets and our system gradually learns algorithms and configurations that perform best for a growing set of known data sets.

C. Algorithm Selection

AutoClust adopts a meta-learning approach to address the problem of algorithm selection. For this purpose, the mainstream technique is to extract a carefully selected set of *meta-features* that serve as a concise representation of a data set. Then, given a new data set, we extract the same set of meta-features and we cast the problem of finding similar data sets to similarity matching of meta-feature descriptions. After finding the most similar data set, we use the information in

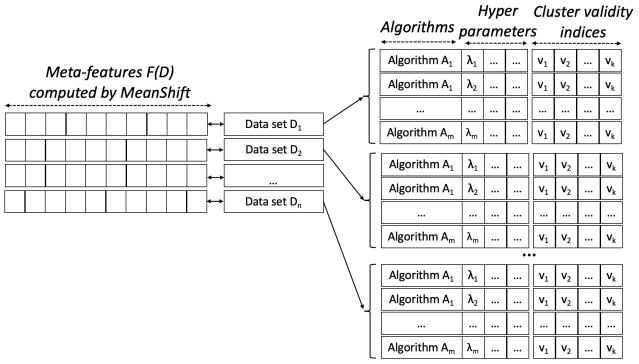

Fig. 1. Example of the data stored in the meta-knowledge repository.

the meta-knowledge repository to retrieve the algorithm with best performance, as recorded in the offline learning phase.

Meta-feature extraction. Existing studies in meta-feature extraction use various types of meta-features [8], [26], but which set of meta-features is more appropriate for each problem is still unclear. AUTOCLUST adopts a novel set of meta-features based on internal cluster validity indices. Intuitively, we expect that similarity based on the values in these internal cluster validity indices is going to reveal data sets with similar structural properties. More formally, given a data set D_{new}, we identify the data set $D' \in \mathcal{D}$, such that $\arg\max sim(F(D'), F(D_{new}))$ among all data sets in the meta-knowledge repository, where sim denotes a similarity function applied on vectors $F(D')$, $F(D_{new})$. In turn, this allows AUTOCLUST to recommend as clustering algorithm for D_{new}, the best performing algorithm in D'. To determine the best performing algorithm we use the Adjusted Rand Index (ARI) [22]. The list of internal cluster validity indices used is listed in Table I. To extract the meta-features $F(D)$ for a given data set D, we need to employ the *same* clustering algorithm for each data set, so that the relative similarity between the produced cluster validity indices is captured in a meaningful way. Moreover, the selected clustering algorithm should not be sensitive to the clusters shape and should require the minimum possible input parameters. Based on this rationale, we choose the *MeanShift* clustering algorithm, which only requires *bandwidth* as input parameter, which is in turn automatically selected [5].

Algorithm selection. For the actual step of algorithm selection, we adopt a k-nearest neighbor approach. When $k = 1$, we identify the most similar meta-features to the data set at hand, and we obtain the best performing clustering algorithm from the meta-knowledge repository. When $k > 1$, we identify the k most similar meta-features, which may correspond to different clustering algorithms. We use majority voting to determine the selected algorithm. As will be shown in Section IV, the k-nearest neighbor approach works better for this task than other classifiers that were tested (Random Forests and Multilayer Perceptron).

TABLE I
ALGORITHMS AND CLUSTER VALIDITY INDICES SUPPORTED BY AUTOCLUST

Algorithms in AUTOCLUST	K-means, DBSCAN, Hierarchical, Spectral, OPTICS, Birch, MeanShift, Agglomerative
Cluster validity indices	Silhouette [16], Dunn [16], C-Index [7], Calinski Harabasz [16], Davies Bouldin [16], SDbw [12], CDBW [13], Tau [4], Ratkowsky Lance [4], McClain Rao [4]

D. Hyperparameter Tuning

We couple our approach for algorithm selection with a method for hyperparameter tuning, which allows setting values of input parameters, a necessary but difficult and time-consuming task. For this purpose, we use a state-of-the-art Bayesian optimization method, an iterative algorithm that is used to optimize expensive black-box functions. We choose to implement a *Tree Structured Parzen Estimator* approach, as described in [1].

The major challenge that arises is choosing an evaluation metric to define as an optimization target. Different from supervised learning, where the labels are known and the optimization goal is straightforward to define, in clustering the lack of supervision makes metrics that rely on the concept of ground truth unavailable. Even though other metrics do exist, such as Silhouette, CDBW, etc., known as internal cluster validity indices, none of them fits as a universal solution to all problem instances.

