SUPPLEMENTARY MATERIAL

Throughout the following sections, we use the notation and references from the paper.

1 RAP MESSAGES

Application of the max-sum algorithm [32] gives two types of messages:

(1) Message from variable to factor/constraint node is simply the sum of all the incoming messages to the variable node except the message from the target factor node. This can be observed in β_{ij}, ϕ_{ij} and ρ_{ij} messages in (18), (20) and (22) respectively.
(2) Message from factor node to variable node depends upon the corresponding factor/constraint function. Since the $E(\cdot)$ constraint is the same in RAP as in AP, the messages α_{ij} in (23) are the same as in (9). Therefore we are left with the messages $\overline{\eta}_{ij}$ and ψ_{ij} that are derived below.

1.1 Derivation of $\overline{\eta}_{ij}$

According to max-sum algorithm:

$$\overline{\eta}_{ij}(h_{ij}) = \max_{k\neq j}\left(\overline{I}(H[i,:]) + \sum_{l\neq j}\beta_{il}(h_{il})\right) \tag{27}$$

When $h_{ij} = 1$: We have met the least requirement of $\overline{I}(.)$ constraint to have at least one entry in ith row equal to 1. For the rest of points q governs whether or not they will be chosen by x_i.

$$\begin{aligned}
\overline{\eta}_{ij}(1) &= \max_{k\neq j}\left(\sum_{m} h_{im}q + \sum_{l\neq j}\beta_{il}(h_{il})\right)\\
&= \max_{k\neq j}\left(\sum_{l\neq j}\left(qh_{il} + \beta_{il}(h_{il})\right)\right) + q\\
&= \sum_{k\neq j}\left(\max_{h_{ik}\in\{0,1\}}\left(qh_{ik} + \beta_{ik}(h_{ik})\right)\right) + q\\
&= \sum_{k\neq j}\max\left(q + \beta_{ik}(1), \beta_{ik}(0)\right) + q\\
&= \sum_{k\neq j}\max\left(q + \beta_{ik}, 0\right) + \sum_{k\neq j}\beta_{ik}(0) + q
\end{aligned} \tag{28}$$

When $h_{ij} = 0$: We need to ensure at least one $h_{ik} = 1; k \neq j$. The rest can decide for themselves based upon the q.

$$\begin{aligned}
&\overline{\eta}_{ij}(0) = \max_{m\neq j}\beta_{im}(1)\\
&+ \max_{k\neq\{m,j\}}\left(\sum_{l\neq\{m,j\}}(qh_{il} + \beta_{il}(h_{il}))\right) + q\\
&= \max_{m\neq j}\beta_{im}(1) + \sum_{k\neq\{m,j\}}\left(\max_{h_{ik}\in\{0,1\}}\left(qh_{ik} + \beta_{ik}(h_{ik})\right)\right) + q\\
&= \max_{m\neq j}\beta_{im}(1) + \sum_{k\neq\{m,j\}}\max\left(q + \beta_{ik}(1), \beta_{ik}(0)\right) + q\\
&= \max_{m\neq j}\beta_{im}(1) + \sum_{k\neq\{m,j\}}\max\left(q + \beta_{ik}, 0\right)\\
&+ \sum_{k\neq\{m,j\}}\beta_{ik}(0) + q
\end{aligned} \tag{29}$$

For binary variable h_{ij}, instead of propagating separate messages for $h_{ij} = 0$ and $h_{ij} = 1$, it is enough to propagate the difference [13]. Therefore the message $\overline{\eta}_{ij}$ is given by:

$$\begin{aligned}
&\overline{\eta}_{ij} = \overline{\eta}_{ij}(1) - \overline{\eta}_{ij}(0)\\
&= \sum_{k\neq j}\max\left(q + \beta_{ik}, 0\right) + \sum_{k\neq j}\beta_{ik}(0) - \max_{m\neq j}\beta_{im}(1)\\
&- \sum_{k\neq\{m,j\}}\max\left(q + \beta_{ik}, 0\right) - \sum_{k\neq\{m,j\}}\beta_{ik}(0)\\
&= -\max_{m\neq j}\left(\beta_{im}(1) - \max\left(q + \beta_{im}, 0\right) - \beta_{im}(0)\right)\\
&= -\max_{m\neq j}\left(\beta_{im} + \min\left(-q - \beta_{im}, 0\right)\right)\\
&= -\max_{m\neq j}\left(\min\left(\beta_{im}, -q\right)\right)\\
&= \max\left(-\max_{m\neq j}\beta_{im}\ ,\ q\right)
\end{aligned} \tag{30}$$

1.2 Derivation of ψ_{ij}

$\psi_{ij}(h_{ii})$ is computed as:

$$\psi_{ij}(h_{ii}) = \max_{\substack{k\in\partial_j\\ k\neq i}}\left(G_j(\partial_j) + \sum_{\substack{l\in\partial_j\\ l\neq i}}\phi_{lj}(h_{ll})\right) \tag{31}$$

