Consensus Clustering Using kNN Mode Seeking

Jonas Nordhaug Myhre(✉), Karl Øyvind Mikalsen, Sigurd Løkse,
and Robert Jenssen

Machine Learning @ UiT Lab,
UiT - The Arctic University of Norway, Tromsø, Norway
jonas.n.myhre@uit.no
http://site.uit.no/ml

Abstract. In this paper we present a novel clustering approach which combines two modern strategies, namely *consensus clustering*, and *two stage clustering* as represented by the *mean shift spectral clustering* algorithm. We introduce the recent kNN mode seeking algorithm in the consensus clustering framework, and the information theoretic kNN Cauchy Schwarz divergence as foundation for spectral clustering. In combining these frameworks, two well known problematic issues are directly bypassed; the kernel bandwidth choice of the kernel density based mean shift and the computational complexity of the mean shift iterations. We demonstrate experiments on both real and synthetic data as a proof of concept for our contributions.

1 Introduction

Clustering is one of the major areas of research in data analysis and related fields, including image analysis. For comprehensive reviews, see for example the textbooks [21], [7], [3].

One prominent methodology in nonparametric clustering, i.e. assuming no pre-defined parametric statistical models for the clusters to be found, is represented by the mean shift algorithm. Mean shift has experienced success in various applications, e.g. tracking [6], and is for example a component in Microsoft's Kinect® computer vision system [20]. It is an iterative nonparametric clustering approach introduced by Fukunaga and Hostetler [13], and is used for *seeking the modes* of a probability density function represented by a finite set of samples. The mean shift formulation is revisited by Cheng [4], which made its potential uses in clustering and global optimization more noticeable, and the mean shift algorithm furthermore gained popularity with the work of Comaniciu and Meer [5] and Georgescu et al. [14].

Particularly interesting developments in this line of research for the purpose of this paper, are recent attempts by Ozertem et al. [18] and by Agersborg and Jenssen [1] to couple the mean shift algorithm with spectral clustering [15,23]. The idea is to merge together the modes found by mean shift by a spectral clustering algorithm based on a matrix encoding similarities between every pair of modes. Results obtained were promising, however, several challenges were

R.R. Paulsen and K.S. Pedersen (Eds.): SCIA 2015, LNCS 9127, pp. 175–186, 2015.
DOI: 10.1007/978-3-319-19665-7_15

evident in both these methods: It is well-known that the mean shift algorithm is very sensitive to the particular size of the window employed for the underlying kernel density estimation procedure. Moreover, the procedure is slow and kernel density estimation in higher dimensions can be troublesome. In addition, the spectral clustering step also relies on a kernel density window size, which e.g. in Ozertem et al. [18] is chosen to equal the window size utilized in the mean shift.

This paper goes several steps further. First, we move away from kernel density estimation-based mean shift, utilizing instead the faster k-nearest neighbor (kNN) approach to mode seeking introduced very recently by Duin et al. [8]. Second, we lift the dependence on critical hyper parameters in the clustering procedure, by leveraging the full power of *evidence-based clustering*, also called *consensus clustering* [10]. This is achieved by running the mode seeking algorithm over a range of k-values. Each value of k is used to accumulate evidence about the clustering structure. There are many possible options, but in this work we limit the focus to two concrete approaches:

- In the first approach, what we will refer to as a *consensus matrix*, is computed. This entails simply counting for each k whether or not pairs of data points in the data set belong to the same basin of attraction (mode), for then to compute the average over all k. Based on the consensus matrix, a hierarchical clustering approach similar to that used in [9] and [10] is utilized in order to obtain the final clustering result.
- The second approach we investigate, is based on for each k to compute an information theoretic divergence measure between pairs of modes resulting in a similarity matrix between modes, for then to average over all k. Then, a spectral clustering procedure is executed on this matrix, similar to [1].

