algorithms

MDPI

Article

Agglomerative Clustering with Threshold Optimization via Extreme Value Theory

Chunchun Li [1], Manuel Günther [2], Akshay Raj Dhamija [1,†], Steve Cruz [1,†], Mohsen Jafarzadeh [1,†], Touqeer Ahmad [1,†] and Terrance E. Boult [1,*]

[1] Department of Computer Science, University of Colorado Colorado Springs, 1420 Austin Bluffs Pkwy, Colorado Springs, CO 80918, USA; cli@uccs.edu (C.L.); adhamija@uccs.edu (A.R.D.); scruz@uccs.edu (S.C.); mjafarzadeh@vast.uccs.edu (M.J.); touqeer@vast.uccs.edu (T.A.)
[2] Department of Informatics, University of Zurich, Andreasstrasse 15, 8050 Zurich, Switzerland; guenther@ifi.uzh.ch
* Correspondence: tboult@vast.uccs.edu; Tel.: +1-719-255-3510
† These authors contributed equally to this work.

Abstract: Clustering is a critical part of many tasks and, in most applications, the number of clusters in the data are unknown and must be estimated. This paper presents an Extreme Value Theory-based approach to threshold selection for clustering, proving that the "correct" linkage distances must follow a Weibull distribution for smooth feature spaces. Deep networks and their associated deep features have transformed many aspects of learning, and this paper shows they are consistent with our extreme-linkage theory and provide Unreasonable Clusterability. We show how our novel threshold selection can be applied to both classic agglomerative clustering and the more recent FINCH (First Integer Neighbor Clustering Hierarchy) algorithm. Our evaluation utilizes over a dozen different large-scale vision datasets/subsets, including multiple face-clustering datasets and ImageNet for both in-domain and, more importantly, out-of-domain object clustering. Across multiple deep features clustering tasks with very different characteristics, our novel automated threshold selection performs well, often outperforming state-of-the-art clustering techniques even when they select parameters on the test set.

Keywords: clustering; machine learning; unsupervised learning; deep learning; extreme value theory

Citation: Li, C.; Günther, M.; Dhamija, A.R.; Cruz, S.; Jafarzadeh, M.; Ahmad, T.; Boult, T.E. Agglomerative Clustering with Threshold Optimization via Extreme Value Theory. *Algorithms* **2022**, *15*, 170. https://doi.org/10.3390/a15050170

Academic Editors: Szymon Łukasik and Frank Werner

Received: 15 April 2022
Accepted: 17 May 2022
Published: 20 May 2022

1. Introduction

Clustering in computer vision has had impacts in a broad range of areas such as face processing, motion analysis, domain transfer, and learning representation [1–11]. Such computer vision applications utilize clustering as a non-interactive module in larger vision systems. Unfortunately, all existing clustering algorithms require users to specify parameters, e.g., the desired number of clusters or a distance threshold [12], which requires some kind of interaction with the system.

The holy grail for clustering is an accurate, fully-automatic, and parameter-free algorithm, which has been elusive since researchers started looking for it in the 1930s [13]. In recent work [4], the First Integer Neighbor Clustering Hierarchy (FINCH) clustering algorithm FINCH was developed. FINCH uses rounds of a simple nearest-neighbor process that at least reduces the problem to selecting an appropriate partition from a set of computed ones, and the authors declared this algorithm to be "parameter-free".

We switched to FINCH in various projects. The results of FINCH made us ask: *Is there a theoretical reason to believe that one or more thresholds exist to determine when clusters should merge?* In general, this seems to be an ill-posed question since thresholds depend on the actual clustering task at hand and sometimes need to be optimized to achieve that task. However, as we show in this paper, under some general conditions, the answer is: *Yes, such a threshold exists.* The first important novelty of this paper is *a proof that the*

distribution of proper cluster linkages follows a Weibull extreme value distribution. Using that theory, we present an initial process for a data-driven automatic selection of maximum inter-cluster spacing in agglomerative clustering. Given that a threshold exists, we expect future research to provide even better approximations.

We contend that clustering is inherently ill-defined without meaningful features and similarity or distance measures between them. When dealing with a problem that utilizes clustering, the number of potential clusters is exponential, and the clusters may only have meaning when a measure of similarity of the features is given. For example, given many face images, it needs to be defined if clustering should be based on subject identities, personal attributes (gender, hair color), style of the photos, or location. For the same images, what kinds of clusters we expect depends on the features and not the input images.

We conjecture that prior work has not produced successful fully-automatic clustering algorithms because similarity, or distance, measures for hand-made features are rarely uniform or consistent. Over the past decade, representation and transfer learning, metric learning, and deep features have shown powerful aspects that address vision problems with a significantly better recognition ability and an almost unreasonable ability to define similarity measures that are perceptually meaningful [14]. In our Unreasonable Clusterability of Deep Features Conjecture in Section 3, we hypothesize that consistent clustering of deep features is unreasonably easy because they are obtained from training, which defines semantics and produces relatively uniform spaces where location and distance have semantic meaning. In this paper, we show that deep features serendipitously have the behavior of producing a representation that supports fully-automatic cluster threshold selection as the approximate uniformity of the space turns out to be one of the general conditions critical to our extreme-linkage theorem.

To show the effectiveness of the Extreme Value Theory-based threshold selection, we evaluate multiple real clustering problems, using both classical Agglomerative Hierarchical Clustering (AHC), and we use our method to select among FINCH partitions. When our Provable Extreme-value Agglomerative Clustering-threshold Estimation (PEACE) is applied to a classical Agglomerative Clustering of Hierarchies, we call the resulting algorithm PEACH. When it is applied to FINCH, we call the result Partition Estimation Technique for FINCH (PET-FINCH).

Evaluation of clustering requires carefully designed experiments to ensure we measure what matters. Many classic clustering papers test on toy datasets with hand-defined features, which generally fail to have the necessary properties to apply our extreme-linkage and unreasonable clusterability theorems. Such toy datasets also fail to match real problems. Since clustering is widely used in vision applications, we prefer to evaluate actual vision problems. This paper is not focused on new clustering techniques, rather, we are focused on obtaining better results when a deep-feature system needs clustering, i.e., we make state-of-the-art systems that use clustering even better. To show this, we cover a wide range of problems where, originally, clustering techniques with hand-tuned parameters were applied, showing that our methods improve results.

One vision problem that formally defines seven different practical clustering protocols of varying sizes is the IARPA (Intelligence Advanced Research Projects Activity) Janus benchmark [15]. We use these as part of our evaluation. Another common use for clustering of unknown classes occurs in life-long, continual, and open-world learning. To address that application, we define a new clustering evaluation protocol that seeks to model a more realistic classification and clustering scenario where the data to be clustered include a mixture of in-distribution and out-of-distribution samples. This better aligns with many real problems since users do not know which, or how many, clusters will occur. We analyze and report on each of in-, out-, and mixed-distributions to showcase the difference. Another related common vision problem using clustering is transfer clustering [5], and we use the provided default evaluation protocols here as well. Finally, to show that the PEACH approach does not depend on pre-defined deep features, we evaluate PEACH in deep clustering for unsupervised feature learning and deep features classification where such

problems do not have a predetermined number of clusters. While prior work [10,11] used hand-tuned clustering parameters, we show that, using our Extreme Value Theory-based threshold selections, we can drop PEACH into these systems and significantly improve performance.

Our contributions include:

1. We prove that, with modest assumptions, ideal and actual nearest-neighbor linkages follow Extreme Value Theory (EVT) and are governed by a Weibull distribution for most datasets.
2. We show that the modest assumptions are generally satisfied by features extracted from deep learning systems.
3. We provide algorithms to efficiently approximate linkage thresholds that are robust to moderate levels of outliers and combine them with agglomerative clustering to introduce a Provable Extreme-value Agglomerative Clustering-threshold Estimation (PEACE).
4. We combine EVT-based threshold selection with the state-of-the-art algorithm FINCH to build PET-FINCH. Over multiple experiments, we show that PET-FINCH generally selects the best partition.
5. Using pre-computed deep features, we show that PEACH provides a statistically significant improvement over the original clustering approach on large face datasets: IARAPA Janus Benchmark B (IJB-B) and Labeled Faces in the Wild (LFW).
6. We introduce a real-world clustering protocol on the ImageNet dataset that is useful for applications of data labeling and incremental learning, and we provide a baseline clustering.
7. We demonstrate how to incorporate PEACH on transfer clustering to provide the new state of the art on those problems.
8. We show that PEACH outperforms the state of the art for Top-1 accuracy of deep clustering on handwritten digits (MNIST).

2. Related Work

This paper is not about a new clustering algorithm but rather about an automated parameter selection for existing algorithms. Thus, most papers are only weakly related and not discussed in detail, and we only review those used in the experiments. Tables in the experiments also include a comparison with a wider range of algorithms that do not perform as well.

Clustering analysis research dates back to the 1930s [13]. In addition, hundreds of survey papers exist on general clustering. However, to date, no general algorithm has been fully automated [4], while FINCH is at least able to provide good default parameters. Another automatically liked clustering approach named Swarm intelligence [16] still requires parameters and the abundance values of a dataset. Some researchers developed autonomous parameter selection for image segmentation techniques, and argue that they can be viewed as a type of clustering [17–21]. Image segmentation is a very special problem with added constraints, and the approaches have failed to generalize to general clustering problems even in other vision applications.

Clustering algorithms are generally divided into centroid-based, density-based, spectral-based, and connectivity-based algorithms. The widely used K-Means [22] is an iterative centroid-based algorithm. DBSCAN [23] is a common clustering algorithm using point densities. MeanShift [24] is another example of density-based algorithms. Spectral Clustering [25] is a clustering algorithm that uses the eigenvalues (spectrum) of the similarity matrix for the given samples. Various algorithms have explored combinations of these approaches, such as BI-Clustering [26], Affinity Propagation [27], Ball Cluster Learning (BCL) [3], BIRCH [28] OPTICS [29], and Power Iteration Clustering (PIC) [30]. Most of these are not included in our experiments, as on our data, they are significantly worse than those presented.

Another widely applied clustering technique is agglomerative clustering [31], a connectivity-based hierarchical clustering approach that forms the basis of various other approaches. Agglomerative Hierarchical Clustering (AHC) [31] builds a hierarchical tree using a bottom-up strategy, successively merging nearest clusters, updating distance, and continuing until all clusters have been merged. The AHC tree can even be cut to obtain any number of clusters. Several different definitions of nearest clusters may be employed; for example, SciPy [32] implements single linkage, complete linkage, average linkage, and centroid linkage. Recent work defines new objectives and near-optimal complexity algorithms [33,34].

Currently, no effective, general-purpose, parameter-free, and fully autonomous clustering algorithm exists, although some techniques do take important steps in that direction. The density-based OPTICS [29] and the related HDBSCAN [35], variants of DBSCAN [23], find the core samples of high density and expand clusters from them to build a cluster hierarchy for variable neighborhood radii. They still require critical parameters, but various implementations provide non-data-driven defaults for such parameters used in our experimental comparisons.

