Equidistant Nodes Clustering: a Soft Clustering Algorithm Applied for Synset Induction

© Mikhail Chernoskutov
Ural Federal University,
Krasovskii Institute of Mathematics and Mechanics,
Yekaterinburg, Russia
mach@imm.uran.ru

© Dmitry Ustalov
Data and Web Science Group
University of Mannheim,
Mannheim, Germany
dmitry@informatik.uni-mannheim.de

Abstract. In this paper, we present an algorithm for fuzzy graph clustering called Equidistant Nodes Clustering (ENC). In the core of the algorithm there is an assumption that the nodes in a community can reach each other within a limited number of steps. We describe the algorithm and apply it to a natural language processing task of synset induction. The experimental results show that ENC outperforms popular clustering algorithms as according to a gold standard evaluation.

Keywords: graph clustering, community detection, natural language processing, synsets.

1 Introduction

The problem of graph clustering arises from social network analysis, natural language processing, bioinformatics, etc. Given an undirected graph $G = (V, E)$, a clustering algorithm outputs a covering for the set of nodes V that *reasonably* groups the nodes into *communities* or *clusters*.

A vast diversity of the application-specific requirements for the cluster structure urges the development of scalable clustering algorithms for extremely large and complex graphs. In particular, we focus on the soft clustering of graph in which the clusters may overlap. This is very useful for handling such phenomena as protein-protein interactions [9], lexical ambiguity [10], etc. The contribution of this paper is ENC, a new graph-based algorithm for soft clustering. Also, we evaluate ENC on a synset induction. Our gold standard-based experiments show that ENC outperforms or performs on par with popular graph clustering algorithms. In some papers, *soft clustering* is also referred to as *fuzzy clustering*; these terms are synonyms.

The rest of the paper is organized as follows. Section 2 surveys the related work. Section 3 describes ENC, a soft graph clustering algorithm that uses the equidistant node finding approach. Section 4 shows the experimental setup and compares ENC to other clustering algorithms on the synset induction task. Section 5 concludes the paper with the final remarks.

2 Related Work

Perhaps, the most well-known soft graph clustering algorithm is the clique percolation method (CPM) [9]. CPM finds communities (or clusters) with the biggest sizes which consist of fixed-sized adjacent cliques. In terms of CPM, two cliques are considered adjacent if they share $(n-1)$ nodes, where n is number of nodes in the clique. This algorithm is well-known in bioinformatics and social network analysis; for most real-world tasks the value of n is usually 2, 3 or 4.

MaxMax [6] is a non-parametric graph clustering algorithm that outputs overlapping clusters designed especially for word sense induction problems. This algorithm constructs an intermediate representation of the input undirected graph as a directed graph according to the maximal affinity of the adjacent nodes. Then, it extracts quasi-strongly connected subgraphs from the intermediate graph.

WATSET is a state-of-the-art overlapping community detection meta-algorithm [10]. Like MaxMax, it is designed especially for addressing the word sense induction task. WATSET internally uses non-overlapping community detection algorithms like Markov clustering (MCL) [4] and Chinese Whispers (CW) [1] by building and clustering an intermediate sparser undirected graph representation.

3 Equidistant Node Finding for Fuzzy Graph Clustering

In the core of the proposed algorithm, Equidistant Nodes Clustering (ENC), there is an assumption that the nodes inside clusters have stronger connectivity between each other than outside the clusters. As opposed to Blondel et al. [2] who treat connectivity of nodes in cluster as the ratio between number of edges falling inside the cluster and number of edges that crossing the cluster border, we consider connectivity in close connection with the length of the shortest path between the nodes in a cluster (the shorter the path, the stronger the connectivity between nodes). Therefore, clusters of nodes can be seen as *k-cliques* (a clique in which the distance between every pair of nodes does not exceed k hops). Such type of cluster structure is chosen to distinguish groups of nodes with short pairwise path length from all other nodes in networks. Hence, we put forward the following *assumption*: a subset of nodes is a cluster if every pair of nodes is reachable within a limited

Proceedings of the XX International Conference "Data Analytics and Management in Data Intensive Domains" (DAMDID/RCDL'2018), Moscow, Russia, October 9-12, 2018

CEUR-WS.org/Vol-2277/paper13.pdf

predefined number of steps.

