Cluster Management of Scientific Literature in HSTOOL

Johan Schubert
Department of Decision Support Systems
Swedish Defence Research Agency
SE-164 90 Stockholm, Sweden
johan.schubert@foi.se

Ulrika Wickenberg Bolin
Department of Decision Support Systems
Swedish Defence Research Agency
SE-164 90 Stockholm, Sweden
ulrika.wickenberg-bolin@foi.se

***Abstract*—In this paper, we expand a methodology for horizon scanning of scientific literature to discover scientific trends. In this methodology, scientific articles are automatically clustered within a broadly defined field of research based on the topic. We develop a new method to allow an analyst to handle the large number of clusters that result from the automatic clustering of articles. The method is based on estimating an information-theoretical distance between all possible pairs of clusters. Each of the scientific articles has a probability distribution of affiliation over all possible clusters arising from the clustering process. Using these, we investigate possible pairwise mergers between all pairs of existing clusters and calculate the entropies of the probability distributions of all articles after each possible merger of two clusters. These entropies are visualized in a dendritic tree and a cluster graph. The merger with minimal total entropy is the proposed cluster pair to be merged.**

Keywords—horizon scanning, scientometrics, Gibbs sampling, Dirichlet multinomial mixture model, entropy, clustering, node reducing

I. INTRODUCTION

Methods for scanning scientific literature to discover new scientific trends are important in research. These methods are designed to discover changes, disruptions, and trends with the potential to significantly affect the development of a certain area of interest. For scientific literature, our goal is to discover emerging or rapidly growing research areas and to identify technologies that have reached a level of preparedness that is suitable for industrial applications.

For this purpose, we have developed a methodology and a computer system called the Horizon Scanning Tool (HSTOOL) [1]. HSTOOL is a system for scanning scientific literature in databases to discover scientific trends within a broadly defined field of research. With search queries specified by subject matter experts and iteratively tested by studying the results of multiple scans, we let HSTOOL retrieve titles and abstracts from the Web of Science (WOS) Core Collection in a format that enables automatic data processing. We can then automatically group research articles into clusters by subject content. The focus is on identifying groups of research articles that together constitute a research topic, studying the development of the topic over time, and using the research community's citation statistics regarding the included articles to identify the most important contributions within each research topic.

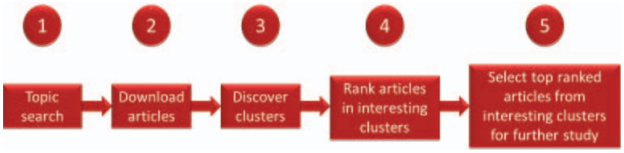

Fig. 1. Workflow for horizon scanning of scientific literature [1].

In Fig. 1 we show the workflow of horizon scanning in five steps. The process facilitates scanning of broad areas defined by a general topic (step 1). Once a search has been performed and articles downloaded (step 2), topics are automatically discovered using a clustering algorithm that groups the scientific articles based upon textual contents (step 3). Clusters of articles can then be selected for further studies. To find the key contributions from a cluster of interest, a ranking method is used (step 4). Once top-rated contributions for a subject area have been identified, a manageable subset of articles can be selected for detailed studies (step 5) [1].

An observation resulting from using HSTOOL has been that there is a need to be able to handle the number of clusters in different ways. This problem emerges when several analysts have to jointly analyze a large number of clusters. The question becomes who takes which clusters for further analysis. Each analyst should take clusters of a similar type. In another situation, an individual analyst may want to merge a large number of clusters into a smaller number of clusters to manage the resolution before starting to analyze the documents. In this situation, the question becomes which clusters should be merged. That is, which clusters are close to each other in an information theory sense?

In this article, we develop a mathematical method for assessing the information theory distance between each pair of clusters. The method is based on output from the completed clustering process and uses each article's probability distribution that each article has about which cluster it belongs to. This method tests different mergers and assesses the effects on the articles' probability distributions. Thereafter, all assessments are aggregated. The merging of clusters that provides the best partitioning of articles is preferable and can be observed in a cluster graph and a dendritic tree.

