Neurocomputing 413 (2020) 193–209

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Hub-based subspace clustering

Priya Mani *, Carlotta Domeniconi **

Department of Computer Science, George Mason University, 4400 University Drive MSN 4A5, Fairfax, VA 22030, USA

ARTICLE INFO

Article history:
Received 13 January 2020
Revised 26 June 2020
Accepted 28 June 2020
Available online 6 July 2020
Communicated by X. Gao

Keywords:
Hubness
Subspace clustering
Graph-based meta-features
Selective sampling

ABSTRACT

Data often exists in subspaces embedded within a high-dimensional space. Subspace clustering seeks to group data according to the dimensions relevant to each subspace. This requires the estimation of subspaces as well as the clustering of data. Subspace clustering becomes increasingly challenging in high dimensional spaces due to the *curse of dimensionality* which affects reliable estimations of distances and density. Recently, another aspect of high-dimensional spaces has been observed, known as the hubness phenomenon, whereby few data points appear frequently as nearest neighbors of the rest of the data. The distribution of neighbor occurrences becomes skewed with increasing intrinsic dimensionality of the data, and few points with high neighbor occurrences emerge as hubs. Hubs exhibit useful geometric properties and have been leveraged for clustering data in the full-dimensional space. In this paper, we study hubs in the context of subspace clustering. We present new characterizations of hubs in relation to subspaces, and design graph-based meta-features to identify a subset of hubs which are well fit to serve as seeds for the discovery of local latent subspaces and clusters. We propose and evaluate a hubness-driven algorithm to find subspace clusters, and show that our approach is superior to the baselines, and is competitive against state-of-the-art subspace clustering methods. We also identify the data characteristics that make hubs suitable for subspace clustering. Such characterization gives valuable guidelines to data mining practitioners.

1. Introduction

Subspace clustering is a fundamental unsupervised learning task which seeks to group data according to their similarity in a combination of features. This is particularly relevant for high-dimensional data, where samples reside in clusters, and different combinations of features are relevant for different clusters. A single feature could be relevant to at least one of the clusters. Global dimensionality reduction methods fail in these scenarios as they cannot capture the local relevance of features within each cluster. Hence, local feature selection techniques are required, which can capture the degree to which each feature contributes to the subspace of a cluster. Several different subspace clustering algorithms have been proposed in the literature [13,32,21,30], based on different definitions of subspaces and methodologies to capture local feature relevance.

However, distance and density estimation in high-dimensional data are negatively affected by the *curse of dimensionality*. In high-dimensional spaces data becomes sparse and pairwise distances become less meaningful, a phenomenon known as distance concentration. Thus, achieving accurate distance and density estimations in high-dimensional data is a challenge.

The hubness phenomenon [34] is another aspect of high-dimensional data. The distribution of k-neighbor occurrences becomes increasingly skewed as dimensionality increases, and few data points with high neighbor occurrence counts emerge as *hubs* in the full-dimensional space (also known as global hubs). Hubness poses both challenges and good potential to enhance subspace clustering. A hub which is a neighbor of data points that belong to a different class than itself can distort subspace estimation, and is detrimental to clustering. Identifying such hubs without access to class labels is a challenge for unsupervised learning tasks. On the other hand, hubs are also known to exhibit useful clustering properties and have been utilized to guide k-means clustering [37]. The value of hubs on detecting subspaces and clusterings therein has not been studied in the literature, and is the focus of this work. As studied in the literature [34], the hubness phenomenon is an inherent property of intrinsically high dimensional data, irrespective of the data distribution and the neighborhood size k. However, none of the subspace clustering algorithms pro-

* Corresponding author.
** Principal corresponding author.
E-mail addresses: pmani@masonlive.gmu.edu (P. Mani), cdomenic@gmu.edu (C. Domeniconi).

https://doi.org/10.1016/j.neucom.2020.06.098
0925-2312/

posed in the literature have considered this phenomenon in their design.

In this paper, we study hubs and their characteristics in the context of subspace clustering. We show that hubs are preserved in local subspaces and enable discriminative initial distance measurements in the full-dimensional space. Hence they form good seeds for iterative, mode-seeking subspace clustering algorithms. The seeding of hubs is non-trivial as different types of hubs exist, and some are detrimental to clustering. We tackle two challenges in unsupervised seed selection, which forms the core of our proposed algorithm: (1) identifying hubs which are detrimental to clustering without access to their class labels, and, (2) the design of a local hubness ranking and selection mechanism to select hubs of different classes. Our experiments on a non-parametric, mean shift based subspace clustering algorithm demonstrate that hub-based seeding improves the quality of clustering and subspace estimation.

We summarize the main contributions of our paper as follows:

1. We present new characterizations of hubs in relation to subspaces, and investigate the conditions under which hubs are effective for clustering data. To the best of our knowledge, this is the first study that analyzes and leverages hubs for subspace clustering.
2. Different types of hubs exist, and not all are suitable for subspace clustering. We propose and evaluate meta-features based on the k-nearest neighbor graph of data to predict the different types of hubs.
3. We propose and evaluate a hubness-driven algorithm to find subspace clusters, which is competitive against state-of-the-art subspace clustering methods. Our experimental evaluation demonstrates that selecting hubs which occur near local density modes can guide and improve the estimation of subspace clusters.

The rest of the paper is organized as follows: Section 2 and Section 3 describe the relevant literature and necessary background for our work, respectively. Section 4 details new characterizations of hubs and their utility for subspace clustering. Our proposed hub-based subspace clustering algorithms are described in Section 5 and their evaluation is provided in Section 6. The paper concludes with a discussion on when hubs should be leveraged, in Section 7.

2. Related work

Several approaches to subspace clustering have been proposed in the literature. Based on the direction of subspace search, they can be classified as top-down and bottom-up algorithms. Top-down subspace clustering algorithms start from the full-dimensional space and prune subspaces based on the *locality assumption*, which assumes that the subspace of a cluster can be derived from a local neighborhood around its centroid or cluster members (e.g., [1,24]. Bottom-up algorithms start with one-dimensional subspaces and expands them based on the *Apriori* principle (e.g., [2,19]. Projected clustering methods (e.g., [1,6] seek to find clusters of data projections on subsets of features, using specialized distance functions. Soft-projected clustering methods (e.g., [8,11] discover subspace clusters by weighting features according to their local feature relevance. Weighting prevents the loss of information incurred in dimensionality reduction. [8] defines a *weighted cluster* as a collection of data points and a weighting of the features along which the data are correlated.

Several hybrid, algebraic and spectral-based subspace clustering approaches have also been proposed. Many of the recent state-of-the-art subspace clustering algorithms are spectral-based and they operate in two stages: (1) construct an affinity matrix of data lying in a union of subspaces, and (2) apply spectral clustering on the affinity matrix to cluster data according to subspaces. The affinity matrix is constructed using the self-expressiveness property of data. Several methods to obtain self-expressive coefficients have been proposed, and they differ mainly in the regularization of the data representation matrix induced by the self-expressive coefficients (e.g., low-rank representation [25,26,38], sparse representation [10,40,39] and block-diagonal representation [27]. A unified optimization framework for affinity construction and spectral clustering is proposed by [23]. [39] find a subset of data points, named *exemplars*, to best represent the data as a linear combination of the selected exemplars. The spectral-based methods embed data in a new global representation as induced by the self-expressive coefficients, and data are segmented according to subspaces by clustering on this embedded space. However, they do not explicitly find the subspaces pertaining to each cluster and hence lack interpretability of features relevant to a subspace. Detailed survey on subspace clustering methods and evaluation can be found in [13,32,21,30].

[34] provides a detailed theoretical study of the hubness phenomenon, its emergence, and its impact on learning tasks. The literature on the application of the hubness phenomenon can be broadly classified into two parts: (1) methods that seek to remove the effect of hubness, and (2) methods that leverage hubness. Some work reports the presence of hubs as detrimental to their tasks (e.g., music retrieval [5], finger-print identification [15], zero-shot learning [41]. [37] is the first work that leverages hubs for a learning task. The paper depicts hubs as good cluster prototypes, and hubs are used to improve k-means clustering in high dimensions. [14] proposed the relative hubness score for hub selection to improve the k-Hubs algorithm proposed in [37].

There are three major differences between the hub selection process in [37] and in [14], and our proposed approach: (1) The previous algorithms rank and select local hubs, i.e. they compute hubs within clusters formed during k-means iterations, while we propose an algorithm to rank and select global hubs; (2) The initial centroids of previous algorithms are selected based on random sampling, while we use a hub-based sample as initial points of a non-parametric clustering algorithm; (3) Existing algorithms perform clustering in full dimensional space, while we focus on hub selection for subspace clustering. The authors in [29] empirically identified geometric relationships between hubs, distance distributions, data density and intrinsic dimensionality. In this paper, we leverage their findings as well as identify new characterizations of hubs, to develop our proposed algorithm for subspace clustering.

3. Background

3.1. The Hubness phenomenon

Let $D = \{\mathbf{x}_i\}_{i=1}^{n}$ be a collection of n data points $\mathbf{x}_i \in \Re^d$. The *hubness score* $N_k(\mathbf{x})$ of a data point $\mathbf{x}$ is defined as the number of times $\mathbf{x}$ occurs in the k-nearest neighbor (kNN) list of the other points [34].

$$N_k(\mathbf{x}) = \sum_{i=1}^{n} I(\mathbf{x}, k\text{NN}(\mathbf{x}_i))$$

$$I(p, Q) = \begin{cases} 1 & p \in Q \\ 0 & otherwise \end{cases}$$

The data points $\mathbf{x}_i$ that contribute to the hubness score of $\mathbf{x}$ are called the *reverse nearest neighbors* of $\mathbf{x}$ (RkNN). A *hub* is a point $\mathbf{x}$ whose hubness score $N_k(\mathbf{x}) > \mu + 2\sigma$, where μ and σ are the mean and standard deviation of the hubness scores N_k of all points in D.

Table 1
Percentage of global hubs (all/good/bad) retained as local hubs within the corresponding weighted clusters. The value N/A denotes the absence of good/bad global hubs in a dataset. Percentages are rounded to two significant digits.

Data	% Total Hubs Retained	% Good Hubs Retained	% Bad Hubs Retained
Toy1	0.93	0.93	N/A
Toy2	0.90	0.90	N/A
Diabetes	0.82	0.91	0.59
Abalone	0.59	1.00	0.59
Letter	0.62	0.66	0.31
Pen-3	0.77	0.77	N/A
Pen-10	0.61	0.61	0
Image	0.46	0.49	0.19
Waveform1	0.85	0.91	0.44
Sonar	0.89	0.95	0.71
Musk1	0.54	0.70	0.10
Musk2	0.86	0.94	0.45
mfeat-factors	0.67	0.67	0
mfeat-pixels	0.66	0.66	0
ISOLET	0.79	0.83	0.31
COIL	0.24	0.24	N/A
Caltech-20	0.80	N/A	0.80
Caltech-100	0.88	N/A	0.88
DrivFace	0.79	0.79	N/A
OVA_ Colon	0.90	0.94	0.33

Hubs can be defined as either *global* or *local*. *Global* hubs are points that emerge as hubs when N_k is computed using the entire data collection (which may include multiple clusters). *Local* hubs are points $\mathbf{x}$ which emerge as hubs when N_k is computed using only data that belong to the same cluster as $\mathbf{x}$. For a unimodal data distribution, global and local hubs are the same, but for multi-modal distributions, local hubs represent hubs within each component (cluster).

Hubs exhibit several geometric properties which are useful for clustering data. Hubs occur near the centroids of uni-modal data distributions. In multi-modal distributions, they occur near the means of the individual component distributions. [34] mathematically proved that this property is amplified in high dimensions and is related to the phenomenon of distance concentration. [37] empirically analyzed the role of hubs w.r.t *k*-means clustering and designed several hub-based variants of *k*-means to leverage local hubs as cluster centroids during *k*-means iterations.

However, the use of hubs to enhance the process of clustering data is challenging due to the potential presence of the so-called *bad* hubs. In fact, hubs are further classified as *good* and *bad* hubs, based on the amount of label mismatch between a hub and its reverse nearest neighbors. Typically, if the label mismatch is above 50%, then the hub is considered a *bad* hub. Bad hubs are detrimental to information retrieval and classification (clustering) tasks, as they are close to many data points from different classes (clusters). The discrimination of good and bad hubs requires label information, which is typically not available in clustering problems.

3.2. Weighted Adaptive Mean Shift

Since we combine our hub-driven methodology with an adaptive version of mean shift, we provide here the necessary background on the latter. Mean shift [7] is a non-parametric algorithm which estimates the modes in the data as the local maxima of a kernel density function. An adaptive kernel density estimator for n instances in d dimensions is defined as follows:

$$\hat{f}_{\kappa}(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{h_i^d}\kappa(||\frac{\mathbf{x}-\mathbf{x}_i}{h_i}||^2)$$

where $\kappa(\mathbf{x})$ is the kernel *profile*, and h_i is an adaptive kernel bandwidth that can be set differently for each data point. Successive estimation points for the modes are derived from the gradient $\nabla\hat{f}_{\kappa}(\mathbf{x})$ as (Eq. 3.4 in [35]:

$$\mathbf{y}_{t+1} = \frac{\sum_{i=1}^{n}\frac{\mathbf{x}_i}{h_i^{d+2}}g(||\frac{\mathbf{y}_t-\mathbf{x}_i}{h_i}||^2)}{\sum_{i=1}^{n}\frac{1}{h_i^{d+2}}g(||\frac{\mathbf{y}_t-\mathbf{x}_i}{h_i}||^2)}$$

where $g = -\kappa'(\mathbf{x})$. Mean shift is applied to each data point, and all points with the same mode are assigned to the same cluster.