The proposed clustering method results in a fast mode seeking based clustering algorithm without the need to heuristically select the value of one critical hyper parameter (in our case the k), enabled by the consensus clustering ideas we adopt in this paper. We show that the resulting *consensus clustering using kNN mode seeking algorithm* obtains promising results.

The remainder of this paper is organized as follows: In section 2 we discuss and reveiw relevant background topics such as *clustering by mode seeking*, *two stage clustering* and *consensus clustering*. In section 3 the algorithms for the two proposed clusterings schemes are explained and specified. The algorithms are tested and compared on different data sets in section 4. concluded with a short conclusion and suggested improvements to be done in further work (see section 5).

2 Relevant Background Topics

2.1 Clustering by Mode Seeking

Mode seeking algorithms cluster data by assigning each data point to its closest local mode. It works by projecting each data point to the closest local mode of the kernel density estimate (KDE) using a gradient ascent approach. Using the standard KDE, $f(x) = \frac{1}{N}\sum_i \mathrm{k}_h(\mathbf{x}, \mathbf{x}_i)$, the mean shift iterations for projecting a single $\mathbf{x}$ to its local mode is given as follows [5]:

$$\mathbf{x} \leftarrow \frac{\sum_{i=1}^{N} \mathbf{x}_i \mathrm{k}_h(\mathbf{x}_i, \mathbf{x})}{\sum_{i=1}^{N} \mathrm{k}_h(\mathbf{x}_i, \mathbf{x})} \tag{1}$$

Looking beyond the available input points, any point within the basin of attraction of a local KDE mode will be in the cluster of that mode.

The kNN mode seeking algorithm, [8], represents a new generation of mode seeking algorithms, whereby the kernel density estimate is replaced by a kNN density estimate showing positive results, both in speed and accuracy. Also, notably different from mean shift, is the fact that projections are only done through the given input points, thus dramatically reducing computational complexity.

Given a kNN-density, where the density at a point $\mathbf{x}$ is simply inversely proportional to the distance to the k-th nearest neighbor, the algorithm can be stated as follows:

1. For each input point $\mathbf{x}_i$:
   - Define a pointer to the point within the k nearest neighbors with the highest kNN-density.
   - Repeat the process by following pointers from the initial pointer until a pointer that points to itself is found. This will be the local mode of $\mathbf{x}_i$.
2. Assign each $\mathbf{x}$ that converged to the same point to the same cluster.

This method is significantly faster and has comparable accuracy, despite only using input points for projections, compared to regular mean shift [8]. In addition, as opposed to k-means, [21], the method still retains the local properties of mean shift making it able to detect slightly non-linear cluster structures.

2.2 Two Stage Clustering

In [18], Ozertem et al. introduced a two step clustering scheme by first partitioning the input space into subsets using mean shift clustering, for then to utilize a variant of spectral clustering to do the final clustering. In the second stage of this process each data point is represented by its local mode as found by mean shift clustering, and the affinity matrix in the second step consists of (dis)similarities between the modes as opposed to the individual data points. From a computational perspective this significantly reduces the complexity as the spectral decomposition is reduced from $O(N^2)$ to $O(M^2)$, where N is the number of data points and M is the number of modes. In the original paper a heuristic approach was used as the final step [18]. Agersborg and Jenssen expanded the concepts and used true spectral clustering and proposed to use different choices of parameters in each step [1].

In addition to the computational advantages of two stage clustering is the fact that strongly nonlinear structures cannot in general be captured by a unimodal density. Thus, a single run of standard mean shift using a kernel density estimate cannot capture nonlinearities that goes beyond a slight bending or stretching of the local structure. A two stage clustering strategy could alleviate this by first decomposing nonlinear structure into subsections of the data represented by local modes, and then in the final stage merge the appropriate modes to obtain a global clustering.

2.3 Consensus Clustering

Consensus clustering is a relatively new methodology which has emerged over the last decade or so. One of the main motivations for introducing consensus clustering, is to acknowledge that there is no single clustering algorithm which will be appropriate for every dataset and different algorithms might produce different partitions for the same data set. This might make the interpretation of the clustering results a challenge. The idea of consensus clustering is to combine the results of several clustering trials to obtain a better partition than each individual trial. This is often done by constructing a similarity matrix, which we have called the *consensus matrix*, but is also referred to as the *co-association-* or *ensemble matrix* in the literature.