Within Horizon Scanning, classes are assumed unknown. Therefore, unsupervised clustering algorithms are used. In the framework of unsupervised clustering, Wang [2] and Wang et

978-1-6654-6283-9/22/$31.00 ©2022 IEEE
DOI 10.1109/ICMLA55696.2022.00062

al. [3] studied aggregation methods to be used before clustering to reduce computational complexity. Other authors have considered the case when classes are pre-specified [4] when supervised techniques such as Support Vector Machines or Naïve Bayes are used for text mining [5].

In Section 2, we describe the method of clustering articles [1] using a Dirichlet multinomial mixture model (GSDMM) algorithm [6, 7] and a method we developed to automatically determine the numbers of clusters. In Section 3, we develop a method that allows an analyst to manage the number of clusters based on visualization of cluster distances in a dendritic tree and a cluster graph. Finally, conclusions are drawn (Section 4).

II. CLUSTERING OF SCIENTIFIC ARTICLES

Once a search result has been downloaded from WOS with HSTOOL, we want to group all articles that touch on the same subject area into a cluster that will be treated as a separate subproblem.

In the following two subsections, we describe how to use a Gibbs sampling algorithm for a Dirichlet multinomial mixture model (GSDMM) [6, 7] to organize articles into clusters with common subject areas and how to determine the optimal number of clusters.

A. Clustering with GSDMM

To group articles within the same subarea, we use the above-mentioned GSDMM algorithm. Simply described, this method starts with a large number of clusters and a random distribution of articles between clusters. The method then examines each article to determine if it is a better fit in any other cluster than in its current placement. This procedure is repeated iteratively for all articles until no further changes are made.

The method proceeds by comparing all words in each article title and abstract with the corresponding words in all other articles. If a word is missing or appears a different number of times than in another article, then the probability that these articles belong together is assigned a lower value. These probabilities are combined for all articles in each cluster. This results in an evaluation of each cluster regarding how well each article fits into all the different clusters. Then, the article is moved to a cluster where it fits well according to these probabilities. The procedure is applied to all articles and iteratively repeated until all articles are placed in their best-matched clusters.

The clustering process is performed by a sequence of Gibbs sampling iterations. During each iteration, we calculate, for each article, the probability that it belongs to each cluster k, which results in the probability that the article will be moved to that cluster.

We have [6]:

$$p_{dki}\left(k_d = k \middle| \vec{k}_{\neg d}, \vec{d}\right) \propto \frac{m_{k,\neg d} + \alpha}{D - 1 + K\alpha} \cdot \frac{\prod_{w \in d} \prod_{j=1}^{N_d^w} \left(n_{k,\neg d}^w + \beta + j - 1\right)}{\prod_{i=1}^{N_d} \left(n_{k,\neg d} + V\beta + i - 1\right)}, \quad (1)$$

where on the left-hand side, k_d is the cluster position of article d, k is the kth cluster, $\vec{k}_{\neg d}$ is the set of cluster positions of all other articles excluding d, and $\vec{d}$ is the set of all articles. In the first factor on the right-hand side, $m_{k,\neg d}$ is the number of articles in cluster k not including d, α is a cluster parameter set to 0.1 in our test case, D is the total number of articles under consideration, and K is the initial number of clusters. In the second factor on the right-hand side, w is the wth word of article d, N_d^w is the number of times word w appears in article d, $n_{k,\neg d}^w$ is the number of times word w appears in cluster k when article d has been removed, β is a cluster parameter that will determine the number of final clusters, N_d is the number of words in article d, $n_{k,\neg d}$ is the number of words in cluster k when article d has been removed, and V is the number of words in the vocabulary. We choose $i = 14$ based on the observation that the algorithm usually converges in 10–12 iterations. Thus, $|\{p_{dki}\}_{i=14}|$ is equal to the final number of clusters.