The authors in [35] proposed a weighted adaptive kernel density estimator for data in subspaces (WAMS). The local feature relevance of each point is learned by a weight distribution on its features such that the dispersion among its nearest neighbors is minimized. The weighted distance of point $\mathbf{x}$ from another point $\mathbf{x}_i$ with local feature relevance $\mathbf{w}_i$ is defined as (Eq. 3.5 in [35] $D_{\mathbf{w}_i}(\mathbf{x}_i,\mathbf{x}) = \sum_{l=1}^{d} w_{il}\frac{|x_{il}-x_{il}|}{s_l}$, where $s_l = \frac{1}{\binom{n}{2}}\sum_{i<j}|x_{il}-x_{jl}|$is the average l^{th} attribute distance. Estimation points $\mathbf{y}_{t+1}$ are computed as (Eq. 3.21 in [35]:

$$\mathbf{y}_{t+1} = \frac{\sum_{i=1}^{n}\frac{\mathbf{x}_i}{h_i^{d+2}}g(||\frac{D_{\mathbf{w}_i}(\mathbf{x}_i,\mathbf{y}_t)}{h_i}||^2)}{\sum_{i=1}^{n}\frac{1}{h_i^{d+2}}g(||\frac{D_{\mathbf{w}_i}(\mathbf{x}_i,\mathbf{y}_t)}{h_i}||^2)}$$

A sampling-based approximation of WAMS, called F-WAMS, was also proposed for large datasets [35]. F-WAMS performs weighted adaptive mean shift on a random subset of the data $\{\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_m\}$, obtaining a set of initial clusters $\{C_1,C_2,\ldots,C_k\}$, and local feature relevance for the sampled data: $\mathbf{w}_1,\mathbf{w}_2,\ldots,\mathbf{w}_m$. Each remaining data point $\mathbf{x}_j$ is assigned to the cluster C_i that minimizes the weighted distance $D_{\mathbf{w}_i}(\mathbf{x}_i,\mathbf{x}_j)$ between $\mathbf{x}_j$ and a cluster member $\mathbf{x}_i$.

4. Characterization of Hubs

In this section, we provide experimental evidence showing the properties of hubs which make them good candidates as seeds of subspace clusters. We present new geometric relationships between hubs and subspace clusters, and propose meta-features based on the *k*-NN graph of data to characterize bad hubs. Based on the identified relationships, we present our proposed hub-based algorithm in Section 5.

We use the simulated and real datasets listed in Table 1 and described in Table 5 to evaluate the characteristics of hubs. We generated two different simulated datasets. Toy1 consists of two Gaussian clusters of 100 dimensions. One Gaussian has a mean of one along each dimension, variance $\sigma^2 = 1$ along the first 40 dimensions $d_1 : d_{40}$, and $\sigma^2 = 5$ along the remaining 60 dimensions $d_{41} : d_{100}$. The other Gaussian has a mean of four along each dimension, variance $\sigma^2 = 5$ along dimensions $d_1 : d_{60}$, and $\sigma^2 = 1$ along $d_{61} : d_{100}$. Both Gaussians have a diagonal covariance matrix. Each cluster consists of 1000 points.

Toy2 consists of two spherical Gaussians of dimensionality 40 and 60, respectively, augmented with noisy features to form an embedding dimensionality of 100. The relevant dimensions of the two subspace clusters are $d_1 : d_{40}$ and $d_{41} : d_{100}$, respectively. The means of each dimension of the two clusters are one and three, respectively. Both Gaussians have a diagonal covariance matrix and the variance of the relevant dimensions for both is four. The augmented dimensions of each cluster are generated from a uniform distribution in the range $[\mu - 3\sigma, \mu + 3\sigma]$, where μ and σ are the mean and standard deviation of the respective Gaussian. Again, each cluster consists of 1000 points.

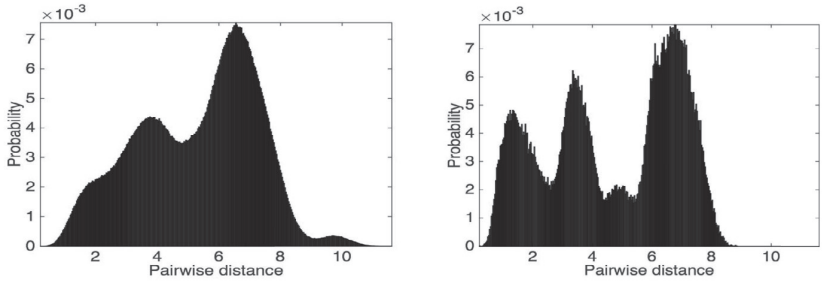


Fig. 1. Pen-3: (a) Histogram of all pairwise distances; (b) Histogram of pairwise distances among hubs.

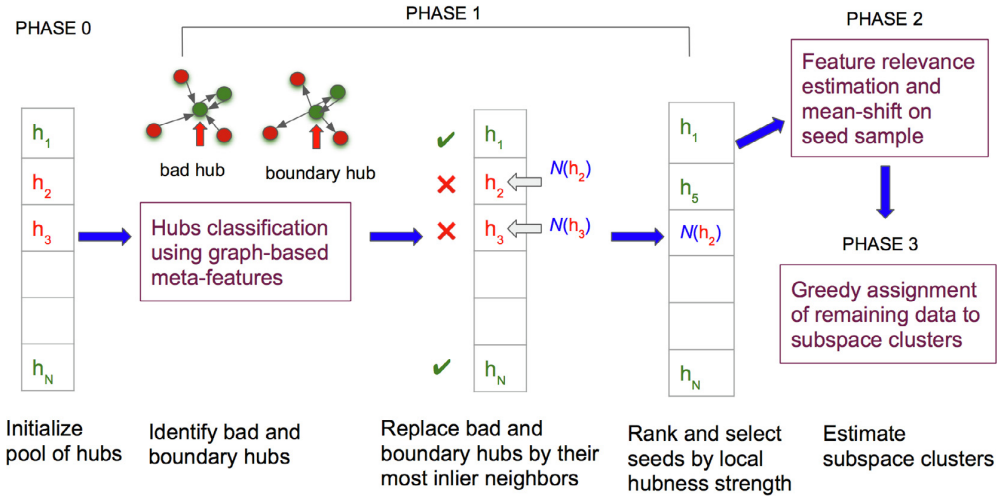


Fig. 2. Overview of the proposed algorithm (H-WAMS).

4.1. Hubs and distance distributions

The authors in [29] have shown that the pairwise distance distribution of hubs manifests a better separation of clusters than the same distribution computed using all points. Fig. 1 illustrates this phenomenon for the real data Pen-3. The histograms in (a) and (b) show the pairwise distances computed using all data and hubs only, respectively. The resulting distributions are multimodal, thus reflecting the clustering structure of the data. The modes obtained from hubs are more pronounced, the corresponding components have reduced variance, and are better separated from one another. This pattern suggests that initializing the clustering process using only hubs can facilitate the finding of good cluster centroids.

4.2. Hubs and subspace clusters

Here we investigate the relationship between hubs and subspace clusters. We adopt the concept of weighted cluster given in [8], defined as a subset of data points, together with a vector **w** of weights, such that the points are closely clustered according to the L_2 norm distance weighted using **w**.

For the analysis, we construct weighted clusters by assigning an ideal weight distribution to the features within each *true* cluster in the data. The weight assigned to a feature is inversely proportional to the variance of the feature within the corresponding cluster. Specifically, the weight w_i for feature f_i is $w_i = \frac{1}{var(f_i)}$. The weights of each cluster are normalized so that their values are within the range $(0, 1)$ and sum to 1. Local hubs in each cluster are computed with respect to the weighted distances. The global hubs, which are the hubs in the full-dimensional space, are computed using unweighted distances.

Table 1 shows the percentage of global hubs retained as local hubs within the corresponding *true* weighted clusters for simulated and real data. The first column gives the total percentage of global hubs retained as local hubs within the corresponding weighted clusters. The second and third columns show the total percentage of good (and bad) global hubs which are retained as local hubs within their weighted clusters. The hubness threshold for real datasets is set to $N_k > \mu + \sigma$ due to the large presence of bad hubs. We observe that a large percentage of global hubs are retained as local hubs within their corresponding weighted clusters. Also, the percentage of global bad hubs which are retained is much lower than the percentage of retained global good hubs. Some datasets such as Pen and COIL do not have any global bad hubs, while Caltech does not have any global good hubs. Hence the percentage of retained good/bad hubs for these datasets is denoted by N/A. On the other hand, a value of 0 means that none of the global good/bad hubs emerge as local hubs within subspaces, hence they are not retained.

The above results suggest that global hubs are good candidates as seeds to estimate the subspaces they belong to. Since global hubs tend to be distributed across clusters, they can be leveraged for a data-driven estimation of subspace clusters, without prior knowledge of the number of clusters in the data.

4.3. Bad and boundary hubs

Bad hubs negatively affect machine learning tasks which rely on nearest neighbor algorithms. Bad hubs perform poorly as cluster

seeds as they could attract many data points of different classes in the same cluster. In addition, a good hub may lie near the boundary of a class; as such, it will also perform poorly as seed of a cluster. We define a hub as *boundary* if more than 50% of its k-nearest neighbors belong to a class that is different from the hub's class.

In this section we describe the design and evaluation of a meta-feature based classifier to identify bad and boundary hubs. The classifier is used to improve seed selection in our proposed subspace clustering algorithm, Hubness-driven Weighted Adaptive Mean Shift (H-WAMS, Phase 1), which is described in detail in Section 5. An overview of H-WAMS, is shown in Fig. 2. H-WAMS involves the following steps: initialize a pool of global hubs to serve as subspace seeds (Phase 0); identify and replace bad and boundary hubs in the seed pool (Phase 1); rank and select seeds from the pool by a measure of local hubness score (Phase 1); and estimate subspace clusters (Phases 2 and 3).

Since clustering is an unsupervised task, the identification of bad and boundary hubs is a difficult challenge. A solution to this problem is to characterize bad and boundary hubs using meta-features, and train a classifier on labeled samples collected from other data sources. We propose to use meta-features defined on the data kNN graph to discriminate bad or boundary hubs from the remaining good hubs. (The latter class includes the good hubs which are not boundary, and as such they are good in a strong sense.) In the following, we introduce the meta-features we define, and an empirical evaluation of the resulting classification process.

4.3.1. Meta-features

A kNN graph representation of the data is used to extract meta-features from the immediate neighborhood of a hub. Let $D = \{\mathbf{x}_i\}_{i=1}^{n}$ be a collection of data points. Let $G = (V, E, w)$ be the kNN graph of D, where $V = \{v_1, v_2, \ldots, v_n\}$ is a set of n vertices corresponding to the data points $\mathbf{x}_i \in D$. $E = \{(v_i, v_j) | v_i \in V \wedge v_j \in k\text{NN}(v_i)\}$ is a set of directed edges, where kNN(v_i) denotes the vertices in G corresponding to the k nearest neighbors of $\mathbf{x}_i$. $w : E \rightarrow I, I = \{1, 2, \ldots, k\}$, is a function that assigns weights to edges: $w((v_i, v_j)) = r$ iff v_j is the r^{th} closest nearest neighbor of v_i $(r = 1, \ldots, k)$. We define the immediate neighborhood of a node v_i as

$$\text{IN}(v_i) = \{v_j | ((v_i, v_j) \in E) \vee ((v_j, v_i) \in E)\}$$

where v_i and v_j are the nodes of G corresponding to data points $\mathbf{x}_i$ and $\mathbf{x}_j$, respectively. The subgraph of the immediate neighborhood of a hub includes its k-nearest neighbors, its reverse k-nearest neighbors, and the edges between them.

We propose 12 meta-features to characterize hubs; they can be divided into three groups: measures of centrality, measures of hubness, and measures of density. The list of all meta-features is given in Table 2. Features 1 through 4 are measures of centrality. They signify whether a hub is located in an interior region or near the boundary of classes. LID and LCC are explained below in detail. EN-r and WE-r are defined based on previous findings [3], which state that power laws govern the distribution of edge count vs. node count, as well as the distribution of edge weight vs. edge count, in the one-step neighborhood of a node (a.k.a. egonet). Hence, the above ratios are a measure of the strength of power laws within the immediate neighborhood of a hub. Lower values indicate neighborhoods which are sparsely connected, and hence a higher possibility for them to be near boundaries. Features 5 through 9 are measures of hubness of the kNNs and RkNNs of a hub, and are included to identify strong good and bad hubs. The intuition is that neighbor nodes of a strong good hub would also have high hubness scores, as they are likely to occur in dense interior regions. On the other hand, the majority of the RkNNs of a strong bad hub may belong to surrounding classes, and they are likely to be positioned near class boundaries and have low hubness scores. The positive skewness in the distribution of hubness scores of RkNNs is likely to be higher for bad hubs, as the majority of its RkNNs is likely to have low hubness scores. Features 10 through 12 are different measures of density, meant to capture the bad or boundary nature of hubs. Diff_d measures the difference in local densities for a hub and its RkNNs. The neighborhood around an interior hub is expected to be dense, and therefore smaller differences in density than for hubs near boundaries are expected. Diff_{rank} is explained below in detail. Var(IN) is introduced to identify the presence of points that belong to different classes within the same neighborhood, as measured by the variance of data features in the immediate neighborhood of a hub. Below we provide the details of features 1, 2, and 11.

Table 2
List of meta-features computed on the immediate neighborhood IN of a hub.

#	Type	Feature	Definition
1		LID	Local Intrinsic Dimensionality
2	Centrality Measures	LCC	Local Clustering Coefficient
3		EN-r	Ratio of edge count to node count in IN
4		WE-r	Ratio of edge weight to edge count in IN
5		Avg (N_{kNN})	Mean hubness score of kNNs
6	Hubness Measures	Avg (N_{RkNN})	Mean hubness score of RkNNs
7		Var (N_{kNN})	Variance of hubness scores of kNNs
8		Var (N_{RkNN})	Variance of hubness scores of RkNNs
9		Skew (N_{RkNN})	Skewness of hubness scores of RkNNs
10		Diff_d	Difference in density: $\text{Diff}_d(h) = q(h) - 1/\lvert R k\text{NN}\rvert \sum_{\mathbf{r} \in R k\text{NN}} q(\mathbf{r}), q(\mathbf{x}) = 1/d_k(\mathbf{x}), d_k(\mathbf{x})$ is the distance of $\mathbf{x}$ to its k^{th} nearest neighbor.
11	Density Measures	Diff_{rank}	Relative difference in rank among the kNNs of a hub, which are also RkNNs of the same.
12		Var(IN)	Normalized mean variance of data features in IN: Var(IN)= avg(var(IN))/|IN|.