There are several proposed algorithms to combine clustering results. Fred and Jain [9–12] suggest to use the k-means clustering algorithm several times with random initial conditions. In each of the clustering trials, the number of clusters, k, is either fixed or chosen randomly in the range $k \in [k_{\min}, k_{\max}]$. The resulting partitions are then created in a voting process. A consensus matrix, $S = \{s_{ij}\}_{N \times N}$, is constructed by counting the number of times the points $\mathbf{x}_i$ and $\mathbf{x}_j$ are assigned to the same cluster in the M different partitions. Each time these data points are clustered together, it counts as one *vote*. This voting process is referred to as *evidence accumulation* [12]. The elements of S are then calculated by

$$s_{ij} = \frac{n_{ij}}{M}, \tag{2}$$

where n_{ij} is the number of times $\mathbf{x}_i$ and $\mathbf{x}_j$ has been assigned to the same cluster.

In the ideal case, we should have

$$s_{ij} = \begin{cases} 1 & \text{if } \mathbf{x}_i \text{ and } \mathbf{x}_j \text{ belong to the same cluster,} \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

This happens when $\mathbf{x}_i$ and $\mathbf{x}_j$ are clustered together in all of the k-means trials. We see that if the data points are ordered according to their final cluster assignment, the consensus matrix will be block diagonal.

The consensus matrix can be considered a similarity matrix. If two data points are clustered together in many of the different clustering solutions, they are considered more similar than two data points that are not clustered together as often. This similarity matrix can then be used to obtain a final partitioning/clustering.

3 The Proposed Clustering Scheme

In this section we present the new clustering scheme proposed in the Introduction. The approaches suggested both use the kNN mode seeking algorithm to build a consensus matrix over a range of k values, but the pairwise affinities in the matrix, as well as the last stage clustering schemes, will be different. This results in two algorithms which we will describe separately in the following subsections.

3.1 kNN Single Link (kNN-SL) Algorithm

In this algorithm, we build a consensus matrix by running the kNN mode seeking algorithm for a range of k values. In [10] Fred and Jain proposed to use several random initializations of the k-means algorithm to build a consensus matrix. The framework was later expanded to also vary the number of clusters in the k-means algorithm [12]. As opposed to k-means, there is no need for initialization in kNN mode seeking. Running the algorithm several times, in addition to adding computational complexity, does not present any variations and thus no benefits. For each iteration and for each pair of data points, $\mathbf{x}_i$ and $\mathbf{x}_j$, that are clustered together, the consensus matrix, S, is updated according to

$$S(i,j) = S(i,j) + \frac{1}{M}, \tag{4}$$

where M is the total number of clusterings.

In [10] a technique similar to single link clustering was introduced to detect consistent clusters within the consensus matrix. The idea is that for each pair of data points, their corresponding clusters are merged if $S(i,j) > t$, where t is a user-defined threshold. From now on we will refer to this as single link. In practice, this threshold has to be varied to obtain good results in different cases with different cluster structures. For simplicity we set the threshold manually in this work, assuming the number of clusters are known. A slightly modified version was introduced in [11] and [12]; instead of the threshold t, the dendrogram was used to find the clusters with the longest lifetime. We experienced that using these two alternatives sometimes resulted in one large cluster and one or more very small clusters[1]. To avoid this problem we propose to use a modified single link on the consensus matrix: For each pair (i,j) that does not belong to the same cluster and s.t. $S(i,j) > t$, we merge the clusters they belong to. After having performed this clustering we iterate through the data set once more and force small clusters (if present in the clustering results) to merge into a larger cluster. We do this for each data point, $\mathbf{x}_i$, that belongs to a cluster that is smaller than some threshold, e.g. $N/10$ datapoints, finding

$$j = \underset{l}{\operatorname{argmax}} \left\{ S(i,l) \right\}, \tag{5}$$

and merge the two clusters.