The computational time complexity for each iteration i of GSDMM was found to be $O(KD\bar{L})$ [6] where $\bar{L}$ is the average length of the articles, which compares to k-means [8] with a time complexity of $O(KDS)$, where S is the maximum number of non-zero elements in the vectors of centroids of the clusters. When clustering short abstracts $\bar{L} \ll S$. Thus, GSDMM outperforms k-mean on short texts.

B. Select the number of clusters

To choose the best number of clusters, we need to evaluate different options. For this purpose, we evaluate different numbers of clusters based on the quality of the clustering.

The GSDMM algorithm does not require a predetermined number of clusters to assign the articles to a given corpus[1]. However, the number of clusters depends on parameter $\beta \in (0, 1)$, as shown in (1). A value of β near zero results in many clusters, while a value of β near one produces fewer clusters.

We focus on the articles that have been clustered and examine how well they fit into the clusters where they have been placed. Each article has a probability distribution across all clusters that indicates the probability that each cluster is the optimal location for that article as defined in (1). This distribution is calculated and used in the clustering process for GSDMM and is recalculated in each step of the clustering process for all articles. At the end of the clustering process, we use the final calculated probability distribution for each article.

We consider $\{p_{dki}\}$, where p_{dki} is the probability that article d belongs to cluster k at iteration i in (1), with:

[1] The collection of all articles from a particular search.

$$\sum_{k=1}^{K} p_{dki} = 1 \tag{2}$$

for any constant d and i, where K is the initial number of clusters.

If the placement of a particular article is almost certain, that article will have a probability value of close to one for that cluster. To study the convergence of the GSDMM algorithm, we calculate at each Gibbs sampling iteration i the entropy [9] for each article d as:

$$Ent_{di} = -\sum_{k=1}^{K} p_{dki}\left(k_d = k|\vec{k}_{\neg d}, \vec{d}\right) \cdot \log\left[p_{dki}\left(k_d = k|\vec{k}_{\neg d}, \vec{d}\right)\right]. \tag{3}$$

To determine the quality of a specific clustering (i.e., the clustering at a specific iteration i for a specific value of β), we calculate its entropy as:

$$Ent_i = \sum_{d=1}^{D} Ent_{di}. \tag{4}$$

A good measure of the quality of the entire partition of all articles for a particular clustering process is the sum of entropy over all articles after the final 14th iteration, where Ent_{14} is the target entropy to be minimized.

As β increases, there is a decline in the final entropy for each clustering process.

The number of clusters keeps decreasing as β approaches 1. Ideally, we want to find a partition that has well-defined clusters that correspond to subject areas and yet has the lowest possible entropy.

To estimate the correct number of clusters, the final entropy derived from clustering with different values of β is calculated. If β is small, then entropy is high; as β increases, entropy decreases with a small residual entropy at high β. This is similar to what we did in [10, 11], where alternative partitions were evaluated using the entropy of another probability measure. The entropy's behavioral change occurs at a point that we believe provides the best number of clusters [12]. In Fig. 2, we observe a change in the behavior of the entropy at a point corresponding to the smallest acute angle between the left and right line segments of the concave lower envelope of entropy. This point corresponds to the best number of clusters, and the β used in this clustering is selected.

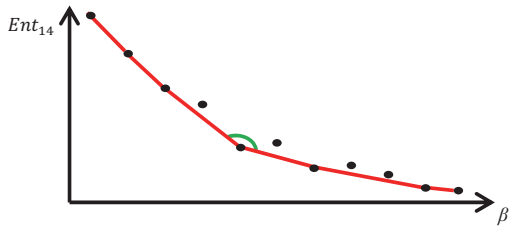

Fig. 2. The red line is the concave lower envelope of the black dots, and green is the minimizing angle.