1. *Local Intrinsic Dimensionality* (LID): it's a local measure of intrinsic dimensionality, defined for each data point. It measures the expansion of the cumulative distribution of pairwise distances for increasing radii around a data point [17,18]. LID has also been characterized as a measure of data inlierness [16,28]. A lower value of LID indicates higher inlierness. [4] proposed an estimator for LID based on the neighborhood of a data point, which we use for our computation. The neighborhood size for LID estimation is set to $k = 5 \times \sqrt{n}$, in order to allow the computation to stabilize. The LID estimator is defined as:

$$\text{LID}(\mathbf{x}) = -\left(\frac{1}{k}\sum_{i=1}^{k} ln \frac{d_i(\mathbf{x})}{d_k(\mathbf{x})}\right)^{-1} \tag{1}$$

   where $d_i(\mathbf{x})$ is the distance of $\mathbf{x}$ to its i^{th} nearest neighbor.
2. *Local Clustering Coefficient* (LCC): it measures the degree to which nodes cluster together. In other words, it indicates the amount of transitivity in the neighborhood, which is measured by the density of triangles in a network. Based on the definition in [31], we compute the value of LCC for data point $\mathbf{x}$ as follows:

$$\text{LCC}(\mathbf{x}) = \frac{2|\{(v_j, v_k) | (v_j, v_k \in \text{IN}(\text{node}(\mathbf{x})) \wedge ((v_j, v_k) \in E)\}|}{|\text{IN}(\text{node}(\mathbf{x}))|(|\text{IN}(\text{node}(\mathbf{x}))| - 1)} \tag{2}$$

   where node$(\mathbf{x})$ denotes the node of G corresponding to $\mathbf{x}$.

Table 3
Evaluation of the SVM classifier with meta-features to detect good vs. bad/boundary hubs. Pen and COIL do not have any bad/boundary hubs.

Test Data	Mean F1-score (Training)	F1-score (Test)	Recall (Bad/Boundary)	Recall (Good)
Diabetes	0.83	0.57	0.45	0.82
Letter	0.85	0.54	0.40	0.59
Pen-3	0.85	0.87	N/A	0.77
Image	0.84	0.54	0.29	0.71
Waveform1	0.84	0.79	0.73	0.40
Sonar	0.83	0.69	0.71	0.25
Musk1	0.84	0.68	0.68	0.62
Musk2	0.83	0.61	0.49	0.86
mfeat-factors	0.85	0.70	0.62	0.69
mfeat-pixels	0.85	0.61	0.60	0.61
ISOLET	0.83	0.80	0.76	0.85
COIL	0.84	0.75	N/A	0.59

3. *Difference in Neighbor Rank* (Diff_{rank}): it measures the relative difference in rank (i.e., edge weight in the kNN graph) for points that are both kNNs and RkNNs of a hub. When such a point has a smaller kNN rank (i.e., it is closer) and a higher RkNN rank w.r.t. a hub, it is indicative of a sparser region around the hub and a denser region around the point, and the hub is likely to be a bad hub. On the other hand, when a point has a higher RkNN rank and a lower kNN rank, it is indicative of a denser region around the hub, and the hub is likely to be a good hub in an interior region. We formally define Diff_{rank} as follows:

$$\text{Diff}_{rank}(\mathbf{x}) = \begin{cases} \frac{\sum_{\mathbf{s}\in S}[w((\mathbf{s},\mathbf{x}))-w((\mathbf{x},\mathbf{s}))]}{w((\mathbf{x},\mathbf{s}))} & \text{if} \quad S \neq \varnothing \\ k & \text{otherwise} \end{cases} \tag{3}$$

where $S = \{\mathbf{s} | \mathbf{s} \in k\text{NN}(\mathbf{x}) \wedge \mathbf{s} \in \text{R}k\text{NN}(\mathbf{x})\}$ and $w()$ is the edge weight defined in Section 4.3.1. The upper-bound of Diff_{rank} is $k-1$, where k is the neighborhood size of the kNN graph. Hence, we set $\text{Diff}_{rank}(\mathbf{x}) = k$ when $S = \varnothing$.

4.3.2. Hubs classification and evaluation

To evaluate the effectiveness of the defined meta-features to classify hubs, we train a support vector machine (SVM) with an RBF kernel. The training and test data consist of hubs represented by the meta-features described in Table 2. As such, each hub becomes a 12-dimensional vector $\mathbf{h}_i$. Labels of hubs are obtained using auxiliary data designed for supervised learning. Bad or boundary hubs are assigned class label 1, and the remaining good hubs are given class label -1. This gives a training dataset $T = \{(\mathbf{h}_i, y_i)\}$, where $y_i \in \{-1, 1\}$ and $\mathbf{h}_i \in \Re^{12}$. The trained classifier can then be used to predict the nature of a hub point in an unsupervised setting.

We use the real datasets given in Table 5 for training and evaluation. For each of the 12 datasets, we construct a training dataset using the hubs of the remaining 11. We then perform testing on the hubs of the left out dataset. For example, when we test the classifier on COIL, we train the model using the hubs of the other datasets (Diabetes, Letter, Pen, Image, Waveform1, Sonar, Musk1, Musk2, mfeat-factors, mfeat-pixels, and ISOLET). For the purpose of evaluation, we assume that while testing on a given dataset, the labels of the remaining datasets are known. However, our classifier can be trained on any sets of data whose ground truth labels are available, i.e. any repository of datasets for supervised learning. In practice, the classifier needs to be trained just once, and can then be deployed to identify bad hubs in data for unsupervised learning. The computation of hub meta-features is unsupervised. The parameters of the classifier are tuned by cross-validation within the training data. Thus, the label information of hubs in test data is not used to train or fine-tune the classifier, and is not required.

The meta-features derived from each dataset are rank-normalized. Each meta-feature value is replaced by its rank among the observed values, and is normalized to the range $(0, 1)$ so that the values sum to 1. Rank-normalization enables the comparison of data generated from different domains, and have been used to normalize gene expression data (e.g., [20]. The best values of the soft-margin parameter v and of the kernel coefficient γ of the SVM are selected using a grid search with 5-fold cross validation on the training data T, where $v \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$ and $\gamma \in [\frac{1}{|T|}, \frac{1}{(|T| \times var(T))}]$, where $var(T)$ is the variance of the data obtained by flattening T into a vector. In order to handle any class imbalance that may occur in different combinations of training data, we under-sample the majority class. We evaluate the performance of the classifier using the weighted F1-score, as the class distribution of hubs in the test data can be imbalanced. The weighted F1-score is defined as the weighted average of the F1-scores of each class, where the weight of a class is the proportion of instances belonging to that class [36].

Table 3 shows the mean weighted F1-score obtained using 5-fold cross-validation on the training data, the F1-score on the test set, and the recall of positive (bad/boundary) and negative (good) classes. The F1-scores on validation and test sets indicate that our designed meta-features are indeed useful for the task. Note that this classifier is designed primarily to identify bad/boundary hubs, as they constitute bad subspace seeds. Table 3 also shows a good recall for both bad/boundary and good hubs for most datasets. We further demonstrate that our classifier can be pre-trained and deployed to classify hubs in previously unseen test data. To this end, we evaluate one of the pre-trained classifiers from Table 3 to classify hubs in the datasets summarized in Table 5, specifically Abalone, Pen-10, Caltech-20, Caltech-100, DrivFace, and OVA_ Colon. The presence or absence of label information in test data does not affect the classifier predictions, as the classifier parameters are not fine-tuned using the test data. In particular, we use the classifier tested on Sonar (i.e., the classifier trained on the datasets of Table 3, Sonar excluded), since it has the largest training data. Note that this classifier includes the hubs of Pen-3 as part of its training data. Hence, in order to classify the hubs of Pen-10, we use the classifier tested on Pen-3. Table 4 shows the results. Our design of the classifier using meta-features facilitates easy access to labeled data needed for its training. One can collect as much hub data as needed from labeled synthetic or real datasets. Further evaluation in Section 6.3 demonstrates the effectiveness of this classifier in improving the seeding for subspace clustering.

5. Hubness-driven weighted adaptive mean shift

Subspace clustering in high-dimensional data is negatively affected by the curse of dimensionality by which distance and den-

Table 4
Evaluation of the SVM classifier as a stand-alone model on test data. In Abalone, Caltech-20, and Caltech-100 all hubs are labeled as bad/boundary.

Test Data	F1-score (Test)	Recall (Bad/Boundary)	Recall (Good)
Abalone	0.62	0.44	N/A
Pen-10	0.73	0.45	0.75
Caltech-20	0.61	0.44	N/A
Caltech-100	0.81	0.68	N/A
DrivFace	0.57	0.44	0.53
OVA_ Colon	0.70	0.89	0.44

Table 5
Summary of datasets.

Data	# instances	# dimensions	# classes
Toy1	2000	100	2
Toy2	2000	100	2
Diabetes	768	8	2
Abalone	4177	8	29
Letter	2263	16	3
Pen-3	3165	16	3
Pen-10	10992	16	10
Image	2310	19	7
Waveform1	5000	21	3
Sonar	208	60	2
Musk1	476	168	2
Musk2	6598	168	2
mfeat-factors	2000	216	10
mfeat-pixels	2000	240	10
ISOLET	1200	617	5
COIL	360	1024	5
Caltech-20	1067	1066	20
Caltech-100	5233	5232	100
DrivFace	606	6400	3
OVA_ Colon	1545	10935	2

sity measurements become less meaningful. In this work we exploit an inherent characteristic of high-dimensional data known as the hubness phenomenon, to improve the quality of subspace clustering in high-dimensional spaces. Building on the characterizations of hubs presented in Section 4, we propose a selective sampling method and an algorithmic framework to leverage hubs as seeds for mode-seeking subspace clustering algorithms that rely on distance or density measurements. We observe that the inherent geometric properties of hubs makes them good seeds for subspace clusters and facilitates discriminative distance measurements. We propose a hub-based subspace clustering algorithm, namely, Hubness-driven Weighted Adaptive Mean Shift (H-WAMS), to estimate subspace clusters. H-WAMS adapts Weighted Adaptive Mean Shift (WAMS) [35], which is a non-parametric, mean shift based subspace clustering algorithm, to leverage a hub-based seed sample. Section 3.2 provides the background details for WAMS.

5.1. Selective sampling

In this section we describe our proposed hub-based seeding strategy. The characterizations provided by [29] suggest that global hubs are not effective to uniformly represent classes with variation in densities. Local hubness ranking is required to identify hubs with high centrality within their respective classes. However, the true local hubness ranking cannot be observed in unsupervised learning, as the data labels are not available. In view of the above challenges, we design a ranking and selection strategy to serve as a proxy for the true local hubness ranking. We consider an initial *seed pool* of data (Phase 0 of Fig. 2), from which seeds are selected. The seed pool includes the data points whose global hubness score N_k is above a certain threshold. The threshold is set to $\mu + 2\sigma$ for simulated data and to $\mu + \sigma$ for real data. The threshold is lowered for real data due to a larger presence of bad hubs. The value of k for nearest neighbor computation is set to $k = \sqrt{n}$, where n is the number of instances in the data. Our seed selection consists of the following steps (Phase 1 of Fig. 2):

1. *Prediction of bad/boundary hubs:* As discussed, using bad hubs as seeds is detrimental to clustering. Hence, we apply the classification framework presented in Section 4.3 to the hubs in the seed pool, and thus predict bad/boundary hubs.
2. *Replacement of predicted bad/boundary hubs:* Some classes in the data may only contain bad/boundary hubs, and hence may not be represented in the pool after the removal of such hubs. Therefore we use a heuristic to represent such classes in the pool. We replace a predicted bad/boundary hub by one of its k-nearest neighbors, which is not already in the pool and has the lowest local intrinsic dimensionality (LID) among the neighbors. Ideally, the selected point belongs to the same class as the bad/boundary hub, and is located far from the class boundary. We leverage the locality assumption which states that the neighborhood of a data point share the same class as the data point. In order to ensure inlierness of the replacing point, we select the neighbor with the smallest LID score (Eq. (1)).
3. *Ranking and selection:* We rank the samples in the seed pool in descending order using a new score, the *local hubness strength*, denoted as Loc-N_k. The Loc-N_k of a point $\mathbf{x}$ measures the percentage of its reverse k-nearest neighbors with global hubness score lower than that of $\mathbf{x}$:

$$\text{Loc} - N_k(\mathbf{x}) = \frac{\sum_{\mathbf{y} \in \text{R}k\text{NN}(\mathbf{x})} I(N_k(\mathbf{x}), N_k(\mathbf{y}))}{|\text{R}k\text{NN}(\mathbf{x})|}$$

$$I(p, q) = \begin{cases} 1 & p \geqslant q \\ 0 & otherwise \end{cases}$$

$\text{Loc} - N_k \in [0, 1]$. This score ensures that data points with high hubness score within their local neighborhood are ranked higher, even if their global hubness score is not among the highest. Since hubs are expected to emerge near centroids of data clusters [37], hubs with a high Loc-N_k value are likely to be found in each cluster. Thus, ranking by the local hubness strength increases the probability of representing all classes in the seed sample. This process is useful for data with unequal class densities.