Swarm [36] is a single-linkage clustering method with semi-automatic parameter selection that has some some superficial similarities with other clustering methods. Swarm's novelty is its iterative growth process and the use of sequence abundance amplicons to delineate clusters. Swarm presents an internal structure where the most abundant amplicon usually occupies a central position and is surrounded by less abundant amplicons. The way Swarm explores the amplicon-space naturally produces a graph representation of the clusters, in the form of a star-shaped minimum spanning tree. Swarm properly delineates large clusters (representing a high recall), and can distinguish clusters with as little as two differences between their centers (high precision). Swarm-inspired clustering can be used in many clustering techniques such as K-Means and AHC [16], and the authors observe that the use of the K-Means and the Fuzzy C-Means [37] in clustering problems are of great importance in the application of the swarm intelligence algorithms. They are largely used for comparing the performance of Swarm-based approaches and creating hybrid proposals. However, Swarm cannot work without abundance values of the dataset, where the user needs to provide both a local parameter and abundance values before using the algorithm—it is not fully-automated.

FINCH [4] is an agglomerative clustering method that only uses the first neighbor of each sample to discover large chains and groups of data. While FINCH may be a parameter-free clustering algorithm, it is not autonomous as it outputs several possible partitions of the data. The user must still select the appropriate partition. The experiments in [4] show the best partition using ground-truth information, which yields an over-optimistic view on the performance of the algorithm in fully-automated systems and the idea of automatic parameter selection.

Evaluation Metrics

While generally, clustering has no explicit way to be evaluated, some evaluation protocols provide the ground truth for the clusters, e.g., when clusters should be formed for separate classes in a classification task. Here, we report the metrics that are often applied in these cases:

Normalized Mutual Information (NMI) Ref. [38] makes it possible to compare the performance of two algorithms that produce different numbers of clusters. Particularly, we compare the obtained clusters C with ground-truth classes G. The NMI provides values between 0 (no overlap between clusters) and 1 (perfect clustering).

Let C be the clustering result for samples X, with C_i representing a specific cluster and G be the ground-truth set, with G_j representing all samples belonging to class j. Then, the NMI is computed as [39]:

$$\mathrm{NMI}(C,G) = \frac{\sum_{C_i \in C} \sum_{G_j \in G} |C_i \cap G_j| \log \frac{|C_i \cap G_j||X|}{|C_i||G_j|}}{\sqrt{\sum_{C_i \in C} |C_i| \log \frac{|C_i|}{|X|} + \sum_{G_j \in G} |G_j| \log \frac{|G_j|}{|X|}}} \quad (1)$$

Adjusted Rand Index (ARI) [40–42] is a variant of the Rand Index (RI) that is corrected for chance. Such a correction establishes a baseline by using the expected similarity of all pairwise comparisons between clustering results specified by a random model [43]. To understand ARI, assume G is the ground-truth and C the clustering result. Then, a, b, and c are defined as:

(a) the number of pairs of samples that are in the same set in G and in the same set in C.
(b) the number of pairs of samples that are in different sets in G and in different sets in C.
(c) the total number of possible pairs in the dataset (without ordering).

The unadjusted RI can be described as:

$$RI = \frac{a+b}{c}. \quad (2)$$

To guarantee that random label assignments obtain a value close to zero, the expected RI of random clustering results is discounted by defining the ARI:

$$ARI = \frac{RI - E[RI]}{\max(RI) - E[RI]} \quad (3)$$

One severe drawback of ARI is that it is strongly impacted by outliers; thus, it may not have a good reflection on the quality of the clusters. In some cases, ARI scores are increased with a decreasing number of detected clusters. Nevertheless, we provide ARI measures so other researchers can compare with our results in the future.

B-Cubed (B^3) F-measure F_b [44] is the harmonic mean of B^3 precision P_b and B^3 recall R_b, and it can be used to evaluate clustering algorithms that yield a different number of clusters. B^3 precision P_b calculates the fraction of points in the same cluster that belongs to the same class. B^3 recall R_b calculates the fraction of points in the same class that is assigned to the same cluster. To compute P_b, R_b, and F_b, a correctness indicator $B(x_i, x_j)$ is defined as 1 when a pair of samples of the same class are in the same cluster, or when a pair of samples of different classes are in different clusters and 0 otherwise. Using this indicator, precision, recall, and F-measure are evaluated as:

$$P_b = \frac{1}{|X|} \sum_{x_i \in X} \sum_{x_j \in C(x_i)} \frac{B(x_i,x_j)}{|C(x_i)|} \quad R_b = \frac{1}{|X|} \sum_{x_i \in X} \sum_{x_j \in G(x_i)} \frac{B(x_i,x_j)}{|G(x_i)|} \quad F_b = \frac{2P_bR_b}{P_b+R_b} \quad (4)$$

where $C(x_i)$ is the cluster that contains sample x_i and $G(x_i)$ is the set of all samples that belong to the same class as x_i.

3. Threshold Estimation Using Extreme Value Theory

The idea that a single threshold can be determined for agglomerative clustering may be rather unexpected and unreasonable. However, there are many properties of deep networks that were initially unexpected and unreasonable to expect [14,45–47]. There is much we do not understand about deep networks and deep features; in this paper, we will add their unreasonable ability to support good clustering threshold determinations to that list.

In a deep feature space, the feature mapping was trained to reduce a loss that results in points with similar semantics, as implicitly defined by the loss function, and which will be mapped closer together. It also produces a continuous space with a data-dependent but normalized dimensional value. If we consider non-degenerate points, i.e., a point that should be clustered with at least one other point, then its nearest neighbors are, with high probability, points with which it should cluster. This observation explains part of the power of FINCH [4], which clusters the first nearest neighbors.

In our analysis, we use approximate (first) nearest neighbors a subset of points that should cluster. The set of approximate nearest neighbors form a distribution from which we will estimate our threshold for the clustering algorithm such as AHC. We note that, since the approach can use Approximate Nearest-Neighbors (ANN), our complexity can be sub-quadratic since first ANN pairs can be found in $O(N \log N)$ time [48], while true nearest neighbors take $O(N^2)$.

Let us formalize our model and then state our core theorems. Let $s_{n,m} \in \mathbb{R}^S, 1 \leq m \leq M, 1 \leq n \leq N_m$; be a set of N_m points drawn from M classes, which represent the ideal cluster label for each input. Let mapping function $f : \mathbb{R}^S \mapsto \mathbb{R}^D$ be a continuous well-behaved feature extraction function. Let $p_i = f(s_{n,m})$ be a mapped sample (Since we do not use the label m during clustering, we here remove class information from the sample index.). Let $\mathcal{P}$ be the set of all mapped points p_i. Allowing for but not assuming clustering in rounds (as in FINCH), we let let $C_j^{(r)} \in C^{(r)}$ be a cluster formed from at least one point in round r, with $C^{(r)}$ being the set of all clusters after round r. Let $d_{ij} = d(p_i, p_j)$ be the dissimilarity or distance between points p_i and p_j. Let $g_{ij}^{(r)} = d(C_i^{(r)}, C_j^{(r)})$ be the dissimilarity or distance gap between clusters $C_i^{(r)}$ and $C_j^{(r)}$ at round r, e.g., distance between their centroids or other cluster distances (linkages) as nearest distances, maximum distances, and average distances. Let the set $\ell_i^{(r)}$ be the set of link distances of points added in round r. Let $\eta_i^{(r)}$ be the set of ground-truth correct link distances of connections that ideally should have been added between ANNs in round r; note this is known only to an oracle.

In agglomerative clustering, we initially consider each point p_i a cluster of size one, and then we merge p_i and p_j into a cluster if their gap distance d_{ij} is sufficiently small. In later rounds, these clusters get further merged if their link distance $g_{ij}^{(r)} \leq \tau$.

Theorem 1 (Extreme-Linkage Theorem)**.** *The distribution of the sets of both the ideal link distances $\eta_i^{(r)}$ and the observed link distance $\ell_i^{(r)}$ are governed by the Fisher–Tippet (Extreme Value) theory, and because they are sets of minima and are bounded from below, the limiting distribution is a Weibull.*

Proof. We consider $\ell_i^{(r)}$, and the proof for $\eta_i^{(r)}$ is similar. We start by noting that, at each level, we choose the (approximate) smallest link distances out of the much larger set of pairwise distances. Thus, each $g_{ij} \in \ell_i^{(r)}$ can be viewed as a minimum over a set of distances that are associated with some sampling of the distance between the initial set of points if $r = 1$ and from clusters from the prior level if $r \geq 2$. If we view the merging of the clusters in level r in decreasing order of pair distance g_{ij}, then each new linkage distance can be seen as a new minimum over an increasing set of items. We can view the set over which we consider the actual minimum as excluding a few items as ignored by the ANN algorithm. Since the Fisher–Tippet Theorem [49] applies to maxima, we perform a change of variables to use the negative of each distance and then consider the distribution of a maximum set of values $\zeta_i = -\ell_i^{(r)}$. Let ϕ be the associated distribution of the maximum of ζ_i in the Fisher–Tippet Theorem (see supplemental material based on [50–52]) and, combined with the knowledge that distances are bounded ($-\ell_i^{(r)} \leq 0$), yields that ϕ converges to a reversed Weibull, as it is the only one of the EVT distributions that are bounded. Changing the variable back yields that the distribution of link distances follows a Weibull distribution. □

Given the Extreme-Linkage Theorem to estimate a threshold, we use a three-parameter Weibull distribution:

$$\mathrm{W}(x;\mu,\sigma,\xi) = \begin{cases} \frac{\xi}{\sigma}\left(\frac{x-\mu}{\sigma}\right)^{\xi-1} e^{-\left(\frac{x-\mu}{\sigma}\right)^{\xi}} & x < \mu - \frac{\sigma}{\xi} \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

where $\mu \in \mathbb{R}$, $\sigma \in \mathbb{R}^{+}$, and $\xi \in \mathbb{R}^{-}$ are locations, scale, and shape parameters [53].

This theorem applies to any clustering algorithm using approximate nearest-neighbor merging, e.g., it applies to both FINCH [4] and any AHC algorithm. This theorem allows us to "reject" any potential linkage that is unlikely. Given the distributional parameters μ, σ, ξ, to compute the merge distance threshold τ_w that yields a given probability $0 < p < 1$ such that probability $P(d_{ij} < \tau_w) \geq p$, we use the inverse CDF to derive:

$$\tau_w = \mu + \sigma \cdot (-\ln(1-p))^{\frac{1}{\xi}} \tag{6}$$

Since the Extreme-Linkage Theorem shows that both the true linkages and the computed linkages follow Weibull distributions, we contend that selecting τ_w to reject improper linkages comes down to fitting the distribution to the set of minima, i.e., the ANN distances. Given the desired level of confidence, such as 99%, a pair merges when consistent with that distribution, we select τ_w as in (6), i.e., τ_w is considered the maximum ANN distance which should merge. Considering correct links, the associated, linked points would still be ANN for some set of points within that cluster. Thus, all valid links should be ANNs and a single τ_w computed from an initial set of ANNs is sufficient since ANN of points within a cluster is still subject to that distribution. In Figures 1 and 2, and figures in the Supplemental Material, we fit and compute τ_w to show that the performance is stable for moderate variations in τ_w.