III. Cluster Management

Often, there is a reason to choose a different number of clusters than what resulted from the automatic determination of the number of clusters. One such reason may be that the subsequent analysis is to be performed by a group of analysts who will carry out different parts of the analysis. In this situation, it may be appropriate to merge several clusters so that each analyst obtains a set of related clusters that cover a broader area rather than many unrelated clusters. Another such reason is that one may want a uniform resolution in all subareas of the subject. Clustering can lead to a resolution within one subarea that differs from another. For example, a search for vehicles may result in a cluster of articles regarding trucks, while articles relating to passenger cars have been divided into different clusters according to car brands. If one wants to keep a consistent resolution in all clusters, then the clusters with articles about different car brands can be merged.

Of course, there may be other reasons why we want to change the number of clusters or why we may want to merge the articles found in certain clusters into one single cluster. Regardless of the reason to merge certain clusters, it is important to obtain information about how consistent the articles are in different clusters. Therefore, we calculate the distance between each pair of clusters and visualize the results of these calculations in a dendritic tree and a cluster graph.

A. Distance between clusters

To estimate the distance between clusters, we examine the consequence of merging each possible pair of clusters. This is done by estimating the change in entropy across all articles with regards to a possible merger of the two clusters. Remember that while each article is placed in a cluster based on the probability that the article belongs to that cluster, each article has a probability distribution of belonging to each cluster.

Let us first write $p_{ds} = p_{dsi}$ because i = 14 is treated as a constant after the clustering process is completed. Second, p_{ds} is a filtered version of p_{dki} of (1) with cluster position for each article containing only R index values ($R \leq K$) corresponding to the R number of clusters that contain at least one article after the last sampling of (1), with i = 14, i.e., $|\{p_{ds}\}| = R$. Note that in most clustering processes, the final number of clusters R is significantly less than the original number of clusters, which is 500 in our settings.

In Fig. 3, article number 1 is placed in cluster number 1, χ_1. However, it has a probability distribution $\{p_{1i}\}_{i=1}^{3}$ over all clusters $\{\chi_i\}_{i=1}^{3}$.

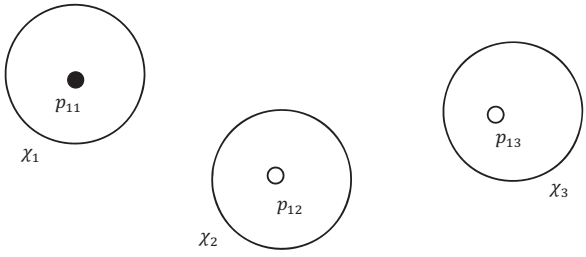

Fig. 3. An article with probability p_{11} placed in cluster χ_1 still has a probability distribution over all clusters.

If we merge clusters χ_1 and χ_2, article 1 would have a probability for the merged cluster of $p_{11} + p_{12}$ while keeping the probability p_{13} for cluster χ_3 unchanged.

We have:

$$\forall dstr | d \in \{1, \dots, |\vec{d}|\}, s \in \{1, \dots, R-1\}, t \in \{s+1, \dots, R\},$$

$$r \in \{1, \dots, R\}. q_{dstr} = \begin{cases} p_{ds} + p_{dt}, & r = s \\ 0, & r = t \\ p_{dr}, & r \neq s, t \end{cases}, \quad (5)$$

where q is the probability of two merged clusters, d is an article index, s and t are indices that control the cluster merging of clusters s and t, and r is the cluster index. Thus, q_{dstr} are elements of a three-dimensional matrix q where each dimension depends on d, the pair (s, t) and r, respectively, with dimension $|\vec{d}|\ x\ \frac{R(R-1)}{2}\ x\ R$. As an example, if we have 100 articles ($|\vec{d}| = 100$) and 10 nonempty clusters ($R = 10$), then q will have dimension $100\ x\ \frac{10 \cdot 9}{2}\ x\ 10 = 100\ x\ 45\ x\ 10 = 45\ 000$.