5.2. Subspace estimation

We now describe the estimation of subspace clusters using the hub-based seed sample. In Phase 2 (see Fig. 2), local latent subspaces are computed for each seed point. We apply the Weighted Bandwidth algorithm introduced in [35] to find local subspaces. A feature weight distribution is learned locally at each seed point to minimize the dispersion within its k-nearest neighbors. We construct an adaptive k-neighborhood graph over the seed sample (described in Section 5.2.1), to define the neighbors of each seed point. The resulting feature weights are in the range $(0, 1)$ and they sum to one. Weighted adaptive mean shift (WAMS, [35] is then performed on the seed sample to find density modes. The adaptive bandwidth for a each seed point is computed as the distance to the farthest neighbor within its local subspace. The modes resulting from mean-shift are grouped to form a clustering of the seed sample, such that if two seed points have the same mode, they belong to the same cluster. In the final phase, a greedy assignment is used to cluster the remaining data points. Each data point which

is not in the seed sample is assigned to the subspace cluster of its nearest seed point. The final clustering uses weighted distances between data points, computed with the feature weight distributions learned on the seed sample. The pseudo code for our algorithm is provided in Algorithm 1.

H-WAMS addresses the following challenges posed by the curse of dimensionality:

Algorithm 1 H-WAMS

input: Data $D = \{\mathbf{x}_i\}_{i=1}^{n} \in \Re^d$, size of seed sample *seedSize*, # mean shift iterations *maxIter*
output: Clustering C, Modes M, Cluster weights W

Phase 0: Hubness Computation
$k := \text{sqrt}(n)$
N_k, Loc-$N_k := \text{computeHubs}(D, k)$

Phase 1: Seeding
seedPool $:= \{x | N_k(x) > \mu(N_k) + \sigma(N_k)\}$ ▷ Initialize seed pool
metaF := extractMetaFeatures(seedPool)
badOrBoundary := SVM(metaF) ▷ Detect bad/boundary hubs
replaceList := ∅
for **b** ∈ badOrBoundary **do** ▷ Replace bad/boundary hubs
  candidates := kNN(**b**) \ seedPool
  replaceList := replaceList ∪ arg min {LID(**c**)|**c** ∈ candidates}
end for

seedPool := seedPool \ badOrBoundary
seedPool := seedPool ∪ replaceList
seedSample := ∅
while size(seedSample) < *seedSize* **do** ▷ Rank and select seed sample
  nextSeed := arg max $\{\text{Loc-}N_k(\mathbf{s}) | \mathbf{s} \in \text{seedPool}\}$
  seedSample := seedSample ∪ nextSeed
  seedPool := seedPool \ nextSeed
end while

Phase 2: Weighted Adaptive Mean Shift
for **s** ∈ seedSample **do**
  $k_{\text{adapt}} := \text{adaptiveNeighborCount}(\mathbf{s}, \text{seedSample})$
  $\mathbf{w}_i, \mathbf{h}_i := \text{WeightedBandwidth}(\mathbf{s}, k_{\text{adapt}})$ ▷ Compute feature weights and bandwidth
end for
$(M, C) := \text{WAMS}(\text{seedSample}, W, \mathbf{h}, \textit{maxIter})$ ▷ Compute modes and seed clusters

Phase 3: Clustering
for **r** ∉ seedSample **do**
  for **s** ∈ seedSample **do**
    $d_w(\mathbf{r}, \mathbf{s}) = D_{W(s)}(\mathbf{s}, \mathbf{r})$ ▷ Compute weighted distances
  end for
  $C(\mathbf{r}) := C(\arg\min \{d_w(\mathbf{r}, \mathbf{s}) | \mathbf{s} \in \text{seedSample}\})$ ▷ Cluster remaining data
end for

1. *High-dimensional data is often embedded as clusters within subspaces, with different combinations of features being relevant to different clusters.* Our proposed subspace clustering algorithm handles this challenge by learning a clustering of the data, as well as feature relevance for each cluster. The feature relevance for a cluster is computed as the mean of the feature weight distributions learned for data points belonging to the cluster. H-WAMS uses weighted distances to assign data points to clusters. This enables the clustering of data points according to their feature relevance, and the discovery of latent subspaces.
2. *Distance and density estimation are less reliable in high dimensional spaces.* H-WAMS computes weighted distances between data points, and weighted adaptive bandwidths of the Gaussian kernel for density estimation in mean-shift. Weighted distances and bandwidths take into account the feature relevance between data points. Hence, they are informative of the subspace structure of the data, and result in more reliable estimates of distance and density. Furthermore, our method leverages hubs, which emerge in high-dimensional data and have useful clustering properties [34,37]. Performing mean-shift only on the seed sample comprised of hubs also facilitates more discriminative distance measurements, as discussed in Section 4.

5.2.1. Adaptive k-neighborhood

The feature weight distribution learned for a seed point is affected by the purity of its k-neighborhood, which in turn is affected by the class imbalance in the seed sample. Hence we compute adaptive k-neighborhoods for each seed point. An initial set of k-neighbors is computed for each seed point, by setting $k = \sqrt{n}$. Let D_k denote the set of average k-neighbor distances of seed points. A radius $r = \mu + \sigma$ is computed, where μ and σ denote the mean and standard deviation of D_k, respectively. Each seed point is assigned those neighbors which are at a distance less than r. This process is useful to prevent the selection of distant neighbors belonging to another class for seed points situated in a low density region. We further refine the adaptive neighbors of a seed point using infor-

Table 6
Evaluation on simulated datasets. Standard deviations are given in parantheses. WAMS on Toy2 did not find any meaningful subspace clusters.

Evaluation Measures	Toy1			Toy2		
	H-WAMS	F-WAMS	WAMS	H-WAMS	F-WAMS	WAMS
Clustering Purity	**1.00**	0.79 (0.2)	**1.00**	**1.00**	0.57 (0.2)	0.50
NMI	**1.00**	0.56 (0.5)	**1.00**	**1.00**	0.12 (0.3)	0
KL Divergence	0.29	0.33 (0.1)	**0.05**	0.24	0.26 (0.04)	**0.03**
Avg. # Mean Shift Iterations	8.2	**5.79** (3.0)	5.28	**6.75**	7.24 (4.7)	8.1
Avg. Length of Mean Shift	**6.00**	12.34 (3.2)	8.14	**7.83**	14.38 (1.2)	13.81
# Clusters	**2**	1.6 (0.5)	**2**	**2**	1.18 (0.4)	1
Running time (sec)	0.48	**0.14** (0.01)	168.63	0.45	**0.21** (0.02)	210.15

Table 7
Clustering purity on real datasets.

Data	H-WAMS	HP-WAMS	F-WAMS	WAMS	ESC-FFS	SSC-OMP	S^3C	BDR-B	BDR-Z	LRR
Diabetes	**0.69**	0.68 (0.01)	0.66 (0.01)	0.67	0.65 (0.004)	0.65	0.65	0.65	0.65	0.65
Abalone	0.26	**0.27** (0.003)	0.24 (0.001)	0.23	0.25 (0.003)	0.22	0.24	0.26	0.26	0.23
Letter	**0.88**	0.87 (0.01)	0.70 (0.04)	0.81	0.67 (0.04)	0.42	0.54	0.74	0.73	0.44
Pen-3	**0.95**	**0.95** (0.001)	0.80 (0.04)	0.69	**0.95** (0.004)	0.43	0.74	0.90	0.90	0.70
Pen-10	**0.88**	0.84 (0.02)	0.63 (0.04)	0.69	0.84 (0.005)	0.26	0.68	0.77	0.77	0.69
Image	0.73	0.74 (0.02)	0.66 (0.03)	0.74	**0.77** (0.02)	0.40	0.69	**0.77**	**0.77**	0.55
Waveform1	**0.61**	0.55 (0.03)	0.57 (0.05)	**0.61**	0.50 (0.003)	0.35	0.34	0.51	0.51	0.35
Sonar	**0.64**	0.57 (0.04)	0.56 (0.04)	0.53	0.57 (0.03)	0.55	0.55	0.61	0.61	0.53
Musk1	**0.60**	**0.60** (0.01)	0.58 (0.02)	0.58	0.57 (8e-16)	0.59	0.57	0.58	0.57	0.57
Musk2	0.86	**0.87** (0.01)	0.85 (0.002)	0.85	0.85 (3e-16)	0.85	0.85	0.85	0.85	0.85
mfeat-factors	0.78	0.77 (0.01)	0.11 (0.02)	0.23	**0.89** (0.03)	0.63	0.83	0.67	0.67	0.60
mfeat-pixels	**0.86**	0.83 (0.01)	0.41 (0.05)	0.71	0.84 (0.05)	0.63	0.74	0.70	0.70	0.52
ISOLET	0.57	0.57 (0.01)	0.23 (0.04)	0.21	0.61 (0.02)	0.51	**0.64**	0.55	0.56	0.20
COIL	0.82	0.82 (0.01)	0.58 (0.1)	**0.84**	0.81 (0.03)	0.79	0.80	0.80	0.80	0.21
Caltech-20	0.14	0.13 (0.004)	0.14 (0.01)	0.11	0.12 (0.01)	**0.15**	**0.15**	0.07	0.12	0.12
Caltech-100	0.06	0.05 (0.005)	**0.08** (0.005)	-	0.06 (0.001)	0.06	0.06	0.03	0.06	0.05
DrivFace	**0.92**	**0.92** (0.001)	0.90 (0.01)	0.90	0.90 (1e-15)	0.90	0.90	0.90	0.91	0.90
OVA_ Colon	**0.94**	**0.94** (0.002)	0.83 (0.04)	-	0.81 (2e-16)	0.81	0.81	0.81	0.81	0.81
Mean	**0.68**	0.67	0.53	0.59	0.65	0.51	0.60	0.62	0.62	0.50

Table 8
NMI on real datasets.

Data	H-WAMS	HP-WAMS	F-WAMS	WAMS	ESC-FFS	SSC-OMP	S^3C	BDR-B	BDR-Z	LRR
Diabetes	**0.06**	0.04 (0.006)	0.02 (0.01)	0.04	0.05 (0.01)	0.001	9e-05	0.02	0.02	0.003
Abalone	**0.18**	0.17 (0.002)	0.15 (0.01)	0.17	0.15 (0.002)	0.10	0.11	0.15	0.16	0.13
Letter	**0.41**	0.39 (0.01)	0.28 (0.03)	0.33	0.40 (0.05)	0.04	0.26	0.37	0.36	0.18
Pen-3	0.54	0.54 (0.01)	0.43 (0.03)	0.44	**0.81** (0.01)	0.05	0.43	0.67	0.69	0.42
Pen-10	0.72	0.72 (0.01)	0.56 (0.02)	0.54	**0.76** (0.01)	0.13	0.64	0.70	0.70	0.67
Image	0.60	0.60 (0.01)	0.57 (0.03)	0.62	**0.73** (0.01)	0.23	0.58	0.70	0.69	0.52
Waveform1	0.34	0.36 (0.01)	0.26 (0.05)	0.32	0.30 (0.01)	7e-04	2e-04	**0.36**	0.37	4e-04
Sonar	0.05	0.02 (0.02)	0.02 (0.04)	0.01	0.02 (0.01)	0.02	0.03	**0.10**	0.08	0.01
Musk1	**0.03**	**0.03** (0.01)	0.02 (0.01)	0.02	0.01 (0.01)	0.02	3e-05	0.02	0.003	0.003
Musk2	**0.07**	**0.07** (0.01)	0.03 (0.01)	0.05	0.02 (0.01)	0.004	0.01	2e-05	0.007	0.01
mfeat-factors	0.75	0.74 (0.01)	0.01 (0.04)	0.22	0.81 (0.02)	0.54	**0.82**	0.60	0.59	0.68
mfeat-pixels	0.72	0.70 (0.01)	0.40 (0.04)	0.62	**0.78** (0.03)	0.59	**0.78**	0.57	0.57	0.59
ISOLET	**0.62**	0.61 (0.02)	0.01 (0.04)	0.004	0.55 (0.02)	0.26	0.53	0.37	0.41	0.01
COIL	0.70	0.69 (0.02)	0.66 (0.18)	**0.88**	0.76 (0.02)	0.83	0.79	0.87	0.87	0.03
Caltech-20	**0.11**	**0.11** (0.01)	0.08 (0.01)	0.08	0.07 (0.004)	0.09	0.10	0.03	0.07	0.08
Caltech-100	0.21	0.18 (0.01)	**0.25** (0.02)	-	0.21 (0.002)	0.21	0.19	0.04	0.20	0.13
DrivFace	0.11	0.11 (0.01)	0.05 (0.04)	0.07	0.01 (0.004)	0.07	0.05	0.06	**0.22**	0.01
OVA_ Colon	**0.27**	**0.27** (0.02)	0.14 (0.08)	-	0.02 (0.005)	0.002	0.07	5e-04	3e-06	0.002
Mean	**0.36**	0.35	0.22	0.28	**0.36**	0.18	0.30	0.31	0.33	0.19

mation from the meta-feature-based classifier. The classifier computes confidence values for each hub being in the good hub class (for an SVM classifier, the margin is converted to a confidence value by Platt scaling [33]. Confidence values are available for each seed hub; the seed points which replace a bad/boundary hub inherit the confidence value of the replaced hub. Neighbors with low confidence values likely belong to a different class than that of the seed point. For each seed point, we filter neighbors whose confidence value is less than $\mu - \sigma$, where μ denotes the mean confidence value of its adaptive neighbors and σ their standard deviation. This filtering process is useful to refine the neighborhoods of seed points near the boundary.

6. Empirical evaluation

6.1. Datasets

We evaluate our proposed algorithm on two simulated datasets and 18 real datasets. A summary of the datasets is given in Table 5. The simulated datasets are described in Section 4.

Table 9
Clustering purity and statistical significance results on real datasets across varying sample sizes.

