Semi-supervised Document Clustering via Loci

Taufik Sutanto[1,2](✉) and Richi Nayak[1]

[1] Queensland University of Technology (QUT), Brisbane, Australia
[2] Syarif Hidayatullah State Islamic University, Jakarta, Indonesia
{taufik.sutanto,r.nayak}@qut.edu.au, taufik.sutanto@uinjkt.ac.id

Abstract. Document clustering is one of the prominent methods for mining important information from the vast amount of data available on the web. However, document clustering generally suffers from the *curse of dimensionality*. Providentially in high dimensional space, data points tend to be more concentrated in some areas of clusters. We take advantage of this phenomenon by introducing a novel concept of dynamic cluster representation named as *loci*. Clusters' loci are efficiently calculated using documents' ranking scores generated from a search engine. We propose a fast loci-based semi-supervised document clustering algorithm that uses clusters' loci instead of conventional centroids for assigning documents to clusters. Empirical analysis on real-world datasets shows that the proposed method produces cluster solutions with promising quality and is substantially faster than several benchmarked centroid-based semi-supervised document clustering methods.

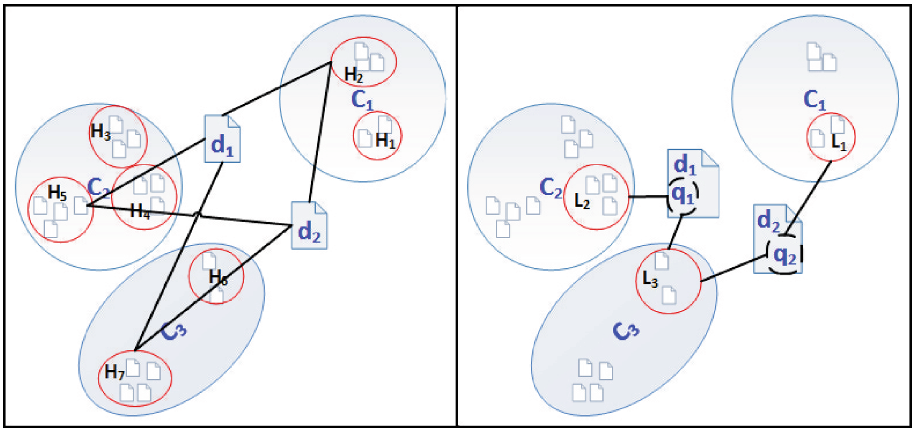
Keywords: Loci · Ranking · Semi-supervised clustering

1 Introduction

In the large document corpora, it is common to find some documents with label or grouping information. Generalizing this valuable prior information to the larger part of the unlabeled documents is relevant to many real-world applications including web information system [7]. Semi-supervised clustering methods have been developed to solve this problem and have been reported to produce a better quality solution than the unsupervised methods [2,7]. Partitional centroid-based semi-supervised clustering algorithms such as seeded k-means or constrained k-means [2] are usually preferred due to their fast performance on large datasets. Nevertheless, these methods do not scale well for the problems with a large number of groupings [7].

Meanwhile, recent clustering studies on high dimensional data have shown the superiority of using the clusters' hub information instead of the conventional centroids for grouping the documents [3,8]. Hubness in clusters (Fig. 1a) is defined as the likelihood of data points in high dimensional data to occur more frequently in k-nearest neighborhood (k-NN) rather than occurring near the centroid point [8]. The behavior of these data points (i.e. hubs) is shown to be a property of high dimensional data and not merely a samples limitation [5]. Unfortunately, the hubness-based clustering algorithms have been limited to apply to a corpus with thousands of documents only [3,8]. The need of calculating the similarity between a document and at least the top-n hub points instead of a single centroid point, representing a cluster, makes the process cumbersome.

In this paper, we propose a novel concept of cluster *locus* and apply it in semi-supervised document clustering setting. The locus of a cluster is a hub-like neighborhood of most *relevant* documents to a target document (e.g. d_1 or d_2 in Fig. 1). In other words, the locus of a cluster is a dynamic cluster representation for a document. A locus can be viewed as a low-dimensional projection of a hub with regards to a document. The relevant documents can be efficiently chosen using the ranking scores generated by a scalable Information Retrieval (IR) system (e.g. search engine).



(a) K-hubs [8] (b) LSDC - the proposed method

Fig. 1. The difference between hubs (a) and loci-based clustering (b).

