# Statistically Rigorous Testing of Clustering Implementations

Xin Yin Vincenzo Musco Iulian Neamtiu Usman Roshan
*Department of Computer Science*
*New Jersey Institute of Technology*
Newark, NJ, USA
{xy258, vincenzo.a.musco, ineamtiu, usman}@njit.edu

***Abstract*—Clustering is a widely used AI technique, but defining clustering correctness, as well as verifying and validating clustering implementations, remains a challenge. To address this, we propose a statistically rigorous approach that couples differential clustering with statistical hypothesis testing, namely we conduct statistical hypothesis testing on the outcome (distribution) of differential clustering to reveal problematic outcomes.**

**We employed this approach on widely-used clustering algorithms implemented in popular ML toolkits; the toolkits were tasked with clustering datasets from the Penn Machine Learning Benchmark. The results indicate that there are statistically significant differences in clustering outcomes in a variety of scenarios where users might not expect clustering outcome variation.**

***Index Terms*—Clustering, Machine Learning, Testing, Statistics**

## I. Introduction

Cluster analysis (*Clustering*) is an unsupervised learning technique used to group together entities that are related or share similar characteristics. While clustering is a well-established area with research going back to the 1950s, there is a stringent and urgent need for approaches to testing Clustering implementations due to several converging factors.

First, supervised and unsupervised learning have started to permeate software products, from "smart" home devices [1] to self-driving platforms [2] and predictive analytics [3]. These implementations make critical decisions themselves or are used to aid decision-making (e.g., autonomous driving or financial fraud detection).

Second, there has been a proliferation of clustering implementations, mostly in the form of software toolkits (e.g., MATLAB and R each offer more than 100 clustering packages [4], [5]). These implementations are run by millions of users [6], [7] including non ML-experts (from life scientists to medical professionals) who should be able to assume that the implementations are correct.

Third, software engineers are under pressure to incorporate/adopt Machine Learning into software products and processes [8]–[10]; engineers should be able to (reasonably) assume that clustering implementations are reliable and interchangeable, i.e., for a given algorithm, its implementation is correct and has no negative impact on the clustering outcome.

However, ensuring clustering correctness, or even specifying clustering correctness, remain distant goals. Therefore, we propose an approach that can expose substantial and systematic issues with clustering algorithms' actual implementations, e.g., wide variations across runs for theoretically-stable, deterministic algorithms, widely different outcomes for different implementations of the same algorithm, or consistently poor performance in specific implementations.

Our approach leverages the wide availability of clustering implementations and datasets with ground truth, coupled with a statistics-driven approach to help developers (or toolkit testers) find statistically significant clustering accuracy differences.

To evaluate our approach and conduct our study, we chose 7 popular clustering algorithms, 4 nondeterministic (K-means, K-means++, Spectral Clustering, Expectation Maximization-GaussianMixture); and 3 deterministic (Hierarchical clustering-Agglomerative, Affinity Propagation, DBSCAN).[1] For uniformity and brevity, we use the following shorthands for the algorithms: *kmeans*, *kmeans++*, *gaussian*, *spectral*, *hierarchical*, *dbscan*, *apcluster*. We chose 7 widely-used clustering toolkits: MATLAB, mlpack, R, Scikit-learn, Shogun, TensorFlow, and WEKA. For uniformity and brevity, we use the following shorthands for the algorithms: *matlab*, *mlpack*, *R*, *sklearn*, *shogun*, *tensorflow*, *weka*. Our clustering inputs are 162 datasets from the Penn Machine Learning Benchmark; the datasets are described in Section II-B. 60% of the datasets come from sciences (medicine, biology, physics) with clusters crafted by domain experts.

Our basic metric of clustering similarity and accuracy is the versatile *adjusted Rand index* (ARI) [11], described in detail in Section II-A. The ARI measures the similarity between two partitions $U$ and $V$ of the same underlying set $S$. The ARI varies between $-1$ and $+1$, where $ARI = +1$ indicates "perfect match", $ARI = 0$ corresponds to independent/random clustering, and $ARI = -1$ indicates "perfect disagreement", that is completely opposite assignment. We use the term *accuracy* to refer to the ARI in the case when $U$ is a clustering produced by an algorithm and $V$ is the Ground Truth (as labeled in PMLB) for that dataset.

*Examples: clustering accuracy in promoters.* The clustering accuracy is measured by ARI between clustering results and

[1] Deterministic clustering algorithms should, in theory, produce the same clustering when run repeatedly on the same input. For nondeterministic (aka randomized) algorithms, the clustering might vary.

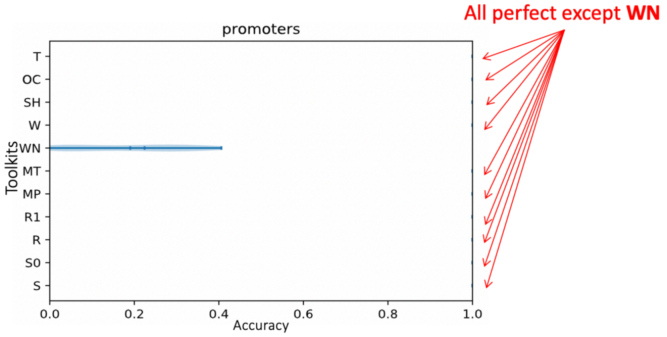

Fig. 1. Kmeans++: Accuracy distributions for 11 toolkits on dataset promoters (WN = WEKA with default normalization).