		Sample Size						
Data	**Algorithm**	**1%**	**2%**	**3%**	**4%**	**5%**	**10%**	**15%**
Diabetes	HP-WAMS	**0.67** (0.02) *†	**0.67** (0.01) *†	**0.68** (0.01) *†	**0.68** (0.01) *†	**0.68** (0.01) *†	**0.68** (0.01) *†	**0.68** (0.01) *†
	ESC-FFS	0.65 (0.01)	0.65 (0.002)	0.65 (0.003)	0.65 (0.002)	0.65 (5e-16)	0.65 (5e-16)	0.65 (5e-16)
	F-WAMS	0.65 (0.01)	0.66 (0.02) ‡	0.66 (0.01) ‡	0.66 (0.01) ‡	0.66 (0.01) ‡	0.66 (0.01) ‡	0.66 (0.01) ‡
Abalone	HP-WAMS	0.25 (0.01)†	**0.25** (0.01) †	**0.25** (0.01)†	**0.26** (0.01) *†	**0.26** (0.01) *†	**0.26** (0.01) *†	**0.27** (0.003) *†
	ESC-FFS	**0.26** (0.004) *‡	**0.25** (0.004) ‡	**0.25** (0.003) ‡	0.25 (0.003) ‡	0.25 (0.003) ‡	0.25 (0.003) ‡	0.24 (0.003)
	F-WAMS	0.22 (0.01)	0.23 (0.01)	0.23 (0.01)	0.24 (0.01)	0.24 (0.01)	0.24 (0.01)	0.24 (0.01)
Letter	HP-WAMS	0.63 (0.1) †	**0.73** (0.1) *†	**0.80** (0.03) *†	**0.81** (0.03) *†	**0.83** (0.02) *†	**0.85** (0.01) *†	**0.87** (0.01) *†
	ESC-FFS	**0.66** (0.04) *‡	0.68 (0.04) ‡	0.67 (0.04) ‡	0.65 (0.03) ‡	0.64 (0.02) ‡	0.65 (0.01)	0.70 (0.02)
	F-WAMS	0.46 (0.1)	0.51 (0.1)	0.57 (0.1)	0.57 (0.1)	0.61 (0.1)	0.66 (0.1) ‡	0.70 (0.04)
Pen-3	HP-WAMS	0.83 (0.1) †	0.90 (0.1) †	0.89 (0.1) †	0.90 (0.1) †	0.92 (0.1) †	**0.94** (0.03) *†	**0.95** (0.001) *†
	ESC-FFS	**0.95** (0.01) *‡	**0.93** (0.1) *‡	**0.95** (0.003) *‡	**0.95** (0.003) *‡	**0.95** (0.002) *‡	0.93 (0.01) ‡	0.91 (0.01) ‡
	F-WAMS	0.74 (0.1)	0.78 (0.1)	0.78 (0.1)	0.78 (0.1)	0.78 (0.04)	0.80 (0.1)	0.80 (0.04)
Pen-10	HP-WAMS	0.76 (0.04) †	0.81 (0.03) †	0.80 (0.02) †	0.81 (0.03)†	0.82 (0.03) †	**0.84** (0.02) *†	**0.84** (0.02) *†
	ESC-FFS	**0.79** (0.003) *‡	**0.84** (0.001) *‡	**0.84** (0.01) *‡	**0.83** (0.003) *‡	**0.83** (0.01) ‡	0.76 (0.01) ‡	0.76 (0.004) ‡
	F-WAMS	0.45 (0.05)	0.49 (0.05)	0.49 (0.05)	0.53 (0.05)	0.54 (0.05)	0.60 (0.03)	0.63 (0.04)
Image	HP-WAMS	0.53 (0.1) †	0.59 (0.04) †	0.63 (0.04) †	0.63 (0.04) †	0.66 (0.03) †	0.71 (0.02) †	**0.74** (0.02) †
	ESC-FFS	**0.76** (0.02) *‡	**0.77** (0.01) *‡	**0.77** (0.02) *‡	**0.77** (0.01) *‡	**0.78** (0.02) *‡	**0.74** (0.01) *‡	**0.74** (0.003) ‡
	F-WAMS	0.35 (0.1)	0.43 (0.1)	0.50 (0.1)	0.50 (0.1)	0.53 (0.1)	0.63 (0.04)	0.66 (0.03)
Waveform1	HP-WAMS	**0.55** (0.1) *†	**0.56** (0.1) *†	**0.58** (0.1) *†	**0.56** (0.1) *†	**0.55** (0.04) *†	**0.55** (0.03) *	-
	ESC-FFS	0.49 (0.02)	0.50 (0.01)	0.50 (0.002)	0.50 (0.003)	0.50 (0.003)	0.50 (0.002)	0.50 (0.002)
	F-WAMS	0.49 (0.1)	0.52 (0.1)	0.53 (0.1) ‡	0.54 (0.1) ‡	0.53 (0.1)	**0.55** (0.1) ‡	**0.57** (0.1) ‡
Sonar	HP-WAMS	**0.59** (0.1) *†	**0.56** (0.04) †	**0.58** (0.1) †	**0.58** (0.1)	**0.57** (0.04) *	**0.57** (0.04) *	-
	ESC-FFS	0.56 (0.04) ‡	**0.56** (0.03) ‡	0.57 (0.03)	0.57 (0.03)	0.55 (0.03)	0.54 (0.01)	0.55 (0.03)
	F-WAMS	0.53 (3e-16)	0.54 (0.02)	0.56 (0.04)	0.56 (0.04)	0.55 (0.03)	0.55 (0.03)	**0.56** (0.04) ‡
Musk1	HP-WAMS	0.57 (0.02)	**0.59** (0.02) *†	**0.60** (0.02) *†	**0.60** (0.02) *†	**0.60** (0.02) *†	**0.61** (0.02) *†	**0.60** (0.01) *†
	ESC-FFS	0.57 (0.01) *	0.57 (3e-16)	0.57 (3e-16) ‡	0.57 (3e-16) ‡	0.57 (3e-16) ‡	0.57 (0.01) ‡	0.57 (0.002)
	F-WAMS	0.57 (0.01) †	0.57 (0.003)	0.57 (0.01)	0.57 (0.01)	0.57 (0.01)	0.57 (0.01)	0.58 (0.02) ‡
Musk2	HP-WAMS	0.85 (0.003) †	**0.86** (0.01) *†	**0.86** (0.01) *†	**0.86** (0.01) *†	**0.86** (0.01) *†	**0.87** (0.01) *†	**0.87** (0.01) *†
	ESC-FFS	0.85 (8e-16) *	0.85 (8e-16)	0.85 (8e-16)	0.85 (8e-16)	0.85 (8e-16)	0.85 (8e-16)	0.85 (9e-16)
	F-WAMS	0.85 (8e-16)	0.85 (8e-16)	0.85 (4e-16)	0.85 (8e-16)	0.85 (8e-16)	0.85 (0.003)	0.85 (0.002)
mfeat-factors	HP-WAMS	0.32 (0.1) †	0.52 (0.1) †	0.61 (0.1) †	0.59 (0.2) †	0.63 (0.2) †	**0.76** (0.03) *†	0.77 (0.01) †
	ESC-FFS	**0.89** (0.03) *‡	**0.90** (0.03) *‡	**0.89** (0.03) *‡	**0.88** (0.03) *‡	**0.86** (0.03) *‡	0.74 (0.03) ‡	**0.82** (0.04) *‡
	F-WAMS	0.11 (0.03)	0.12 (0.03)	0.11 (0.03)	0.11 (0.03)	0.11 (0.03)	0.11 (0.03)	0.11 (0.02)
mfeat-pixels	HP-WAMS	0.48 (0.1) †	0.63 (0.1) †	0.72 (0.04) †	0.76 (0.04) †	**0.79** (0.04) †	**0.83** (0.03) †	**0.83** (0.01) *
	ESC-FFS	**0.81** (0.1) *‡	**0.86** (0.1) *‡	**0.84** (0.1) *‡	**0.80** (0.04) *‡	0.75 (0.04) ‡	0.34 (0.03)	0.33 (0.03)
	F-WAMS	0.29 (0.1)	0.29 (0.1)	0.31 (0.1)	0.33 (0.1)	0.34 (0.1)	0.37 (0.1) ‡	0.41 (0.1) ‡
ISOLET	H-WAMS	0.48 (0.1) †	0.55 (0.1) †	0.49 (0.1) †	0.55 (0.1) †	0.51 (0.1) †	0.53 (0.1) †	0.57 (0.01) †
	ESC-FFS	**0.60** (0.01) *‡	**0.61** (0.01) *‡	**0.61** (0.02) *‡	**0.62** (0.02) *‡	**0.62** (0.04) *‡	**0.64** (0.04) *‡	**0.60** (0.1) *‡
	F-WAMS	0.21 (0.03)	0.21 (0.04)	0.21 (0.03)	0.21 (0.04)	0.21 (0.04)	0.21 (0.04)	0.22 (0.04)
COIL	HP-WAMS	0.48 (0.1) †	0.57 (0.1) †	0.61 (0.1) †	0.71 (0.1) †	0.74 (0.1) †	**0.82** (0.002) *†	-
	ESC-FFS	**0.75** (0.1) *‡	**0.80** (0.03) *‡	**0.81** (0.03) *‡	**0.79** (0.02) *‡	**0.79** (0.02) *‡	0.65 (0.1) ‡	**0.60** (0.004)
	F-WAMS	0.29 (0.1)	0.27 (0.1)	0.25 (0.1)	0.39 (0.2)	0.39 (0.2)	0.52 (0.1)	0.58 (0.1)
Caltech-20	HP-WAMS	0.09 (0.01)	0.10 (0.01) †	0.11 (0.01) †	0.11 (0.01) †	**0.12** (0.01) †	**0.13** (0.004) *†	-
	ESC-FFS	**0.12** (0.01) *‡	**0.12**(0.01) *‡	**0.12** (0.01)*‡	**0.12** (0.01) *‡	**0.12** (0.01) *‡	0.12 (0.01)	0.12 (0.01)
	F-WAMS	0.09 (0.01)	0.09 (0.01)	0.10 (0.01)	0.11 (0.01)	0.11 (0.01)	0.12 (0.01)	**0.14** (0.01). ‡
Caltech-100	HP-WAMS	0.03 (0.001)	0.04 (0.002) †	0.04 (0.002) †	0.05 (0.003) †	0.05 (0.004) †	-	-
	ESC-FFS	**0.06**(0.002) *‡	**0.06** (0.002) *‡	**0.06** (0.002) *‡	**0.06** (0.001) *‡	**0.06** (0.001) *‡	0.06 (0.001)	0.06 (0.001)
	F-WAMS	0.03 (0.003)	0.03 (0.003)	0.04 (0.003)	0.04 (0.003)	0.05 (0.004)	**0.07** (0.004) ‡	**0.08** (0.01) ‡
DrivFace	HP-WAMS	**0.90** (1e-15)	**0.90** (0.01) †	**0.91** (0.01) *†	**0.90** (0.01) †	**0.91** (0.01) *†	**0.91** (0.01) *†	**0.92** (0.01) *†
	ESC-FFS	**0.90** (1e-15)	**0.90** (1e-15) *	0.90 (1e-15)	**0.90** (1e-15) *	0.90 (1e-15) ‡	0.90 (1e-15) ‡	0.90 (1e-15) ‡
	F-WAMS	**0.90** (1e-15)	**0.90** (0.002)	0.90 (0.001)	**0.90** (0.003)	0.90 (0.004)	0.90 (0.001)	0.90 (0.001)
OVA_ Colon	HP-WAMS	0.84 (0.04)*	0.87 (0.05)*	**0.90** (0.05)*	**0.90** (0.04)*	**0.92** (0.04)*†	**0.94** (0.002)*†	-
	ESC-FFS	0.82 (0.01)	0.81 (2e-16)	0.81 (2e-16)	0.81 (2e-16)	0.81 (2e-16)	0.81 (2e-16)	0.81 (2e-16)
	F-WAMS	**0.90** (1e-15)‡	**0.90** (0.002)‡†	**0.90** (9e-4)‡†	**0.90** (0.003)‡†	0.90 (0.004)‡	0.90 (0.005)‡	**0.90** (0.006)‡
Mean	HP-WAMS	0.55 (0.3)	0.60 (0.3)	0.61 (0.3)	0.63 (0.3)	**0.64** (0.3)	**0.66** (0.3)	**0.67** (0.3)
across	ESC-FFS	**0.64** (0.3)	**0.65** (0.3)	**0.65** (0.3)	**0.64** (0.3)	**0.64** (0.3)	0.59 (0.3)	0.59 (0.3)
datasets	F-WAMS	0.45 (0.3)	0.46 (0.3)	0.47 (0.3)	0.49 (0.3)	0.49 (0.3)	0.51 (0.3)	0.53 (0.3)

We chose real datasets with sufficient variation in density, class distribution, and dimensionality. Except for COIL[1], Caltech[2], Diabetes[3], and OVA_ Colon[4], the remaining datasets are obtained from the UCI Machine Learning Repository [9]. COIL, Letter, and Pen-3 are sub-sampled as described in [35]. COIL originally consists of 100 classes, out of which the first 5 classes are selected, with 72 images in each class. Three digit classes (3, 8, and 9) are chosen for Pen-3, and letter classes (I, J and L) are chosen for Letter. Pen-10 comprises of the all samples in UCI Pen dataset. For ISOLET, the training data (isolet1 + 2+3 + 4 in the UCI repository) of five classes (A,B,C,D,E) are selected. The attributes 'musk name' and 'conformation name' are omitted for Musk1 and Musk2. The Caltech images are converted to 240×240 grayscale, and transformed by kernel PCA with Gaussian kernel. The number of images from each class is restricted to a maximum of 100. For Caltech-100, the classes {Background_Faces_Easy} are left out. The following classes are selected for Caltech-20: {dollar_ bill, pizza, stop _ sign, lamp, ceiling_ fan, soccer_ ball, metronome, watch, sunflower, yin_ yang, airplanes, strawberry, barrel, camera, brain, umbrella, accordion, scissors,

[1] www.cad.zju.edu.cn/home/dengcai
[2] www.vision.caltech.edu/Image_ Datasets/Caltech101.
[3] www.kaggle.com/uciml/pima-indians-diabetes-database
[4] www.openml.org/d/1161

Table 10
Running times (in seconds) on real datasets.