[1] By very small clusters we mean clusters that only contain a single or a few data points.

3.2 kNN Cauchy Schwarz (kNN-CS) Algorithm

In this algorithm the votes in the consensus matrix S, are replaced by kNN Cauchy Schwarz (CS) divergences [22]. The general CS divergence is an information theoretic similarity measure between two densities, [19]. In this work we use the *symmetric Cauchy Schwarz* measure which lets us use kNN density estimates in the measure. This was first introduced in [22]:

$$d_{CS}\left(p_1, p_2\right) = \frac{\frac{1}{2}\left(\int p_1(\mathbf{x})p_2(\mathbf{x})\mathrm{d}\mathbf{x} \int p_2(\mathbf{x})p_1(\mathbf{x})\mathrm{d}\mathbf{x}\right)}{\sqrt{\int p_1^2(\mathbf{x})\mathrm{d}\mathbf{x} \int p_2^2(\mathbf{x})\mathrm{d}\mathbf{x}}}. \tag{6}$$

The quantities $\int p_i(\mathbf{x})p_j(\mathbf{x})\mathrm{d}\mathbf{x}$ and $\int p_i^2(\mathbf{x})\mathrm{d}\mathbf{x}$ are calculated using kNN density estimates. The symmetry was introduced to avoid the effects of differences in expected values when using kNN densities, see [22] for further details and analysis. We calculate the Cauchy Schwarz divergences between each of the modes found by the kNN mode seeking algorithm. The CS divergence between each point is then represented as the CS divergence between the modes they belong to. The consensus matrix is built by, for each k, adding the pairwise divergences and finally averaging over all k. After building S, we do a spectral decomposition of the matrix, $S = E\Lambda E^T$ and, similar to [17], perform k-means clustering in the feature space represented by the eigenvectors in order to obtain the final clustering.

We note that this algorithm has in effect two parameters that need to be set, i.e. the number of clusters in the k-means algorithm and the number of eigenvectors to use in the spectral decomposition of the consensus matrix. In this work we assume for simplicity that we know the number of clusters and that, by convention, the same number of eigenvectors as clusters is a reasonable choice [23]. The choice of neighborhood size k is avoided in using the consensus strategy, thus only leaving an upper and lower bound to be set.

To summarize this section we include pseudocode for the two algorithms in Figure 1.

4 Experiments

In this section we present results illustrating the benefits and potential of the proposed methods. We have used both toy data and real datasets.

The first data set is a toy data set created to illustrate that the algorithms can handle nonlinear structure and clusters of different shape and geometry. The second data set is a subset of the 10K subset of the MNIST image data set and is used to illustrate that the algorithm can handle high dimensional data. The third experiment we include is the widely used Frey faces, to illustrate the potential of the algorithms in a completely unsupervised setting. We also include a small set of UCI benchmark datasets. In all experiments an initial range of k values where set to $k \in \{5, 6, \ldots, 20\}$, and then manually tuned to obtain the best results.

If nothing is stated we assume that the number of clusters is known.

kNN-SL algorithm:

```
– Input: Data set $X$, range of $k$-values $K$ and threshold $t$.
– Initialize $S$ as $\mathbf{0}_{N \times N}$
– Step 1: For each $k \in K$:
    • Use kNN mode seeking to obtain a clustering of $X$.
    • For each pair of data points, $(i, j)$ , update $S$ by $S(i, j) = S(i, j) + \frac{1}{|K|}$ if $\mathbf{x}_i$ and $\mathbf{x}_j$ belong to the same cluster.
– Step 2: Initially let each $\mathbf{x}_i$ be one individual cluster.
    • For each pair $(i, j)$ that does not belong to the same cluster and s.t. $S(i, j) > t$, merge the clusters they belong to.
    • If datapoint $\mathbf{x}_i$ belongs to a "small" cluster, find $j = \underset{l}{\operatorname{argmax}} \{S(i, l)\}$ and merge the clusters that $\mathbf{x}_i$ and $\mathbf{x}_j$ belong to.
```