ground truth. The promoters [12] dataset essentially contains two clusters that partition the underlying E.coli DNA (gene) sequences into “promoters” [2] and “non-promoters”. We ran our statistical approach on clustering outcomes of algorithm K-means++ in 11 toolkit configurations (sample size = 30, corresponding to the 30 repeated runs of that configuration on the same dataset). The resulting accuracy range is shown in Figure 1. Note that 10 toolkits achieve identical, perfect clusterings in each of the 30 runs, i.e.,

$$minAccuracy = maxAccuracy = 1$$

However, the toolkit WEKA’s accuracy across 30 runs varied:

$$minAccuracy = -0.01; maxAccuracy = 0.4$$

This allows us to make two observations: (1) WEKA’s accuracy varies substantially across re-runs on the same input dataset; and (2) all toolkits achieve perfect clusterings in every run, whereas’s WEKA’s best run has accuracy 0.4. This clearly indicates a clustering implementation issue.[3]

Our approach exposes such issues (and more) in a statistically rigorous way. *This is the first statistically rigorous approach to testing clustering implementations.*

Our approach has 4 components; we present each component in the context of using that component to test a null hypothesis:

1) How different runs of the same algorithm in the same implementation lead to different clusterings (Section III).
2) How different implementations of the same algorithm in different toolkits lead to different clusterings (Section IV).
3) The toolkit’s impact when comparing algorithms (Section V).
4) How different toolkits “disagree” (Section VI).

TABLE I
CATEGORIES FOR THE PMLB DATASETS.

| Category | Percentage |
|---|---|
| Medical/Health | 24% |
| Biology, Biochemistry, Bioinformatics | 15% |
| Physics, Math, Astronomy | 11% |
| Social, Census | 10% |
| Sports | 7% |
| Financial | 7% |
| Image recognition | 6% |
| Synthetic datasets | 6% |
| IT, AI | 4% |
| Linguistics | 3% |
| Miscellaneous | 7% |

## II. BACKGROUND

### A. Definitions

**Clustering.** Given a set $S$ of $n$ points ($d$-dimensional vectors in the $\mathcal{R}^d$ space), a clustering is a partitioning of $S$ into $K$ non-overlapping subsets (clusters) $S_1, \ldots, S_i, \ldots, S_K$ such that intra-cluster distance between points (that is, within individual $S_i$’s) is minimized, while inter-cluster distance (e.g., between centroids of $S_i$ and $S_j$ where $i \neq j$) is maximized.

**Accuracy.** The *adjusted Rand index* (ARI), introduced by Hubert and Arabie [11] is an effective and intuitive measure of clustering outcomes: it allows two different partitioning schemes of an underlying set $D$ to be compared. Multiple surveys and comparisons of clustering metrics have shown that ARI is the most widely used [13], most effective, as well as very sensitive [14]. Concretely, assuming two clusterings (partitionings) $U$ and $V$ of $S$, the ARI measures how similar $U$ and $V$ are. The ARI varies between $-1$ and $+1$, where $ARI = +1$ indicates perfect agreement, $ARI = 0$ corresponds to independent/random clustering, $ARI = -1$ indicates “perfect disagreement”, that is completely opposite assignment, and specially $-1 < ARI < 0$ is defined as “worse-than-random”, because it in worse than randomness clustering. Concretely, assuming two clusterings (partitioning) $U$ and $V$ of $S$, $ARI(U, V)$ measures how similar $U$ and $V$ are:[4]

$$ARI(U, V) = \frac{2(N_{00}N_{11} - N_{01}N_{10})}{(N_{00} + N_{01})(N_{01} + N_{11}) + (N_{00} + N_{10})(N_{10} + N_{11})}$$

### B. Datasets

We chose PMLB (Penn Machine Learning Benchmark) [16], a benchmark suite that includes “real-world, simulated, and toy benchmark datasets” [16]. PMLB was designed to benchmark ML implementations and avoid imbalance across meta-features (which often plagues handpicked datasets). PMLB is a collection of 166 datasets, of which we

[2] A *promoter* sequence marks the DNA region where the transcription of that gene into RNA should begin.

[3] WEKA developers were able to attribute this low accuracy to WEKA’s default normalization setting.

[4] Where “$N_{11}$ is the number of pairs that are in the same cluster in both $U$ and $V$; $N_{00}$ is the number of pairs that are in different clusters in both $U$ and $V$; $N_{01}$ is the number of pairs that are in the same cluster in $U$ but in different clusters in $V$; and $N_{10}$ is the number of pairs that are in different clusters in $U$ but in the same cluster in $V$” [15].

used 162; we excluded connect-4, poker, mnist, and kddcup due to their excessive size – running these hundred of times would be prohibitive. The following table contains descriptive statistics for the 162 datasets.

| | Min | Max | Geom. mean |
|---|---|---|---|
| Instances | 32 | 105,908 | 809.25 |
| Features (attributes) | 2 | 1,000 | 15.41 |
| K (# of clusters) | 2 | 26 | 3.18 |

Datasets have, on average, 809 instances (that is, points to be clustered) and the mean number of features (the number of attributes, or dimensions $d$) is 15. PMLB comes with ground truth, which allows us to measure clustering accuracy. About half the datasets have two clusters ($K = 2$), while for the rest we have $3 \leq K \leq 26$.

We categorized the nature of each dataset and present the category breakdown in Table I. We point out several things: the datasets are quite representative, as they cover a wide range of domains, from scientific to social to financial. About 60% of the datasets come from sciences (medicine, biology, physics) with clusters crafted by domain experts in those branches of science (not ML experts).

### C. Clustering Algorithms

We now present a brief overview of each algorithm.

$K$**-means.** The algorithm aims to cluster the observations (points in $S$) into $K$ distinct clusters, where observations belong to the clusters with the nearest mean. The goal is to minimize the sum of all intra-cluster distances. The algorithm starts from $K$ selected initial points as "centroids" (cluster centers). These centroids play a crucial role in the algorithm's effectiveness: the algorithm is not guaranteed to converge to a global minimum, so with "bad" centroids the algorithm can "fall" into local minima.