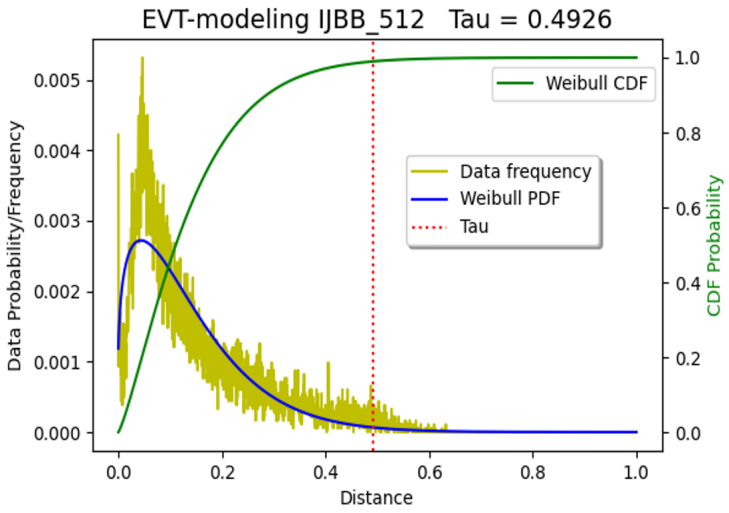
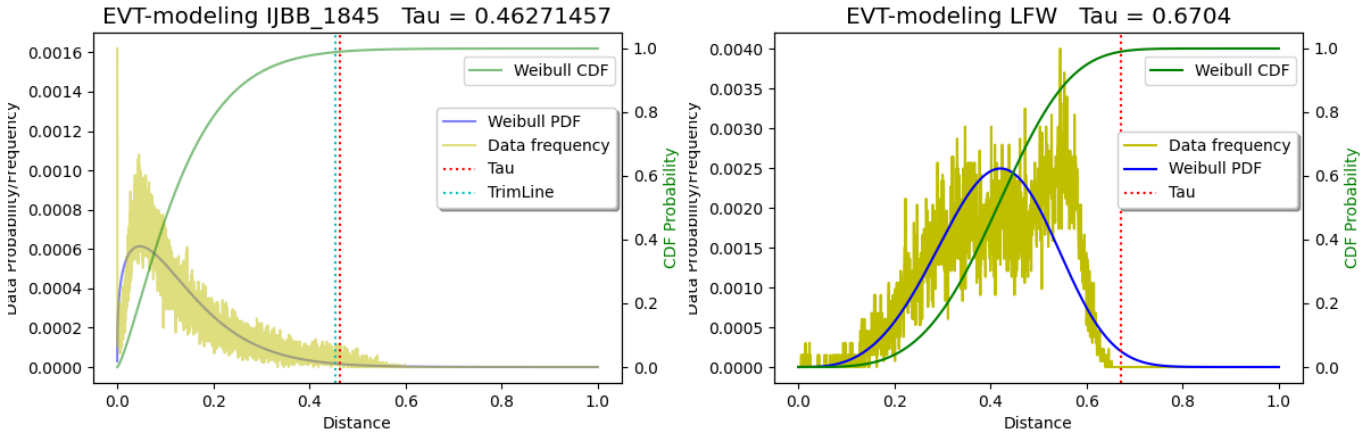


Figure 1. *EVT-based Clustering Threshold Selection.* We prove that nearest-neighbor distances and valid cluster link distances will follow an EVT distribution, and show how that leads to a good threshold selection for clustering. Yellow shows the data frequency histrogram, blue the probability density function, and green the cumulative density fit of a Weibull distribution to the approximate nearest-neighbor distance for IJB-B (protocol 512, see Section 5). While noisy, it is a good fit and supports choosing the threshold τ_r/τ_w at for the Cumulative Distribution Function (CDF) of the Weibull for 99% probability.



(a) IJB-B **(b)** LFW

Figure 2. *Weibull Modeling.* The distribution of ANN distances is proven Weibull. (**a**) shows the histogram of distances from IJB-B dataset, the resulting Weibull fit, its CDF, and the resulting τ_w. (**b**) shows the plot for LFW data, which is impacted from outliers and mode shift is used as one heuristic for outlier detection. Note that there are ANN distances above τ_w, which would be rejected as outliers and not merged.

With τ_w being defined, we can now state our main conjecture:

Conjecture 1 (Unreasonable Clusterability of Deep Features)**.** *Let mapping function $f' : \mathbb{R}^S \mapsto \mathbb{R}^{D'}$ be a function from a deep learning classifier trained to separate its training data $t_{n,m} \in \mathcal{T}$ that yields a D'-dimensional representation of an input sample. This sample then goes through either a linear or a nearest-neighbor-like classifier.*

Let $f(x) = W^\top f'(x)$ be a transformation, e.g., PCA, of the features into D-dimensional deep features associated with the input sample with the set of all such points being $\mathcal{P}$. Let τ_w be set by (6) from the Weibull fit to the $\alpha(p_i) \forall p_i \in \mathcal{P}$. If each ideal cluster has sufficiently many points, then using this τ_w as the threshold will, with high probability, produce good automatic clustering with semantics given by the function f' that underlies the deep feature space. The probability of good clustering improves with increased similarity of the training and clustering domains.

For clusters that match the network training classes, this conjecture follows from the assumption for a good classifier. However, clustering only data from training classes is pointless since we already have a classifier for that. For generalization to sufficiently close domains, the intuition of the conjecture follows from our Extreme-Linkage Theorem combined with the many known results on how well deep features support domain adaption and transfer learning, including unsupervised and universal adaption [54–59]. Proving the conjecture in these cases would require detailed formalization of the exact problem and what it means to be a sufficiently close domain, which is beyond this paper's scope.

Theory Limitations

We note the proof of the Extreme-Linkage Theorem depends on the Fisher–Tippet Theorem, which is an asymptotic theorem that requires some general smoothness assumptions for convergence—it does not apply in spaces where values occur only on a set of measure zero. The assumptions of the conjecture are sufficient to provide that smoothness, but if a high-dimensional feature space is highly non-uniform, the semantics will be dominated by the dominate directions—hence we often add PCA to normalize. We also note there is a second EVT theorem, with slightly different assumptions that leads to Generalized-Pareto-Distributions, e.g., see [60]; which EVT model is better for cluster threshold selection is left for future work.

Deep learning methods effectively develop feature spaces where distances have semantic meaning that reflect the distribution of labels used in training. Generic nearest neighbors in high dimensions are often problematic [61], but deep networks use a fully connected layer to reduce dimensionality with their linear or NN classification stage and create meaning in their feature space. Since clustering does not have supervised data to learn to reduce dimensions, in our work, we often use a PCA projection of features to reduce dimensionality as in [10].

In any fitting, there is a question of the impact of outliers. If outliers dominate the tail of the distance distribution, the resulting fit of τ_w will result in merging many points that should be outliers. With few outliers, the Weibull fit may end up with a τ_w that rejects the outliers. While the theorem and conjecture have assumptions that exclude singletons, in practice, it is hard to know when that assumption will hold. Therefore, in the next section, we develop some approaches to manage outliers and discuss parameter selection.

4. Approach

4.1. PEACE Parameter Selection

Our Provable Extreme-value Agglomerative Clustering-threshold Estimation (PEACE) uses threshold τ_w computed by fitting a Weibull. We note, however, that the fitting and threshold selection still requires a choice of parameters: amount of data to trim, tail-size, and confidence threshold. Thus, unlike FINCH, our approach is not parameter-free. However, we believe our parameters, which are about generic data quality, are more intuitive and better subject to defaults than choosing the actual number of clusters (K in K-Means), choosing between a partition of clusters (FINCH), or estimating maximum point separation within a cluster (ϵ in DBSCAN). Our free parameters for Weibull fitting are about the quality of the data. For example, in most of the experiments herein, we use (6) with a tail-size of 1% of the data, which reduces the impact of stray outliers even without trimming the tail. For confidence to select that actual threshold given the fit, we use $p = 0.99$. If one has a rough estimate of the frequency of outliers, i.e., points that should not cluster, of say $q > 1\%$, then one can trim $q\%$ of the data and use the remaining default parameters for the Weibull fitting. This type of approach is sufficient for many applications, and for most problems where one expects to cluster nearly all the data, our defaults (drop 1%, tail 1%, confidence 99%) will work. In general, slight overestimation of outliers is not a problem as results are stable for variations in τ_w; see plots in Supplemental Material.

However, we observed that some datasets have way more than 1% outliers, and searching for the amount to trim would require some feedback on outliers or fit quality. We tried basic goodness-of-fit measures, such as sample-based Kolmogorov–Smirnov (KS). Unfortunately, those measures were still too weak—with 1000 generated samples from the Weibull fit, none of the distributions in experiments failed a KS test even with the many outliers in Labeled Faces in the Wild (LFW), where we have >50% outliers. Thus, for problems where one is uncertain about the potential for outliers, we developed a heuristic approach to normalize for outliers without a prior guess on the fraction of outliers.

Recognizing that outliers also impact the raw distribution of all pair distances, we set up an equivalent relation to estimating a robust version of our threshold. If we let $d(\mathcal{S})$ be a set of pairwise distances, our equivalent relation for robust threshold estimate τ_r divided by our raw Weibull estimate τ_w should equal the ratio of the median (a robust operator on all points) divided by the outlier-sensitive maximum distance on all points:

$$\frac{\tau_r}{\tau_w} = \frac{\text{median}\{d(\mathcal{S})\}}{\max\{d(\mathcal{S})\}} \tag{7}$$

Solving that relation for τ_r gives our normalization factor. We can apply this with EVT estimates of the ANN. We normalize by ω as follows:

$$\omega = \frac{\text{median}\{d(\mathcal{S})\}}{\max\{d(\mathcal{S})\}}; \quad \tau_r = \omega \cdot \tau_w; \tag{8}$$

where τ_w employs (6) with 99% of the data and $p = 0.999$ then is scaled yield robust τ_r. While approximate nearest neighbors can be computed quickly, if $\mathcal{S}$ includes all points, the normalizations would require all pairs, which would require complexity $O(N^2)$. The normalization, however, does not really need all pairs, and $\mathcal{S}$ as a random sample of a few thousand points produces a good normalization, keeping total complexity to $O(N \log N)$. While this is just a heuristic, our experiments find it works well, often even better than our theoretically justified approach with default parameters. Figure 3 shows PEACH with its automatically selected τ_r is better than K-Means & Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH) [28] even when those algorithms optimize over the test set; and is close to the optimal value produced when searching for AHC parameters over the test set. We also show the state-of-the-art FINCH algorithm, which produces eight clusters (not all visible), and with EVT theory, PET-FINCH automatically chooses a good partition.