When $h_{ii} = 1 \implies h_{kk} \overset{!}{=} 0\ \ \forall k\in\partial_j$. So:

$$\psi_{ij}(1) = \sum_{\substack{l\in\partial_j\\ l\neq i}}\phi_{lj}(0) \tag{32}$$

When $h_{ii} = 0$: There are two possibilities: either $h_{kk} = 0\ \ \forall k \in \partial_j$ or one of $k \in \partial_j$ is 1 and rest are 0. We need to look for the possibility that maximizes the message $\psi_{ij}(0)$. So:

$$\begin{aligned}
\psi_{ij}(0) &= \max\left(\sum_{\substack{l\in\partial_j\\ l\neq i}}\phi_{lj}(0), \max_{\substack{l\in\partial_j\\ l\neq i}}\left(\phi_{lj}(1) + \sum_{\substack{m\in\partial_j\\ m\notin\{i,l\}}}\phi_{mj}(0)\right)\right)\\
&= \max\left(0, \max_{\substack{l\in\partial_j\\ l\neq i}}\left(\phi_{lj}(1) - \phi_{lj}(0)\right)\right) + \sum_{\substack{l\in\partial_j\\ l\neq i}}\phi_{lj}(0)\\
&= \max\left(0, \max_{\substack{l\in\partial_j\\ l\neq i}}\phi_{lj}\right) + \sum_{\substack{l\in\partial_j\\ l\neq i}}\phi_{lj}(0)
\end{aligned} \tag{33}$$

From (32) and (33), we get:

$$\psi_{ij} = \psi_{ij}(1) - \psi_{ij}(0) = -\max(0, \max_{\substack{l\in\partial_j\\ l\neq i}}\phi_{lj}) \tag{34}$$

1.3 Complexity Analysis of Messages

We now look at the equations from (18) to (23).

- β_{ij}: From Eq. (18), it is evident that non-diagonal messages are simply a summation of s_{ij} and α_{ij}. For diagonal messages, we also need sum over the involved neighborhoods $\mathcal{M}_i$. In worst case $\mathcal{M}_i$ can have $n-1$ points indicating that for diagonal, β_{ii} can be of $O(n)$. Hence computation of all β_{ii} messages is of $O(n^2)$ in worst case. Since non-diagonal

messages computation is also of $O(n^2)$, we can state that computation of all β_{ij} is $O(n^2)$.
- ϕ_{ij}: Using the same argument as in case of β_{ij} we can state that these messages are also of $O(n^2)$.
- ρ_{ij}: These messages are also $O(n^2)$. The argument is the same as for β_{ij} and ϕ_{ij}.
- $\overline{\eta}_{ij}$: The computation of original messages η_{ij} of AP is of $O(n^2)$ [13]. Since $\overline{\eta}_{ij}$ is just a maximum with a scalar, we can state that these messages are also $O(n^2)$.
- ϕ_{ij}: These messages are computed only for diagonal entries h_{ij}. Each message can be $O(n)$ in worst case. So computation of all messages is $O(n^2)$.
- α_{ij}: These messages are the same as in AP since we did not change the consistency constraint of AP. So computation of these messages is also $O(n^2)$.

Since all involved messages can be computed in $O(n^2)$ time, we can state that the worst time complexity of RAP is $O(n^2)$. In practice it reduces to $O(|\mathcal{E}|)$ where $\mathcal{E}$ is the set of edges along which $s(i,j)$ values are defined.

2 ADDITIONAL FEATURES OF RAP

Besides the features that we already mentioned in Sec. 5, we will elaborate on some additional features of RAP in this section.

2.1 Pruning for Refined Inter-exemplar Connections:

Sec. 5.3 briefs how the inter-exemplar connections histogram can be used to analyse cluster strength in different regions of a cluster. An analyst can manually look at the weakly connected pairs to decide whether the links should be retained or *pruned*. We can go one step further to automate this process of pruning. Once RAP messages have converged, we select m nearest exemplars for every exemplar because we are interested in only the connections between nearby exemplars. For these $|\mathcal{E}|m$ exemplar pairs, we can simply prune all the boundary connections where connection strength is less than a predefined threshold n_t. The computational complexity of this process is $O(nm|\mathcal{E}|)$ which is less than $O(n^2)$ for all practical purposes. This step is performed during the decision phase before finding the connected components in Alg. 1. Hence, if needed, it can even be iterated for multiple values of n_t without having to run the message passing phase every time. In the example of Fig. 6 in our work, pruning with $n_t = 2$ separates all exemplar pairs with single boundary connection, thus breaking the single link in the weak strength region of the red cluster and resulting in 3 clusters. However, for 19 out of 20 exemplar pairs, both local exemplars belonging to the pair still fall in the same cluster. Only one exemplar pair now has its members in different clusters. Hence, by comparing the results of RAP with $n_t = 1$ (i.e. no pruning) and $n_t = 2$, this one pair can be identified programmatically without going through all 20 pairs manually. Therefore, by running the pruning process for different values of n_t, we not only split the discovered clusters at the weaker regions, but also find the data points that were responsible for merging these regions. In our work, we fix $n_t = 3$ independent of the dataset and other settings.