*Variation due to starting points:* the algorithm requires "starting points", that is, initial centroids. To ensure consistency across toolkits, we fed all toolkits the same starting points. We explored the variation in outcome by randomly picking different starting points from the datasets. That is, if a dataset has $K = 2$, for each run we pick two different points in $S$, $s_1$ and $s_2$ as centroids, and run the algorithm from there, with all toolkits starting with $s_1$ and $s_2$.

$K$**-means++** was designed to improve $K$-means by choosing the starting points more carefully so they are farther apart. Theoretically, this improved version ensures that the algorithm is less likely to converge to local minima.

**EM/Gaussian.** Gaussian mixture clustering is a model-based approach: clustering is first done using a model (i.e., a parametric distribution such as a Gaussian). Each cluster is defined by an initial random model and the dataset is composed of the mixture of these models. Then, the model/data fitness is optimized – a common optimization is Expectation-Maximization.

**Spectral clustering** computes eigenvalues of the similarity matrix between the data points to reduce the dimensionality of the original data. After the reduction, a clustering algorithm, e.g., $K$-means, is applied on the reduced-dimensionality data.

TABLE II
TOOLKIT/ALGORITHM CONFIGURATIONS

| | MATLAB | mlpack | R | Scikit-learn | Shogun | TensorFlow | WEKA |
|---|---|---|---|---|---|---|---|
| kmeans++ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| kmeans | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | |
| spectral | | | ✓ | ✓ | | | |
| hierarchical | ✓ | | ✓ | ✓ | | | |
| gaussian | ✓ | | | ✓ | | ✓ | ✓ |
| dbscan | | ✓ | ✓ | ✓ | | | |
| apcluster | | | ✓ | ✓ | | | |

TABLE III
LEVENE'S TEST RESULTS: THE NUMBER OF DATASETS, OUT OF 162, WITH SIGNIFICANT VARIANCE ($p < 0.05$)

| Algorithm | Toolkit | # Datasets |
|---|---|---|
| kmeans++ | sklearn | 126 |
| kmeans++ | R | 111 |
| kmeans++ | mlpack | 144 |
| kmeans++ | matlab | 125 |
| kmeans++ | shogun | 143 |
| kmeans++ | tensorflow | 144 |
| kmeans++ | weka | 157 |
| spectral | sklearn | 93 |
| spectral | sklearnfast | 97 |
| spectral | R | 113 |
| kmeans | sklearn | 148 |
| kmeans | R | 153 |
| kmeans | mlpack | 146 |
| kmeans | matlab | 141 |
| kmeans | shogun | 146 |
| kmeans | tensorflow | 145 |
| hierarchical | sklearn | 71 |
| hierarchical | R | 63 |
| hierarchical | matlab | 63 |
| gaussian | sklearn | 136 |
| gaussian | matlab | 153 |
| gaussian | tensorflow | 151 |
| gaussian | weka | 123 |

The aforementioned 4 algorithms were nondeterministic; we now discuss the 3 deterministic algorithms.

**Hierarchical clustering** is a deterministic algorithm, based on building a hierarchy of clusters using one of two approaches: (i) a bottom-up approach named "agglomerative": each observation is initially put in its own cluster and then they are merged, (ii) a top-down approach named "divisive": all observations are initially placed in the same cluster and then they are split. We use the first variant.

**DBSCAN** is a deterministic algorithm based on density, that is, high density regions of data are grouped together in a neighbor graph and form a data cluster.

**Affinity Propagation (AP)** is based on propagation on a graph in which each data point is a node. The algorithm builds clusters by iteratively passing messages from a node to another until determining which one is part of a specific cluster.

Some toolkits do not support all 7 algorithms; Table II shows the supported algorithm/toolkit combinations; in all, there were 27 algorithm-toolkit configurations.

TABLE IV
TOP-10 WIDEST DIFFERENCES IN ACCURACY ACROSS RUNS

| Algorithm | Toolkit | Dataset | Min | Max |
|---|---|---|---|---|
| gaussian | tensorflow | prnn_crabs | -0.005 | 1 |
| gaussian | matlab | prnn_crabs | -0.005 | 0.979 |
| gaussian | matlab | analcatdata_cr. | -0.024 | 0.958 |
| gaussian | tensorflow | twonorm | 0 | 0.908 |
| gaussian | matlab | twonorm | 0.003 | 0.910 |
| gaussian | tensorflow | ionosphere | 0.004 | 0.772 |
| spectral | R | breast-w | 0.056 | 0.818 |
| gaussian | matlab | analcatdata_aut.p | 0.041 | 0.794 |
| gaussian | matlab | wdbc | 0.007 | 0.754 |
| gaussian | tensorflow | breast-cancer-wsc. | 0.032 | 0.760 |

TABLE V
HIGHEST STANDARD DEVIATIONS IN ACCURACY ACROSS RUNS

| Algorithm | Toolkit | Dataset | Stddev |
|---|---|---|---|
| gaussian | tensorflow | twonorm | 0.400 |
| gaussian | tensorflow | prnn_crabs | 0.345 |
| gaussian | tensorflow | ionosphere | 0.298 |
| gaussian | sklearn | breast | 0.281 |
| kmeans++ | weka | australian | 0.236 |
| gaussian | matlab | house-votes-84 | 0.236 |
| gaussian | matlab | tokyo1 | 0.216 |
| gaussian | sklearn.0_tol | breast | 0.213 |
| gaussian | matlab | twonorm | 0.212 |
| gaussian | matlab | analcatdata_cr. | 0.206 |
| gaussian | matlab | wine-recognition | 0.205 |
| spectral | R | appendicitis | 0.204 |
| kmeans++ | shogun | house-votes-84 | 0.201 |

### D. Runs

Our analysis is based on clustering results achieved by each algorithm-toolkit configuration on each of the 162 datasets 30 different times (i.e., more than 160,000 clustering tasks; for all toolkits default settings were used).