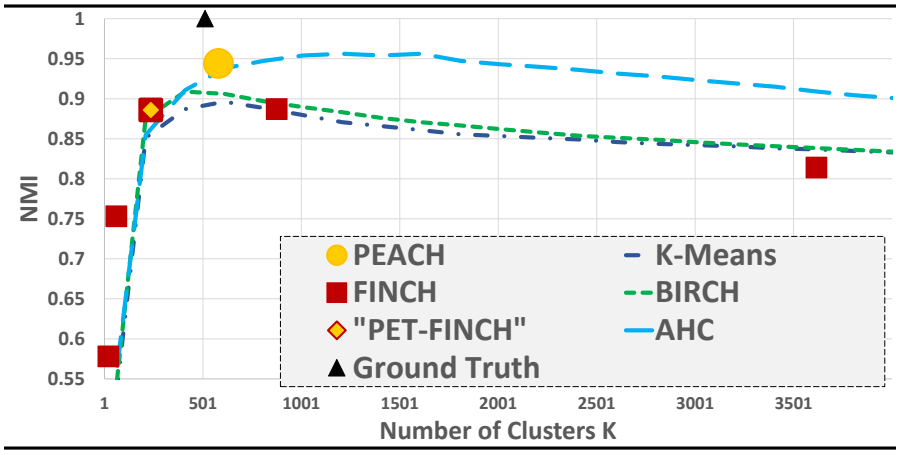


Figure 3. *EVT-based Threshold Works Well.* We show multiple clustering algorithms' NMI performance on the IJB-B 512 face dataset.

4.2. Overall System Operation

The overall system operation shown in Figure 4. The processing starts with training data with labels (that we will only use for evaluation), then we extract deep features from a deep learning system. We compute the nearest neighbors of extracted deep features. The nearest neighbors are used to generate initial pairs for PEACH, and we compute the pairwise distance $d(\mathcal{S})$. Now, we can use Weibull fitting to compute our Weibull estimated threshold τ_w as given in (6). In order to obtain out robust threshold τ_r, we estimate the median and maximum distance $d(\mathcal{S})$ of all pairs of data points.

By using $\tau \in \{\tau_w, \tau_r\}$, we decide if any set of clusters should merge. At the beginning of merging, as typical Agglomerative clustering, PEACH treats every sample as a single cluster, then we generate initial clusters from these single clusters. All paired clusters with link distance below the threshold τ merge. Next, we find the nearest cluster to each cluster and generate a merging list for this level. Unpaired clusters just pass upward unmerged until they are the closest neighbor of some cluster. We repeat finding all nearest neighbors, building a merge list, and merging until the list of clusters does not change. After merging, we assign cluster labels C_i to the grouped data.

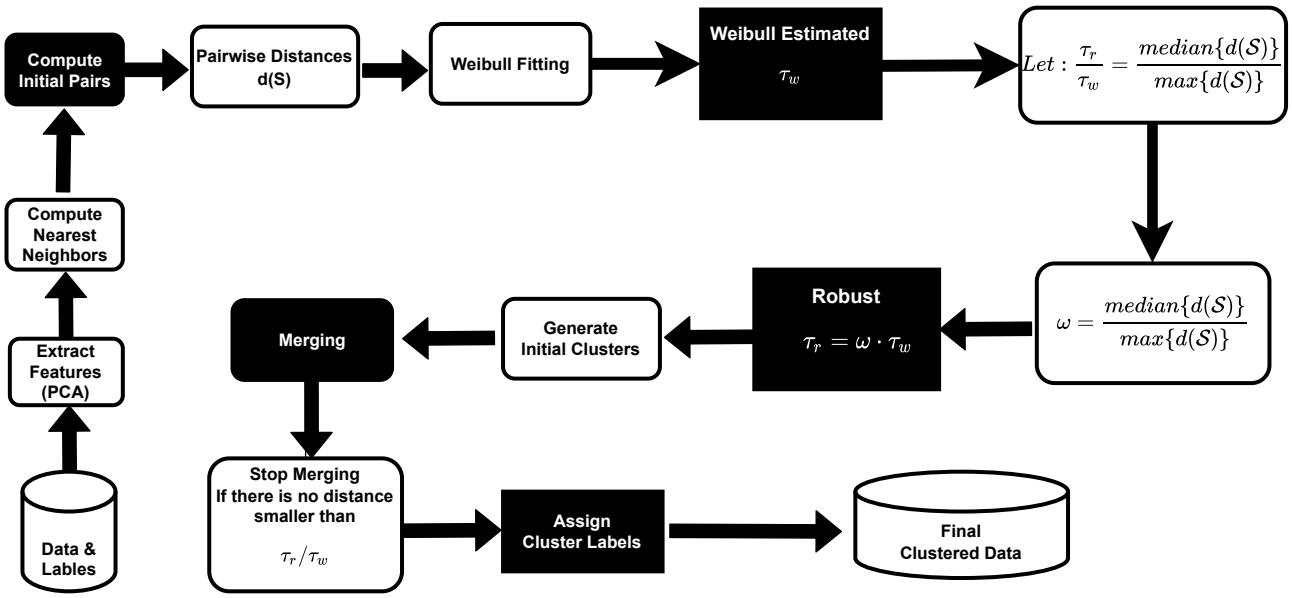


Figure 4. *Diagram of Overall System Function.* This figure shows the overall pipeline of PEACH. First, we extract deep features, then we compute the initial pairs, median{d(S)} and max{d(S)} for the EVT Estimation to compute the threshold τ_r/τ_w. Next, we use the nearest neighbors to generate initial clusters and the τ_r/τ_w will be used for merging to obtain final clustered data.

4.3. PET-FINCH

FINCH is an agglomerative clustering method that does not require an input parameter. Instead, FINCH leaves some potential results as output in order to remove an input parameter. The cost is that the user has to find a way to choose one of these partitions as their final result, which is sometimes difficult if there is no ground-truth. This could be called an extra "parameter" that users need to insert into the algorithm. Hence, while the authors of FINCH claim that the algorithm is parameter-free, it is actually not. For some types of experimental datasets, e.g., MNIST, CIFAR, and ImageNet, we already know how many classes the dataset has; however, for some real world problems, there is no way to know the number of clusters contained in the problem. Therefore, knowing how to choose the appropriate partition/result for the FINCH approach presents an issue.

In this section, we introduce the datasets and evaluation protocols of our experiments. Clustering of deep features from pre-trained classification networks has become a common practice. While there are various clustering protocols for a variety of datasets, existing clustering protocols for deep features have used only data from the same domain as training, e.g., face clustering. In many practical applications, the training domain is different from the clustering domain. Therefore, we provide a novel data-distribution view of clustering protocols and then detail our evaluation protocols.

4.4. ImageNet-Based Evaluation Protocols

In-, Out-, Mixed- Distribution Protocol: We base our new in/mixed/out-of-domain protocols on the ILSVRC partitions of 2010 and 2012 [62] from the ImageNet dataset, using the data splits of [63]. While previously these splits were used for classification, we adapt them to a novel clustering protocol. Since many available pre-trained deep networks were trained on the 1000 classes of the 2012 dataset, we use the validation set of the 2012 dataset as our in-distribution data. We consider out-of-distribution data to be the 166 classes from the 2010 dataset that have not been re-used in the 2012 dataset. Finally, the mixed-distribution data are a mix of the in- and out-of-distribution data containing images from a total of 1166 classes.

Out-of-Domain Protocol: We use a network that is pre-trained on ILSVRC 2012 training set and apply it to face identity clustering, showing that the clustering properties

of deep features transfer across domains. In particular, we use the Labeled Faces in the Wild (LFW) dataset for testing and call this protocol Out-of-domain LFW (OLFW). Labeled Faces in the Wild (LFW) [64] contains mostly frontal facial images from 5749 people, while only 1680 of them have multiple images. The remaining subjects are singletons and should not be clustered with other subjects—they produce many outliers for our theory. Since performance on OLFW is not comparable to prior work that used in-domain face features for clustering on LFW, we report those in-domain experiments in the supplemental material.

Unsupervised Network Training Protocol: The furthest out-of-domain, with the greatest uncertainty of the feature space and similarity, occurs for unsupervised network training or deep clustering. Initially, the features are random and all data are out-of-distribution. As the networks learn from the samples, they start moving toward in-distribution. However, there is no independent concept of correct classes/clusters and, thus, no practical way to estimate clustering parameters. We follow the protocol of [10,11], which requires a fully autonomous clustering subsystem, but where that past work guessed an overestimate of the number of clusters.

4.5. Face Identity Clustering Experiments

The IARPA Janus Benchmark B (IJB-B) [15] data contain images of up to 1845 subjects in different qualities and face poses, each with at least three images. IJB-B provides seven clustering protocols with an increasing number of subjects. This is one of the largest available datasets for clustering. Since deep networks used for face recognition are usually trained on large corpora of facial images disjoint from the evaluation datasets, these protocols can be regarded as an in-domain and out-of-distribution.

5. Experimental Results

We report the results of PEACH, which is fully automatic and does not use the test set to optimize parameters. Ideally, we would not need to compare against results optimized on the test set or by using ground-truth parameters. Still, to be consistent with past work, we report results of other algorithms from their papers for classic algorithms optimized over ground-truth. In particular, if they require specifying the number of clusters, e.g. K-Means, we report their performance on either the ground-truth number or the PEACH detected number of classes, whichever provides the best score. For FINCH, we report on their best partition results, despite there not being a way to choose that in practice, and note that PET-FINCH automatically selects that partition in all cases with one exception. Note that these are optimistic results for the algorithms as the performance would be worse, often much worse, if we would guess their parameters or obtain them via validation.

PEACH can be applied using many different distance metrics and dissimilarity measures, such as Euclidean distance, Mahalanobis distance, or cosine dissimilarity. Since most previous publications indicate a higher performance with cosine for deep features, we employ the cosine dissimilarity for all approaches when supported and fall back to Euclidean distance when required. In addition, we reduce dimensionality and apply whitening using PCA as in [10].

5.1. Face Clustering on IJB-B

We evaluate several clustering algorithm on all seven of IJB-B protocols. We advance the state of the art on all seven standard protocols but show only four in Table 1 in the main paper, the remainder in later tables. To facilitate comparison, we use the 128-dimensional features that have been used for identity clustering in PAHC [1,65], provided by the authors of those papers. We also compare with the current state-of-the-art [2], which utilizes more advanced features with hard negative mining, so they are not directly comparable on clustering alone as PEACH may even do better with their features. Nevertheless, we still outperform DCC-NEG without tuning clustering parameters on the test set as was done in [2]. Classic algorithms other than PEACH and PET-FINCH use *hyperparameters optimized for best performance on the test set*, yet it is observed that PEACH (τ_r) provides the best NMI

and B^3 performance for all of the existing algorithms on the IJB-B protocols. Moreover, both PEACH versions significantly advance over the best baselines with those features. We also evaluated a smaller clustering problem, randomly choosing three subjects—over ten runs, we always got perfect clustering (NMI = 1) for PEACH, which shows that it works fine with a small number of clusters.

Table 1. *Clustering Out-of-Distribution/Out-of-Domain data.* This table shows NMI and B^3 F-scores for various clustering algorithms on the IJB-B and LFW face datasets. For IJB-B experiments, everything other than *DCC-NEG* uses older features from [1]; *DCC-NEG* uses its own (better) features and clustering from the same research group. For OLFW, we use a ResNet-101 network trained on ILSVRC 2012. We mark the **best results**, as well as algorithms that optimize parameters on the test set.