To be able to compare how suitable different partitions of articles are, we need to be able to evaluate each possible merger of two clusters. The idea is that we can find which clusters are close to each other in terms of information theory. Directly comparing different clusters with each other when they have widely differing numbers of articles can be difficult to do objectively. Instead, we view the problem from an article's perspective and study how different articles are affected by merging two clusters. Since each article has a probability distribution over R clusters, we can measure the effect when two clusters are merged using the entropy for the new resulting probability distribution after the merger. A merger that entails minimum entropy is preferable because it corresponds to a probability distribution that is closest to a determination of the article's affiliation with a particular cluster.

We calculate the entropy for each article's probability distribution given each possible merger of two clusters, i.e., for the entire set $\{(s, t)\}$. We have:

$$\forall dst | d \in \{1, \dots, |\vec{d}|\}, s \in \{1, \dots, R-1\}, t \in \{s+1, \dots, R\}.$$

$$Ent_{dst} = -\sum_{r=1}^{R} q_{dstr} \cdot \log q_{dstr}, \quad (6)$$

where Ent_{dst} is the entropy of article d given a merger of clusters s and t. Thus, Ent_{dst} are elements of a two-dimensional matrix Ent where each dimension depends on d and (s, t), respectively, with dimension $|\vec{d}|\ x\ R(R-1)/2$.

To evaluate all possible mergers of two clusters against each other, we sum for each possible merger of two clusters the entropies for the new probability distribution of all articles after the merger. In this way, we can observe how each alternative merging of two clusters affects the resulting probability distributions for all articles. The merger that has the lowest sum of entropy calculated over the probability distribution for all articles is the preferred merger.

We calculate the sum of entropy from all articles d and each possible cluster merging (s, t). We have:

$$\forall st | s \in \{1, \dots, R-1\}, t \in \{s+1, \dots, R\}. SEnt_{st}$$

$$= \sum_{d \in \{1, \dots, |\vec{d}|\}} Ent_{dst}, \quad (7)$$

where $SEnt_{st}$ is the total entropy over all articles given a merger of clusters s and t. Thus, $SEnt_{st}$ are elements of a one-dimensional vector $SEnt$ whose length depends on the pair (s, t). The length of $SEnt$ is $R(R-1)/2$.

The estimated sum of entropy $SEnt_{st}$ over all articles d given the merger of clusters s and t is the distance sought between the two clusters. The two closest clusters are the pair (s, t) given by $\text{argmin}_{s,t}\ SEnt_{st}$, where $s \in \{1, \dots, R-1\}, t \in \{s+1, \dots, R\}$.

An example of a graph of clusters with arcs is presented in Fig. 4. The width of the arcs corresponds to the distance between the clusters, $SEnt_{st}$. Based on the width of the arcs in the cluster diagram, the analyst can choose which clusters to merge. After each selection, (5–7) are recalculated, and the cluster diagram is updated. Usually, we do not work directly with such sizable cluster diagrams. In the next section, we will introduce a dendritic tree that presents a proposed order for cluster mergers.

The pseudocode of an algorithm for calculating (5–7) is given in TABLE I. This algorithm calculates the sum of entropy over the probability distribution of all articles for each possible merger of two clusters. That sum is considered the distance between each pair of clusters.

The computational time complexity of (5–7) is $O(DR^2)$. This makes the proposed cluster management process as a post-cluster aggregation computationally reasonable since GSDMM has a time complexity of $O(KD\bar{L})$, where usually $R \ll K$ and $R < \bar{L}$.