2.2 Local Exemplars Connected to a Point

When a data point is connected to more than one well separated local exemplars, this signifies that the data point shares a mix of properties of these local exemplars. The local exemplars connected to x_i are given in $\mathcal{L}[i]$ which can be computed in $O(|\mathcal{E}|)$. Hence we can find the number of local exemplars connected to each data point in $O(n|\mathcal{E}|)$. This information can be visualized using a histogram where the x-axis represents the number of local exemplars and the height of each bar represents the number of data points connected to these many local exemplars. Such a histogram can be used for various purposes. It can allow for efficient hyperparameter tuning which we will discuss in Sec. 3. It can also be used to recognize the relative densities of different clusters in a dataset. We illustrate this on different datasets shown in Fig. 10. For this purpose we choose $\epsilon > 100$ percentile of $s(i,j)$, i.e., $G_j(\cdot)$ constraints are rendered inactive. As mentioned in Sec. 4.4 this often has a negligible impact on the global cluster discovery. At the same time this setting encourages multiple local exemplars to appear close by in dense regions, allowing a better analysis of relative densities.

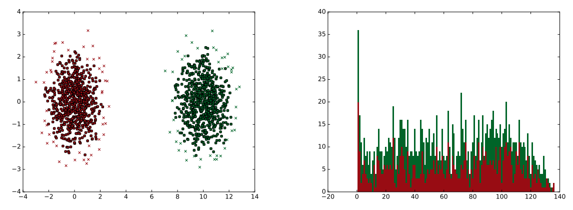
Fig. 10(a) shows that for *Blobs* toy dataset [21] with clusters having equal densities, the histogram bars are co-located. On the other hand, the bars for red cluster in Fig. 10(b) are located further to the right of the bars for the green cluster, indicating that for red cluster the data points are connected to more local exemplars, which in turn implies a higher density of the red cluster. Similar behavior can be seen for the blue cluster in Fig. 10(c) and green cluster in Fig. 10(d).

2.3 Clustering New Data Points

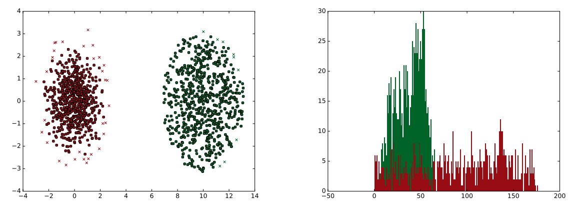
The local exemplars, discovered by RAP , tend to lie in the dense regions of the clusters rather than the boundaries that may have noisy data. Therefore, compared to nearest neighbors, they serve as better reference points for analyzing cluster assignment confidence as well for clustering new points. Suppose that we want to cluster a new data point $x_{new} \notin \mathcal{X}$ based upon the clustering results of $\mathcal{X}$. For this purpose, we can compare x_{new} to the already found local exemplars ($\mathcal{E}$) and assign it to the cluster which contains the closest local exemplar. This usually lowers the complexity by orders of magnitude when compared to assigning a cluster by finding the closest neighbor(s) in the already clustered dataset, as would be the case for algorithms that do not provide any local information. The new data point x_{new} can be treated based upon its similarity with its potential exemplars.

- If x_{new} has a high enough similarity with two or more local exemplars belonging to the same cluster, connecting x_{new} to all of them will not affect cluster assignments.
- If x_{new} has a high enough similarity with two or more local exemplars belonging to different clusters, the case may be forwarded to a human analyst for expert opinion in order to decide whether x_{new} should be assigned to one of the two clusters or should the clusters be merged.
- If the new data point has low enough similarity to all of the local exemplars, it can be treated as an outlier.

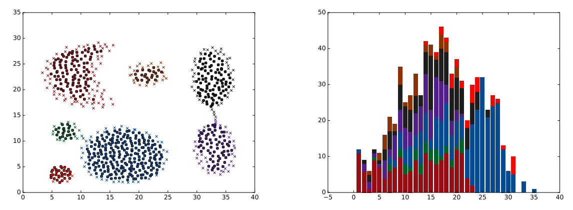
Fig. 11 illustrates an example of clustering new data points where already discovered local exemplars are circled in blue color and new points to be clustered are bordered in red and green colors. We