Algorithm	**IJB-B Protocols**								**OLFW**	
# Subjects ⇒	***256***		***512***		***1024***		***1845***		***1680 (5479)***	
	NMI	**B^3**	**NMI**	**B^3**	**NMI**	**B^3**	**NMI**	**B^3**	**NMI**	**B^3**
K-Means True K	0.889	0.713	0.898	0.701	0.900	0.675	0.777	0.663	0.643	0.606
DCC-NEG [2]	0.926	0.816	0.921	0.802	0.922	0.751	0.925	0.746	—	—
PAHC [1]	0.865	0.648	—	—	0.890	0.639	0.890	0.610	—	—
FINCH (Best Partition) [4]	0.891	0.731	0.887	0.693	0.891	0.678	0.894	0.677	0.873	0.540
FINCH (Closest to GT) [4]	0.645	0.286	0.648	0.273	0.667	0.270	0.690	0.259	0.894	0.611
PET-FINCH	0.876	0.599	0.888	0.608	0.892	0.594	0.892	0.576	0.894	0.611
PEACH(τ_w)	0.940	0.842	0.941	0.814	0.943	0.817	0.941	0.821	0.945	0.783
PEACH(τ_r)	**0.960**	**0.907**	**0.956**	**0.886**	**0.956**	**0.875**	**0.956**	**0.870**	**0.957**	**0.860**
# Clusters PEACH (τ_r, τ_w) ⇒	*329*, **251**		*643*, **474**		*1298*, **902**		*2278*, **1817**		**1221**, *1166*	

5.2. ImageNet-Based Experiments

For all our ImageNet experiments, we use the ResNet-101 network pre-trained on ILSVRC 2012 [62] data, which achieves a top-1 error rate of 20.69% on the ILSVRC 2012 validation set. While these features are no longer state-of-the-art on ImageNet, our focus is on clustering not classification performance. We extract features for samples from ILSVRC 2010 and 2012 as detailed in Section 4.3. We chose to use the 1000 dimensional logits, i.e., the features before Softmax activation, and reduce their dimensionality to 128 by PCA.

The performance of our clustering algorithm compared to other approaches can be observed for different distributions in Table 2. While the parameter-dependent clustering algorithms need to know the actual number of clusters present in the data, they still cannot match the performance of some of the claimed parameter-independent approaches (FINCH) and our completely automatic approach (PEACH). The results depict that PEACH provides the best performance even when compared to parameter-dependent approaches that use ground-truth information to provide the same number of clusters as the number of classes. In addition, PEACH outperforms the best performing partition of FINCH, the current state of the art, and claimed parameter-free algorithm. The experiments on ImageNet also demonstrate the feasibility of our clustering approach to work with large-scale datasets.

We use OLFW to test out-of-domain clustering, where we employ the same ILSVRC-2012 pre-trained deep feature extraction, but use these features to cluster faces. Results can be found in the last column in Table 1. PEACH outperforms PAHC even when PAHC uses ground-truth number of clusters. The heuristically-scaled version PEACH is the most robust to outliers and has the best performance. Additionally, PEACH is able to identify 1221 clusters, which is close to the 1680 subjects with more than one image, *even though the features were not trained on face data.* Note that FINCH uses PyFlann to find nearest neighbor, which results in slightly different NMIs each time, and we report what we actually obtain.

Table 2. *Clustering for Out/In/Mix Distribution ImageNet data.* This table shows NMI and B^3 F-scores on ImageNet 2010 and 2012 validation sets with 166 (Out-of-Distribution data), 1000 (In-Distribution data), and 1166 classes (Mixed-Distribution data). We mark the second-best and **best results**, as well as algorithms that optimize parameters on the test set.

Param. Dep.	**Approach** *GT # Classes*	**Out-of-Distribution** *ImageNet:166*			**In-Distribution** *ImageNet:1000*			**Mixed-Distribution** *ImageNet:1166*		
		#C	**NMI**	B^3	**#C**	**NMI**	B^3	**#C**	**NMI**	B^3
Ground-truth	K-Means	given	0.643	0.268	given	0.825	0.576	given	0.755	0.367
Ground-truth	Spectral	given	0.798	0.609	given	0.630	0.298	given	0.763	0.367
Ground-truth	BIRCH	given	0.798	0.573	given	0.832	0.545	given	0.808	0.468
Ground-truth	AHC [32]	given	0.632	0.253	given	0.805	0.444	given	0.730	0.264
Closest to GT	FINCH [4]	247	0.819	0.604	**1018**	0.821	0.575	1441	0.768	0.343
Automatic	**PEACH(τ_r)**	**265**	**0.822**	**0.662**	896	**0.863**	**0.813**	**1031**	**0.837**	**0.487**

In a final ImageNet comparison, we compare a transfer clustering task. Results are provided in Table 3. The prior state of the art, Deep Transfer Clustering (DTC) [5], uses a ResNet-18 backbone and a heuristic for estimating the number of clusters. It works well with only 30 classes to cluster, but we find that its performance drops significantly with 118 classes. Using a ResNet-18 architecture trained on the 800 classes and 128 PCA dimensions, PEACH outperforms DTC, even by a large margin for the 118 classes problem. PEACH also outrivals the other algorithms, even when they use the ground-truth number of clusters.

In Table 3, we do not compare against approaches that develop self-supervised features on the target dataset [66], as such work focuses on better features not just on clustering. With their improved features, they report an NMI of 0.82, slightly better than our 0.818. However, also using better features, e.g., using a ResNet-50 PEACH, significantly improves to an NMI of 0.90.

Table 3. *PEACH Transfer Clustering*. This table shows NMI on transfer clustering for ImageNet with partitions from [67] with either 800 training and 82 validation classes and three random samples with 30 classes for testing, or 882 training and 118 test classes. Results with **K** used the ground-truth number of clusters, while **U** indicates estimated cluster counts, partially by [5]. We mark the **best results**.

Algorithm	**30 K**	**30 U**	**118 K**	**118 U**
Large Spectral Clustering	0.733	0.655	—	—
KL-divergence Contrastive	0.750	0.715	—	—
MetaClassification Likelihood	0.762	0.765	—	—
Deep Transfer Clustering	0.791	0.786	0.38	0.53
FINCH(*Best*) ResNet-18	0.696	—	0.63	—
PEACH(τ_r) ResNet-18	—	**0.818**	—	**0.68**

5.3. Deep Clustering

Recent work [10,11] proposes an unsupervised feature learning approach, combining self-supervision and clustering to capture complementary statistics from large-scale data. During network training, the traditional clustering algorithms with hand-tuned parameters K-Means [22] and PIC [30] are used to cluster features at every epoch. We replace these clustering algorithms with the specified approach (AHC, FINCH, PEACH) and train for 100 epochs.

We report the deep clustering results for training the ILSVRC 2012 validation set in Table 4. When used alongside network training to cluster features at each epoch for an unsupervised deep learning approach, PEACH provides the best results. The improvement over FINCH would be even great if we would not have manually selected the best partition. More sensitive parameter-dependent algorithms are worse with bad guesses when used

inside of deep- and deeper-clusters. This highlights that clustering approaches like PEACH are indispensable since it is impossible to know beforehand how many classes are present in unsupervised learning. We also report the top-1 accuracy of deep clustering on MNIST in Table 5, which shows PEACH outperforms K-Means and FINCH at mature epochs.

Table 4. *PEACH Improves Unsupervised Training.* We show the NMI for various clustering algorithms used during the training process of deep clustering, and gains over [10,11], respectively. We mark the **best results**.

Algorithm	Deeper Clustering		Deep Clustering	
	NMI	% Gain	NMI	% Gain
K-Means	0.447	–	0.446	–
AHC [32]	0.392	−12.3%	0.421	−5.8%
FINCH(GT) [4]	0.346	−22.6%	0.371	−17.0%
PEACH(τ_r)	**0.756**	**69.1%**	**0.604**	**35.2%**

Table 5. *Top-1 Accuracy of Linear Classifier of Deep Clustering.* We employ LeNet++ for deep clustering of MNIST, and we report the top-1 accuracy of a converged linear classifier after feature extraction. PEACH(τ_r) achieves higher accuracy than K-Means and FINCH at each of the deep clustering epochs examined.

Algorithm	MNIST/Deep Clustering Epochs			
	Epoch 100	Epoch 125	Epoch 150	Epoch 200
K-Means	94.44%	93.09%	92.11%	90.56%
FINCH(*Best*) [4]	96.63%	96.12%	96.39%	96.07%
PEACH(τ_r)	**97.72%**	**96.23%**	**96.47%**	**96.46%**

5.4. Timing

As shown in Table 6, PEACH(τ_r) is faster than all tested algorithms for large face clustering problems of IJB-B, with the exception of FINCH which is the fastest by far. While both PEACH and FINCH have overall time complexity of $O(N \log N)$, PEACH has a larger constant because of a greater number of rounds of merging and its need to estimate the scaling ω and threshold.

Table 6. *Running Time of IJB-B Protocols.* Running time in seconds for various clustering algorithms is provided for all IJB-B protocols. Missing values (—) indicate that results could not be obtained within one day. We mark the second-fastest and **fastest algorithms**.

Algorithm	Number of Classes						
	32	64	128	256	512	1024	1845
Agglomerative	**0.04**	0.13	0.91	4.18	18.67	99.51	462.51
BIRCH	0.11	0.31	1.51	5.46	21.31	102.61	444.20
K-Means	0.84	2.82	12.43	57.19	258.74	877.89	3784.34
Spectral Clustering	2.96	9.68	69.43	591.54	3238.38	—	—
OPTICS	1.58	4.61	26.23	113.83	411.38	1254.61	—
Affinity Propagation	2.71	11.33	84.59	265.01	—	—	—
FINCH	0.06	**0.08**	**0.27**	**0.71**	**2.16**	**7.78**	**25.71**
PEACH	2.32	3.59	8.84	18.56	36.13	79.35	176.92

For example, consider IJB-B 1024, the rough timing is FINCH∼8 s, PEACH∼79 s, AHC∼100 s, BIRCH∼102 s, K-Means∼877 s, OPTICS∼1255 s, and Spectral clustering was stopped after running for more than a day. While FINCH is faster, PEACH is more accurate

and does not require user selection among the final partitions. For large problems, PEACH is faster than standard Agglomerative clustering since it stops merging sooner.

6. Ablation Study

6.1. Sensitivity to Variations in Threshold

The prime inner parameter of PEACH is the automatically computed distance threshold τ_r/τ_w. Thus, variations in EVT fitting (or in maximum/median values used for τ_r) will impact the threshold. As the critical parameter, it is important to have an ablation study to assess the sensitivity to variations in τ_r/τ_w and show that clustering is stable to reasonable variations around our estimation of the distance threshold.

We scale the automatically estimated threshold and replace it with the threshold $s \cdot \tau_r$ with values for s between 0.9 to 1.1 and observe how severely the clustering results are impacted in terms of NMI. We applied this on the IJB-B 1845, LFW, and ImageNet 166 protocols. The resulting NMI scores in Figure 5 show that PEACH is stable with respect to the prime inner parameter τ_r/τ_w and the performance of the system is not very sensitive—the difference is less than 5% in NMI. This is an important observation—not only must the threshold exist, but empirically we see that clustering results tend to be robust to moderate variations in that threshold.