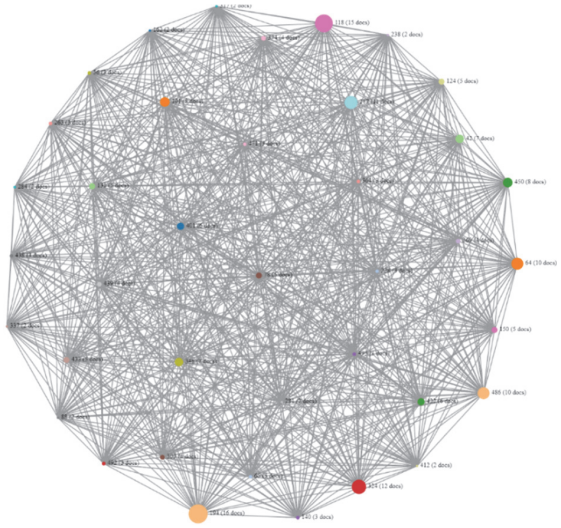

Fig. 4. A cluster graph where the node size corresponds to the number of articles in the cluster and the width of the arcs corresponds to the distance between the clusters (wide arches mean that the clusters are close).

TABLE I. PSEUDOCODE OF AN ALGORITHM FOR CALCULATING THE DISTANCES BETWEEN ALL CLUSTERS.

Algorithm: The sum of the entropy of the probability distribution for all paired clusters of articles.

```
algorithm calculateSENT(D,R,P)
   input: D integer,
          R integer,
          P array(D,R)
   var:   Q array(D,R),
          ENT array(D),
          SENT array(R,R)
   output: SENT array

   for s := 0 to R - 1
      for t := s + 1 to R
         for d := 0 to D
            for r := 0 to R
               // Compute (5)
               if r = s
                  Q(d,r):= P(d,s) + P(d,t)
               elseif r = t
                  Q(d,r):= 0
               else
                  Q(d,r):= P(d,r)
            end
            // Compute (6)
            ENT(d):= 0.0
            for r := 0 to R
               if Q(d,r) > 0.0
                  ENT(d) := ENT(d) - Q(d,r) * log(Q(d,r))
            end
         end
         // Compute (7)
         SENT(s,t) := 0.0
         for d := 0 to D
            SENT(s,t) := SENT(s,t) + ENT(d)
         end
      end
   end
   return SENT
end
```

Wang [2] and Wang et al. [3] proposed an alternative approach, where pre-clustering aggregation using node proximity [13] was used before spectral clustering [14]. This is another approach where aggregation before clustering is used to reduce the time complexity of clustering. Our approach of aggregation after clustering is done to manage the number of clusters and improve analysis.

The computational complexity of pre-clustering by Wang [2] was found to be $O(K^2 logK)$. With $R \ll K$, the computational time complexity of (5–7), being $O(DR^2)$, is less than pre-clustering for medium-sized clustering problems. For large clustering problems with several thousand documents (D), the method of Wang [2] can be used as pre-clustering before GSDMM, followed by the post-clustering method proposed in this paper.

B. *Managing the number of clusters*

Clustering can be visualized with both a cluster graph and a dendritic tree in HSTOOL software. This allows the analyst to see how closely related different clusters are to each other and choose which clusters to merge.

A dendritic tree corresponding to the cluster graph of Fig. 4 is shown in Fig. 5.

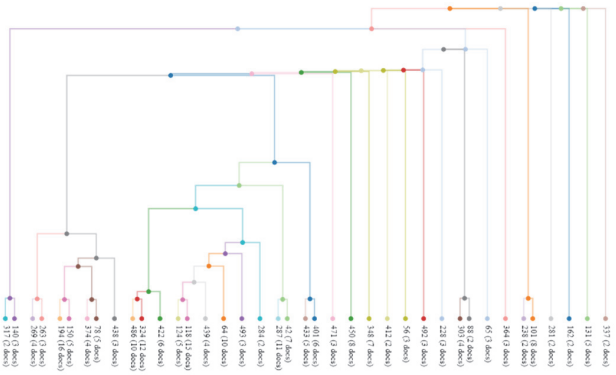

Fig. 5. A dendritic tree of the same problem as shown in the cluster graph in Fig. 4. The length of the vertical branches corresponds to the distance between the respective clusters. The node labels contain the cluster-ID.