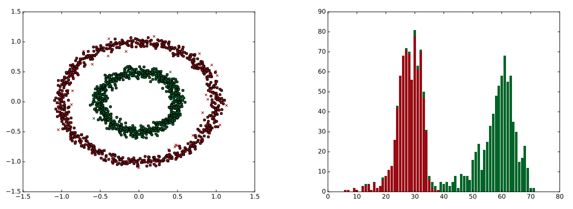
(a) Blobs with equal cluster densities



(b) Blobs with unequal cluster densities



(c) Aggregation



(d) Concentric Circles

Figure 10: Relative Densities with right column showing RAP results for $\epsilon > 100$ percentile of $s(i, j)$. Left column shows the relative cluster density histograms. For (c) and (d) same parameters are used as mentioned in Sec. 6.1. The figure shows displaced histogram bars for clusters with different relative densities. Best viewed in high resolution.

can observe that the red point is most similar to the bottom-right exemplar while the green point is visually similar to both top-left and center-left exemplars. Therefore based upon the given pairwise similarities, we may choose one or multiple local exemplars to represent the new data point. We can apply this procedure to multiple new data points as well. In that case we should just be more careful that if we notice a sufficiently large number of new data points being declared as outliers this may be an indication of a new cluster being formed which was not present in the clustering results for $\mathcal{X}$ using RAP . Also if we notice enough data points having close enough similarities to two exemplars from different

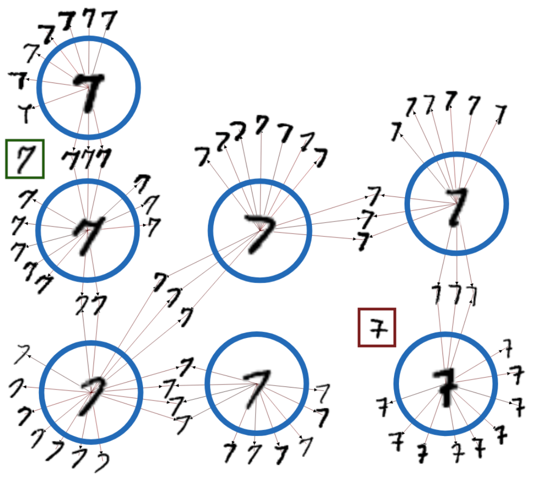


Figure 11: RAP results sample on MNIST with local exemplars encircled in blue. We cluster new data points, bordered in red and green, by assigning them to the most similar exemplars.

clusters, they may form a strong enough bridge between the two clusters to merge them to one. In these cases it is a good idea to either re-cluster the whole dataset or a partial subset of the dataset.

2.4 Outliers and Noisy Data

We briefly mentioned for in Sec. 6.1 how RAP handles noisy boundaries between clusters for different synthetic datasets. We will now explore how RAP handles outliers. It starts by considering all data points as potential local exemplars and then explores the neighborhood of each data point to later decide if a data point should become a local exemplar or not. This provides RAP the ability to discover outliers as they do not tend to connect to other local exemplars that belong to well-formed bigger far away clusters. This is depicted in Fig. 12, where we have the half-moons dataset with 2 clusters and we have now added some outliers to it. Specifically, we have a lone outlier on the top right side of the figure and a small cluster of three outliers on the bottom left of the figure. RAP recognizes the lone outlier as a single point cluster, represented by the purple color. Similarly, RAP also puts together the three outliers on the lower-left corner as one small cluster, designated by blue color. It is also important to note that the presence of these outliers does not impact the clustering of the two well-formed bigger clusters. This can be observed by comparing Fig. 12 to Fig. 3(b), where we have the same dataset but without the outliers.

3 HYPERPARAMETERS - TUNING AND SENSITIVITY

3.1 Successive Tuning Methodology

By keeping ϵ fixed to a small value e.g. 99 percentile of input $s(i, j)$ values, we are left with two parameters, p and q. We can perform efficient successive tuning of the hyperparameters easily to obtain desired results in a few steps. This is due to the following reasons:

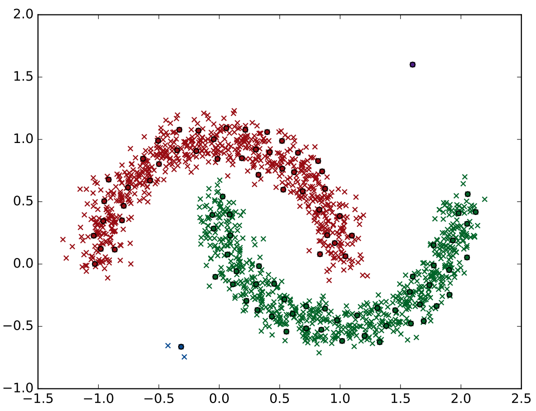


Figure 12: Half-moons dataset with outliers. These results are obtained using the same parameters as Fig. 3(b), i.e., $p = 0.6$, $q = -0.97$ and $\epsilon = 0.99$.