Data	H-WAMS	HP-WAMS	F-WAMS	WAMS	ESC-FFS	SSC-OMP	S^3C	BDR-B	BDR-Z	LRR
Diabetes	0.49	0.45 (0.02)	0.23 (0.02)	4.30	**0.12** (0.004)	0.30	14.68	0.86	0.85	0.47
Abalone	6.78	7.22 (0.24)	2.47 (0.43)	142.61	1.97 (0.07)	**1.72**	3e+03	489.78	483.72	31.48
Letter	1.79	1.73 (0.03)	0.62 (0.08)	17.67	0.60 (0.02)	**0.38**	104.68	7.70	6.88	3.66
Pen-3	5.65	5.52 (0.3)	1.60 (0.20)	73.54	1.22 (0.03)	**0.60**	556.75	35.84	36.15	13.90
Pen-10	93.42	91.13 (0.96)	18.32 (1.76)	1.4e+03	17.31 (0.29)	**4.71**	1.2e+04	1e+03	1e+03	656.39
Image	3.54	3.49 (0.06)	2.28 (0.08)	21.07	2.17 (0.05)	**1.96**	175.78	117.21	117.89	6.62
Waveform1	18.37	21.31 (0.90)	7.43 (2.30)	382.68	3.52 (0.06)	**1.49**	1e+03	81.02	81.83	55.54
Sonar	0.14	0.10 (0.01)	0.06 (0.01)	0.70	**0.04** (0.001)	0.07	0.98	0.70	0.69	0.35
Musk1	1.15	0.62 (0.10)	0.23 (0.03)	9.79	**0.05** (0.002)	0.18	3.56	2.44	2.86	2.15
Musk2	94.60	69.25 (9.20)	68.93 (5.80)	-	97.50 (2.40)	**5.73**	4e+03	1e+03	1e+03	140.21
mfeat-factors	21.65	14.14 (2.30)	5.53 (0.80)	354.08	1.42 (0.04)	**0.53**	217.33	31.89	31.90	11.43
mfeat-pixels	25.11	17.00 (2.90)	6.06 (0.40)	449.53	1.28 (0.04)	**0.52**	257.56	15.73	15.77	8.53
ISOLET	16.45	10.72 (0.70)	5.45 (0.70)	323.25	**0.45** (0.01)	1.50	81.92	8.79	8.78	28.95
COIL	0.91	0.64 (0.07)	0.82 (0.30)	71.03	**0.08** (0.003)	0.22	22.49	1.91	1.91	8.76
Caltech-20	3.91	4.20 (0.35)	6.77 (0.79)	3e+03	**0.30** (0.11)	0.85	326.05	0.73	1.45	37.01
Caltech-100	259.23	270.91 (17.18)	1e+03 (110.68)	-	**6.70** (0.14)	90.46	2e+05	48.03	48.36	5e+03
DrivFace	25.61	21.84 (2.23)	29.73 (5.12)	3e+03	**0.44** (0.02)	0.88	1e+03	0.48	0.57	27.35
OVA_ Colon	209.05	211.38 (25.27)	632.94 (171.76)	-	**2.17** (0.04)	8.20	2e+04	2.81	3.42	284.98
Mean	43.29	41.72	105.86	616.68	7.63	**6.68**	1.3e+04	186.63	186.85	325.92

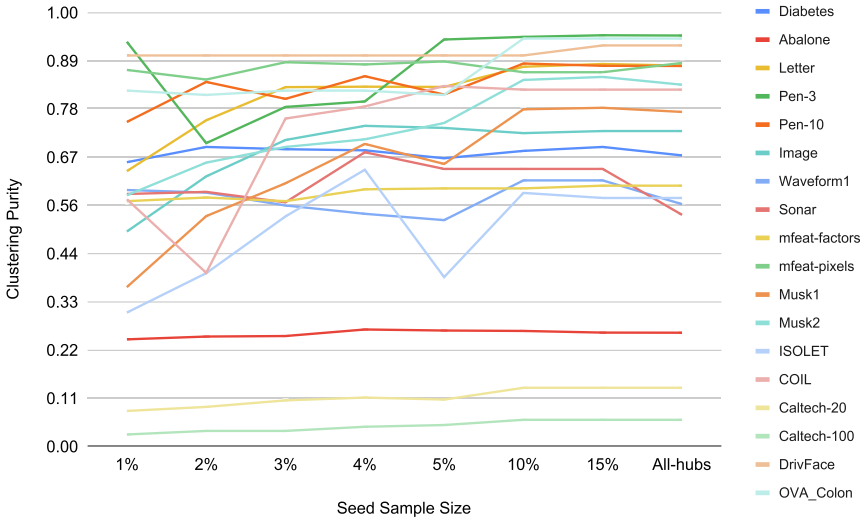


Fig. 3. Clustering purity vs. seed sample size for H-WAMS (best viewed in color).

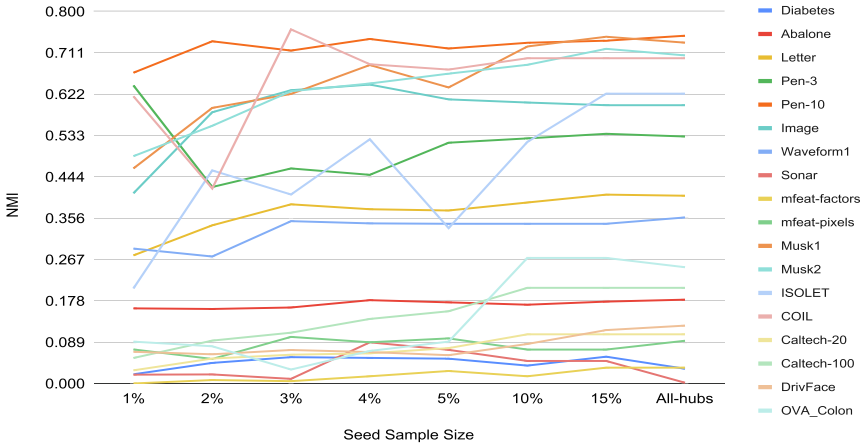


Fig. 4. NMI vs. seed sample size for H-WAMS (best viewed in color).

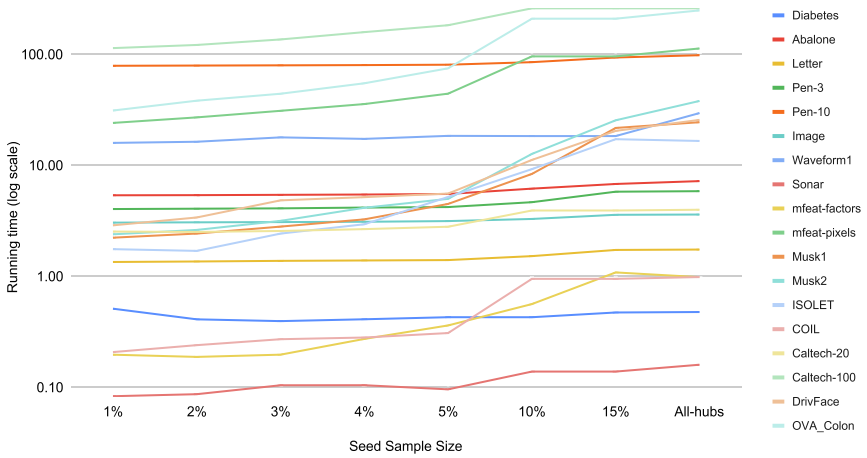


Fig. 5. Running time (in log scale) vs. seed sample size for H-WAMS (best viewed in color).

chair, cup}. OVA_ Colon is a benchmark gene expression data having 286 instances with colon tumor and 1259 instances without tumor. The features of the datasets are standardized using the Z-score, except for Sonar, COIL, and ISOLET, whose features did not vary in scale. Duplicate instances and invariant features are removed from each dataset.

6.2. Evaluation on simulated data

We evaluate the quality of clusterings and subspaces computed by H-WAMS on the simulated datasets. We compare H-WAMS against WAMS and F-WAMS algorithms. WAMS applies mean shift to all data points. F-WAMS [35] selects seed points randomly to approximate WAMS. For H-WAMS, the threshold for the seed pool is set to $\mu + 2\sigma$. We evaluate the algorithms using the following criteria: (1) Clustering purity [42]; (2) Normalized Mutual Information (NMI, [22]; (3) KL divergence between the learned feature weight distributions of clusters and their ideal feature weight distributions; (4) average number of mean shift iterations needed for the seed data points; (5) average length of mean shift for the seed points; (6) number of clusters found; and (7) running time.

Table 6 shows the results obtained for a seed sample size of 1% for Toy1 and 2% for Toy2, which are the respective minimum sample sizes that resulted in a perfect clustering for H-WAMS. Results of F-WAMS are averaged across 50 runs, and the corresponding standard deviations are reported. H-WAMS, instead, is deterministic. H-WAMS outperforms F-WAMS and achieves perfect clustering on the simulated data, while using a small sample of hubs.

To evaluate the quality of the estimated subspaces we proceed as follows. For each *true* cluster in the data, we compute the KL divergence between the learned feature weight and the ideal feature weight distributions. An ideal feature weight distribution for a cluster has weights inversely proportional to the variance of the features. We compute the learned feature weights for a *true* cluster as the average of the feature weights learned by an algorithm for the data points of the cluster. In the case of H-WAMS and F-WAMS, the learned weight distribution is averaged across the seed points of a cluster. The weight values learned by H-WAMS and F-WAMS are in the range $(0, 1)$ and sum to one. We then compute the KL divergence between the ideal and the learned weight distributions across clusters, as follows:

$$\mathrm{KL} = \sum_{c=1}^{n_c} \mathrm{KL}(P_c \| Q_c) = \sum_{c=1}^{n_c} \sum_{i} P_c(i) log \frac{P_c(i)}{Q_c(i)}$$

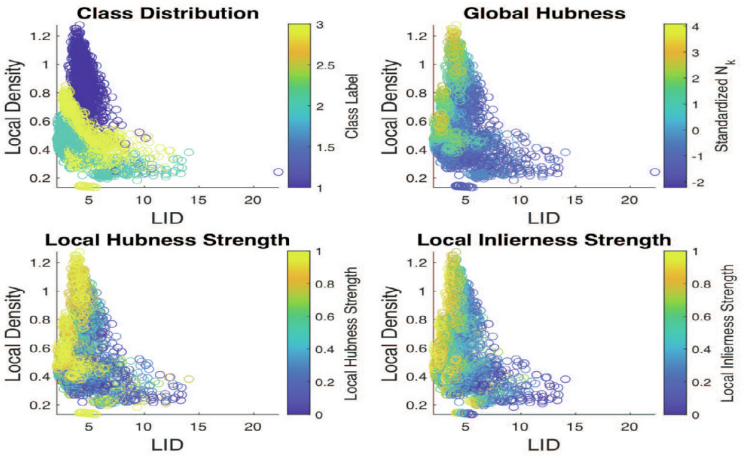
where n_c is the number of true clusters, P_c is the ideal weight distribution for cluster c, and Q_c is the learned weight distribution for cluster c.

Table 6 shows that the KL divergence values for the weight distributions learned by H-WAMS are smaller than those learned by F-WAMS. This shows that H-WAMS can learn weight distributions which are closer to the ideal ones compared to those found by F-WAMS. WAMS has the overall lowest KL divergence. This is because it uses all the data to compute the learned weight distribution. In contrast, the learned weight distribution for H-WAMS is averaged across the seed hubs only. Hence its KL divergence is higher than that of WAMS, but, as WAMS, it achieves a perfect clustering purity. Compared to F-WAMS, H-WAMS has smaller average length of mean shift, and finds the actual number of clusters in the data. This indicates that hubs are close to their respective modes (i.e., centroids of subspace clusters), and this enables the finding of more accurate clusters. The average number of mean shift iterations of the seed sample is lower for F-WAMS on Toy1. However, due to the lower clustering quality and higher average mean shift length, it's possible that mean shift on a random sample gets stuck in local maxima. Both WAMS and H-WAMS achieve perfect clustering on Toy1, but H-WAMS is significantly faster. We observe that WAMS on Toy2 did not find any meaningful clusters, hence its NMI is 0. This latter result indicates that mean-shift applied on selected data can achieve improved performance on noisy data.

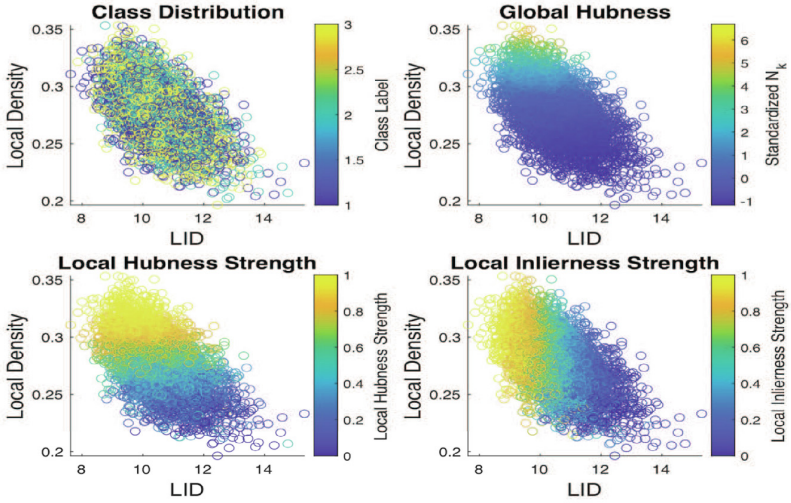
6.3. Evaluation on real data

We evaluate the clustering quality of hub-based seeding on real datasets using clustering purity and NMI. Due to a large presence of bad/boundary hubs in real data, for H-WAMS we set the threshold to choose the pool of hubs to $\mu + \sigma$. To evaluate the robustness of H-WAMS, we also consider a probabilistic variant of hub-based seeding; in this approach, each data point in the seed pool is assigned a weight equal to its local hubness strength (Loc-N_k). A seed sample is selected from the pool by *weighted random sampling* using Loc-N_k. We name this algorithm *Hubness Proportional Weighted Adaptive Mean Shift* (HP-WAMS). We compare our proposed hub-based methods, H-WAMS and HP-WAMS, with F-WAMS, WAMS, subspace segmentation based on low-rank representation (LRR, [25], and several state-of-the-art subspace clustering algorithms: ESC-FFS [39], SSC-OMP [40], S^3C [23], and BDR [27].