To address this challenge, we present a novel method to define a target function fit for Bayesian optimization in the context of clustering. We aim to explore the relationship between internal cluster validity indices $\{v_1, v_2, \dots\}$ and the Adjusted Rand Index (ARI) and learn a mapping between them. To this end we train a Regression model $RM(v_1, v_2, \dots)$ that takes cluster validity indices as input and provides prediction for ARI. In order to better capture nonlinear dependencies, we opt to use a Multilayer Perceptron Regressor (MLP) that uses 10 different cluster validity indices as predictor variables (listed in Table I). After different tests, we employ a neural network (illustrated in Figure 2) that follows a simple architecture with 10 input nodes that correspond to the cluster validity indices, 3 hidden layers of 60, 30 and 10 neurons respectively and a 1-node output layer of the ARI prediction. All of the hidden layers and the output layer use a ReLU activation function and a Normal Kernel initializer. The neural network uses an ADAM optimizer in order to minimize the Mean Squared Error (MSE).

After training, the aforementioned model is used at the evaluation steps of the iterative Bayesian optimization algorithm. We perform regression on the available cluster validity indices, in order to combine them into one evaluation metric, as depicted in Figure 2. First the search space is narrowed down by the algorithm selection process. Afterwards the Bayesian optimization procedure is initiated, where a configuration is selected at time and is executed on the data set at hand. This

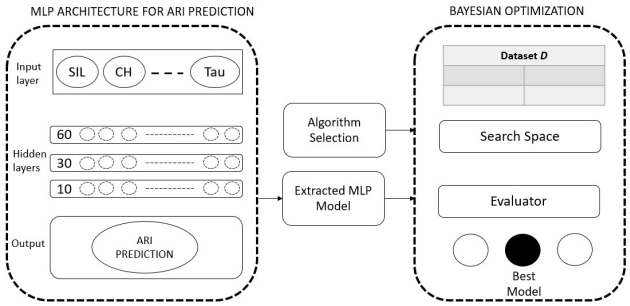

Fig. 2. Implementation of the MLP predictive model during the Bayesian Optimization procedure.

configuration is evaluated according to a set of 10 internal cluster validity indices. These measures are provided as input to the MLP regressor and the ARI prediction provided takes the role of the target function. Finally the surrogate model is updated and the process is iterated with the next point selected until the budget of evaluations is reached.

Consequently, this method enables the generalized applicability of the Bayesian optimization procedure on clustering new, unseen problem instances, without the need of prior knowledge of a best evaluation metric.

IV. Experimental Evaluation

The AutoClust framework has been developed using Python 3, on top of the HyperOpt and Scikit-Learn libraries, deployed in the BigDataStack environment [14].

A. Experimental Setup and Methodology

Data sets. A set $\mathcal{D}$ of 24 data sets (Table III) was used, obtained from the UCI Machine Learning Repository. These data sets include information on the natural clustering of the data (ground truth). All of the data sets were pre-processed to be transformed as more appropriate input for the task of clustering.

Methodology. To assess the performance of AutoClust we define two set of experiments. In the first one (Section IV-B), we assess the ability of the framework to select the best performing algorithm, determined offline by brute-force evaluation of all clustering algorithms, indicated by the Adjusted Rand Index (ARI) [22]. We measure the quality of results by evaluating in how many cases the meta-learner selects the best performing clustering algorithm. We call this metric the top-1 accuracy. We also report a relaxed metric (top-3 accuracy), where the meta-learner is allowed to return a set of 3 algorithms and measure again how many times the best performing algorithm is included in this set.

In the second set of experiments (Section IV-C), we address the overall performance of the framework, after hyperparameter tuning. We compare the results of AutoClust after applying Bayesian optimization with a limited evaluation budget (i.e., number of iterations) and our novel target function, to the results achieved by two different methods, which are considered as baselines. The first method simulates a basic practice for selecting a clustering model. From the execution of all of the available algorithms with their default hyperparameters, the best one is selected based on the optimization of the Silhouette index. The second one refers to exhaustively searching configurations and using each of the 10 internal cluster validity indices to indicate the best performing one (but each time using a *single* index). This target for comparison is thought to simulate the case of a Data Science practitioner, who selects one index from the available ones for evaluation of clustering results.

TABLE II
Domain of hyperparameters Λ_i for each algorithm A_i.