- It is easy and intuitive to understand the impact of changing any of the two hyperparameters on the clustering outcome.
- The abundance of insightful information, beyond just cluster assignments, provided by RAP at each step helps the analyst in deciding how to adapt the hyperparameters to improve the results.

Some ways to utilize local information to guide the subsequent steps in hyperparameter tuning are as follows:

- **Too few local exemplars:** A small number of local exemplars indicates that we have set p too low and we should increase it such that we obtain enough local exemplars that are spread throughout the dataset.
- **Boundary connections:** Once we have an adequate number of local exemplars available, we focus on building boundary connections between the local exemplars for global structure discovery. For this we use inter-exemplar connections histogram to tune q as follows
  - **Inter-exemplar connections histogram shifted too much to the left:** This configuration implies that the maximum linkage penalty is too high and hence not allowing sufficient boundary connections to form between the local exemplars. This will also manifest itself in terms of many more clusters than expected. In this scenario, we should decrease the linkage penalty so that more boundary connections can appear.
  - **Inter-exemplar connections histogram shifted too much to the right:** This scenario is the opposite of the previous one. Due to the low linkage penalty data points also get connected to far off local exemplars (which possibly may belong to a different "ground truth" cluster). This can result in too few clusters than expected and violates the basic principle of introducing the penalty q that we want to form only boundary connections between close enough local exemplars to facilitate global cluster discovery. Allowing points to connect to even far off local exemplars rather distorts the global cluster discovery by even merging well-separated clusters. In this scenario, we should increase the linkage penalty.
- **Confidence values:** The confidence values discussed in Sec. 5.2 can also be used to determine if the results obtained for a specific choice of hyperparameters are satisfactory. Analysis of a few points with low confidence values can point out whether we have two clusters lying close by, or that the algorithm has not been able to form boundary connections to merge sub-clusters where it should have. In the second case we need to reduce the linkage penalty to merge these sub-clusters.

Finally, we want to emphasize that all the results presented in this work, unless explicitly stated otherwise, are obtained via successive tuning of hyperparameters rather than any kind of parameter sweeps. Hence these results are a good representation of what one can expect in the normal application of RAP and do not correspond to the ceiling performance on the respective datasets. Furthermore, we also notice that the final results shown in Sec. 6.1 and Sec. 6.2 correspond to hyperparameter values that are very similar for different datasets although the datasets have very different characteristics. This also points to the ease of finding suitable hyperparameters for a wide variety of datasets with different cluster characteristics by exploring a very narrow range of hyperparameter values, as long as the datasets employ the same (or similar) pairwise similarity metric.

3.1.1 *Successive Tuning Example.*

We will now present a step by step example of using the insights from Sec. 4 and Sec. 3.1 to do hyperparameter tuning for the half-moons dataset earlier used in Fig. 1 and Fig. 7. We keep $\epsilon = 99$ percentile of $s(i, j)$. These steps are illustrated in Fig. 13 where each row corresponds to a step. The columns list clustering results with exemplars, inter-exemplar connection strength, exemplars connected to data points and cluster assignment confidence values respectively. Note that RAP only needs $s(i, j)$ values and the plots in the first and fourth columns are for sake of illustration only. The relevant information in these plots is also available otherwise for the analyst.

(1) As the first step, we apply RAP to the dataset for $p = 60$ percentile of $s(i, j)$ while keeping $q = -\infty$. This gives us 60 clusters as shown in the first row of Fig. 13. The results show that there is only one exemplar connection per point as we have set the penalty q to maximum. As a result there is no need for inter-exemplar links as there is no exemplar pair with common points. The analysis of cluster assignment confidence shows that there are many similar points that have been assigned to different clusters. These points can easily be obtained programmatically for an analyst to decide whether they should be connected or not. We accept this relatively large number of clusters in the first step because we know that they will be merged in subsequent steps. If required, we can also check the dependence of results on p by playing around with this parameter. For instance, the current dataset yields 73 and 38 clusters for $p = 80$ and $p = 20$ percentile of $s(i, j)$ respectively.