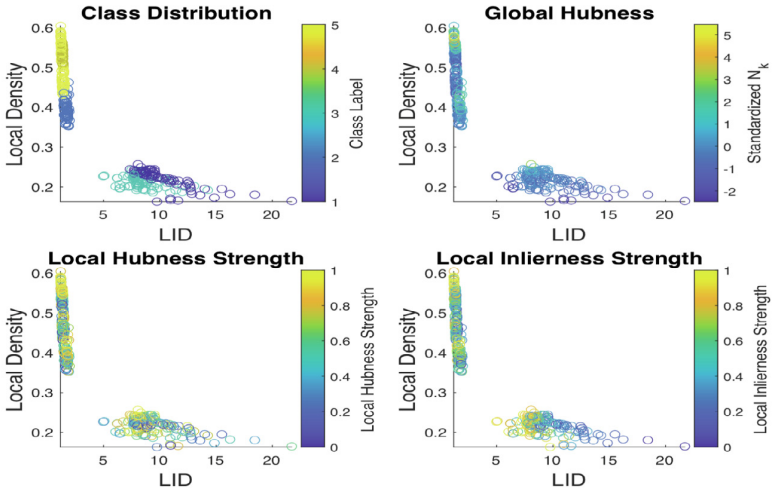
Since the methods H-WAMS, HP-WAMS, F-WAMS, and ESC-FFS require a sample of data, we run these algorithms for different sample sizes, namely (1%, 2%, 3%, 4%, 5%, 10%, 15%), and report



(a) Pen-3



(b) Waveform1



(c) COIL

Fig. 6. Characterization of real datasets (best viewed in color): (a) Pen-3; (b) Waveform1, and (c) COIL.

Table 11
Characterizing the usefulness of hubs in real datasets.

Data	$S_{N_k}(k=\sqrt{n})$	% Hubs in the seed pool predicted as bad/boundary	% Inlier hubs in seed pool (Loc-IS > 0.5)
Diabetes	0.44	0.42	0.90
Abalone	0.06	0.44	0.99
Letter	0.43	0.41	0.88
Pen-3	0.51	0.28	0.92
Pen-10	0.39	0.28	0.96
Image	0.07	0.29	0.88
Waveform1	1.82	0.72	0.87
Sonar	1.10	0.71	0.93
Musk1	1.12	0.59	0.93
Musk2	1.41	0.46	0.93
mfeat-factors	0.67	0.38	0.98
mfeat-pixels	0.67	0.46	0.96
ISOLET	1.15	0.54	0.87
COIL	1.07	0.41	0.68
Caltech-20	2.86	0.75	0.74
Caltech-100	5.48	0.68	0.58
DrivFace	0.51	0.46	0.93
OVA_ Colon	2.83	0.76	1.00

results for the sample size with best average across the datasets. The reported sample sizes are 15% for H-WAMS, HP-WAMS, and F-WAMS, and 3 % for ESC-FFS. Wave, Sonar, COIL, Caltech-20, and OVA_ Colon have fewer global hubs, thus no results are available for them for a seed sample size of 15%. We report the results for Wave, Sonar, COIL, Caltech-20, and OVA_ Colon at 10%. Caltech-100 has fewer than 10% global hubs, hence we report results for Caltech-100 using all hubs. HP-WAMS, F-WAMS, and ESC-FFS use probabilistic sampling and hence their results are averaged across 50 iterations. In order to learn a good weight distribution, features with low variance (< 0.01) on the seed sample are removed for all the seeding based algorithms. H-WAMS and HP-WAMS use the SVM classifier described in Section 4.3.2 to predict and filter bad and boundary hubs. To predict bad/boundary hubs on the datasets which are not evaluated in Table 3, we use the following classifiers, as described in Section 4.3.2: for Abalone, Caltech-20, Caltech-100, DrivFace, and OVA_ Colon, we use the classifier tested on Sonar (i.e., the classifier trained on the datasets shown in Table 3 excluding Sonar). For Pen-10, we use the classifier tested on Pen-3. The parameter λ for ESC-FFS is set to 100 as suggested by the authors in their paper [39]. For SSC-OMP, we tune the parameter k_{max} in the range $\{5, 10, 15, 20\}$. We use default values for S^3C, as suggested by the authors in their paper [23]: $\nu = 1.2, T_{max} = 10$, and the parameter α is tuned in the range $\{0.1, 0.3, 0.5, 0.7, 1.0\}$. For BDR we fix γ = 0.01 and vary λ in the range $\{0.001, 0.01, 0.1, 1, 10, 100\}$. Note that BDR-B and BDR-Z use the same parameters. The regularization parameter λ for LRR is changed in the range $\{0.5, 1.0, 1.5, 2.0\}$. We set the number of subspace clusters equal to the number of classes in the data in spectral and low-rank based algorithms.

Table 7 gives the clustering purity values obtained for all algorithms under comparison. WAMS resulted in an out-of-memory error on Caltech-100 and OVA_ Colon, hence the evaluation of WAMS on these datasets are not reported. We observe from the Table that H-WAMS is the most robust approach among the competitors. H-WAMS achieves the best average purity value across all datasets, and performs better than its competitors on the majority of the datasets. HP-WAMS is the overall second-best. Table 8 shows the NMI for all algorithms. H-WAMS and ESC-FFS have the highest NMI on average. Both H-WAMS and HP-WAMS outperform F-WAMS in clustering purity and NMI across the datasets, with the exception of Caltech-100. This can be explained by the fact that Caltech-100 has fewer than 10% global hubs, while F-WAMS is evaluated at sample size 15% in the given tables. Overall, the superior performance of hub-based algorithms shows that a hub-based sample finds more accurate modes than a random sample. Among the non–hub based algorithms, ESC-FFS is the most competitive. However, our method has the advantage of finding the subspace of each cluster via weighting the features, while the subspaces of clusters found by spectral-based methods are not explicit. For example, the knowledge pertaining the subspace of a cluster enables automated topic assignments and is useful in applications such as document retrieval.

We further analyze the statistical significance of the results of the probabilistic algorithms across the sample sizes. We ran a two-sided, two-sample t-test with $\alpha = 0.05$ to compute the statistical significance between each pair of algorithms. Table 9 gives the clustering purity values and their statistical significance for HP-WAMS, ESC, and F-WAMS. We report the mean clustering purity and standard deviations computed across 50 runs. The symbol (*) denotes the result which is statistically superior between HP-WAMS and ESC-FFS; the symbol (†) denotes the result which is statistically superior between HP-WAMS and F-WAMS; and (‡) denotes the statistically superior result between ESC-FFS and F-WAMS. The highest clustering purity values across the three algorithms is bold-faced. We observe that HP-WAMS and ESC-FFS are statistically superior to F-WAMS in general. ESC-FFS achieves better clustering purity at lower sample sizes; its clustering quality degrades with increasing sample size, while HP-WAMS has the opposite trend. We observe that HP-WAMS is statistically superior to ESC-FFS across sample sizes on several datasets, and achieves the highest mean clustering purity across the datasets.

6.4. Time complexity and trend plots

Table 10 compares the running times (in seconds) of all the algorithms. All experiments are run on a 2.3 GHz Intel Core i5 processor with 16 GB RAM. WAMS could not complete on Musk2, Caltech-100, and OVA_ Colon using the above resources, hence their running time is not reported. This is because WAMS uses all the data to compute the weighted bandwidth and mean-shift, and resulted in an out-of-memory error on these datasets. We ran WAMS on these datasets using a larger computational resource to obtain results on clustering purity and NMI. However Caltech-100 and OVA_ Colon still resulted in an out-of-memory error. Hence we do not report the clustering purity and NMI for Caltech-100 and OVA_ Colon. We observe that SSC-OMP is the fastest. Comparing the mean-shift algorithms, we see that H-WAMS and HP-WAMS are significantly faster than WAMS. The computation of hubness scores largely contributes to the running time of H-WAMS and HP-WAMS. However, the time complexity of hubness computation can be significantly reduced by computing approximate nearest neighbors, as in Locality Sensitive Hashing [12].

We also analyze the theoretical complexity of H-WAMS. Let n be the number of instances, m the seed sample size, and d the dimensionality of a dataset. Phase 0 of Algorithm 1 has a time complexity of $\mathcal{O}(n^2 d)$ to compute hubness scores. Phase 1 consists of initializing the seed pool of hubs ($\mathcal{O}(n)$), meta-feature extraction and hubs classification ($\mathcal{O}(m^2)$), and local ranking and selection from the seed pool ($\mathcal{O}(m^2)$). Thus, the time complexity of Phase 1 is $\mathcal{O}(n) + \mathcal{O}(m^2)$. Phase 2 performs weighted adaptive mean-shift. Suppose each seed point requires at most t_W steps to estimate its feature weight distribution, and at most t_{MS} steps for mean-shift to converge. Then, the time complexity of the weighted bandwidth algorithm is $\mathcal{O}(m^2 d t_W)$; the complexity of mean-shift is $\mathcal{O}(m^2 d t_{MS})$; and the complexity of grouping mean-shift modes into clusters is $\mathcal{O}(m^2 d)$. Phase 3 clusters non-seed data points with

complexity $\mathcal{O}((n-m)md)$. The combined time complexity of Phases 2 and 3 is $\mathcal{O}(m^2dt_W)+\mathcal{O}(m^2dt_{MS})+\mathcal{O}(nmd)$. The time complexity of Phase 1 is subsumed within that of Phase 0. Hence, the overall time complexity of H-WAMS is $\mathcal{O}(n^2d)+\mathcal{O}(m^2dt_W)+\mathcal{O}(m^2dt_{MS})$. On the other hand, the time complexity of WAMS is $\mathcal{O}(n^2dt_W)+\mathcal{O}(n^2dt_{MS})+\mathcal{O}(n^2d)$, where the additive components represent the weighted bandwidth computation, and mean-shift and clustering of mean-shift modes, respectively. Typically the value of t_{MS} for hubs is smaller than the value for non–hubs, due to the hubs' geometric property of occurring near the center of compact sub-clusters. Given $m \ll n$, and due to the faster convergence of mean-shift on hubs, H-WAMS achieves a significant improvement in running time over WAMS.

The trends of clustering purity, NMI, and running time of H-WAMS across sample sizes are shown in Figs. 3–5 respectively. We observe a general upward trend for the purity, NMI, and the running times as the sample size increases. However, using all the hubs for seeding reduced the clustering purity and NMI for most of the datasets. Hence, we find that 15% is the best sample size for H-WAMS. In general, H-WAMS is superior to WAMS in running time and clustering quality for the majority of the datasets. In cases where H-WAMS does not outperform WAMS in clustering quality (Image, Waveform1, and COIL), it is still faster and sacrifices the clustering quality only mildly.

7. Discussion: when should hubs be leveraged?

Comparing the results in Table 7, we observe that H-WAMS and HP-WAMS does not outperform WAMS across all datasets. Therefore, it's important to understand when hubs should or should not be leveraged. In order for hubs to have a positive impact on clustering, they must be good and inlier hubs. To analyze this phenomenon, we compare the characteristics of datasets using scatter plots. In Fig. 6, we plot three representative datasets with different performance compared to WAMS, namely, Pen-3, Waveform1, and COIL. Pen-3 is representative of an ideal dataset having *good* hubs and distinct density modes corresponding to each class. Hub-based methods on Pen perform better than WAMS by a large margin, while they had similar or lower performance than WAMS on Waveform1 and COIL. In Fig. 6, each dataset depicts four subplots, each representing *local density* vs. LID. LID measures the inlierness of a point; the local density of a point is computed as the inverse distance to its k^{th} nearest neighbor. Each subplot represents a different attribute, whose values are color-coded: *class distribution*, *global hubness*, *local hubness strength* (Loc-N_k), and *Local inlierness strength* (Loc-IS). All the data points are plotted in each subplot and are ranked by color-coding. We define the local inlierness strength of a point as the percentage of its reverse k-nearest neighbors with higher LID values than itself.

$$\text{Loc}-\text{IS}(\mathbf{x}) = \frac{\sum_{\mathbf{y}\in \text{R}k\text{NN}(\mathbf{x})} I(\text{LID}(\mathbf{x}), \text{LID}(\mathbf{y}))}{|\text{R}k\text{NN}(\mathbf{x})|}$$

$$I(p,q) = \begin{cases} 1 & p \leqslant q \\ 0 & otherwise \end{cases}$$

The scatter plot of class distribution for Pen-3 (Fig. 6 (a)) shows three classes with distinct density modes. The seed pool for this data can be viewed from the plot of global hubness score (standardized $N_k > 1$). The global hubs are located near the local density modes and have high inlierness. The plot of Loc-N_k depicts the order in which seed selection would be performed from the seed pool. Comparing this plot with the plot of Loc-IS, we observe that hubs with high local hubness also have high local inlierness, and hence form good seeds. The results of H-WAMS for Pen-3 support this claim.

The subplots for Waveform1 in Fig. 6 (b) depicts a different pattern. All the classes have a similar distribution of local density and LID and do not exhibit density modes. From the plot of global hubness, we observe that the data points with high local density or high hubness are not the most pronounced inliers. Comparing the plots of Loc-N_k and Loc-IS, we observe that these measures have very different distributions, i.e., the strong local hubs are not strong local inliers and vice versa. Hence, H-WAMS may not find accurate density peaks through mean-shift. This also explains why HP-WAMS performed worse than WAMS on Waveform1, as a probabilistic selection from the seed pool may select hubs which are not inliers for this data.

In Fig. 6 (c), the class distribution of COIL shows five classes with widely varying density. Comparing the plots of global hubness and Loc-N_k, we see that hubs with high Loc-N_k emerge in all classes, while the low density classes did not have strong global hubs. Hence seed selection based on Loc-N_k ensures a better representation of classes. This is confirmed by the superior peformance of H-WAMS and HP-WAMS over F-WAMS on COIL and across the datasets. However, comparing the plots of Loc-N_k and Loc-IS, we observe that there are data points with high Loc-IS and high local density, which do not have a corresponding high Loc-N_k. This suggests that there could be data points other than hubs which form better seeds.