The idea of ENC can be illustrated using the word synonymy data from the English Wiktionary on Fig. 1. A synonymy graph is an undirected graph which nodes are lexical units (words); a pair of nodes is connected by an edge *iff* these words are synonyms. Due to the data sparsity, some edges are missing, e.g., an edge between the words "make" and "build", or an edge between the words "produce" and "assemble". However, sets of words from Fig. 1 can be treated in terms of k-cliques with $k = 2$. It means that each word pair in this set is connected by one edge {make, construct} or two edges {produce, engender}, {engender, assemble}.

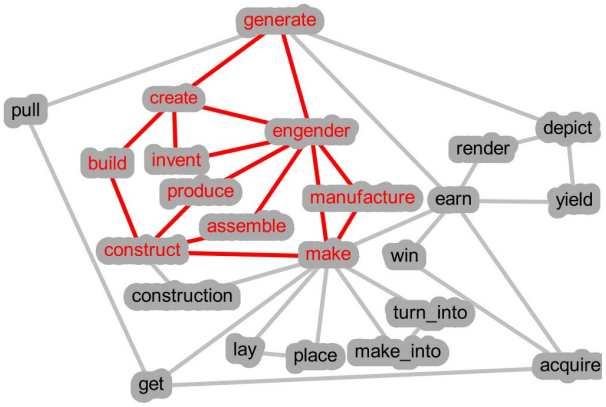

Figure 1. An example of graph with a cluster of synonyms (synset)

In this formulation, cluster overlap is possible due to node sharing between several equidistant node sets (Fig. 2). In this case, clusters of words indicated by nodes incident to edges with same color. As seen, the word "break" falls into three different clusters simultaneously: two of them are cliques and one is k-clique with $k = 2$. We design our algorithm, ENC, in order to detect such fuzzy k-cliques. Considering the task of synset induction, we assume that $k = 2$.

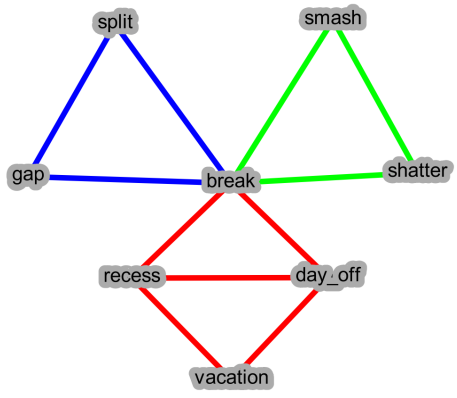

Figure 2 Example of fuzzy clusters

3.1 Overview of the ENC Algorithm

The ENC algorithm has four steps. At the first step, we extract all the regions of interest from the graph, i.e., two-hops neighborhood for each node in the graph. Such a neighborhood size is chosen to consider all possible k-cliques involving corresponding node. This is crucial for the reduction the problem size by not considering the non-informative parts of the graph (outside of two-hops neighborhood for corresponding node), which is useful in processing very large graphs. Aforementioned regions of interest will be used in subsequent steps to extract k-cliques.

At the second step, we extract all the k-cliques from each ego network (i.e., node and all its two hops neighborhood). In order to do this, it is possible to reduce task of finding k-cliques to the well-known task of finding simple cliques [3] by topology modification of each initial ego network. This topology modification implies reducing all two-hops paths to single hop paths by applying transitive closure procedure. After that, cliques that we have in transitive closure-based graph is at the same time k-cliques in corresponding ego network. At this point, we have a large number of k-cliques with small structural differences in the adjacent ego networks since these ego networks can share most of its nodes.