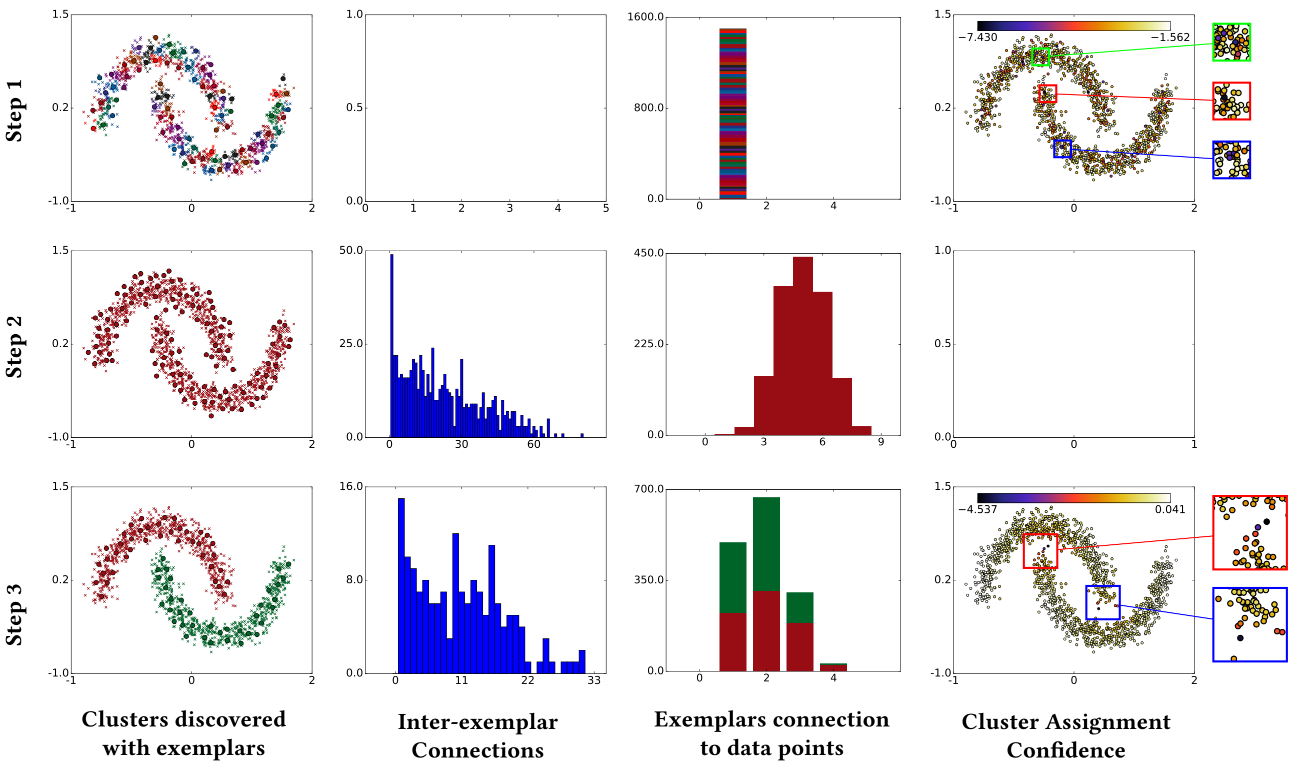


Figure 13: Successive tuning of hyperparameters p and q for half-moons dataset. We use the insights provided by current step to refine our choice of p and q for next step.

(2) First step shows that there is a need to connect many closely located clusters. For second step, we fix $p = 60$ percentile of $s(i, j)$ and select $q = 95$ percentile of $s(i, j)$ to encourage multiple exemplars per data point. However this results in a single cluster. We can see from the histograms of second row in Fig. 13 that on average 5 points are connected to a point. Moreover, inter-exemplar connections are quite high. There are exemplar pairs with more than 60 common points. This indicates that we need to increase the penalty to force lesser common connections between exemplars. There is no need for a confidence plot here as we have only one cluster.

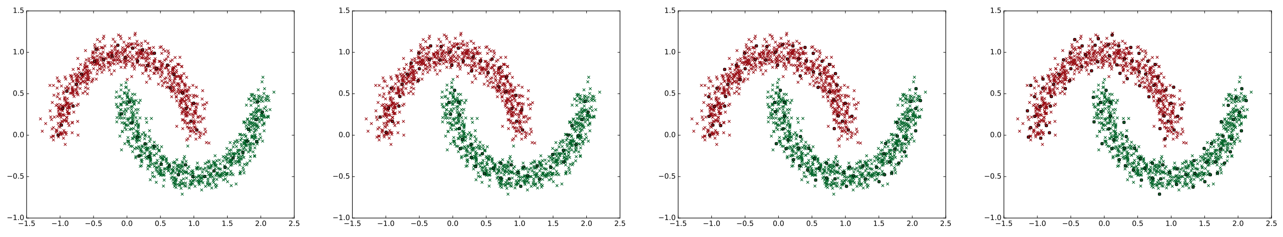
(3) Based upon insights from the first and the second step, we now need to have q somewhere between 100 and 95 percentile of $s(i, j)$ while keeping p fixed. We choose $q = 97.5$ (i.e. mean of 100 and 95) percentile and get 2 clusters as shown in the third row of Fig. 13. The histograms now show that most data points are now connected to 1, 2 or 3 local exemplars which is a reasonable result. Moreover, the inter-exemplar connections are also reduced compared to the second step, although there are still some pairs with more than 22 common points. It now depends upon the analyst if he wants to further reduce these connections or if he is content with the results. The co-located bars in the histogram of exemplar connections per point also indicate that the two discovered clusters have similar density.