Specifically we analyzed the distribution of accuracy (i.e., ARI when comparing the resulting clustering with ground truth) achieved in these 30 different clustering runs. Hence for our subsequent statistical analyses we have sample size $n = 30$ for one-sample tests (and $n_1 = n_2 = 30$ for two-sample tests). Note that $K$-means requires "starting points" – initial centroids. Hence in configuration *kmeans* each of the 30 runs used a different, randomly-picked, different starting points from the dataset.

## III. VARIATION ACROSS RUNS

This testing procedure is shown in Figure 2: a single toolkit is run on a single dataset multiple times (30 in our case), and a statistical analysis is performed on the resulting accuracy distribution.

***Null hypothesis:*** *accuracy does not vary across runs.*

In other words, for a certain algorithm and dataset, we set out to measure *non-determinism*. To test this hypothesis, we use Levene's test as follows: one sample contains the actual accuracy values for the 30 runs, the other sample has the same mean, size, and no variance, that is, all 30 elements are equal to the mean of the first set. We ran this on all datasets. Rejecting the null hypothesis means that accuracy varies in a statistically significant way across runs. We report results at $p < 0.05$.

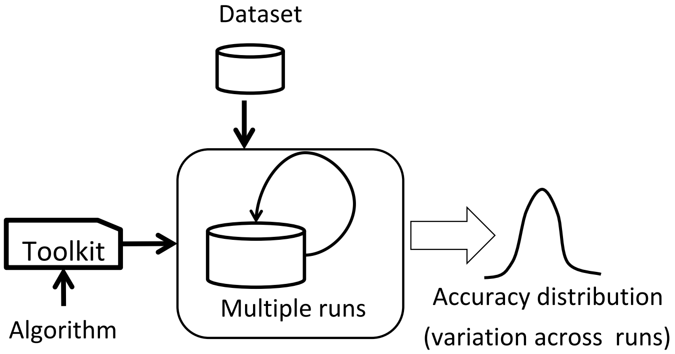

Fig. 2. Testing for variation across runs.

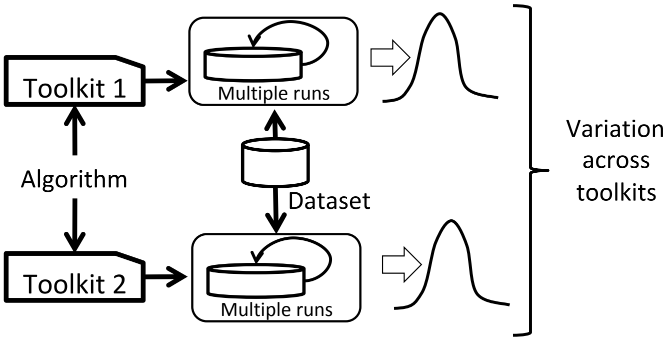

Fig. 3. Testing for variation across toolkits.

In Table III we show the number of datasets where variance is statistically significant at $p < 0.05$; recall that we have a total of 162 datasets. We observe that Spectral is the most stable nondeterministic algorithm; for Spectral, only 93–113 datasets show significant variance. Hierarchical, which should be deterministic, still has 63–71 datasets with significant variance. In contrast, *K-means, K-means++, and Gaussian Mixture, have significant variance from run to run.*

In Table IV we show how broad the accuracy range (difference between minimum accuracy and maximum accuracy) can be. The first three columns show the algorithm, toolkit and dataset. The last two columns show the minimum and maximum accuracy attained over the 30 runs. For example, Gaussian has quite a large range on some datasets: accuracy on the dataset prnn_crabs has a min-max range of *more than 1*, with one run's accuracy below 0 and another run having perfect or (close to perfect) accuracy.

In Table V we show how high the standard deviation of the accuracy can be across runs. For example, accuracy on the dataset twonorm can have a *standard deviation of 0.4*. More than a dozen other toolkit/algorithm setups have *standard deviation higher than 0.2.*

## IV. VARIATION ACROSS TOOLKITS

This testing procedure is shown in Figure 3: two toolkits implementing the same algorithms are run on the same dataset multiple times (30 in our case), and a statistical analysis is performed to compare the two accuracy distributions.

TABLE VI
MANN-WHITNEY U-TEST RESULTS FOR TOOLKITS: NUMBER OF DATASETS WITH SIGNIFICANTLY DIFFERENT ACCURACY DISTRIBUTIONS ($p < 0.05$)