At the third step, we rank all the k-cliques by mapping them to a real number. Given a k-clique C and a ranking function in general form as $f : \mathrm{C} \to \mathbb{R}$, we denote as $\alpha \in \mathbb{R}$ the score of the k-clique. In particular, f assesses whether the k-clique looks like synset or not, and its output α number is "degree" of such similarity. There are a number of ways to determine f depending on what type of clusters the algorithm should detect. Considering the task of synset induction, we chose the following options for determining f function (the list is not limited by the following functions and may be augmented in further research):

- Average edge weight in k-clique (in this case k-cliques with higher values of average edge weights are more likely to be synsets). According to the synset induction task, edge weight corresponds to the strength of connection between different nodes. Hence, the higher edge weight is, the more similar two nodes are. We refer to this implementation of function f as *rank_w* in the results section.
- Transitivity of k-clique (relative number of triangles in k-clique). It shows how dense is corresponding k-clique. The denser graph is, the higher values of transitivity it has. We refer to this function as *rank_t*.
- Density of k-clique (ratio of the number of edges in k-clique to total number of possible edges). Actually, it is another way to assess how dense graph is. We refer to this function as *rank_d*.
- Also we test some combinations of aforementioned metrics like multiplication of average edge weight to transitivity coefficient (*rank_wt*) and multiplication of average edge weight to graph density (*rank_wd*). The reason of considering these combinations is because we believe that k-cliques both with higher values of edge weights and dense inner structure have much more probability to be "successful" synset candidates.

Last thing on this step is sorting all k-cliques by its α value in decreasing order, thus, obtaining list of k-cliques sorted by its likelihood to be a proper community.

At the fourth step, we successively map the k-cliques from the sorted list to the topology of the input graph according to its α value. It means that we take the k-clique with highest α value from the list. Then, we mark all the nodes in the input graph that correspond nodes in selected k-cliques with id of that clique. Then we take next k-clique (according to its α value) and mark corresponding nodes in the input graph again. At some point, there may be a situation when nodes in selected k-clique has already been marked with another clique ids in one of the previous steps (i.e., different k-cliques are overlapped). It is important to consider the overlapping degree (number of nodes in k-clique that marked by other k-cliques) of each k-clique with other already mapped k-cliques, since it may affect the correctness of the algorithm results. For instance, if k-clique candidate overlaps with 50 % of all its nodes with another k-clique that has already been mapped to input graph then it is highly likely to be excluded (depending of the k-clique size). The reason is because different synsets tend to overlap by only small part of its nodes (like in example depicted on fig. 2).

After mapping all k-cliques from the sorted list to the topology of initial graph we treat the successful ones as the output set of soft clusters (so called synsets in the synset induction task).

3.2 Pseudocode

The pseudocode for the proposed overlapping community detection algorithm, ENC, is presented below.

```
Input: an undirected graph G
Output: a set of clusters clusters
1   for each v in G do
2       Gs ← ExtractSubgraph(G,v,2)
3       T ← TransitiveClosure(Gs,2)
4       cliques_local ← {K:Ci∈T,i∈N}
5       cliques_global ← cliques_global ∪
                         cliques_local
6   end for
7   cliques_ranked ← {(αi,Ci):α∈R, 0≤αi≤1,
                   αi ← GetRank(Ci), i∈N,
                  1≤i≤len(cliques_global)}
8   cliques_sorted ←
              SortCliques(cliques_ranked)
9   for each v in G do
10      v.β ← 0
11  end for
12  for i = 0 to |cliques_sorted| do
13      (αi,Ci) ← cliques_sorted[i]
14      if CheckOverlap(G,Ci) is true then
15      clusters ← clusters ∪
            {vj:vj∈Ci.V, j∈N, 1≤j≤|Ci.V|}
16          for each v in Ci do
17              v.β ← v.β + 1
18          end for
19      end if
20  end for
```

ENC starts from the procedure of determining k-cliques in the neighborhood of each node. Line 2 describes extraction of a subgraph of G with the center in node v and its two hops neighborhood. Line 3 describes the transitive closure for the nodes having the path length less or equal than two steps. In line 4, simple cliques are extracted from the transitive closure graph. All the nodes forming cliques are collected in the shared list in line 5. In lines 1–6 we extract all possible k-cliques candidates.

The second step ranks all the k-cliques formed after transitive closures. In line 7 each k-clique is initialized with some real number $0 \leq \alpha \leq 1$ using the GetRank function. GetRank function represent one of the procedures described in Section 3.1 that takes k-clique as input and returns its corresponding α-value. It may be, for instance, density or transitivity of considered k-clique. Then, after forming the list consisting of all possible k-cliques, we sort it by the descending order of α in line 8 using SortCliques function.