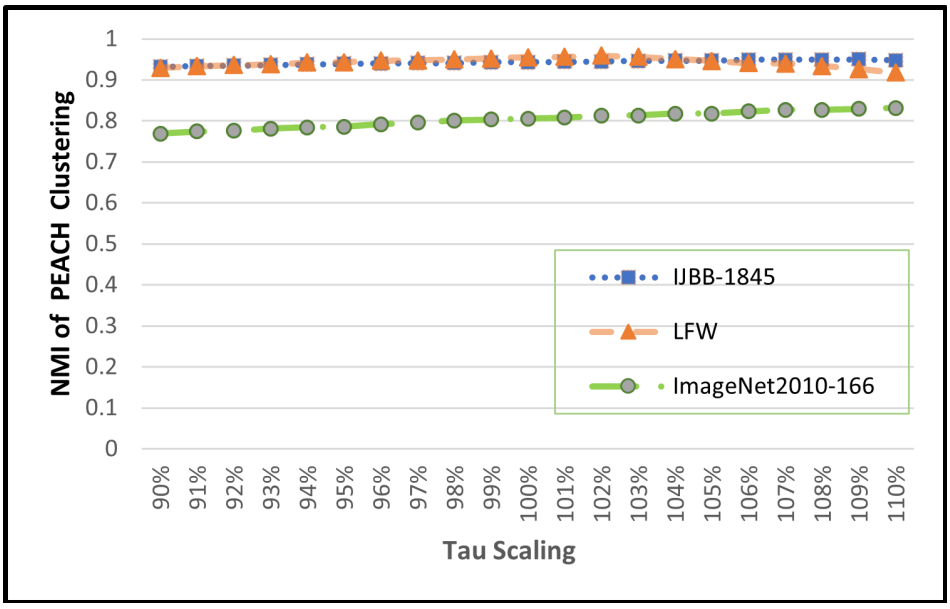


Figure 5. *Different Scaling of* τ_r. The figure shows how stable the final NMI values are with respect to up to 10% perturbation of the threshold τ_r used in PEACH(τ_r). Evaluating the IJB-B 1845, LFW, and ImageNet 360 protocols, we scale the automatically estimated τ_r/τ_w within the range of 90% to 110%. The figure shows that even a moderate change in threshold computation, e.g., by a few outliers, is not likely to break the PEACH algorithm.

6.2. Comparison with AHC Using a Robust Threshold

Given that we have proven the EVT distribution of various link distances, it is natural to ask what happens if we use that distance threshold in standard linkage. In Table 7, we show the NMI, B^3, and ARI [39,40,44] scores for the three most common AHC [31] linkages using the τ, which is also shown. A 1-sided paired t-test across the NMI, B^3 and ARI for all 11 datasets shows that PEACH(τ_r) is statistically significantly better than each AHC version with $p = 0.01$.

Table 7. *AHC Linkages.* These tables show NMI, B^3, and ARI scores for AHC with classic linkages when using τ_r to limit merges, while PEACH uses centroid linkage. Results are provided for all experiments. We mark the second-best and **best results**.

(a) NMI

	IJ32	IJ64	IJ128	IJ256	IJ512	IJ1024	IJ1845	OLFW	IN166	IN1000	IN1166
$\tau =$	0.515	0.502	0.469	0.455	0.440	0.404	0.395	0.456	0.504	0.447	0.454
AHC (Average)	**0.933**	0.923	0.937	0.937	0.935	0.932	0.934	0.935	0.779	**0.863**	**0.836**
AHC (Single)	0.803	0.765	0.786	0.709	0.883	0.796	0.636	0.951	0.001	0.166	0.215
AHC (Complete)	0.837	0.859	0.855	0.879	0.883	0.879	0.885	0.924	0.793	0.837	0.812
PEACH(τ_w)	0.963	**0.958**	**0.963**	**0.960**	0.950	0.951	0.952	0.933	0.870	0.812	0.812
PEACH(τ_r)	**0.967**	0.954	0.962	**0.960**	**0.956**	**0.956**	**0.956**	**0.957**	**0.876**	0.838	0.812

(b) B^3

	IJ32	IJ64	IJ128	IJ256	IJ512	IJ1024	IJ1845	OLFW	IN166	IN1000	IN1166
AHC (Average)	0.904	0.837	0.859	0.841	0.804	0.767	0.761	0.742	0.554	**0.552**	0.423
AHC (Single)	0.711	0.657	0.718	0.604	0.784	0.680	0.533	0.832	0.013	0.172	0.188
AHC (Complete)	0.589	0.597	0.541	0.597	0.553	0.485	0.480	0.696	0.399	0.352	0.225
PEACH(w)	0.937	**0.918**	**0.924**	**0.907**	0.872	0.865	0.861	0.814	0.713	0.460	**0.488**
PEACH(r)	**0.947**	0.909	0.923	**0.907**	**0.886**	**0.875**	**0.870**	**0.860**	**0.727**	0.543	**0.488**

(c) ARI

	IJ32	IJ64	IJ128	IJ256	IJ512	IJ1024	IJ1845	OLFW	IN166	IN1000	IN1166
AHC (Average)	0.943	0.847	**0.955**	**0.955**	0.907	0.832	0.756	0.126	0.308	0.514	0.367
AHC (Single)	0.345	0.176	0.272	0.093	0.253	0.048	0.009	0.536	0.001	0.0002	0.0001
AHC (Complete)	0.478	0.551	0.462	0.691	0.557	0.407	0.322	0.053	0.384	0.333	0.183
PEACH(w)	0.695	0.759	0.728	0.754	0.780	0.777	0.765	0.801	0.737	0.677	0.587
PEACH(r)	**0.961**	**0.941**	0.951	0.944	**0.932**	**0.926**	**0.924**	**0.805**	**0.816**	**0.727**	**0.683**

6.3. Optimized or Ground-Truth Number of Clusters

An important question is what happens if we use the number of clusters computed using our robust threshold τ_r in classic algorithms that take the number of clusters as their primary parameters. Here, we investigate classic K-Means and AHC as well as more recent algorithms BIRCH [28], Spectral [25], and Bi-clustering [26], all as implemented in the SciKit-Learn Python library. For comparison, in Table 8, we show results when algorithms are provided the true number of clusters. For these algorithms, it is an optimistic view since the number of classes is usually unknown. However, PEACH(τ_r) is generally performs better.

In Table 9, we show the NMI, B^3, and ARI scores for various experiments when supplying the algorithm with our optimized number of clusters. A 1-sided paired t-test across the NMI, B^3, and ARI [39,40,44] for all 11 datasets shows that, even when they are provided with the ground-truth number of classes or the number of clusters computed by PEACH(τ_r), we see that PEACH(τ_r) is statistically significantly better than K-Means, Spectral, and Bi-clustering with $p = 0.01$. While PEACH generally has better scores, the difference over AHC and Birch is not statistically significant, though this is not that surprising since both are agglomerative algorithms. Of course, they only obtain this level of performance when given a ground-truth number of classes or when PEACH provides an estimate of the number of clusters, so they offer no real advantage over PEACH.

Table 8. *Utilizing Ground-Truth Number of Classes.* These tables include NMI and B^3 results for various classic algorithms when using the ground number of classes in the face clustering experiments. We mark the second-best and **best results**.

(a) NMI

	IJ32	IJ64	IJ128	IJ256	IJ512	IJ1024	IJ1845	OLFW
Number of Classes	32	64	128	256	512	1024	1845	1680 (5479)
K-Means	0.928	0.898	0.892	0.889	0.897	0.900	0.777	0.643
AHC(average)	**0.959**	0.921	0.928	0.924	0.927	0.928	0.931	0.941
BIRCH	0.929	0.920	0.909	0.904	0.909	0.913	0.913	0.903
Spectral-Clustering	0.867	0.883	0.883	0.870	0.864	0.867	0.868	0.790
BI	0.781	0.767	0.728	0.719	0.723	0.740	0.749	0.834
Affinity-Prop	0.848	0.855	0.845	0.857	0.872	0.882	0.886	0.717
FINCH	0.888	0.874	0.879	0.891	0.887	0.891	0.894	0.873
PEACH(τ_r)	**0.967**	**0.954**	**0.962**	**0.960**	**0.956**	**0.956**	**0.956**	**0.957**

(b) B^3

	IJ32	IJ64	IJ128	IJ256	IJ512	IJ1024	IJ1845	OLFW
K-Means	0.859	0.793	0.751	0.713	0.701	0.675	0.663	0.606
AHC (Average)	0.943	0.828	0.838	0.798	0.792	0.775	0.771	0.754
BIRCH	0.869	0.842	0.791	0.750	0.739	0.725	0.711	0.632
Spectral	0.766	0.785	0.762	0.726	0.705	0.698	0.676	0.606
BI	0.630	0.554	0.446	0.389	0.343	0.334	0.328	0.475
Affinity-Prop	0.622	0.595	0.537	0.534	0.544	0.542	0.538	0.402
FINCH	0.811	0.740	0.727	0.731	0.693	0.678	0.677	0.540
PEACH(τ_r)	**0.967**	**0.954**	**0.962**	**0.960**	**0.956**	**0.956**	**.956**	**.957**

Table 9. *Utilizing Robust Number of Clusters.* These tables show NMI, B^3, and ARI results for various classic algorithms when using the number of clusters as defined by our robust threshold estimation τ_r. We mark the second-best and **best results**.