| Algorithm | Toolkits | # Datasets |
|---|---|---|
| kmeans++ | sklearn vs. R | 50 |
| kmeans++ | sklearn vs. matlab | 107 |
| kmeans++ | sklearn vs. weka | 134 |
| kmeans++ | sklearn vs. mlpack | 104 |
| kmeans++ | sklearn vs. shogun | 110 |
| kmeans++ | sklearn vs. tensorflow | 109 |
| kmeans++ | R vs. matlab | 104 |
| kmeans++ | R vs. weka | 134 |
| kmeans++ | R vs. mlpack | 101 |
| kmeans++ | R vs. shogun | 108 |
| kmeans++ | R vs. tensorflow | 115 |
| kmeans++ | matlab vs. weka | 141 |
| kmeans++ | matlab vs. mlpack | 105 |
| kmeans++ | matlab vs. shogun | 120 |
| kmeans++ | matlab vs. tensorflow | 124 |
| kmeans++ | weka vs. mlpack | 96 |
| kmeans++ | weka vs. shogun | 113 |
| kmeans++ | weka vs. tensorflow | 125 |
| kmeans++ | mlpack vs. shogun | 15 |
| kmeans++ | mlpack vs. tensorflow | 57 |
| kmeans++ | shogun vs. tensorflow | 60 |
| spectral | sklearn vs. R | 109 |
| kmeans | sklearn vs. R | 41 |
| kmeans | sklearn vs. matlab | 9 |
| kmeans | sklearn vs. mlpack | 8 |
| kmeans | sklearn vs. shogun | 9 |
| kmeans | sklearn vs. tensorflow | 9 |
| kmeans | R vs. matlab | 48 |
| kmeans | R vs. mlpack | 39 |
| kmeans | R vs. shogun | 43 |
| kmeans | R vs. tensorflow | 42 |
| kmeans | matlab vs. mlpack | 2 |
| kmeans | matlab vs. shogun | 1 |
| kmeans | matlab vs. tensorflow | 0 |
| kmeans | mlpack vs. shogun | 1 |
| kmeans | mlpack vs. tensorflow | 2 |
| kmeans | shogun vs. tensorflow | 1 |
| hierarchical | sklearn vs. R | 53 |
| hierarchical | sklearn vs. matlab | 58 |
| hierarchical | R vs. matlab | 57 |
| gaussian | sklearn vs. matlab | 129 |
| gaussian | sklearn vs. weka | 146 |
| gaussian | sklearn vs. tensorflow | 120 |
| gaussian | matlab vs. weka | 146 |
| gaussian | matlab vs. tensorflow | 104 |
| gaussian | weka vs. tensorflow | 120 |
| dbscan | sklearn vs. R | 0 |
| dbscan | sklearn vs. mlpack | 7 |
| dbscan | R vs. mlpack | 7 |
| apcluster | sklearn vs. R | 40 |

***Null hypothesis:*** *For a given algorithm, accuracy does not vary across toolkits.*

To test this hypothesis, we use the Mann-Whitney U test as follows. We fix the algorithm, e.g., $K$-means++, and the dataset. Next, we compare the distributions of accuracy values pairwise, for all possible toolkits pairs, that is, if we have $N$ toolkits for a given algorithm, for a given dataset there will be $\binom{N}{2}$ Mann-Whitney U tests; hence for each algorithm there will be $162 \times \binom{N}{2}$ tests. Rejecting the null hypothesis means that accuracy varies significantly between toolkits. We report results at $p < 0.05$.

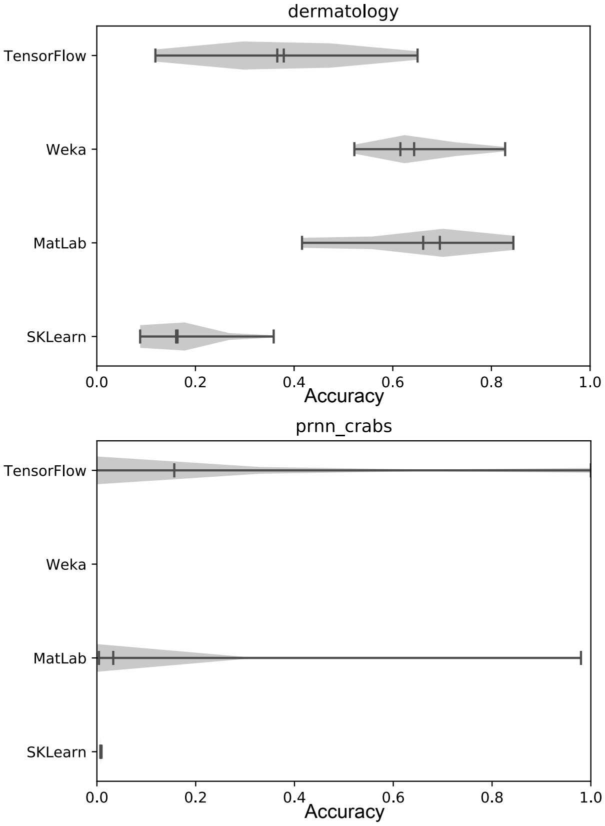

Fig. 4. EM (Gaussian Mixture): differences between toolkits on two datasets, dermatology and prnn-crabs.

In Table VI we show the number of datasets where accuracy distributions between two toolkits are statistically significant at $p < 0.05$. We observe that Gaussian Mixture and $K$-means++ induce most differences in toolkit outcomes' distributions (generally over 100 out of 162). Even for apcluster (deterministic), on 40 out of 162 datasets we found statistically significant differences between Sklearn and R.

### *A. Non-overlaps*

In Table VII we show the largest gaps between accuracy intervals, computed as follows: we find all dataset/algorithm combinations where the accuracy intervals for two toolkits, say $[ARI_{1min}, ARI_{1max}]$ and $[ARI_{2min}, ARI_{2max}]$ are non-overlapping, that is, $ARI_{1min} > ARI_{2max}$. In other words, *for any run of toolkit 1, its accuracy floor (min.) is higher than the accuracy ceiling (max.) of any run of toolkit 2*. We call that difference "gap", i.e., $gap = ARI_{1min} - ARI_{2max}$. We show the top-10 gaps in Table VII. Notice that this gap can be as large as 0.966.

We found that 1,776 such gaps exist (out of 34,987 runs of the same algorithm/dataset combinations). This is very problematic, as it shows how toolkits are not "created equal" – even after multiple runs, in 1,776 scenarios, a toolkit's *best* accuracy cannot even reach another toolkit's *worst* accuracy.