kNN-CS algorithm:

```
– Input: $X$, $K$ and number of clusters $K_c$.
– Initialize $S$ as $\mathbf{0}_{N \times N}$
– Step 1: For each $k \in K$:
    • Use kNN mode seeking to obtain a clustering of $X$.
    • For each pair of modes, $(c_r, c_s)$, calculate the CS divergences $d_{rs} = d_{CS}(c_r, c_s)$ from (6).
    • For each pair of data points, $(i, j)$ , update $S$ by adding the CS divergence between the two modes, $c_{i'}$ and $c_{j'}$, that represent the two data points; $S(i, j) = S(i, j) + \frac{d_{i'j'}}{|K|}$.
– Step 2: Calculate the eigendecomposition of $S$ and perform $k$-means with input value $K_c$ on the top $K_c$ eigenvectors.
```

Fig. 1. The proposed algorithms

4.1 Toy Data: Two Moons and a Gaussian Blob

The first dataset is a two dimensional toy data set consisting of two moon shaped clusters with 400 data points in each and a spherical Gaussian cluster consisting of 200 data points. This is a clear example of a nonlinear dataset where standard methods like e.g. k-means perform poorly. Figure 2 shows 4 different clusterings of this dataset: k-means; a single run of the kNN mode seeking algorithm; and the two algorithms presented in this paper. We see that both k-means and the single run of the kNN mode seeking do not find the correct cluster structure. The kNN-SL gives a clustering that has no errors, whereas the kNN-CS gives a clustering with only a few errors in the moon shaped clusters.

4.2 MNIST Images

We chose a subset of the 10K MNIST image dataset, [16], containing the digits 3, 6 and 9. We used the vectorized images as features and used no feature selection methods, giving a 784 × 3024 input matrix for the clustering algorithm.

The clustering error percentages are presented in Table 1. We see that the two proposed algorithms give promising results; the error is 4.10% for both kNN-SL and kNN-CS, which is a notable improvement compared to a single run of

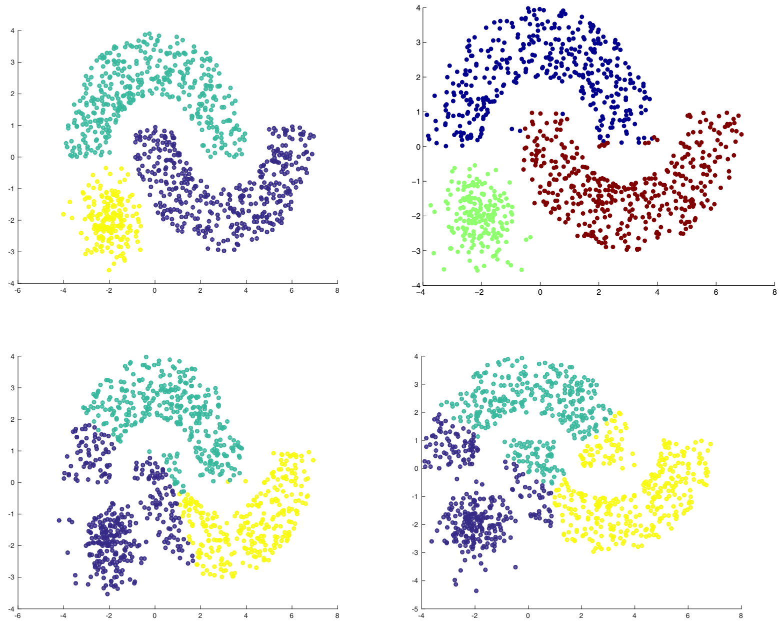


Fig. 2. Clustering of the two moon and Gaussian blob dataset. Upper left: kNN-SL. Upper right: kNN-CS. Lower left: Single run of mode seeking. Lower right: k-means.