When two clusters are merged through interaction with the dendritic tree or with the cluster graph, the clustering algorithm (TABLE I) is initiated recursively with all documents and the new cluster. If two clusters were merged in the interaction, the new cluster would contain all documents from the merged clusters but no other documents.

1) *Interaction with a dendritic tree*

The dendritic tree is produced by iteratively calling the clustering algorithm. First, the cluster pairs with the lowest entropy (i.e., distance) according to $SEnt_{st}$ (7) are identified. The probability distributions for these clusters are added to a joint cluster, and the clustering algorithm is then called again. This procedure is repeated until all distances are obtained, and a dendritic tree can be constructed, as shown in Fig. 6.

The interaction in the dendritic tree is done by double-clicking on a node. If the action is performed on an unlabeled node, the child clusters will merge, and the dendritic tree will be recreated based on the new entropies (Fig. 7). The action can be reversed by double-clicking on the merged node.

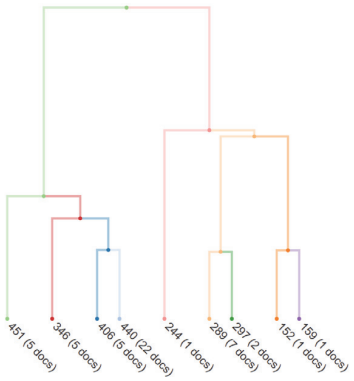

Fig. 6. The dendritic tree from HSTOOL clustering where the entropies between clusters correspond to the distances in the tree. The node labels contain the cluster-ID (which varies between 0 and 499) with the cluster number of documents within parentheses.

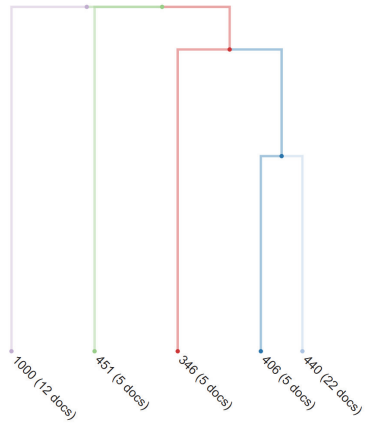

Fig. 7. A dendritic tree where several clusters (with cluster-IDs 244, 289, 297, 152, and 159) from the example in Fig. 5 have been merged into one cluster with 12 documents (cluster-ID 1000).

2) *Interaction with a cluster graph*

A cluster graph enables the visualization of the relationship between all clusters according to $SEnt_{st}$ and makes it possible to merge clusters that are further apart according to the dendritic tree; for example, clusters that belong to different branches. Interaction with the cluster graph is performed either by double-clicking on an arc to merge two clusters or by double-clicking on a node corresponding to two merged clusters to unmerge them. Fig. 8 shows the cluster graphs for the corresponding HSTOOL clustering shown in Fig. 6 and 7.

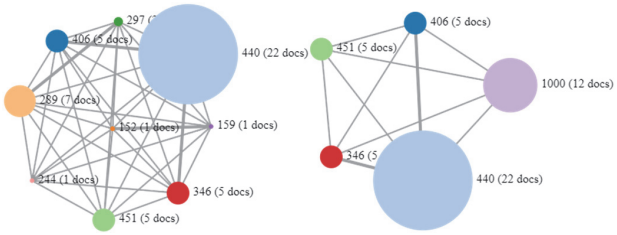

Fig. 8. The figure on the left shows the cluster graph corresponding to the dendritic tree in Fig. 6. The figure on the right shows the cluster graph after merging the same five clusters as in Fig. 7.

IV. Conclusions

In this paper, we develop an approach to managing the large number of clusters resulting from the clustering of articles using a GSDMM algorithm. The method is based on estimating an information-theoretical distance between all possible pairs of clusters. Instead of making a direct comparison based solely on the content of the clusters, we take a reverse approach where we see possible mergers between all pairs of clusters from the perspective of the articles. These articles have a complete probability distribution of affiliation that spans all clusters. When we evaluate a possible merger between two clusters, we compute the effect it has on the probability distributions of affiliation for all articles in the given corpus.