Algorithm (A_i)	Hyperparameter	Search Space (Λ_i)
Affinity Propagation	damping	[0.5,1]
Agglomerative	No_Clusters	[2,30]
	Affinity	["Euclidean", "l1", "l2" , "manhattan", "cosine"]
	Linkage	["ward", "complete", "average","single"]
Birch	Threshold	[0.2,0.7]
	No_Clusters	[2,30]
DBSCAN	Eps	[0.1,0.7]
	Min_Samples	[3,7]
K-means	No_Clusters	[2,30]
MeanShift	Bandwidth	Automatically Selected
OPTICS	Min_Samples	[3,7]
	Cluster_Method	["xi", "DBSCAN"]
Spectral Clustering	No_Clusters	[2,30]
	gamma	[0.5,1.5]

Parameters. The configurations search space for hyperparameter tuning that we explore is set by discretizing – when necessary – each individual domain of the parameters of the algorithms. Additionally, when these domains include continuous values we include a random selection technique of 20 evaluations uniformly at random. The minimum and maximum values for each parameter are presented in Table II with steps of 1 and 0.1 for discrete and continuous variables respectively.

B. Results on Algorithm Selection

In this experiment, we evaluate the performance of AutoClust with the K-Nearest Neighbors (KNN) meta-learner. We also compare its performance to two different meta-learners: Multilayer Perceptron (MLP) and Random Forest (RF). Both MLP and RF are used with the default parameters provided by Scikit-Learn. KNN is executed for different values of k (from 1 to 10) and evaluated for the best performing value of k.

As for the meta-features used for data representation, we compare two different sets; our novel set of meta-features based on internal cluster validity indices, against the state-of-the-art approach for algorithm selection in clustering proposed by Ferrari et al. [8], which is based on the pairwise distance distribution of objects in the data set.

The set of available algorithms that are to be predicted is listed in Table III. We report that 2 of the 8 algorithms (namely MeanShift and OPTICS) were not indicated as best performing algorithm in any of the 24 data sets used in our study. According to this observation, a simplistic competitive approach

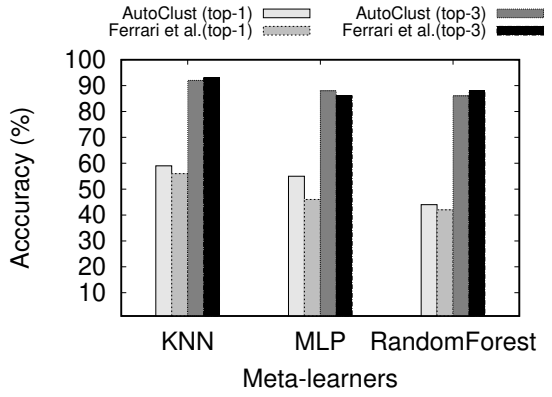

Fig. 3. Top-1 and top-3 accuracy for algorithm selection as achieved for different combinations of 3 meta-learners and 2 sets of meta-features.

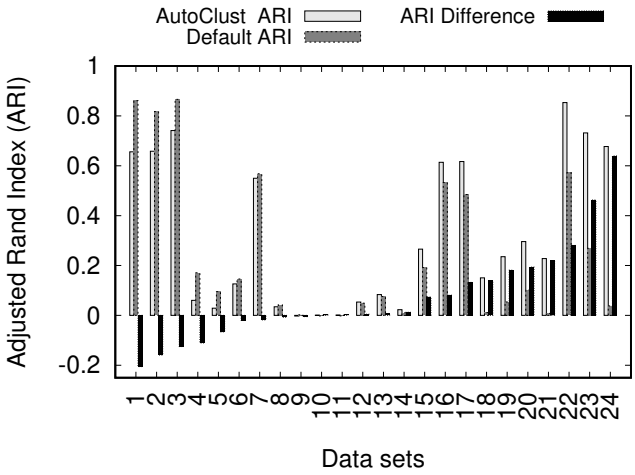

Fig. 4. Comparison of AUTOCLUST performance vs. Exploration of algorithms with default parametric values and silhouette index optimization.

is that of *random algorithm selection* that is equivalent to throwing a dice to select an algorithm out of the remaining 6 algorithms, thus expecting accuracy of 16.6%.