Lines 9–11 initialize the k-cliques counter for each involved node. In real-world graphs, a node does not participate in a very big number of k-cliques, so we aim at preventing this behavior with counting them.

Finally, in the last loop we determine all the communities by choosing the most important k-cliques. For that, we apply the CheckOverlap function in line 14 that determines whether the given k-clique is suitable to be a community, or we should discard it. In terms of synset induction task, we check compliance with the conditions in Section 3.1. It means that we discard k-cliques that overlaps in almost all its nodes, and, vice versa, accept k-cliques that has only small part of its nodes participating in other k-cliques. In case if k-clique is suitable, we put it into the final list of clusters in line 15 and mark its nodes in lines 16–18 to prevent unnecessary overlapping with the k-cliques provided with the lower values of α. Reasoning about choosing appropriate conditions for the CheckOverlap functions are described in Section 3.1.

4 Evaluation

We evaluate ENC by comparing its performance to several state-of-the-art graph clustering algorithms on a synset induction task. A synset induction task is a natural language processing task of clustering a synonymy graph to group the words having the same meaning into sets of synonyms—synsets. In this task, the input is a synonymy graph $G = (V, E)$, where V is the vocabulary and $E \subseteq V^2$ is a reflexive symmetric relation (synonymy) defined on the vocabulary. The performance of the algorithms is compared to the pre-defined gold standard clustering.

We compare ENC to the following algorithms: CPM [9], MaxMax [6] and WATSET [10]. Our evaluation approach is the same as in [6] and [10]. We run clustering algorithms on the same input graph and compare the result to a gold standard clustering. For that, we decompose every synset of n words into a set of $\frac{n(n-1)}{2}$ word pairs and compute pairwise precision, recall and their harmonic mean also known as F_1-score. We used the same input graph as in [10] which has 77 906 nodes, 71 880 edges, and based on the English Wiktionary. The edges were weighted using cosine similarity between the Skip-gram vectors trained on the Google News corpus [7]. As the gold standard, we use the same synsets

from WordNet [5] and BabelNet [8] for the English language as in [10].

4.1 ENC Parameter Tuning

A simple way to improve the community detection quality by ENC is to prune the input graph by deleting edges with small weight. In terms of synset induction, we found that edges with the weight smaller than 0.3 may be useless because it is highly unlikely that the nodes incident to such edges are really synonyms. When this filter is enabled, we denote it as *filt_on*. Otherwise, we denote it as *filt_off*.

Another way to tune the ENC algorithm is to normalize k-clique ranks according to their sizes. Normalization should help to keep balance between k-clique size and density of edges inside it when computing the k-clique rank. When this filter is enabled, we refer to it as *norm_on*. Otherwise, we denote it as *norm_off*.

Also, we selected the following conditions for k-clique overlapping (where n is number of nodes in k-clique and m is number of overlapped nodes, which participate in other k-cliques as well, see Section 3.1 for details):

- $\begin{cases} m = 1,\ 2 \le n \le 3 \\ m = 2,\ 4 \le n \le 6 \\ m = 3,\ 7 \le n \le 10 \\ m = 4,\ 11 \le n \le 15 \\ m = 5,\ n \ge 16 \end{cases}$, we refer to these conditions as *cond_1* in results section.
- $\begin{cases} m = 1,\ 2 \le n \le 3 \\ m = 2,\ 4 \le n \le 6 \\ m = 3,\ 7 \le n \le 9 \\ m = 4,\ 10 \le n \le 12 \\ m = 5,\ n \ge 13 \end{cases}$, as *cond_2*.
- $\begin{cases} m = 1,\ 2 \le n \le 3 \\ m = 2,\ 4 \le n \le 5 \\ m = 3,\ 6 \le n \le 8 \\ m = 4,\ 9 \le n \le 10 \\ m = 5,\ 11 \le n \le 15 \\ k = 6,\ n \ge 16 \end{cases}$, as *cond_3*.
- $\begin{cases} m = 1,\ \mathrm{n} = 2 \\ m = 2,\ 3 \le n \le 5 \\ m = 3,\ 6 \le n \le 7 \\ m = 4,\ 8 \le n \le 9 \\ m = 5,\ 10 \le n \le 14 \\ k = 6,\ n \ge 15 \end{cases}$, as *cond_4*.