In Figure 4 we show violin plots of toolkits' accuracy distributions in the EM algorithm on two datasets. On set

TABLE VII
TOP-10 LARGEST ACCURACY GAPS BETWEEN TOOLKITS

| Algorithm | Dataset | Toolkit 1 | Floor (Min) | Toolkit 2 | Ceiling (Max) | Gap |
|---|---|---|---|---|---|---|
| gaussian | promoters | sklearn | 1 | tensorflow | 0.034 | 0.966 |
| gaussian | promoters | sklearn | 1 | tensorflow | 0.034 | 0.966 |
| spectral | promoters | R | 0.962 | sklearn | 0.001 | 0.962 |
| spectral | analcatdata_cred. | sklearn | 0.84 | R | -0.002 | 0.842 |
| kpp | breast | weka | 0.813 | sklearn | -0.003 | 0.815 |
| kpp | breast | weka | 0.813 | mlpack,matlab,R | 0.02 | 0.792 |
| kpp | breast | weka | 0.813 | tensorflow,shogun | 0.02 | 0.792 |
| gaussian | promoters | sklearn | 1 | matlab | 0.234 | 0.766 |
| kpp | promoters | tensorflow | 1 | weka | 0.406 | 0.594 |

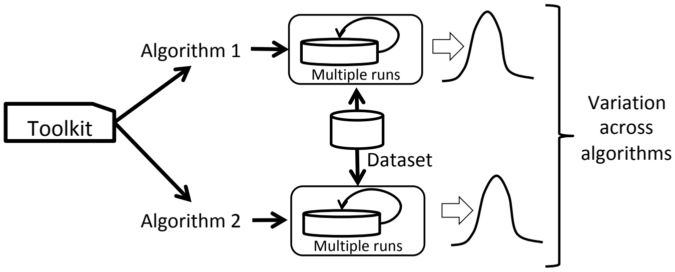

Fig. 5. Testing for variation across algorithms.

dermatology (top) note the wide ranges of TensorFlow and the gap between WEKA and Sklearn. On set prnn-crabs (bottom) note the high-end accuracy of 1 (TensorFlow, MATLAB) and the consistently low accuracy in WEKA and Sklearn.

## V. VARIATION ACROSS ALGORITHMS

This testing procedure is shown in Figure 5: implementations of two different algorithms but in the same toolkit are run on the same dataset multiple times (30 in our case), and a statistical analysis is performed to compare the two accuracy distributions.

***Null hypothesis:*** *For a given toolkit, accuracy does not vary across algorithms.*

To test this hypothesis, we again use the Mann-Whitney U test. We fix the toolkit, e.g., MATLAB, and the dataset. Next, we compare the distributions of accuracy values pairwise, for all possible algorithm pairs. Rejecting the null hypothesis implies that, for a given toolkit, algorithms' accuracy varies significantly.

In Table VIII we show the number of datasets where accuracy distributions between two algorithms are significantly different. Typically, algorithms' accuracies differ on more than 110 datasets; we expected to see such differences between algorithms. However, we did not expect wide differences when looking at the *same algorithm pairs in different toolkits*. For example, $K$-means and $K$-means++ differ on 105/101/115/120 datasets in Sklearn/R/MATLAB/WEKA but only on 27 datasets in MLpack and only 31 datasets in Shogun. This again shows that toolkits are not interchangeable (though users might expect them to be).

## VI. TOOLKIT DISAGREEMENT

We next set out to study whether toolkits "agree" or "disagree" on those points that are misclassified w.r.t. ground truth. Specifically, we are interested in those cases where two toolkits have relatively high accuracy w.r.t. ground truth, but there are large disagreements between the toolkits on the remaining, or misclassified points (i.e., where toolkits' clustering differs from ground truth).

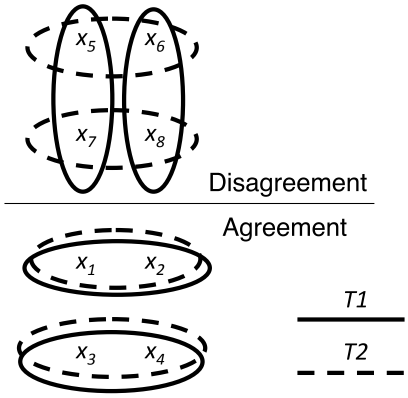

Fig. 6. Toolkit disagreement.

We illustrate this in Figure 6. Assuming two toolkits $T1$ and $T2$, their clustering of $x_1, x_2, x_3, x_4$ (on the bottom) is in agreement, and let us assume this clustering agrees with ground truth as well. We want to measure the disagreement on the remaining points $x_5, x_6, x_7, x_8$ (on top).

Intuitively, datasets that induce this disagreement between $T1$ and $T2$ on the top points manage to expose differences in toolkit implementations "at the margin"; since agreement with ground truth is high, users might expect the toolkits will be in agreement on the reminaing points, too.

Let $ARI_{T1G}$ and $ARI_{T2G}$ be the accuracy of two different toolkits on the same algorithm and same dataset. Let $ARI_{T1T2}$ be the ARI when comparing the two clusterings (rather than with ground truth). There were 14,831 $ARI_{T1T2}$ comparisons. Out of these, we found 928 cases where:

$$ARI_{T1G} > ARI_{T1T2} \wedge ARI_{T2G} > ARI_{T1T2}$$

That is, there were *928 cases where toolkits' clusterings disagree with each other more than they disagree with ground*