3.2 Sensitivity of Clusters Discovery to Hyperparameters

We look at how global structure discovery is impacted by the variation of individual hyperparameters. For this purpose, we use the example in Sec. 3.1.1 and start with the hyperparameters set selected in the last step of successive tuning i.e. $p = 60$, $q = -97.5$ and $\epsilon = 99$ percentile of $s(i, j)$ values. We then change one parameter at a time to see the sensitivity of the results.

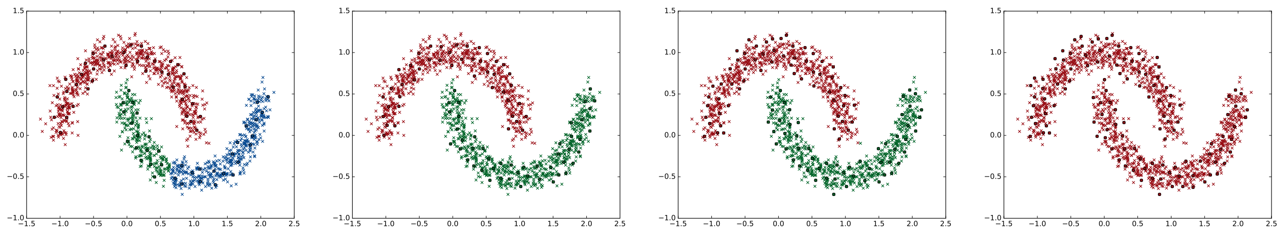
3.2.1 Sensitivity to p: For fixed value of q and ϵ, p controls the number of local exemplars. In general as long as p is not too low the global cluster results don't get impacted as shown in Fig. 14. This is unlike AP where variations in p have a strong impact on the number of clusters obtained and the cluster assignments. In the case of RAP , as long as p is varied in a reasonable range, this only changes the number of local exemplars appearing in any region but as long as the number is not too small and they can get connected via boundary connections, the number of globally discovered clusters does not change. Only when p is too low, we are no longer able to form bridge connections between the appropriate exemplars and we lose the ability to discover the global structure.

3.2.2 Sensitivity to q: q is arguably the hyperparameter which needs the most care but there is a reasonable range where the global clusters obtained are not affected and successive tuning can discover a suitable value of q in this range quickly. When we have



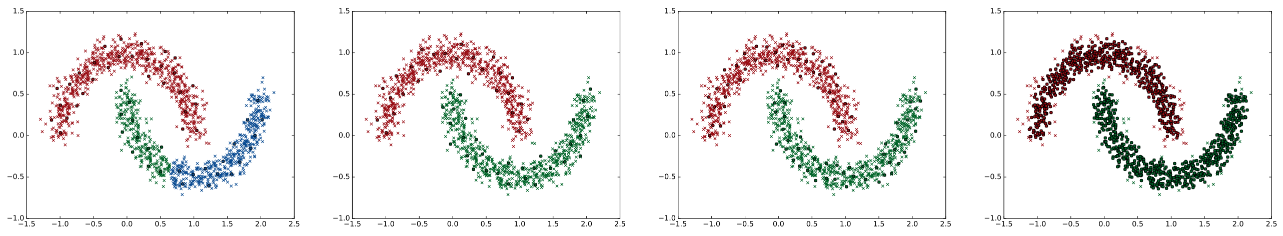
(a) $p = 10$ **(b)** $p = 30$ **(c)** $p = 60$ **(d)** $p = 80$

Figure 14: Sensitivity to p while keeping q and ϵ constant. All values are in percentiles of input $s(i, j)$ values. Best viewed in high resolution



(a) $q = 96$ **(b)** $q = 97$ **(c)** $q = 99$ **(d)** $q = 99.5$

Figure 15: Sensitivity to q while keeping p and ϵ constant. All values are in percentiles of input $s(i, j)$ values. Best viewed in high resolution



(a) $\epsilon = 98$ **(b)** $\epsilon = 98.5$ **(c)** $\epsilon = 99$ **(d)** $\epsilon = 100$

Figure 16: Sensitivity to ϵ while keeping p and q constant. All values are in percentiles of input $s(i, j)$ values. Best viewed in high resolution

a too high linkage penalty, clusters start to break at their weakly connected regions as seen in Fig. 15. Note that still in Fig. 15 there have not been drastic changes in the results beyond the green cluster being broken where it was most weakly connected.

3.2.3 Sensitivity to ϵ: Fig. 16 shows the effect of changing ϵ on global clusters discovery while keeping p and q fixed. It can be seen that for small values of ϵ, the global clusters discovery is not affected. However, if the purpose of ϵ, which is only to avoid crowding of local exemplars in dense regions, is not kept in mind and the values are changed beyond a limit, they start affecting the clusters. This can be seen in the case when $\epsilon = 98$. It is worth noticing that again, the cluster gets split at the weakly connected region. Practically, ϵ can be considered as a fixed parameter at 99 percentile of input $s(i, j)$.