By calculating the entropy of all articles for each possible merger of two clusters, we can estimate how close two clusters are to each other. Merged clusters that result in lower entropies of the probability distributions for all articles are close in an information theory sense.

We conclude that by using the two visualization models known as dendritic trees and cluster graphs, based on the calculated entropy-based information theory distances, an analyst can better manage the number of clusters by selecting proposed cluster pairs to merge in a sequence of decisions.

References

[1] M. Karasalo and J. Schubert, "Developing horizon scanning methods for the discovery of scientific trends," in Proc. 15th Int. Conf. Doc. Anal. Recognit. Piscataway, NJ: IEEE, 2019, pp. 1055–1062. doi:10.1109/ICDAR.2019.00172

[2] Y. Wang, "Improving spectral clustering using spectrum-preserving Node reduction," arXiv preprint arXiv:2110.12328. doi:10.48550/arXiv.2110.12328

[3] Y. Wang, Z. Zhao and Z. Feng, "Scalable graph topology learning via spectral densification," in Proc. 15th ACM Int. Conf. Web Search and Data Min., February 2022, Pages 1099–1108. doi:10.1145/3488560.3498480

[4] S. Sulova, L. Todoranova, B. Penchev, and R. Nacheva, "Using text mining to classify research papers," in Int. Multidiscip. Sci. GeoConference Surv. Geol. Min. Ecol. Manag. Sofia: SGEM, vol. 17, 2017, pp. 647–654. doi:10.5593/sgem2017/21/S07.083

[5] V. Gupta and G. S. Lehal, "A survey of text mining techniques and applications," J. Emerg. Technol. Web Intell., vol. 1, pp. 60–76, August 2009.

[6] J. Yin and J. Wang, "A Dirichlet multinomial mixture model-based approach for short text clustering," in Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min. New York: ACM, 2014, pp. 233–242. doi:10.1145/2623330.2623715

[7] K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using EM," Mach. Learn., vol. 39, pp. 103–134, May 2000. doi:10.1023/A:1007692713085

[8] S. P. Stuart, "Least squares quantization in PCM," IEEE Trans. Inf. Theory, vol. 28, pp. 129–137, March 1982. doi:10.1109/TIT.1982.1056489

[9] C. E. Shannon, "A mathematical theory of communication," The Bell Syst. Tech. J., vol. 27, pp. 379–423, 623–656, July–October 1948. doi:10.1002/j.1538-7305.1948.tb01338.x

[10] J. Schubert, "Constructing and evaluating alternative frames of discernment," Int. J. Approx. Reason., vol. 53, pp. 176–189, February 2012. doi:10.1016/j.ijar.2011.09.009

[11] J. Schubert, "Constructing multiple frames of discernment for multiple subproblem," in Inf. Process. Manag. Uncertain. Knowl.-Based Syst. Theory and Methods. Berlin: Springer (CCIS 80), 2010, pp. 189−198. doi:10.1007/978-3-642-14055-6_20

[12] S. Ahlberg, P. Hörling, K. Johansson, K. Jöred, H. Kjellström, C. Mårtenson, G. Neider, J. Schubert, P. Svenson, P. Svensson, and J. Walter, "An information fusion demonstrator for tactical intelligence processing in network-based defense," Inf. Fusion, vol. 8, pp. 84–107, January 2007. doi:10.1016/j.inffus.2005.11.002

[13] O. E. Livne and A. E. Brandt, "Lean algebraic multigrid (LAMG): Fast graph Laplacian linear solver," SIAM J. Sci. Comput., vol. 34, pp. B499–B522, August 2012. doi:10.1137/110843563

[14] J. Shi and J. Malik, "Normalized cuts and image segmentation" in IEEE Trans. Pattern Anal. Mach. Intell., vol. 22, pp. 888–905, August 2000. doi:10.1109/34.868688