Figure 3 shows the results of the meta-features used by AUTOCLUST against the meta-features used in [8], in terms of top-1 and top-3 accuracy, respectively. In the case of top-1 accuracy, AUTOCLUST outperforms the approach of Ferrari et al. [8] when using KNN (59.25% vs. 56.6%). Recall that the random algorithm selection achieves a poor 16.6%. When using other meta-learners (MLP and RF), this improvement increases, although the absolute performance drops for both approaches. Thus, our first finding is that our meta-features work better than the distance-based meta-features of [8]. Our second finding is that KNN works better than MLP and RF, thus validating our choice of KNN as the meta-learner used in AUTOCLUST. When considering top-3 accuracy, we observe that the absolute accuracy values are higher, as expected. This means that much higher accuracy (in the order of 85%–92.5% can be achieved, if we allow AUTOCLUST to select 3 algorithms out of the 8 available ones. Again, KNN outperforms all the other meta-learners, but the approach of [8] is marginally better than AUTOCLUST in the case of KNN. Notice however that the differences between the two approaches are very small in this experiment.

C. Results on the Overall Performance of AUTOCLUST

In this experiment, we evaluate the overall quality achieved by the end-to-end execution of AUTOCLUST, including the hyperparameter tuning. The quality of results is measured using the Adjusted Rand Index (ARI), a similarity measure that indicates how similar the clustering obtained by AUTOCLUST is compared to the ground truth. The evaluation budget set for the Bayesian optimization procedure of the framework is set to 40 trials and the MLP Regressor responsible for ARI prediction is trained for 100 epochs.

Comparison to Default Parameters. In this experiment, we compare AUTOCLUST against a baseline approach (called *Default ARI*). We consider as competitor the best performing clustering algorithm from the 8 available ones, using their default hyperparameter values provided by Scikit-Learn. The best performing clustering algorithm is determined by the Silhouette coefficient, a metric that we choose due to its popularity for evaluating clustering results.

Figure 4 shows the obtained ARI measurements for AUTOCLUST (in light grey), the competitor Default ARI (in grey), and the difference (in black), for the 24 data sets (depicted on the x-axis). AUTOCLUST outperforms the baseline in half of the cases, and in several cases the difference is substantial (up to 0.62). The baseline is better in 7 cases, albeit the difference is much smaller (at most 0.20). In the remaining cases, the result is almost the same. Essentially, this experiment demonstrates that AUTOCLUST usually outperforms the common practice of running multiple clustering algorithms and keeping the best.

Comparison to Exhaustive Search. In this experiment, we demonstrate the usefulness of our new optimization goal that combines existing cluster validity indices. To this end, we compare against a method called *Exhaustive Search*, which corresponds to the brute force evaluation of the hyperparameter values in the search space indicated in Table II. The optimization goal of *Exhaustive Search* is set to be a *single* cluster validity index, thus we have 10 variants of *Exhaustive Search* each optimizing a different index. We compare the ARI achieved by AUTOCLUST against each of these 10 variants.

Table III presents the results of this comparison for each data set, with the last column ("wins") measuring the number of variants of *Exhaustive Search* that AUTOCLUST outperforms for each data set. On average, AUTOCLUST outperforms 7.5 out of the 10 variants. This shows that optimizing a single cluster validity index is not a good solution, thereby supporting our approach for an optimization goal that combines different cluster validity indices.

V. CONCLUSIONS AND FUTURE WORK

In this paper, we proposed AUTOCLUST, a novel solution for automated clustering. We introduced a new set of meta-

TABLE III

DETAILED COMPARISON OF EXHAUSTIVE SEARCH AND AUTOCLUST IN TERMS OF ADJUSTED RAND INDEX, ON A SET OF 24 REAL-LIFE DATA SETS. FOR EXHAUSTIVE SEARCH, WE SELECT THE BEST CONFIGURATION BASED ON EVERY SINGLE ONE OF THE 10 POPULAR INTERNAL CLUSTER VALIDITY INDICES. THE COLUMN "WINS" RECORDS HOW MANY TIMES (OUT OF 10) AUTOCLUST PRODUCES BETTER RESULTS.