Our previous clustering method, CICR [7], also uses the ranking scores produced by a search engine to improve the performance when the number of clusters in the dataset is large. However, there are substantial differences between CICR [7] and the proposed method LSDC. Not only does CICR use conventional centroids, it needs to update the cluster information in the search engine (*realtime index*) [7]. On the other hand, LSDC can work as a *plugin/add-on* to the existing IR system without any alteration to the IR system. Consequently, LSDC is more applicable at a production level.

The LSDC method has some important features that make it suitable for large datasets with a large number of clusters. It does not need centroid initialization or updates. LSDC does not need the input parameter k (i.e. the total number of clusters) like most other clustering algorithms. The total number of clusters that fit the dataset is automatically calculated by the algorithm. Moreover, as shown in Fig. 1b, in order to group a target document, the comparison is done with some clusters only and with some documents only within those clusters. This selective comparison reduces the need to scan all the documents (or clusters) in the corpus. As shown in our experiments in the latter section, a small size of relevant documents is enough to accurately cluster the documents. LSDC is able to generate new clustering labels that are not present in the initial labeled data (i.e.seed). Finally, a locus is not necessarily at the center of a cluster, hence LSDC does not have the tendency to form (hyper)sphere-like clusters.

Several real-world document datasets, exhibiting a large number of clusters, were used in the experiments. A comparison of LSDC with centroid-based semi-supervised clustering methods that utilize initial labels (seed) is reported. Empirical analysis reveals that LSDC is not only able to cluster data with a large number of clusters, but also produces fast and accurate clustering solution.

2 Loci Based Semi-supervised Document Clustering

Let $D = \{d_1, d_2, d_3, \ldots, d_N\}$ be the set of all N documents in a dataset. We denote $D^\ell \subset D$ as a set of labeled documents (i.e. seed) and $D^\mu \subset D$ as a set of documents that needs be clustered (i.e. target documents). Note that $D^\mu \cap D^\ell = \emptyset$ and $D^\mu \cup D^\ell = D$. Since LSDC is a *hard clustering* method, every document in D^μ will be mapped to a single cluster.

Using a search engine (e.g. *Sphinx*[1]), it is possible to get a set of ordered terms based on their frequency of occurrence in D. Let $F^t = \{(t_i, f_{t_i}) : t_i \in d, d \in D\}$ be a set of terms and their frequencies in the whole corpus imported from a search engine where $f_{t_i} \leq f_{t_j}$ for $i \leq j$. Given a document $d \in D^\mu$, *LSDC* extracts a set of s distinct terms from d to form a query q based on their frequency information in F^t. In this paper, the s terms in q are chosen from d as terms with s lowest frequencies and it occur more than once in the whole corpus.

A set of at most m relevant documents to the query q and its ranking score vector $\boldsymbol{r}$ is then generated by a search engine. Clusters' loci are calculated from the prior grouping information found in the relevant documents. Finally, similarities between d and the clusters' loci are then used to determine which cluster d should be grouped into.

2.1 Document Ranking Schemes

Let $q = \{t_1, t_2, \ldots, t_s\} \subseteq d$ be the query generated from d. Using a search engine, a set of m most relevant documents to q is identified and its ranking scores vector $\boldsymbol{r}$ is generated. A ranking function R_f employed in a search engine calculates the ranking scores r of m documents for query q generated from d as:

$$R_f : q \rightarrow D^q = \{(d_j^q, r_j) : j = 1, 2, \ldots, m'\}, \tag{1}$$

where $0 \leq m' \leq m$. If $m' = 0$, it indicates that there is no relevant document found in D for the given query q. In LSDC, this will trigger a new cluster

[1] http://sphinxsearch.com.

formation and enables LSDC to cluster the documents in D^{μ} to clusters beyond the existing initial prior information in D^{ℓ}.