Dataset	Points	True clusters
Aggregation	788	7
Flame	240	2
R15	600	15
Circles	1500	2
Optdigits	1797	10
MNIST	1000	10
Skin	5000	2
PPI	1171	276

Table 4: Statistics of the synthetic and real-world datasets

4 EXPERIMENT DETAILS

4.1 Datasets

Tab. 4 gives the number of data points and number of ground truth clusters for all the datasets used in the experiments. For synthetic datasets, these are the only relevant statistics as all the synthetic datasets are in $\mathbb{R}^2$ and the plots have already been given in Fig. 7. The real-world datasets are further explained below

- **Optdigits** consists of grey scale 8×8 images of handwritten digits from 0 to 9. We only use the test set, consisting of 1797 images, for clustering.
- **MNIST** is also a similar dataset of 28×28 gray scale images of handwritten digits. We choose 1000 images from the dataset at random to define $\mathcal{X}$.
- **Skin** dataset has of 245, 057 points constructed over (B, G, R) color space, consisting of two classes: skin tones and non-skin tones. The skin tones are obtained using skin textures from face images of people of different ages, gender and ethnicity. We randomly choose a subset of 5000 points that are equally distributed between the two classes.
- **Protein-Protein Interactions** is the Saccharomyces Cerevisiae gold standard dataset. The pairwise similarity values for this dataset represent protein-protein interactions. We use the probabilistic measure, proposed in [24], to define our pairwise similarity matrix. This dataset is different from the previously discussed synthetic and real datasets in the sense that the available ground truth does not necessarily assign each data point to only one ground truth cluster. Furthermore, some ground truth clusters are a subset of other ground truth clusters. To reduce such overlap in the discovered clusters, we threshold the protein-protein interactions such that the interactions below 0.4 are set to 0[2]. Furthermore, we only consider proteins that are common with CYC2008. This leaves us with 1171 proteins (data points) assigned to 276 ground truth clusters.

4.2 Implementation Settings

For AP, we report the performance using the default parameter selection ($p = 50$ percentile of input $s(i, j)$) as well as the ceiling performance obtained by sweeping p from 10 to 90 percentile of input similarities with a percentile step size of 5. All the algorithms, except AP, require tuning of two hyperparameters. For HAP and SCAP, although there exist good intuitions for the hyperparameters involved, there is no simple way of choosing them. Therefore we tune the parameters of HAP and SCAP manually by searching over preference ranges between 10 and 80 percentile of $s(i, j)$ values with a percentile step size of 5 and present the average accuracy score of the top 3 results. For MEAP, we use the default settings of parameters as suggested by the authors [30] i.e. the preference and linkage parameters are set to the medians of pairwise similarities and linkage matrix respectively, where linkage matrix is computed by normalizing pairwise similarity matrix by n. It is a common practice in message-passing based algorithms to aid convergence by *damping* every message m in iteration $(t + 1)$ by a factor of $\lambda \in [0, 1]$ i.e.

$$m^{t+1} := (1 - \lambda)m^{t+1} + \lambda m^t. \tag{35}$$

The damping factor λ is kept at 0.8 in all experiments for all the message passing based algorithms. It is worth noticing that within reasonable ranges (e.g. $\lambda \geq 0.5$), the results remain the same and λ only affects the speed of convergence. Moreover, maximum iterations are fixed to 1000 and convergence-iterations are fixed to 100. i.e. if the results remain the same for 100 iterations, the algorithm stops irrespective of the number of maximum iterations. We keep the number of maximum-iterations high to encourage all the competitors to converge as some algorithms might take a bit longer to converge compared to the others. For MCL, we sweep inflation from 1 to 5 with step-size of 1,. and report the best results. For DBSCAN and OPTICS, we use the default parameter settings of `sklearn` python package. The boundary points for DBSCAN and OPTICS are clustered using 5 nearest neighbors.

For RAP , in all the experiments, we used successive tuning for hyperparameter selection, unless specified otherwise. We can see that successive tuning may lead to a self-preference value higher than the default for AP i.e. 50-percentile of $s(i, j)$. This is intuitive because in RAP , we have a relatively higher number of the local exemplars with sub-clusters that are subsequently merged using boundary connections. All the messages are initialized to 0. For inter-exemplar connections histogram, we keep $m = 5$ i.e. we target only 5 nearest exemplars for every exemplar. In addition to the heuristic performance, we also present the ceiling performance obtained by sweeping p over the percentile range from 10 to 80 percentile of $s(i, j)$ with the percentile step size of 5, and q from 90 to 99 percentile of $s(i, j)$ with the percentile step size of 1. Rest of the settings are kept the same as other message passing based competitors. Implementation is available at [3].

[2]This approach has been adopted in previous studies too e.g. [29] and [22], although different thresholds have been used there.