Data set	AutoClust	CIndex	Silhouette	SDbw	Ratkowsky Lance	mcclain_Rao	Dunn	davies_bouldin	CDBW	Calinski Harabasz	Tau	Wins
arrythmia	0.053338	-0.02388	0.004002	0.043805	0.102981	-0.02388	0.03179	-0.00797	-0.04677	0.050228	0.178618	**8**
creditapproval	0.126332	0.002388	0.043704	0.011888	0.402086	-0.00246	0.100307	0.011888	-0.00715	0.043704	0.043704	**9**
balance_scale	0.06021	0.032206	0.122753	0.169141	0.143459	0.032206	0.034252	0.065231	0.029474	0.122753	0.122753	**4**
breast_cancer	0.655808	-0.08346	0.697157	0.571898	0.849761	-0.08346	0.002575	0.456599	-0.08342	0.849761	0.692396	**6**
cpu_pre.csv	-0.00244	0.091799	0.000147	0.003683	0.002813	0.094248	0.000507	0.053573	0.092846	0.252434	0.09651	**0**
dermatology	0.853281	0.006382	0.080987	0.352213	0.572695	0.006382	6.17E-05	0.650824	0.056232	0.570933	0.456818	**10**
ecoli	0.677235	0.004318	0.004373	0.030322	0.756124	0.004318	0.037978	0.053926	0.014391	0.753814	0.803048	**7**
german	0.022979	-0.01959	-0.00665	-0.00313	0.025637	-0.00434	0.002665	0.016203	-0.00665	-0.00313	-0.00313	**9**
glass	0.22764	-0.00961	0.010579	-0.03442	0.267645	-0.03771	-0.00235	0.037543	0.018231	0.202337	0.230373	**8**
haberman	0.000956	-0.04943	-0.01296	0.026567	-0.00196	-0.04943	0.011577	0.152523	-0.01296	-0.00252	-0.00252	**7**
heart-statlog	0.235252	-0.00253	-0.00122	0.106031	0.348755	-0.0026	0.001852	0.007016	-0.00253	0.357626	0.219844	**8**
iono	0.265788	-0.03304	-0.03695	-0.02921	0.177607	-0.03304	0.004473	0.628927	-0.05573	0.177607	0.099985	**9**
iris	0.549894	0.009058	0.568116	0.569446	0.568116	0.009058	0.568116	0.461129	0.030176	0.568116	0.568116	**4**
segment	0.296011	0.0004	0.0004	0.313976	0.104139	0.0004	0	0.000186	0.007267	-0.00038	-0.00038	**9**
sonar	0.001087	0.002295	0.002049	-0.00436	0.000101	-0.00421	-0.00121	-0.00436	0.001163	0.008546	0.014516	**5**
tae	0.034381	0.007541	0.015761	0.004679	0.089964	0.007541	0.042051	0.030829	0.015761	0.085451	0.042051	**6**
thy	0.73179	0.060863	0.11144	-0.06784	0.74776	-0.06784	0.031407	0.060863	0.094704	0.409613	0.954922	**8**
vehicle	0.083593	-5.89E-05	-8.19E-05	0.010517	0.085913	-5.89E-05	9.54E-05	-0.00052	0.05131	0.085913	0.084687	**7**
vowel	0.028881	0.035002	0.001861	-0.00403	-0.00306	0.000236	-0.00462	0.02341	0.015206	-0.00403	-0.00462	**9**
wdbc	0.613728	-0.00555	0.002403	0.019982	0.742286	-0.00555	0.002403	0.358676	0.013982	0.742286	0.600206	**8**
wine	0.616866	0.011538	0.011526	-0.00517	0.948669	0.011538	-0.00196	-0.00517	0.035201	0.948669	0.825004	**7**
wisc	0.741323	-0.07053	-0.04252	0.35786	0.866065	-0.09911	0.002575	0.738869	-0.09911	0.8444	0.691639	**8**
yeast	0.15036	-0.01501	-0.00092	0.169666	0.093953	-0.01501	0.00298	0.02294	0.005218	0.022202	0.062958	**9**
zoo	0.65808	0.033596	-0.03218	-0.00762	0.649351	-0.04324	0.250182	0.432342	-0.0458	-0.00762	0.883374	**9**

features, based on internal cluster validity indices and developed a method based on a regression technique that combines this set of indices to provide a clear optimization target, which can be in turn used as a target function for hyperparameter tuning with Bayesian optimization. We demonstrate the merits of our approach experimentally using 24 real-life data sets. Regarding future work, our plans are to further improve the algorithm selection step, by exploring extended sets of meta-features that will serve as good indicators of similar clustering structure in the data sets. Moreover, we intend to build a scalable version of our framework that will be readily applicable for Big Data.

ACKNOWLEDGMENT

This work is supported by EU/H2020 project BigDataStack, which has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 779747.

REFERENCES

[1] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl. Algorithms for hyper-parameter optimization. In *Proc. of NIPS'11*, pages 2546–2554, 2011.

[2] T. D. Bie, L. D. Raedt, H. H. Hoos, and P. Smyth. Automating Data Science (Dagstuhl Seminar 18401). *Dagstuhl Reports*, 8(9):154–181, 2019.