(a) NMI

	IJ32	IJ64	IJ128	IJ256	IJ512	IJ1024	IJ1845	OLFW	IN166	IN1000	IN1166
Clusters	38	82	157	326	637	1304	2315	1216	265	677	1031
K-Means	0.909	0.891	0.882	0.885	0.894	0.896	0.899	0.862	0.810	0.837	**0.825**
AHC (Average)	0.956	0.944	0.944	0.943	0.935	0.935	0.937	0.886	0.774	0.797	0.781
BIRCH	0.909	0.903	0.899	0.896	0.905	0.909	0.910	0.886	0.803	0.813	0.803
Spectral	0.875	0.880	0.875	0.867	0.869	0.870	0.869	0.854	0.798	0.816	0.806
BI	0.807	0.781	0.750	0.737	0.749	0.758	0.766	0.783	0.626	0.590	0.571
PEACH(τ_r)	**0.967**	**0.954**	**0.962**	**0.960**	**0.956**	**0.956**	**0.956**	**0.957**	**0.876**	**0.838**	0.812

(b) B^3

	IJ32	IJ64	IJ128	IJ256	IJ512	IJ1024	IJ1845	OLFW	IN166	IN1000	IN1166
K-Means	0.810	0.744	0.688	0.665	0.665	0.641	0.635	0.508	0.546	0.540	0.509
AHC (Average)	0.921	0.883	0.877	0.859	0.816	0.793	0.783	0.554	0.506	0.403	0.386
BIRCH	0.823	0.766	0.735	0.699	0.701	0.681	0.672	0.589	0.542	0.476	0.461
Spectral	0.750	0.734	0.719	0.662	0.652	0.643	0.623	0.566	0.539	**0.580**	**0.531**
BI	0.664	0.568	0.459	0.378	0.360	0.333	0.328	0.324	0.229	0.179	0.124
PEACH(τ_r)	**0.947**	**0.909**	**0.923**	**0.907**	**0.886**	**0.875**	**0.870**	**0.860**	**0.727**	0.543	0.488

(c) ARI

	IJ32	IJ64	IJ128	IJ256	IJ512	IJ1024	IJ1845	OLFW	IN166	IN1000	IN1166
K-Means	0.709	0.639	0.440	0.415	0.441	0.413	0.344	0.189	0.525	0.470	0.456
AHC (Average)	0.949	0.915	**0.965**	**0.961**	0.905	0.851	0.781	0.453	0.442	0.352	0.337
BIRCH	0.699	0.678	0.550	0.497	0.513	0.473	0.395	0.307	0.527	0.404	0.401
Spectral	0.593	0.580	0.543	0.356	0.296	0.142	0.151	0.102	0.299	0.079	0.072
BI	0.617	0.529	0.285	0.251	0.257	0.253	0.185	0.067	0.202	0.137	0.098
PEACH(τ_r)	**0.961**	**0.941**	0.951	0.944	**0.932**	**0.926**	**0.924**	**0.805**	**0.816**	**0.727**	**0.683**

6.4. FINCH Partition Analysis

Unlike PEACH, which provides one definitive clustering solution, FINCH—which is claimed to be a parameter-free approach—provided six potential clustering solutions for the IJB-B 1845 dataset, all of which are listed in Table 10. The final decision of which solution to use has to be made by the user. As shown in Table 10, this choice of solution can greatly impact the performance of the resulting clustering.

Table 10. *Sensitivity of FINCH to Partition Selection.* In this table, we show the NMI scores and the number of clusters of each FINCH partition. In our optimistic evaluation of Table 1, we manually selected partition three since the number of clusters of this partition is 880, which is the closest one to the ground-truth 1845. If we would not have the ground-truth but pick a random partition, the resulting NMI values could drop dramatically. In all experiments in this work, we always chose the FINCH partition closest to the ground-truth.

Partition	# Clusters	NMI
1	13406	0.833
2	3223	0.892
3	880	0.894
4	208	0.803
5	43	0.642
6	10	0.439

The authors of FINCH [4] did not provide an indication regarding which of these partitions to select without knowing the ground-truth number of classes. Even when we choose the FINCH partition closest to the ground-truth, a 1-sided paired t-test across the NMI/BCubed/ARI for all 11 datasets shows PEACH(τ_r) is statistically significantly better than FINCH. For our PET-FINCH, we do experiment on a novel data-distribution view of clustering protocols and then detail our evaluation protocols. The Table 11 shows that our robust τ_r works with FINCH and made significant results.

Table 11. *PET-FINCH.* In this table, we show the NMI scores of FINCH partition which were selected by using our τ_r. We estimate the number of clusters of AHC by using τ_r and then we choose the FINCH partition, which is closest to the estimated number of clusters. We compare the results with PEACH. We mark the **best results**.

NMI	IJ32	IJ64	IJ128	IJ256	IJ512	IJ1024	IJ1845	OLFW	IN166	IN1000	IN1166
$\tau_r =$	0.456	0.513	0.467	0.467	0.471	0.448	0.424	0.507	0.597	0.502	0.454
PET-FINCH	0.851	0.874	0.865	0.876	0.888	0.892	0.892	0.894	0.852	**0.821**	0.798
PEACH(τ_r)	**0.963**	**0.958**	**0.963**	**0.960**	**0.950**	**0.951**	**0.952**	**0.933**	**0.870**	0.812	**0.812**

7. Limitations

Our Extreme-Linkage Theory and resulting threshold selection provide a transformative step in clustering; it is the first effective fully automatic clustering algorithm for deep features to improve vision or learning systems' performance. However, PEACE/PEACH is not a panacea; our theory's assumptions generally do not hold on many raw or hand-crafted features used in classic clustering evaluation, and the resulting threshold is not good. PEACE is also not parameter-free; it just provides good defaults. While we tested on many different types of problems, there are a plethora of clustering problems in vision that could be used in evaluation.

8. Conclusions

In most problems, one cannot know the number of clusters or choose between potential clustering options—real-world problems, especially in computer vision, need fully automatic clustering. This paper introduced our Provable Extreme-value Agglomerative Clustering-threshold Estimation (PEACE), and the fully automated PEACH, which applies

the fully automatic clustering threshold to classic agglomerative clustering. We proved that nearest neighbors of correct links follow a EVT-distribution, in particular a Weibull distribution, and used that to derive a threshold, and then develop a more robust threshold with heuristics automatically reducing an outlier's impact. We also show empirically *that this approach can select the best partition for FINCH* [4], the current state of the art in a visual clustering algorithm. For evaluation, we focused on clustering deep features relevant for advancing the state of the art in vision.

Our experiments show that PEACH significantly advances the state of the art in clustering through outperforming recent results from top venues [2,4,5,10,11,68–70]. Looking across all of the different experimental domains, distribution, and problems, PEACH was consistently better than FINCH and other algorithms. We encourage downloading our Github code (released after acceptance) and try either PET-FINCH or PEACH on your problem. If you like the previous state of the art, FINCH, then free yourself from the selection problem with PET-FINCH with virtually no change to your code. However, also note that PEACH offers superior performance, advancing the state of clustering, as well as providing fully automatic parameter selection.

Supplementary Materials: The following supporting information can be downloaded at: https://www.mdpi.com/article/10.3390/a15050170/s1, Figure S1: Weibull fit and resulting τ. Example plots showing Weibull fit and resulting threshold τ estimated from the distribution of ANN distances. In this figure, we show a histogram of raw data from the named dataset, the resulting Weibull fit, its CDF, and the resulting τ_w for 98% of the data and 99% confidence, except for ImageNet and LFW, where we show the "robust" version based on the mode heuristic, which results in using only 88% and 63% of the data, respectively. For LFW, the resulting fit is quite different from the example fit for all LFW data, which is shown in the main paper. These plots show the trim-line of what data was ignored in fitting the Weibull.

Author Contributions: Author contributions: Conceptualization, C.L. and T.E.B.; methodology C.L. and M.G. ; software C.L.; validation, A.R.D., S.C., M.J., and T.A.; formal analysis, T.E.B.; investigation, C.L., M.G., and M.J.; writing—original draft preparation, C.L. and T.E.B.; writing—review and editing, all authors; supervision, T.E.B.; funding acquisition, T.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded in part by DARPA SAIL-ON Contract # HR001120C0055.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Publicly available datasets were analyzed in this study. This data can be found here: ImageNet: http://image-net.org/download.php (accessed on 10 April 2022); Labeled Faces in the Wild: http://vis-www.cs.umass.edu/lfw (accessed on 10 April 2022); IJB-B (part of IJB-C): http://www.nist.gov/programs-projects/face-challenges (accessed on 10 April 2022).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Lin, W.A.; Chen, J.C.; Chellappa, R. A proximity-aware hierarchical clustering of faces. In Proceedings of the International Conference on Automatic Face & Gesture Recognition (FG), Washington, DC, USA, 30 May–3 June 2017.
2. Lin, W.A.; Chen, J.C.; Castillo, C.D.; Chellappa, R. Deep density clustering of unconstrained faces. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake, UT, USA, 18–22 June 2018.
3. Tapaswi, M.; Law, M.T.; Fidler, S. Video Face Clustering With Unknown Number of Clusters. In Proceedings of the International Conference on Computer Vision (ICCV), Seoul, Korea, 27 October–2 November 2019.
4. Sarfraz, S.; Sharma, V.; Stiefelhagen, R. Efficient Parameter-free Clustering Using First Neighbor Relations. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019.
5. Han, K.; Vedaldi, A.; Zisserman, A. Learning to discover novel visual categories via deep transfer clustering. In Proceedings of the International Conference on Computer Vision (ICCV), Seoul, Korea, 27 October–2 November 2019.
6. Liu, B.; Wu, Z.; Hu, H.; Lin, S. Deep metric transfer for label propagation with limited annotated data. In Proceedings of the International Conference on Computer Vision Workshops (CVPRW), Seoul, Korea, 27 October–2 November 2019.