TABLE VIII
MANN-WHITNEY U-TEST RESULTS FOR ALGORITHMS: NUMBER OF DATASETS WITH SIGNIFICANTLY DIFFERENT ACCURACY DISTRIBUTIONS ($p < 0.05$)

| Toolkit | Algorithms | # Datasets |
|---|---|---|
| sklearn | kmeans vs. kmeans++ | 105 |
| sklearn | kmeans vs. gaussian | 123 |
| sklearn | kmeans vs. hierarchical | 134 |
| sklearn | kmeans vs. spectral | 112 |
| sklearn | kmeans vs. dbscan | 150 |
| sklearn | kmeans vs. apcluster | 115 |
| sklearn | kmeans++ vs. gaussian | 132 |
| sklearn | kmeans++ vs. hierarchical | 153 |
| sklearn | kmeans++ vs. spectral | 109 |
| sklearn | kmeans++ vs. dbscan | 155 |
| sklearn | kmeans++ vs. apcluster | 117 |
| sklearn | gaussian vs. hierarchical | 145 |
| sklearn | gaussian vs. spectral | 108 |
| sklearn | gaussian vs. dbscan | 150 |
| sklearn | gaussian vs. apcluster | 113 |
| sklearn | hierarchical vs. spectral | 120 |
| sklearn | hierarchical vs. dbscan | 155 |
| sklearn | hierarchical vs. apcluster | 122 |
| sklearn | spectral vs. dbscan | 122 |
| sklearn | spectral vs. apcluster | 115 |
| sklearn | dbscan vs. apcluster | 117 |
| shogun | kmeans vs. kmeans++ | 31 |
| R | kmeans vs. kmeans++ | 101 |
| R | kmeans vs. hierarchical | 139 |
| R | kmeans vs. spectral | 94 |
| R | kmeans vs. dbscan | 149 |
| R | kmeans vs. apcluster | 117 |
| R | kmeans++ vs. hierarchical | 150 |
| R | kmeans++ vs. spectral | 97 |
| R | kmeans++ vs. dbscan | 157 |
| R | kmeans++ vs. apcluster | 123 |
| R | hierarchical vs. spectral | 113 |
| R | hierarchical vs. dbscan | 154 |
| R | hierarchical vs. apcluster | 123 |
| R | spectral vs. dbscan | 115 |
| R | spectral vs. apcluster | 122 |
| R | dbscan vs. apcluster | 123 |
| tensorflow | kmeans vs. kmeans++ | 74 |
| tensorflow | kmeans vs. gaussian | 117 |
| tensorflow | kmeans++ vs. gaussian | 121 |
| matlab | kmeans vs. kmeans++ | 115 |
| matlab | kmeans vs. gaussian | 135 |
| matlab | kmeans vs. hierarchical | 141 |
| matlab | kmeans++ vs. gaussian | 146 |
| matlab | kmeans++ vs. hierarchical | 155 |
| matlab | gaussian vs. hierarchical | 146 |
| mlpack | kmeans vs. kmeans++ | 27 |
| mlpack | kmeans vs. dbscan | 135 |
| mlpack | kmeans++ vs. dbscan | 130 |
| weka | kmeans++ vs. gaussian | 120 |

*truth* – in other words, toolkits disagree strongly on those points that are not clustered perfectly.

In Table IX we show the top-10 such disagreements, excluding the trivial cases where one toolkit's accuracy is 1. These datasets are particularly important as they manage to "drive wedges" between toolkits; this has many applications, from differential toolkit testing to constructing adversarial datasets.

## VII. RELATED WORK

There is a surprising scarcity of studies measuring clustering accuracy/outcome across toolkits and across same-toolkit runs.

Fränti [17] has compared performance on clustering basic benchmark, and measure performance on four factors: overlap of clusters, number of clusters, dimensionality and unbalance of cluster sizes. However, they only consider synthesis data and their datasets have simple structures. Our work is based on PMLB which includes mainly real-world datasets allowed for comparing ML methods comprehensively.

The study closest to us in breadth of algorithm/toolkit combinations is Kriegel et al.'s [18]. They have also pointed out the peril of assuming that "toolkits don't matter": an algorithm's implementation is *not* standardized across all toolkits. They have compared several algorithm and implementations; algorithms include $K$-means, $K$-means++, DBSCAN; toolkits include WEKA, SKlearn and R, Shogun, ELKI, Julia, etc. However, Kriegel et al. have a narrower benchmark set: a single dataset of 500k Twitter locations, and subsets thereof. Their goal is different: they measure runtime efficiency, and show orders-of-magnitude differences across toolkits for the same algorithm and same input dataset.

Hamerly [19] has proposed a new algorithm for accelerating K-means, and performed an evaluation on efficiency similar to Kriegel et al.'s (time and memory). Our focus is on accuracy rather than efficiency.

Chen et al. [20] have compared four clustering algorithms – hierarchical clustering, $K$-means, Self-organizing Map (SOM) and Partitioning around Medoids (PAM) on a single dataset, mouse genomic data. Unlike us, they varied the $K$, whereas we used the ground truth's $K$. Our focus is different: varying runs of the same algorithm, and a breadth of datasets. Abu [21] has compared four clustering algorithms – $K$-means, hierarchical, SOM, and Expectation Maximization (EM), each implemented in two toolkits LNKnet and Cluster/TreeView; they used a single 600-instance dataset, and compared performance/accuracy on this dataset, and a 200-instance subset thereof. Our setup is substantially larger and our focus is substantially broader.

Clustering stability has been defined by Tilman et al. [22] as "solutions [that] are similar for two different data sets that have been generated by the same (probabilistic) source". Our definition of stability is different: similarity of solutions on the same dataset, but produced by two different runs.

## VIII. CONCLUSIONS

We present the first approach for testing clustering implementations via rigorous statistical analysis. We demonstrate our approach via statistical analysis of clustering outcomes across multiple runs, toolkits and algorithms. We found statistically significant variations across all these dimensions, which might violate users' determinism and invariance assumptions. Our results point out the need for improving the correctness and determinism of clustering implementations.