Aforementioned conditions are selected only to agree with task of synset induction (i.e., the assumption that different synset can share only small part of its nodes). These conditions may be the subject of tuning in the further research.

We conclude this subsection with summary table for all tunable parameters in proposed algorithm and in reference algorithms. Table 1 shows the combination of the parameters used in our study.

Table 1 Parameter tuning; the best-found combination is highlighted (for WATSET and CPM)

Method	Parameters
ENC	Edge filtering ∈ {filt_on, filt_off}
	k-clique ranking function f ∈ {rank_w, rank_t, rank_d, rank_wt, rank_wd}
	k-clique rank normalization ∈ {norm_on, norm_off}
	Overlapping condition ∈ {cond_1, cond_2, cond_3, cond_4}
WATSET	$\text{Cluster}_{\text{Local}}$ ∈ {**MCL**, CW_{top}, **CW_{log}**, CW_{nolog}}
	$\text{Cluster}_{\text{Global}}$ ∈ {**MCL**, CW_{top}, CW_{log}, CW_{nolog}}
CPM	k ∈ {**2**, 3, 4}

4.2 Results

We carried out 100 experiments for different configurations of the evaluated algorithms. The results obtained on WordNet are present in Table 2; the results obtained on BabelNet are present in Table 3. The values of precision, recall and F_1-score are provided. For brevity reasons, each table contains results only for ten experiments. Also, only best achieved results of WATSET and CPM are provided. All rows in tables are sorted in decreasing order according to the value of F_1.

Table 2 Results for WordNet

№	Method	Pr	Re	F_1
1	WATSET [CW_{log}, MCL]	0,398	0,294	**0,338**
…	…	…	…	...
14	ENC [filt_on, rank_d, norm_off, cond_4]	0,429	0,272	0,333
15	ENC [filt_on, rank_wd, norm_off, cond_4]	0,423	0,272	0,331
16	ENC [filt_on, rank_d, norm_on, cond_4]	0,359	0,302	0,328
17	ENC [filt_on, rank_wd, norm_on, cond_4]	0,359	0,302	0,328
18	ENC [filt_on, rank_wd, norm_off, cond_3]	0,424	0,267	0,328
19	ENC [filt_on, rank_w, norm_off, cond_4]	0,350	0,307	0,327
20	ENC [filt_off, rank_wd, norm_on, cond_2]	0,340	0,315	0,327
…	…	…	…	…
97	CPM [k2]	**0,557**	0,154	0,241
98	MaxMax	0,133	**0,367**	0,195

Table 3 Results for BabelNet

№	Method	Pr	Re	F_1
1	ENC [filt_on, rank_d, norm_off, cond_4]	0,492	0,313	**0,382**
2	ENC [filt_on, rank_wd, norm_off, cond_4]	0,489	0,314	0,382
3	ENC [filt_on, rank_wd, norm_off, cond_3]	0,492	0,310	0,380
4	ENC [filt_on, rank_d, norm_off, cond_3]	0,495	0,306	0,379
5	ENC [filt_on, rank_wd, norm_on, cond_4]	0,439	0,332	0,378
6	ENC [filt_on, rank_d, norm_on, cond_4]	0,438	0,332	0,378
7	ENC [filt_on, rank_w, norm_off, cond_4]	0,429	0,336	0,377
8	ENC [filt_off, rank_wd, norm_off, cond_4]	0,442	0,328	0,377
18	WATSET [MCL, MCL]	0,416	0,342	0,375
…	…	…	…	…
93	CPM [k2]	**0,522**	0,245	0,333
…	…	…	…	…
98	MaxMax	0,164	**0,382**	0,229

On WordNet, the ENC algorithm shown lower values of F_1 comparing to WATSET. However, ENC shows high values of precision for the cases of turned off normalization, graph density-based ranking alongside with filtering low-weight edges and softest conditions of *k*-clique overlapping (cliques can overlap by more nodes). Despite the fact that CPM shows the highest precision and MaxMax shows the highest recall, both algorithms demonstrate imbalance between its precision and recall, which resulting in lower values of F_1.