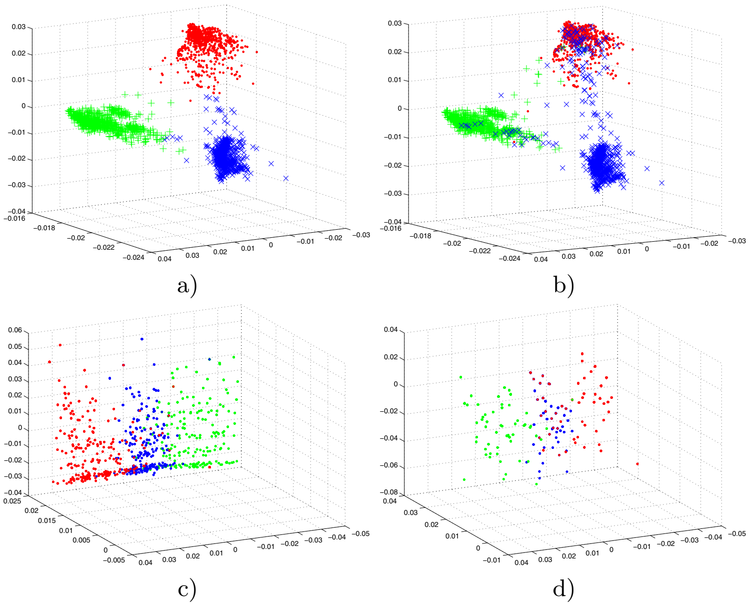
the kNN mode seeking algorithm. We compare to the k-means algorithm which is considerably slower and has linear cluster boundaries, and a single run of the kNN mode seeking algorithm. As an example the kNN-SL algorithm took 0.4 seconds on the MNIST example, while a test run of the consensus framework with random initializations of k-means took 42 seconds.

Table 1. Clustering results for the MNIST images. Error in percentages.

Dataset	# features	# dim.	k	k-means (av.)	kNN mode seeking	kNN-SL	kNN-CS
MNIST	3024	784	3	5.79	9.39	4.10	4.10

We see a clear improvement over k-means, and note that the consensus stage is clearly relevant as the single run of kNN mode seeking gives poor results.

One of the benefits of using the kNN-CS method is that results can be visualized by plotting the eigenvectors of the Cauchy-Schwarz matrix. In Figure 3, a), we see the spectral decomposition consisting of the top three eigenvectors of the kNN CS matrix. The color coding corresponds to the results after running k-means on the eigenvectors. In b) we see the true labels. It is evident that the class structure is well represented by the eigenvectors. c) and d) shows the top three eigenvectors of the kNN CS matrix for two individual runs of the kNN mode seeking algorithm with color coding representing the true labels. The class structure is evident in both cases, but the separability is not as strong as in the consensus case where the structures are much more compact and distanced from each other.



a)

b)

c)

d)

Fig. 3. a): Results from k-means on top three Cauchy-Schwarz concensus eigenvectors. b): True labels of MNIST data. c): Top three eigenvectors of CS matrix of a single run of kNN mode seeking with $k = 2$. d): Same as left, but $k = 5$.

Looking at the true labels we see that there is some overlap that an algorithm like k-means in the eigenvector feature space cannot capture. This is due to the fact that some of the images are overlapping in the input space, and our choice of not using any feature extractions.