## ACKNOWLEDGMENTS

This material is based upon work supported by the NSF Grant No. CCF-1629186. Research was sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-13-2-0045 (ARL

TABLE IX
TOP-10 LARGEST DISAGREEMENTS BETWEEN TOOLKITS YET HAVING HIGH AGREEMENT WITH GROUND TRUTH

| **Algorithm** | **Dataset** | **Toolkit1** | | **Toolkit2** | | |
|---|---|---|---|---|---|---|
| | | | $ARI_{T1G}$ | | $ARI_{T2G}$ | $ARI_{T1T2}$ |
| spectral | promoters | sklearnfast | 0.889 | R | 0.962 | 0.853 |
| gaussian | iris | weka | 0.759 | sklearn | 0.904 | 0.693 |
| gaussian | wine-recognition | weka | 0.915 | sklearn | 0.607 | 0.568 |
| gaussian | analcatdata_authorship | weka | 0.951 | sklearn | 0.740 | 0.719 |
| gaussian | wine-recognition | sklearn | 0.607 | matlab | 0.724 | 0.469 |
| spectral | breast-w | sklearnfast | 0.809 | R | 0.552 | 0.477 |
| gaussian | wine-recognition | weka | 0.915 | matlab | 0.724 | 0.718 |
| gaussian | iris | sklearn | 0.904 | matlab | 0.560 | 0.550 |
| gaussian | iris | tensorflow | 0.562 | sklearn | 0.904 | 0.555 |
| gaussian | texture | tensorflow | 0.694 | sklearn | 0.742 | 0.614 |
| gaussian | dermatology | weka | 0.615 | matlab | 0.695 | 0.519 |
| kpp | analcatdata_authorship | weka | 0.777 | shogun | 0.718 | 0.700 |

Cyber Security CRA). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

## REFERENCES

[1] D. Bell, "5 great ai-powered home devices that will improve your life today," https://www.t3.com/features/5-great-ai-powered-home-devices-that-will-improve-your-life-today.

[2] Nvidia, "World's first functionally safe ai self-driving platform," https://www.nvidia.com/en-us/self-driving-cars/drive-platform/.

[3] PAT RESEARCH, "Top 15 artificial intelligence platforms in 2018," 2018, https://www.predictiveanalyticstoday.com/artificial-intelligence-platforms/.

[4] "Cran task view: Cluster analysis & finite mixture models," November 2018, https://cran.r-project.org/web/views/Cluster.html.

[5] "Matlab file exchange:clustering," November 2018, https://www.mathworks.com/matlabcentral/fileexchange/?term=clustering+product%3A%22MATLAB%22&utf8=%E2%9C%93.

[6] "Mathworks fast facts," https://www.mathworks.com/company/aboutus.html.

[7] M. Hornick, "Oracle r technologies overview," https://www.oracle.com/assets/media/oraclertechnologies-2188877.pdf.

[8] Janakiram MSV, "Why do developers find it hard to learn machine learning?" 2017, https://www.forbes.com/sites/janakirammsv/2018/01/01/why-do-developers-find-it-hard-to-learn-machine-learning/.

[9] Carlton E. Sapp, "Gartner: Preparing and architecting for machine learning," 2017, https://www.gartner.com/binaries/content/assets/events/keywords/catalyst/catus8/preparing_and_architecting_for_machine_learning.pdf.

[10] N. M. do Nascimento, C. Lucena, P. S. C. Alencar, and D. D. Cowan, "Software engineers vs. machine learning algorithms: An empirical study assessing performance and reuse tasks," *CoRR*, vol. abs/1802.01096, 2018.

[11] L. Hubert and P. Arabie, "Comparing partitions," vol. 2, pp. 193–218, 02 1985.

[12] D. Dheeru and E. Karra Taniskidou, "UCI machine learning repository," 2017. [Online]. Available: http://archive.ics.uci.edu/ml

[13] D. Steinley, "Properties of the hubert-arable adjusted rand index." *Psychological methods*, vol. 9, no. 3, p. 386, 2004.

[14] G. W. Milligan and M. C. Cooper, "A study of the comparability of external criteria for hierarchical cluster analysis," *Multivariate Behavioral Research*, vol. 21, no. 4, pp. 441–458, 1986.

[15] N. X. Vinh, J. Epps, and J. Bailey, "Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance," *JMLR*, vol. 11, no. Oct, pp. 2837–2854, 2010.

[16] R. S. Olson, W. La Cava, P. Orzechowski, R. J. Urbanowicz, and J. H. Moore, "Pmlb: a large benchmark suite for machine learning evaluation and comparison," *BioData Mining*, vol. 10, no. 1, p. 36, Dec 2017.

[17] P. Fränti and S. Sieranoja, "K-means properties on six clustering benchmark datasets," *Applied Intelligence*, vol. 48, no. 12, pp. 4743–4759, Dec 2018. [Online]. Available: https://doi.org/10.1007/s10489-018-1238-7

[18] H.-P. Kriegel, E. Schubert, and A. Zimek, "The (black) art of runtime evaluation: Are we comparing algorithms or implementations?" *Knowl. Inf. Syst.*, vol. 52, no. 2, pp. 341–378, Aug. 2017.

[19] G. Hamerly, *Making k-means even faster*, pp. 130–140. [Online]. Available: https://epubs.siam.org/doi/abs/10.1137/1.9781611972801.12

[20] G. Chen, S. A. Jaradat, N. Banerjee, T. S. Tanaka, M. S. H. Ko, and M. Q. Zhang, "Evaluation and comparison of clustering algorithms in analyzing es cell gene expression data," *Stat. Sinica*, pp. 241–262, 2002.

[21] O. Abu Abbas, "Comparison between data clustering algorithm," *Int. Arab Journal of Information Technology*, vol. 5, no. 3, 2008.

[22] T. Lange, V. Roth, M. L. Braun, and J. M. Buhmann, "Stability-based validation of clustering solutions," *Neural Computation*, vol. 16, no. 6, pp. 1299–1323, 2004.