There is a number of functions that can be used to calculate ranking score of a document with regards to a query. However, we have shown previously that the *Sphinx* search engine's specific ranking function called *SPH04* results in superior clustering quality compared to other ranking schemes such as weighted *tf-idf*, *BM25*, and *BM25* with proximity [7]. Therefore, we have used *SPH04* in all of our experiments. The ranking score *SPH04* of $d \in D$ given a query q of length s is defined in [1] as:

$$R_f^{SPH04}(q) = 1000 f_w(q,d) + \lceil 999 R_f^{BM^*}(q,d) \rfloor, \tag{2}$$

with

$$f_w(q,d) = 4\max\{LCS(q,d)\}(1+U_w) + \begin{cases} 3, \text{exact query match} \\ 2, \text{first query term match} \end{cases} \tag{3}$$

and

$$R_f^{BM^*}(q,d) = \frac{1}{2} + \frac{\sum_{i=1}^{s}(\frac{t_f * idf}{t_f + 1.2})}{2s}, \text{ where } idf = \frac{log(\frac{N - n_t + 1}{N})}{1+N}. \tag{4}$$

$LCS(q,d)$, the Longest Common Sub-sequence, is defined as the number of keywords that are present in d in the exact same order as in q. U_w is a user defined constant, t_f is a term frequency, the number of terms occurring inside document d, and n_t is the number of documents in D that have the term t.

2.2 Loci Based Document Clustering

Let D^q be the set of documents returned by a search engine in response to the query q. We obtain a set of relevant and labeled documents as $D_\ell^q = D^\ell \cap D^q$. Let C_ℓ^q be the set of distinct cluster label information within D_ℓ^q. For each cluster found in C_ℓ^q and assuming $D_\ell^q \neq \emptyset$, clusters' loci L_k of q are calculated as follows:

$$L_k = \{d : d \in D_k^q, k \in C_\ell^q\}. \tag{5}$$

Using the formulation given in (5), each relevant cluster in C_ℓ^q has only one cluster locus (shown in Fig. 1b). If $D_\ell^q = \emptyset$ then a new cluster is formed with d as its member. We would like to emphasize that in (5) the role of document ranking scores in calculating L_k is implicit. This would result in not only more efficient computation but also less communication cost.

Once the clusters' loci are formed, similarity values between d and each locus in L_k are calculated to decide the document's final clustering decision. Let ϕ denotes a similarity function, then the cluster decision is based on the solution of the following constrained optimization:

$$\begin{array}{ll} \max\limits_{k} & \{\phi(d, L_k) : k \in C_\ell^q\} \\ \text{subject to} & \phi(d, L_k) \geq \rho \end{array}, \tag{6}$$

```
input  : A set of indexed documents $D$, labeled documents $D^\ell$, distinct initial
         cluster labels $C^\ell$ extracted from $D^\ell$, documents that are going to be
         clustered $D^\mu$, a similarity threshold $\rho$, and the maximum number of
         relevant documents $m$.
output: Disjoint partitions of $D^\mu$.
for each $d \in D^\mu$ do
    Extract $q$ from $d$;
    $D^q = R_f(q) = \{(d_j^q, r_j) : j = 1, 2, \ldots, m\}$;          // Search engine's query
    $D_\ell^q = D^q \cap D^\ell$;       // Consider only labeled and relevant documents
    if $D_\ell^q = \emptyset$ then
        $C^\ell = C^\ell \cup \{d\}$;                                 // Form a new cluster
    else
        $L_k = \{d : d \in D_k^q, k \in C_\ell^q\}$;
        $\phi^* = \max_k \{\phi(d, L_k) : k \in C_\ell^q\}$;
        if $\phi^* \geq \rho$ then
            $c_k = c_k \cup \{d\}$;                                   // Assign $d$ to $c_k$
        else
            $C^\ell = C^\ell \cup \{d\}$;                             // Form a new cluster
        end
    end
    $D^\ell = D^\ell \cup d$;
end
```

Algorithm 1. LSDC Algorithm.

where ρ is a user defined threshold to determine whether the similarity value between a document and a locus is considered significant. The set of labeled documents D^ℓ is then updated using the optimal decision (i.e. c_k). When the optimization in (6) does not have a feasible solution, then d will form a new unit cluster. The process is incrementally done for all of the documents in D^μ. Algorithm 1 details the overall process.

3 Experiments

We used four openly available datasets: Reuters 21578[2], MediaEval Social Event Detection (SED) 2013, SED 2014 development data[3], and Wikipedia (September 2014 dumps). The Reuters data has five different categories: topics, exchanges, organizations, people, and places. Since LSDC is a *hard clustering* method, only Reuter documents that have single topic category were used. Similarly, single category documents from Wikipedia dumps were used. Short length documents (usually pages referring to image files) were filtered. All of the text information from SED 2013 and 2014 metadata were concatenated and used (except *URL*). The datasets summary is given in Table 1.