4.3 Frey Faces

We tested both algorithms on the Frey faces with an arbitrary set of parameters to illustrate the algorithms in a setting where nothing is known in advance. Due to space limitations we only show the kNN-SL algorithm which, with the given parameters, gave a very clear partitioning of the face images. In Figure 4, 24 randomly selected images from each cluster found by the kNN-SL algorithm is shown in the top row. In the bottom row we show a k-means clustering with

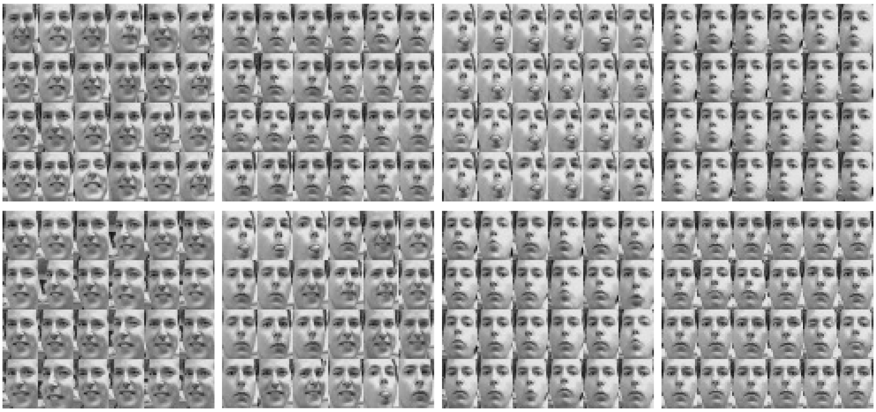


Fig. 4. First row: 4 clusters obtained using the kNN-SL algorithm. Second row: a clustering obtained using k-means.

the same number of clusters as found by the kNN-SL algorithm. All in all, the kNN SL algorithm seems to give much more visually intuitive results, whilst the results of k-means does not give as much sense and is harder to interpret visually.

4.4 UCI Datasets

To conclude the experiments, we tested our algorithms on three datasets from the UCI repository [2]; Iris, Wine and Wisconsin breast cancer data. The performance of the different methods is presented in Table 2. We note that the results of our algorithms are combarable to that of k-means, indicating that the datasets are linearly separable, leaving the improvements by nonlinear considerations less notable. We also see that a single run of the kNN mode seeking algorithm does not give good results, indicating that the consensus stage is of clear benefit.

Table 2. Clustering results on a selection of UCI datasets. Error in percentages.

Dataset	# features	# dim.	k	k-means (av.)	kNN mode seek	kNN-SL	kNN-CS
Cancer	699	10	2	4.29	30.04	5.01	5.01
Iris	150	4	3	10.7	33.3	10.0	9.3
Wine	178	13	3	3.37	12.92	8.99	8.99

5 Conclusion

In this paper we have presented two new clustering algorithms that shows good potential in both strongly nonlinear and high dimensional data. We have investigated the kNN mode seeking algorithm in the consensus clustering framework.

In introducing the consensus clustering principles to the two-stage clustering scheme we see that critical parameter choices can be rendered unnecessary and greater robustness to scale and setting can in principle be achieved. To conclude this work we include a few critical points and possible directions of future research.

5.1 Future Work

- In principle any clustering algorithm can be used in the first stage, and algorithms such as quick-shift should be tested.
- The kNN mode seeking results in fewer number of clusters as the neighborhood parameter k increases, which if set too high will give large clusters that are not intuitive in the input space. So a theoretical threshold for the upper bound of k should be investigated.
- The kNN mode seeking algorithm is able to handle much larger data set sizes than traditional mean shift, so the algorithms should be investigated in a larger scale setting than this work.
- The individual steps in the kNN-CS algorithm leads to matrices of different size. In this work we simply expanded the matrix with all points in the same cluster having the same value compared to another cluster. A less memory intensive strategy should be investigated in addition to looking into eigenvector summation instead of matrix summation.
- The threshold parameter in the kNN-SL algorithm is not straight-forward to choose and needs to be investigated further.
- The speedup factor of the kNN mode seeking algorithm compared to regular mean shift and also k-means, which has been used extensively in consensus clustering, is considerable and should be investigated and presented further.