[3] P. Brazdil, C. G. Giraud-Carrier, C. Soares, and R. Vilalta. *Metalearning - Applications to Data Mining*. Cognitive Technologies. Springer, 2009.

[4] M. Charrad, N. Ghazzali, V. Boiteau, and A. Niknafs. *An examination of indices for determining the number of clusters : NbClust Package*. 09 2013.

[5] D. Comaniciu and P. Meer. Mean shift: a robust approach toward feature space analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(5):603–619, 2002.

[6] M. C. P. de Souto, R. B. C. Prudencio, R. G. F. Soares, D. S. A. de Araujo, I. G. Costa, T. B. Ludermir, and A. Schliep. Ranking and selecting clustering algorithms using a meta-learning approach. In *Proc. of IJCNN'08*, pages 3729–3735, 2008.

[7] B. S. Everitt, S. Landau, and M. Leese. *Cluster Analysis*. 2009.

[8] D. G. Ferrari and L. N. de Castro. Clustering algorithm selection by meta-learning systems: A new distance-based problem characterization and ranking combination methods. *Inf. Sci.*, 301:181–194, 2015.

[9] M. Feurer, A. Klein, K. Eggensperger, J. T. Springenberg, M. Blum, and F. Hutter. Efficient and robust automated machine learning. In *Proc. of NIPS'15*, page 2755–2763, 2015.

[10] D. Golovin, B. Solnik, S. Moitra, G. Kochanski, J. Karro, and D. Sculley. Google vizier: A service for black-box optimization. In *Proc. of KDD'17*, pages 1487–1495. ACM, 2017.

[11] M. Halkidi, Y. Batistakis, and M. Vazirgiannis. On clustering validation techniques. *J. Intell. Inf. Syst.*, 17(2-3):107–145, 2001.

[12] M. Halkidi and M. Vazirgiannis. Clustering validity assessment: Finding the optimal partitioning of a data set. In *Proc. of ICDM'01*, pages 187–194, 2001.

[13] M. Halkidi and M. Vazirgiannis. A density-based cluster validity approach using multi-representatives. *Pattern Recognition Letters*, 29(6):773 – 786, 2008.

[14] D. Kyriazis et al. BigDataStack: A holistic data-driven stack for big data applications and operations. In *Proc. of 2018 IEEE BigData Congress*, pages 237–241, 2018.

[15] C. Lemke, M. Budka, and B. Gabrys. Metalearning: a survey of trends and technologies. *Artif. Intell. Rev.*, 44(1):117–130, 2015.

[16] Y. Liu, Z. Li, H. Xiong, X. Gao, and J. Wu. Understanding of internal clustering validation measures. In *Proc. of ICDM'10*, pages 911–916, 2010.

[17] M. Maher and S. Sakr. SmartML: A meta learning-based framework for automated selection and hyperparameter tuning for machine learning algorithms. In *Proc. of EDBT'19*, pages 554–557, 2019.

[18] H. Mendoza, A. Klein, M. Feurer, J. T. Springenberg, and F. Hutter. Towards automatically-tuned neural networks. In *Proc. of ICML/AutoML'16 workshop*, pages 58–65, 2016.

[19] S. Muravyov and A. Filchenkov. Meta-learning system for automated clustering. In P. Brazdil, J. Vanschoren, F. Hutter, and H. H. Hoos, editors, *Proc. of PKDD/AutoML'17 workshop*, pages 99–101, 2017.

[20] M. V. Nural, H. Peng, and J. A. Miller. Using meta-learning for model type selection in predictive big data analytics. In *Proc. of IEEE Big Data'17*, pages 2027–2036, 2017.

[21] R. S. Olson and J. H. Moore. TPOT: A tree-based pipeline optimization tool for automating machine learning. In *Proc. of ICML/AutoML'16 workshop*, pages 66–74, 2016.

[22] W. M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

[23] K. Smith-Miles. Cross-disciplinary perspectives on meta-learning for algorithm selection. *ACM Comput. Surv.*, 41(1):6:1–6:25, 2008.

[24] J. A. Sáez and E. Corchado. A meta-learning recommendation system for characterizing unsupervised problems: On using quality indices to describe data conformations. *IEEE Access*, 7:63247–63263, 2019.

[25] C. Thornton, F. Hutter, H. H. Hoos, and K. Leyton-Brown. Auto-weka: combined selection and. hyperparameter optimization of classification algorithms. In *Proc. of KDD'13*, pages 847–855, 2013.

[26] J. Vanschoren. Meta-learning: A survey. *CoRR*, abs/1810.03548, 2018.