Table 11 quantifies additional indicators on the usefulness of hubs. The first column measures the skewness of hubness. The skewness of hubness is higher for intrinsically high dimensional data [34], and the geometric properties of hubs are more pronounced on such data. Hub-based methods are less effective on data with very low skewness (e.g., Image). Abalone has a very low skewness as well, however, it also has a very high percentage of inlier hubs, which contributes to a better performance compared to Image. Datasets with extreme skewness are also less suitable for hub-based methods, as this suggests that there are very few hubs in the data. For example, Caltech-100 has only 7% hubs which was not sufficient to represent 100 classes. The second column measures the percentage of hubs in the seed pool which are predicted as bad/boundary. A large percentage of predicted bad/boundary hubs for a dataset indicates lower utility of hub-based seeding (e.g., Waveform1). The third column measures the percentage of hubs in the seed pool which are also inliers (i.e., with Loc-IS > 0.5). Hub-based methods are effective when the percentage of inlier seed hubs is high.

Note that in Fig. 6, the label information is used only for the class distribution subplots. Hence, the visualizations of Loc-N_k, Loc-IS, and global hubness can be used, along with the combination of measures in Table 11, to decide whether hub-based seeding should be leveraged for clustering.

8. Conclusion

We presented a new characterization of hubs in relation to subspaces, and proposed meta-features to identify bad and boundary hubs. Based on our findings, we introduced a hubness-driven algorithm to find subspace clusters. Our experimental results show the effectiveness of our technique, both in terms of accuracy and speed. The analysis and results presented in this work shed light on the role of hubs for subspace clustering. In the future, we plan to further investigate the use of hubs in manifold regularization and deep learning.

CRediT authorship contribution statement

Priya Mani: Conceptualization, Methodology, Software, Validation, Writing - original draft. **Carlotta Domeniconi:** Supervision, Conceptualization, Methodology, Validation, Writing - review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] C.C. Aggarwal, C.M. Procopiuc, J.L. Wolf, P. Yu, J.S. Park, Fast algorithms for projected clustering, in: SIGMOD 1999, Proceedings ACM SIGMOD International Conference on Management of Data, June 1-3, 1999, Philadelphia, Pennsylvania, USA, pp. 61–72. doi:10.1145/304182.304188.

[2] R. Agrawal, J. Gehrke, D. Gunopulos, P. Raghavan, Automatic subspace clustering of high dimensional data for data mining applications, in: SIGMOD 1998, Proceedings ACM SIGMOD International Conference on Management of Data, June 2–4, 1998, Seattle, Washington, USA, pp. 94–105. doi:10.1145/276304.276314.

[3] L. Akoglu, M. McGlohon, C. Faloutsos, Oddball: Spotting anomalies in weighted graphs, in: Advances in Knowledge Discovery and Data Mining, 14th Pacific-Asia Conference, PAKDD 2010, Hyderabad, India, June 21–24, 2010. Proceedings. Part II, pp. 410–421. doi:10.1007/978-3-642-13672-6_40.

[4] L. Amsaleg, O. Chelly, T. Furon, S. Girard, M.E. Houle, M. Nett, Estimating local intrinsic dimensionality, in: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Sydney, NSW, Australia, August 10–13, 2015, pp. 29–38. doi:10.1145/2783258.2783405.

[5] A. Berenzweig, Anchors and Hubs in Audio-based Music Similarity Ph.D. thesis, Columbia University, 2007.

[6] C. Böhm, K. Kailing, H.P. Kriegel, P. Kröger, Density connected clustering with local subspace preferences, in: Proceedings of the 4th IEEE International Conference on Data Mining (ICDM 2004), 1–4 November 2004, Brighton, UK, pp. 27–34. doi:10.1109/ICDM.2004.10087.

[7] D. Comaniciu, P. Meer, Mean shift: A robust approach toward feature space analysis, IEEE Trans. Pattern Anal. Mach. Intell. 24 (2002) 603–619, https://doi.org/10.1109/34.1000236.

[8] C. Domeniconi, D. Gunopulos, S. Ma, B. Yan, M. Al-Razgan, D. Papadopoulos, Locally adaptive metrics for clustering high dimensional data, Data Min. Knowl. Discov. 14 (2007) 63–97, https://doi.org/10.1007/s10618-006-0060-8.

[9] D. Dua, C. Graff, UCI machine learning repository. URL: http://archive.ics.uci.edu/ml. University of California, Irvine, School of Information and Computer Sciences, 2019.

[10] E. Elhamifar, R. Vidal, Sparse subspace clustering: Algorithm, theory, and applications, IEEE Trans. Pattern Anal. Mach. Intell. 35 (2013) 2765–2781, https://doi.org/10.1109/TPAMI.2013.57.

[11] J.H. Friedman, J.J. Meullman, Clustering objects on subsets of attributes, J. Royal Statist. Soc. Series B (Statistical Methodology) 66 (2000) 825–849.

[12] A. Gionis, P. Indyk, R. Motwani, Similarity search in high dimensions via hashing, 1999. pp. 518–529.

[13] J.A. Hartigan, Direct clustering of a data matrix, J. Am. Stat. Assoc. 67 (1972) 123–129.

[14] Z. He, Hub selection for hub based clustering algorithms, in: 11th International Conference on Fuzzy Systems and Knowledge Discovery, FSKD 2014, Xiamen, China, August 19–21, 2014, pp. 479–484. URL: doi: 10.1109/FSKD.2014.6980881, doi:10.1109/FSKD.2014.6980881.

[15] A. Hicklin, C. Watson, B. Ulery, The myth of goats: How many people have fingerprints that are hard to match. Internal Report 7271, National Institute of Standards and Technology (NIST), 2005.

[16] M.E. Houle, Inlierness, Outlierness, Hubness and Discriminability: an Extreme-Value-Theoretic Foundation. Technical Report 2015–002E. NII, 2015.

[17] M.E. Houle, Local intrinsic dimensionality I: an extreme-value-theoretic foundation for similarity applications, in: Similarity Search and Applications - 10th International Conference, SISAP 2017, Munich, Germany, October 4-6, 2017, Proceedings, 2017a. pp. 64–79. doi:10.1007/978-3-319-68474-1_5.

[18] M.E. Houle, Local intrinsic dimensionality II: multivariate analysis and distributional support, in: Similarity Search and Applications - 10th International Conference, SISAP 2017, Munich, Germany, October 4-6, 2017, Proceedings, 2017b. pp. 80–95. doi:10.1007/978-3-319-68474-1_6.

[19] K. Kailing, H.P. Kriegel, P. Kröger, Density-connected subspace clustering for high-dimensional data, in: Proceedings of the Fourth SIAM International Conference on Data Mining, Lake Buena Vista, Florida, USA, April 22–24, 2004, pp. 246–256. URL: doi: 10.1137/1.9781611972740.23, doi:10.1137/1.9781611972740.23.

[20] K.Y. Kim, D.H. Ki, H.C. Jeung, H.C. Chung, S.Y. Rha, Improving the prediction accuracy in classification using the combined data sets by ranks of gene expressions, BMC Bioinformatics 9 (2008), https://doi.org/10.1186/1471-2105-9-283.

[21] H.P. Kriegel, P. Kröger, A. Zimek, Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering, TKDD 3, 1:1–1 (2009) 58, https://doi.org/10.1145/1497577.1497578.

[22] L.I. Kuncheva, S.T. Hadjitodorov, Using diversity in cluster ensembles, in: Proceedings of the IEEE International Conference on Systems, Man & Cybernetics: The Hague, Netherlands, 10–13 October 2004, IEEE. pp. 1214–1219, doi:10.1109/ICSMC.2004.1399790.

[23] C.G. Li, R. Vidal, Structured sparse subspace clustering: A unified optimization framework, in: IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7–12, 2015, pp. 277–286. doi:10.1109/CVPR.2015.7298624.

[24] B. Liu, Y. Xia, P.S. Yu, Clustering through decision tree construction, in: Proceedings of the 2000 ACM CIKM International Conference on Information and Knowledge Management, McLean, VA, USA, November 6–11, 2000, pp. 20–29. doi:10.1145/354756.354775.

[25] G. Liu, Z. Lin, Y. Yu, Robust subspace segmentation by low-rank representation, in: Fürnkranz, J., Joachims, T. (Eds.), Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel, Omnipress. 2010a. pp. 663–670. URL: https://icml.cc/Conferences/2010/papers/521.pdf.

[26] G. Liu, Z. Lin, Y. Yu, Robust subspace segmentation by low-rank representation, in: Proceedings of the 27th International Conference on Machine Learning (ICML-10), June 21-24, 2010, Haifa, Israel, 2010b. pp. 663–670. URL: https://icml.cc/Conferences/2010/papers/521.pdf.

[27] C. Lu, J. Feng, Z. Lin, T. Mei, S. Yan, Subspace clustering by block diagonal representation, IEEE Trans. Pattern Anal. Mach. Intell. 41 (2019) 487–501, https://doi.org/10.1109/TPAMI.2018.2794348.

[28] X. Ma, B. Li, Y. Wang, S.M. Erfani, S.N.R. Wijewickrema, G. Schoenebeck, D. Song, M.E. Houle, J. Bailey, Characterizing adversarial subspaces using local intrinsic dimensionality, in: 6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings. URL: https://openreview.net/forum?id=B1gJ1L2aW.

[29] P. Mani, M. Vazquez, J.R. Metcalf-Burton, C. Domeniconi, H. Fairbanks, G. Bal, E. Beer, S. Tari, The hubness phenomenon in high-dimensional spaces, in: E. Gasparovic, C. Domeniconi (Eds.), Research in Data Science, Associations for Women in Mathematics Series, vol. 17, Springer, 2019, pp. 15–45, https://doi.org/10.1007/978-3-030-11566-1_2.

[30] E. Müller, S. Günnemann, I. Assent, T. Seidl, Evaluating clustering in subspace projections of high dimensional data. PVLDB 2, 2009. 1270–1281. URL: http://www.vldb.org/pvldb/2/vldb09-600.pdf, doi:10.14778/1687627.1687770.

[31] M.E.J. Newman, The structure and function of complex networks, SIAM Rev. 45 (2003) 167–256, https://doi.org/10.1137/S003614450342480.

[32] L. Parsons, E. Haque, H. Liu, Subspace clustering for high dimensional data: a review, SIGKDD Explorations 6 (2004) 90–105, https://doi.org/10.1145/1007730.1007731.

[33] J.C. Platt, Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods, in: Advances in Large Margin Classifiers, MIT Press, 1999, pp. 61–74.

[34] M. Radovanovic, A. Nanopoulos, M. Ivanovic, Hubs in space: popular nearest neighbors in high-dimensional data, J. Mach. Learn. Res. 11 (2010) 2487–2531, URL: http://portal.acm.org/citation.cfm?id=1953015.

[35] Y.R. Ren, C. Domeniconi, G. Zhang, G.X. Yu, A weighted adaptive mean shift clustering algorithm, in: Proceedings of the 2014 SIAM International Conference on Data Mining, Philadelphia, Pennsylvania, USA, April 24–26, 2014, pp. 794–802. doi:10.1137/1.9781611973440.91.

[36] M. Skinner, Product categorization with lstms and balanced pooling views, in: The SIGIR 2018 Workshop On eCommerce co-located with the 41st International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2018), Ann Arbor, Michigan, USA, July 12, 2018. URL: http://ceur-ws.org/Vol-2319/ecom18DC_paper_9.pdf.

[37] N. Tomasev, M. Radovanovic, M. Mladenic, M. Ivanovic, The role of hubness in clustering high-dimensional data, IEEE Trans. Knowl. Data Eng. 26 (2014) 739–751, https://doi.org/10.1109/TKDE.2013.25.

[38] R. Vidal, P. Favaro, Low rank subspace clustering (LRSC), Pattern Recognition Lett. 43 (2014) 47–61, https://doi.org/10.1016/j.patrec.2013.08.006.

[39] C. You, C. Li, D.P. Robinson, R. Vidal, A scalable exemplar-based subspace clustering algorithm for class-imbalanced data, in: Computer Vision - ECCV 2018–15th European Conference, Munich, Germany, September 8–14, 2018, Proceedings, Part IX, 2018, pp. 68–85, https://doi.org/10.1007/978-3-030-01240-3_5.

[40] C. You, D.P. Robinson, R. Vidal, Scalable sparse subspace clustering by orthogonal matching pursuit, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27–30, 2016, pp. 3918–3927. URL: doi: 10.1109/CVPR.2016.425, doi:10.1109/CVPR.2016.425.

[41] L. Zhang, T. Xiang, S. Gong, Learning a deep embedding model for zero-shot learning, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21–26, 2017, pp. 3010–3019. doi:10.1109/CVPR.2017.321.

[42] Y. Zhao, G. Karypis, Criterion Functions for Document Clustering: Experiments and Analysis. Technical Report, 2002.

Priya Mani is pursuing the Ph.D degree in Computer Science at George Mason University. She received the B. Tech degree in Computer Science and Engineering from University of Kerala and the M.S. degree in Computer Science and Engineering from Indian Institute of Technology Madras. Her research interests include machine learning, deep learning and data science, applied to unsupervised learning problems.

Carlotta Domeniconi is an Associate Professor in the Department of Computer Science at George Mason University. Her research interests include machine learning, data mining, classification, clustering, anomaly detection, and big data, with applications in text mining, social network analysis, and learning analytics. She has published extensively in premier journals and conferences in machine learning and data mining. She was the program co-Chair of SIAM Data Mining in 2012, and is currently serving as General Chair for the same conference. Dr. Domeniconi has served as PC member for KDD, ICDM, SDM, ECML-PKDD, IJCAI, and AAAI, and she is an Associate Editor of the IEEE Transactions on Knowledge and Data Engineering Journal. Dr. Domeniconi is a recipient of an ORAU Ralph E. Powe Junior Faculty Enhancement Award. She has worked as PI or co-PI on projects supported by the US Army, the Air Force, and the DoD. Her research has been in part supported by an NSF CAREER Award.