References

1. Agersborg, J., Jenssen, R.: Mean shift spectral clustering using kernel entropy component analysis. In: Proceedings of IEEE Workshop on Machine Learning for Signal Processing, Reims, France, September 21-24, 2014
2. Bache, K., Lichman, M.: UCI machine learning repository (2013). http://archive.ics.uci.edu/ml
3. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer (2006)
4. Cheng, Y.: Mean shift, mode seeking, and clustering. IEEE Transactions on Pattern Analysis and Machine Intelligence **17**(8), 790–799 (1995)
5. Comaniciu, D., Meer, P.: Mean shift: A robust approach toward feature space analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence **24**(5), 603–619 (2002)
6. Comaniciu, D., Ramesh, V., Meer, P.: Kernel-based object tracking. IEEE Transactions on Pattern Analysis and Machine Intelligence **25**(5), 564–577 (2003)
7. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification, 2nd edn. John Wiley & Sons, New York (2001)

8. Duin, R.P.W., Fred, A.L.N., Loog, M., Pękalska, E.: Mode seeking clustering by knn and mean shift evaluated. In: Gimel'farb, G., Hancock, E., Imiya, A., Kuijper, A., Kudo, M., Omachi, S., Windeatt, T., Yamada, K. (eds.) SSPR & SPR 2012. LNCS, vol. 7626, pp. 51–59. Springer, Heidelberg (2012)
9. Fred, A.: Finding consistent clusters in data partitions. In: Kittler, J., Roli, F. (eds.) MCS 2001. LNCS, vol. 2096, pp. 309–318. Springer, Heidelberg (2001)
10. Fred, A.L.N., Jain, A.K.: Data clustering using evidence accumulation. In: Proceedings of the 16th International Conference on Pattern Recognition, vol. 4, pp. 276–280. IEEE (2002)
11. Fred, A., Jain, A.K.: Evidence accumulation clustering based on the k-means algorithm. In: Caelli, T., Amin, A., Duin, R.P.W., de Ridder, D., Kamel, M. (eds.) SSPR&SPR 2002. LNCS, vol. 2396, pp. 442–451. Springer, Heidelberg (2002)
12. Fred, A.L.N., Jain, A.K.: Combining multiple clusterings using evidence accumulation. IEEE Transactions on Pattern Analysis and Machine Intelligence **27**(6), 835–850 (2005)
13. Fukunaga, K., Hostetler, L.: The estimation of the gradient of a density function, with applications in pattern recognition. IEEE Trans. Inf. Theor. **21**(1), 32–40 (2006)
14. Georgescu, B., Shimshoni, I., Meer, P.: Mean shift based clustering in high dimensions: a texture classification example. In: Proceedings of the Ninth IEEE International Conference on Computer Vision, pp. 456–463. IEEE (2003)
15. Jenssen, R.: Kernel Entropy Component Analysis. IEEE Transactions on Pattern Analysis and Machine Intelligence **33**(5), 847–860 (2010)
16. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE **86**(11), 2278–2324 (1998)
17. Ng, A.Y., Jordan, M.I., Weiss, Y., et al.: On spectral clustering: Analysis and an algorithm. Advances in Neural Information Processing Systems **2**, 849–856 (2002)
18. Ozertem, U., Erdogmus, D., Jenssen, R.: Mean shift spectral clustering. Pattern Recognition **41**(6), 1924–1938 (2008)
19. Principe, J.C., Xu, D., Fisher, J.: Information theoretic learning. Unsupervised Adaptive Filtering **1**, 265–319 (2000)
20. Shotton, J., Sharp, T., Kipman, A., Fitzgibbon, A., Finocchio, M., Blake, A., Cook, M., Moore, R.: Real-Time Human Pose Recognition in Parts from Single Depth Images. Communications of the ACM **56**(1), 116–124 (2013)
21. Theodoridis, S., Koutroumbas, K.: Pattern Recognition, 4th edn. Academic Press, San Diego (2009)
22. Vikjord, V.V., Jenssen, R.: Information theoretic clustering using a k-nearest neighbors approach. Pattern Recognition **47**(9), 3070–3081 (2014)
23. von Luxburg, U.: A Tutorial on Spectral Clustering. Statistics and Computing **17**(4), 395–416 (2007)