[2] http://www.daviddlewis.com/resources/testcollections/reuters21578/.
[3] http://www.multimediaeval.org/.

Table 1. Summary of datasets in the experiments.

Dataset	#Docs	K	#terms	Raw size (MB)
Reuters	9446	66	28,614	5.4
SED'13	437,370	16,711	189,164	179
SED'14	362,578	17,834	158,272	105
Wikipedia	701,141	59,600	4,553,408	2,074

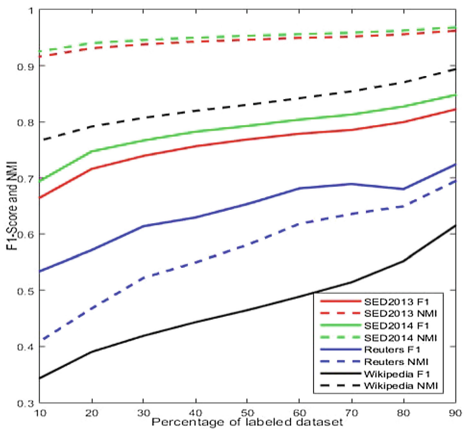
3.1 Pre-processing and Evaluation Criteria

Standard text pre-processing such as English stopwords and non-alpha-numeric characters filtering were applied. The document length normalized *tf-idf* (term frequency-inverse document frequency) weighting [6] was used to represent the document vectors. In all of the experiments the query length s was set to 30 and $\rho = 0$. The clustering evaluation metrics used were pairwise F1-score and Normalized Mutual Information (NMI) [4]. Running time was calculated for all of the clustering methods without the pre-processing steps. Included in the time measurements were initialization (in centroid based methods), communication cost (CICR and LSDC), and updating of the cluster information. Under the distributed computing environment, the time was measured using the real *CPU* time instead of wall time. Experiments were done using Matlab in a local area network connected machine (1Gbps) and no parallel processing has been utilized in all of the clustering algorithms.

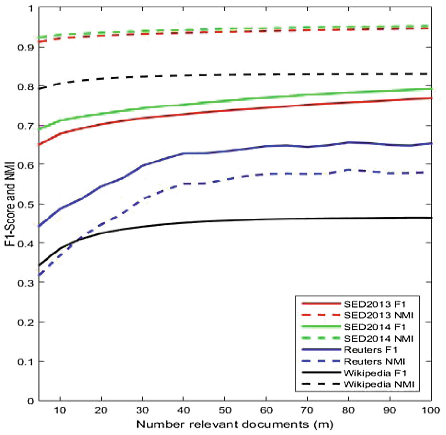
3.2 Results and Discussion

Use of Label Information: *LSDC* as a semi-supervised clustering method depends on the availability of the prior information in the form of grouping information (labels) contained in D^ℓ. We evaluated *LSDC* with representing D^ℓ of different sizes from the data corpus D in order to analyze the impact of labeled data to the clustering quality. As shown in Fig. 2a, the *LSDC* performance is consistent with the increasing proportion of the supervision provided. Most importantly, *LSDC* does not require a large portion of labeled data to give a relatively acceptable clustering quality. With the exception of Reuters (i.e. small dataset), the NMI scores are all above 0.75 by using merely 10 % of D.

Loci Size: Across all four used datasets running time, NMI, and F1-score were recorded for all clustering solutions generated with varying values of m (i.e. the number of relevant documents returned from a search engine to be used in loci calculation). Figure 2b shows that as the size of m increases, the clustering quality is improves. However, the increment is getting smaller and reaches a plateau at around $m = 50$. This indicates that a relatively small value of m is generally enough for *LSDC* to produce *near-optimal* clustering quality.



(a) Varying proportions of D^ℓ.



(b) Different loci sizes.

Fig. 2. The effect of different sizes of D^ℓ and m to LSDC clustering results.

Table 2. Performance comparison on different datasets, methods, and metrics.

	Time (hours:minutes)				NMI				F1-score			
	LS	CR	CK	SK	LS	CR	CK	SK	LS	CR	CK	SK
Reuters	00:01	00:02	00:01	00:04	.660	.714	.703	.676	.721	.747	.699	.657
SED 2013	00:18	29:41	182:56	>240:00	.946	.935	.965	.964	.768	.725	.838	.831
SED 2014	00:29	22:40	117:31	>240:00	.953	.946	.979	.978	.792	.767	.905	.899
Wikipedia	00:40	>240:00	>240:00	>240:00	.830	–	–	–	.464	–	–	–
Method: LSDC (LS), CICR (CR), constrained (CK) and seeded (SK) k-means.												