7. Feng, Z.; Xu, C.; Tao, D. Self-Supervised Representation Learning From Multi-Domain Data. In Proceedings of the International Conference on Computer Vision (ICCV), Seoul, Korea, 27 October–2 November 2019.
8. Jiang, H.; Grauman, K. Seeing invisible poses: Estimating 3D body pose from egocentric video. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.
9. You, C.; Li, C.; Robinson, D.P.; Vidal, R. Scalable Exemplar-based Subspace Clustering on Class-Imbalanced Data. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018.
10. Caron, M.; Bojanowski, P.; Joulin, A.; Douze, M. Deep Clustering for Unsupervised Learning of Visual Features. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018; Springer: Berlin/Heidelberg, Germany, 2018.
11. Caron, M.; Bojanowski, P.; Mairal, J.; Joulin, A. Unsupervised Pre-Training of Image Features on Non-Curated Data. In Proceedings of the International Conference on Computer Vision (ICCV), Seoul, Korea, 27 October–2 November 2019.
12. Xu, D.; Tian, Y. A comprehensive survey of clustering algorithms. *Ann. Data Sci.* **2015**, *2*, 165–193. [CrossRef]
13. Tryon, R. *Cluster Analysis: Correlation Profile and Orthometric (Factor) Analysis for the Isolation of Unities in Mind and Personality*; Edwards Brothers: Ann Arbor, MI, United States 1939.
14. Zhang, R.; Isola, P.; Efros, A.A.; Shechtman, E.; Wang, O. The unreasonable effectiveness of deep features as a perceptual metric. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 18–22 June 2018.
15. Whitelam, C.; Taborsky, E.; Blanton, A.; Maze, B.; Adams, J.; Miller, T.; Kalka, N.; Jain, A.K.; Duncan, J.A.; Allen, K.; et al. IARPA Janus Benchmark-B Face Dataset. In Proceedings of the Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 21–26 July 2017, Honolulu, HI, USA, 2017.
16. Figueiredo, E.; Macedo, M.; Siqueira, H.V.; Santana, C.J.; Gokhale, A.; Bastos-Filho, C.J. Swarm intelligence for clustering—A systematic review with new perspectives on data mining. *Eng. Appl. Artif. Intell.* **2019**, *82*, 313–329. [CrossRef]
17. Lee, S.W.; Ryu, D.S. Parameter-free geometric document layout analysis. *Trans. Pattern Anal. Mach. Intell.* **2001**, *23*, 1240–1256.
18. Paragios, N.; Rousson, M.; Ramesh, V. Knowledge-based registration & segmentation of the left ventricle: A level set approach. In Proceedings of the Workshop on Applications of Computer Vision (WACV), Orlando, FL, USA, 4 December 2002.
19. Wolf, S.; Pape, C.; Bailoni, A.; Rahaman, N.; Kreshuk, A.; Kothe, U.; Hamprecht, F. The mutex watershed: efficient, parameter-free image partitioning. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018.
20. Hou, J.; Gao, H.; Li, X. DSets-DBSCAN: A parameter-free clustering algorithm. *IEEE Trans. Image Process.* **2016**, *25*, 3182–3193. [CrossRef] [PubMed]
21. Spencer, J.; Chen, K.; Duan, J. Parameter-free selective segmentation with convex variational methods. *IEEE Trans. Image Process.* **2018**, *28*, 2163–2172. [CrossRef] [PubMed]
22. MacQueen, J. Some methods for classification and analysis of multivariate observations. In *Berkeley Symposium on Mathematical Statistics and Probability*; University of California Press: Oakland, CA, USA, 1967; Volume 1.
23. Ester, M.; Kriegel, H.P.; Sander, J.; Xu, X. A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD), Portland, OR, USA, 2–4 August 1996; AAAI Press: Palo Alto, CA, USA, 1996.
24. Cheng, Y. Mean shift, mode seeking, and clustering. *IEEE Trans. Pattern Anal. Mach. Intell.* **1995**, *17*, 790–799. [CrossRef]
25. von Luxburg, U. A Tutorial on Spectral Clustering. *Stat. Comput.* **2007**, *17*, 395–416. [CrossRef]
26. Govaert, G.; Nadif, M. *Co-Clustering: Models, Algorithms and Applications*; Wiley: Hoboken, NJ, USA, 2013.
27. Frey, B.J.; Dueck, D. Clustering by passing messages between data points. *Science* **2007**, *315*, 972–976. [CrossRef]
28. Zhang, T.; Ramakrishnan, R.; Livny, M. BIRCH: An Efficient Data Clustering Method for Very Large Databases. In Proceedings of the International Conference on Management of Data (SIGMOD), Montreal, QC, Canada, 4–6 June 1996; ACM: New York, NY, USA, 1996.
29. Ankerst, M.; Breunig, M.M.; Kriegel, H.P.; Sander, J. OPTICS: Ordering Points to Identify the Clustering Structure. In Proceedings of the International Conference on Management of Data (SIGMOD), Philadelphia, PA, USA, 1–3 June 1999; ACM: New York, NY, USA, 1999.
30. Lin, F.; Cohen, W.W. Power Iteration Clustering. In Proceedings of the International Conference on Machine Learning (ICML), Haifa, Israel, 21–24 June 2010; Omnipress: Madison, WI, USA, 2010.
31. Kaufman, L.; Rousseeuw, P.J. *Finding Groups in Data: An Introduction to Cluster Analysis*; Wiley: Hoboken, NJ, USA, 1990.
32. Müllner, D. fastcluster: Fast hierarchical, agglomerative clustering routines for R and Python. *J. Stat. Softw.* **2013**, *53*, 1–18. [CrossRef]
33. Dasgupta, S. A Cost Function for Similarity-Based Hierarchical Clustering. 2016. Available online: https://doi.org/10.1145/2897518.2897527 (accessed on 10 April 2022).
34. Moseley, B.; Wang, J. Approximation bounds for hierarchical clustering: Average linkage, bisecting K-Means, and local search. In Proceedings of the Advances in Neural Information Processing Systems (NIPS), Long Beach, CA, USA, 4–9 December 2017. Available online: https://proceedings.neurips.cc/paper/2017/file/d8d31bd778da8bdd536187c36e48892b-Paper.pdf (accessed on 10 April 2022).
35. McInnes, L.; Healy, J.; Astels, S. HDBSCAN: Hierarchical density based clustering. *J. Open Source Softw.* **2017**, *2*, 205. [CrossRef]

36. Mahé, F.; Rognes, T.; Quince, C.; de Vargas, C.; Dunthorn, M. Swarm: robust and fast clustering method for amplicon-based studies. *PeerJ* **2014**, *2*, e593. [CrossRef] [PubMed]
37. Bezdek, J.C.; Ehrlich, R.; Full, W. FCM: The fuzzy C-means clustering algorithm. *Comput. Geosci.* **1984**, *10*, 191–203. [CrossRef]
38. Meilă, M. Comparing clusterings—An information based distance. *J. Multivar. Anal.* **2007**, *98*, 873–895. [CrossRef]
39. Strehl, A.; Ghosh, J. Cluster ensembles—A knowledge reuse framework for combining multiple partitions. *J. Mach. Learn. Res.* **2003**, *3*, 583–617.
40. Rand, W. Objective criteria for the evaluation of clustering methods. *J. Am. Stat. Assoc.* **1971**, *66*, 846–850. [CrossRef]
41. Hubert, L.; Arabie, P. Comparing partitions. *J. Classif.* **1985**, *2*, 193–218. [CrossRef]
42. Vinh, N.X.; Epps, J.; Bailey, J. Information Theoretic Measures for Clusterings Comparison: Is a Correction for Chance Necessary? In Proceedings of the International Conference on Machine Learning (ICML), Montreal, QC, Canada, 14–18 June 2009.
43. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
44. Amigó, E.; Gonzalo, J.; Artiles, J.; Verdejo, F. A Comparison of Extrinsic Clustering Evaluation Metrics Based on Formal Constraints. *Inf. Retr.* **2009**, *12*, 461–486. [CrossRef]
45. Sun, C.; Shrivastava, A.; Singh, S.; Gupta, A. Revisiting unreasonable effectiveness of data in deep learning era. In Proceedings of the International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017.
46. Sejnowski, T.J. The unreasonable effectiveness of deep learning in artificial intelligence. *Proc. Natl. Acad. Sci. USA* **2020**, *117*, 30033–30038. [CrossRef]
47. Krause, J.; Sapp, B.; Howard, A.; Zhou, H.; Toshev, A.; Duerig, T.; Philbin, J.; Fei-Fei, L. The unreasonable effectiveness of noisy data for fine-grained recognition. In Proceedings of the European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 11–14 October 2016.
48. Muja, M.; Lowe, D.G. Scalable nearest neighbor algorithms for high dimensional data. *TRansactions Pattern Anal. Mach. Intell.* **2014**, *36*, 2227–2240. [CrossRef]
49. Kotz, S.; Nadarajah, S. *Extreme Value Distributions: Theory and Applications*; World Science: Singapore, 2001.
50. Scheirer, W.J.; Rocha, A.; Micheals, R.J.; Boult, T.E. Meta-Recognition: The Theory and Practice of Recognition Score Analysis. *TRansactions Pattern Recognit. Mach. Intell.* **2011**, *33*, 1689–1695. [CrossRef] [PubMed]
51. Scheirer, W.; Jain, L.; Boult, T. Probability Models for Open Set Recognition. *Trans. Pattern Recognit. Mach. Intell.* **2014**, *36*, 2317–2324. [CrossRef] [PubMed]
52. Carpentier, A.; Valko, M. Extreme bandits. In Proceedings of the Advances in Neural Information Processing Systems (NIPS), Montreal, QC, Canada, 8–13 December 2014. Available online: https://papers.nips.cc/paper/2014/file/8c7bbbba95c1025975e548cee86dfadc-Paper.pdf (accessed on 10 April 2022).
53. Coles, S. *An Introduction to Statistical Modeling of Extreme Values*; Springer: Berlin/Heidelberg, Germany, 2001.
54. Long, M.; Zhu, H.; Wang, J.; Jordan, M.I. Unsupervised Domain Adaptation with Residual Transfer Networks. In Proceedings of the Advances in Neural Information Processing Systems (NIPS), Barcelona, Spain, 5–10 December 2016. Available online: https://proceedings.neurips.cc/paper/2016/file/ac627ab1ccbdb62ec96e702f07f6425b-Paper.pdf (accessed on 10 April 2022).
55. Ghifary, M.; Kleijn, W.B.; Zhang, M.; Balduzzi, D.; Li, W. Deep reconstruction-classification networks for unsupervised domain adaptation. In Proceedings of the European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 11–14 October 2016.
56. Venkateswara, H.; Eusebio, J.; Chakraborty, S.; Panchanathan, S. Deep hashing network for unsupervised domain adaptation. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.
57. Haeusser, P.; Frerix, T.; Mordvintsev, A.; Cremers, D. Associative domain adaptation. In Proceedings of the International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017.
58. You, K.; Long, M.; Cao, Z.; Wang, J.; Jordan, M.I. Universal domain adaptation. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019.
59. Singh, R.; Vatsa, M.; Patel, V.M.; Ratha, N. *Domain Adaptation for Visual Understanding*; Springer: Berlin/Heidelberg, Germany, 2020.
60. Oza, P.; Patel, V.M. C2AE: Class conditioned auto-encoder for open-set recognition. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019.
61. Beyer, K.; Goldstein, J.; Ramakrishnan, R.; Shaft, U. When is "nearest neighbor" meaningful? In Proceedings of the International Conference on Database Theory, Jerusalem, Israel, 10–12 January 1999; Springer: Berlin/Heidelberg, Germany, 1999.
62. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. ImageNet Large Scale Visual Recognition Challenge. *Int. J. Comput. Vis.* **2015**, *115*, 211–252. [CrossRef]
63. Jafarzadeh, M.; Ahmad, T.; Dhamija, A.R.; Li, C.; Cruz, S.; Boult, T.E. Automatic open-world reliability assessment. In Proceedings of the Winter Conference on Applications of Computer Vision (WACV), Waikoloa, HI, USA, 3–8 January 2021.
64. Huang, G.B.; Ramesh, M.; Berg, T.; Learned-Miller, E. *Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments*; Technical Report 07-49; University of Massachusetts: Amherst, MA, USA, 2007.
65. Lin, W.A.; Chen, J.C.; Ranjan, R.; Bansal, A.; Sankaranarayanan, S.; Castillo, C.D.; Chellappa, R. Proximity-Aware Hierarchical Clustering of unconstrained faces. *Image Vis. Comput.* **2018**, *77*, 33–44. [CrossRef]

66. Han, K.; Rebuffi, S.A.; Ehrhardt, S.; Vedaldi, A.; Zisserman, A. Automatically Discovering and Learning New Visual Categories with Ranking Statistics. In Proceedings of the International Conference on Learning Representations (ICLR), New Orleans, LA, USA, 6–9 May 2019.
67. Vinyals, O.; Blundell, C.; Lillicrap, T.; Wierstra, D. Matching Networks for One Shot Learning. In Proceedings of the Advances in Neural Information Processing Systems (NIPS), Barcelona, Spain, 5–10 December 2016; Volume 29. Available online: https://proceedings.neurips.cc/paper/2016/file/90e1357833654983612fb05e3ec9148c-Paper.pdf (accessed on 10 April 2022).
68. Hsu, Y.C.; Lv, Z.; Schlosser, J.; Odom, P.; Kira, Z. Multi-class classification without multi-class labels. In Proceedings of the International Conference on Learning Representations (ICLR), Vancouver, BC, Canada, 30 April–3 May 2018.
69. Deng, J.; Guo, J.; Xue, N.; Zafeiriou, S. Arcface: Additive angular margin loss for deep face recognition. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019.
70. Wang, Z.; Zheng, L.; Li, Y.; Wang, S. Linkage based face clustering via graph convolution network. In Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR), Long Beach, CA, USA, 16–20 June 2019.