On BabelNet, the ENC algorithm shows the best overall results. The only stable parameter is filtering, which is always turned on for each testing instance. Much softer conditions (*cond_3* and *cond_4*) meet much more often than more stronger conditions (*cond_1* and *cond_2*). As seen from the Table 3, ENC shows nearly the same precision of overlapping community detection with and without normalization as well as for different *k*-cliques ranking functions. The results for CPM and MaxMax on BabelNet demonstrate imbalance between precision and recall, which leads to lower values of F_1, like in the case of WordNet.

As according to the experimental results obtained on WordNet and BabelNet, ENC shows generally better results for synset induction task when algorithm parameters are oriented on detecting clusters with dense edge structure (*rank_d*, *rank_w* and its combination *rank_wd*). The reason for this is a fact that set of words tend to be synset if it has a lot of connections between its words. Also, ENC works well when there is preprocessing in the form of discarding low-weight edges before computation. This helps filter useless connections between clusters thus highlighting clusters inner dense structure. Another important feature for ENC-based synset induction is soft overlapping conditions (*cond_3* or *cond_4*). It allows to have much more variability when forming and mapping *k*-cliques on the input graph.

5 Conclusion

As according to the experimental results, the assumptions in our algorithm, ENC, makes it possible to successfully capture the structure in synonymy graphs for English. ENC showed the performance comparable to the state-of-the-art algorithm, WATSET, although using a different treatment for the overlapping community detection problem. Further studies include scaling of the algorithm for larger graphs and applying ENC for synset induction from other languages.

Acknowledgments. The reported study was funded by RFBR according to the research projects No. 16-37-00203 мол_а and No. 16-37-00354 мол_а. The research was supported by the Ministry of Education and Science of the Russian Federation Agreement no. 02.A03.21.0006. We used the "Uran" supercomputer of the Krasovskii Institute of Mathematics and Mechanics in this study.

References

[1] Biemann, C.: Chinese Whispers: an Efficient Graph Clustering Algorithm and Its Application to Natural Language Processing Problems. In: Proceedings of the First Workshop on Graph Based Methods for Natural Language Processing, pp. 73–80. TextGraphs-1. Association for Computational Linguistics, New York, NY, USA (2006)

[2] Blondel, V.D., Guillaume, J.-L., Lambiotte, R., Lefebvre, E.: Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment. 2008 (10): P10008. doi: 10.1088/1742-5468/2008/10/P10008

[3] Bomze I.M., Budinich M., Pardalos P.M., Pelillo M. (1999) The Maximum Clique Problem. In: Du DZ., Pardalos P.M. (eds) Handbook of Combinatorial Optimization. Springer, Boston, MA, USA. doi: 10.1007/978-1-4757-3023-4_1

[4] van Dongen, S.: Graph Clustering by Flow Simulation. Ph.D. thesis, University of Utrecht (2000)

[5] Fellbaum, C.: WordNet: An Electronic Database. MIT Press, Cambridge (1998)

[6] Hope, D., Keller, B.: MaxMax: A Graph-Based Soft Clustering Algorithm Applied to Word Sense Induction. In: Gelbukh, A. (ed.) CICLing 2013. LNCS, vol. 7816, pp. 368–381. Springer, Heidelberg (2013) doi: 10.1007/978-3-642-37247-6_30

[7] Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing

Systems, vol. 26, pp. 3111–3119. Curran Associates Inc., Harrahs and Harveys, NV, USA (2013)

[8] Navigli, R., Ponzetto, S.P.: BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. Artificial Intelligence 193, 217–250 (2012) doi: 10.1016/j.artint.2012.07.001

[9] Palla, G., Derenyi, I., Farkas, I., Vicsek, T.: Uncovering the overlapping community structure of complex networks in nature and society. Nature. 2005. Vol. 435. P. 814–818

[10] Ustalov, D., Panchenko, A., Biemann, C.: Watset: Automatic Induction of Synsets from a Graph of Synonyms. In: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1579–1590. ACL 2017. Association for Computational Linguistics, Vancouver, BC, Canada, (2017). doi: 10.18653/v1/P17-1145