Benchmarks: Comparisons were made with the centroid-based semi-supervised document clustering methods such as *CICR* [7], seeded k-means [2], and constrained k-means [2]. All documents in D^ℓ are used for centroid initialization in constrained k-means and seeded k-means. However, in constrained k-means only documents in D^μ are used to produce final cluster results while seeded k-means uses all of the documents in D. D is randomly split into two equal size of D^ℓ and D^μ. The evaluation on all methods are done using documents in D^μ against the available gold standard. We add an additional time stopper (240 h) as a time threshold.

Table 2 shows that overall LSDC is faster while producing comparable clustering quality. It appears that the discrepancy of speed between LSDC and the other methods is higher as the datasets get larger. This indicates that LSDC is more suitable for large size datasets. All other methods reach the time limit without producing any results for the large dataset (Wikipedia) while *LSDC* took only 40 min to finish. Seeded k-means even reached the time limit for medium size datasets (SED 2013 and SED 2014). The F-score and NMI values show that *LSDC* gives high-quality clustering solutions. In fact, *LSDC* gives marginally better results than *CICR* and almost similar results to constrained k-means and seeded k-means. We also compared the performance with the Euclidean distance measure, these clustering solutions consumed more time and yielded lower accuracy in comparison to the cosine similarity measure.

Further analysis on clustering quality: Although not shown in this paper (due to the space constraint), empirical analysis shows that longer queries do not necessarily produce better clustering quality. We also found that LSDC's clustering quality can be further improved by using the available structure information in the documents (e.g. *title* and *tags*) to build a customized ranking scheme.

4 Conclusions and Future Work

A novel concept of *loci* is introduced in this paper to dynamically represent document clusters. We proposed and evaluated a fast incremental loci based document clustering method (LSDC) on several real-world datasets. By extending a search engine capability to process a large set of documents and provide relevant documents to a query, LSDC is capable of clustering a large set of documents substantially faster than the benchmarked algorithms while retaining a comparable clustering quality. Analysis on the quality of prior information to the LSDC clustering quality and a possible extension of loci concept in classification problems with a large number of categories are subjects of our future investigations.

References

1. Aksyonoff, A.: Introduction to Search with Sphinx: From Installation to Relevance Tuning. O'Reilly, Sebastopol (2011)
2. Basu, S., Banerjee, A., Mooney, R.J.: Semi-supervised clustering by seeding. In: Proceedings of the Nineteenth International Conference on Machine Learning. ICML 2002, San Francisco, CA, USA, pp. 27–34 (2002)
3. Hou, J., Nayak, R.: The heterogeneous cluster ensemble method using hubness for clustering text documents. In: Lin, X., Manolopoulos, Y., Srivastava, D., Huang, G. (eds.) WISE 2013, Part I. LNCS, vol. 8180, pp. 102–110. Springer, Heidelberg (2013)
4. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval, vol. 1. Cambridge University Press, Cambridge (2008)
5. Radovanović, M., Nanopoulos, A., Ivanović, M.: Hubs in space: popular nearest neighbors in high-dimensional data. J. Mach. Learn. Res. **11**, 2487–2531 (2010)
6. Singhal, A., Buckley, C., Mitra, M.: Pivoted document length normalization. In: Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 1996, New York, NY, USA, pp. 21–29 (1996)
7. Sutanto, T., Nayak, R.: The ranking based constrained document clustering method and its application to social event detection. In: Bhowmick, S.S., Dyreson, C.E., Jensen, C.S., Lee, M.L., Muliantara, A., Thalheim, B. (eds.) DASFAA 2014, Part II. LNCS, vol. 8422, pp. 47–60. Springer, Heidelberg (2014)
8. Tomašev, N., Radovanović, M., Mladenić, D., Ivanović, M.: The role of hubness in clustering high-dimensional data. In: Huang, J.Z., Cao, L., Srivastava, J. (eds.) PAKDD 2011, Part I. LNCS, vol. 6634, pp. 183–195. Springer, Heidelberg (2011)