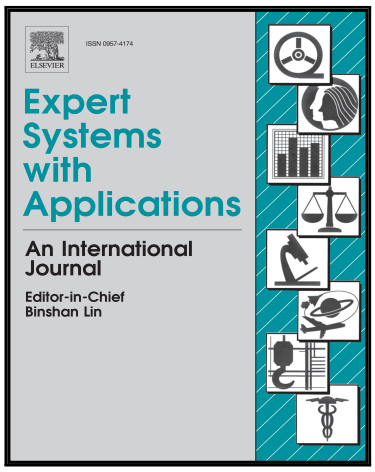


Triadic Co-Clustering of Users, Issues and Sentiments in Political Tweets

Sefa Şahin Koç, Mert Özer, İsmail Hakkı Toroslu, Hasan Davulcu, Jeremy Jordan

PII: S0957-4174(18)30056-3
DOI: 10.1016/j.eswa.2018.01.043
Reference: ESWA 11791

To appear in: *Expert Systems With Applications*

Received date: 7 October 2017
Revised date: 25 January 2018
Accepted date: 26 January 2018

Please cite this article as: Sefa Şahin Koç, Mert Özer, İsmail Hakkı Toroslu, Hasan Davulcu, Jeremy Jordan, Triadic Co-Clustering of Users, Issues and Sentiments in Political Tweets, *Expert Systems With Applications* (2018), doi: 10.1016/j.eswa.2018.01.043

Highlights

- Person-issue-keyword relationships models the sentiment of a person on an issue.
- On this signed 3-partite graph a new strong co-clustering problem is defined.
- A novel algorithm has been proposed and compared with spectral clustering method.
- Experiments are done on real-life data using twits of UK politicians on Brexit.
- Scalability of the new algorithm has been shown through syntactic data.

Triadic Co-Clustering of Users, Issues and Sentiments in Political Tweets

Sefa Şahin Koç[a], Mert Özer[b], İsmail Hakkı Toroslu[a], Hasan Davulcu[b], Jeremy Jordan[c]

[a] *Computer Engineering Department, Middle East Technical University, Ankara, Turkey 06530 ({sefa.koc,toroslu}@metu.edu.tr)*
[b] *School of Computing,Informatics, Decision Systems Eng., Arizona State University, Tempe, AZ 85287 ({mozer,hdavulcu}@asu.edu)*
[c] *Dept. of Mathematics and Statistics, Air Force Institute and Technology, Dayton, OH 45433 (Jeremy.Jordan@afit.edu)*

Abstract

Social network data contains many hidden relationships. The most well known is the communities formed by users. Moreover, typical social network data, such as Twitter, can also be interpreted in terms of three-dimensional relationships; namely the users, issues discussed by the users, and terminology chosen by the users in these discussions. In this paper, we propose a new problem to generate co-clusters in these three dimensions simultaneously. There are three major differences between our problem and the standard co-clustering problem definition: a node can be a member of more than one clusters; all the nodes are not necessarily members of some cluster; and edges are signed and cluster are expected to have high density of positive signed edges, and low density of negative signed edges. We apply our method to the tweets of British politicians just before the Brexit referendum. Our motivation is to discover clusters of politicians, issues and the sentimental words politicians use to express their feelings on these issues in their tweets.

Keywords:
Social Network Analysis, Co-clustering, Hypergraph, 3 partite graph, Sentiment Analysis

1. Introduction

Although social networks have been around for quite some time, Facebook and Twitter have actually made them readily available to mainstream users. These top social network sites are among the most visited websites in the world. This trend introduces new and interesting problems, the most well-known being the determination of communities or user clusters within these social networks.

The community detection problem or user clustering are among the most popular recent research topics, and many different versions of the problem have been studied. However, co-clustering is still relatively new and less explored than other clustering problems. Most of the research is limited to the 2-dimensional version of the co-clustering problem, also known as 2-mode clustering or biclustering, see (Angiulli and Pizzuti, 2005; Dhillon, 2001; Giannakidou et al., 2008; Zha et al., 2001). In this work, we focus on the 3-dimensional co-clustering problem, which is motivated by real-life social network data.

On many social network sites, users share their views on a variety of subjects. This simple data set can naturally be modeled as a 3-dimensional relationship which includes users, topics on which users share their ideas, and the keywords they use to express their feelings on the subjects. Users may be clustered according to the similarity of feelings on the same issues. Also, users with different political views or from different social groups, may prefer to use different keywords with the same negative or positive sentiments in order to express the same kind of feeling on the same issue. Therefore, in clustering the people, it is important to use the feelings of the people on the issues, as well as utilizing keywords they use to express these views.

In this work, we have collected tweets of preselected groups of users on specific issues, and subsequently created a 3-dimensional data model by extracting the sentiment keywords used by these users on the selected issues. More specifically, our selected group of people are politicians from the UK, and the issues are hot subjects discussed just before the "Brexit" referendum. Using this data, we have created a 3-D data set, whose dimensions correspond to the people, the issues and keywords, and the values of each element of this 3-D data. If the person had written a tweet on the issue including that keyword, the values are the sign of the sentiment keyword (as positive or negative), else the value is empty. This 3-D data can be modeled as a tri-partite hypergraph.

The aim of this paper is to propose an effective method which generates tri-clusters from hypergraphs with signed tripartite hyperedges. Our focus data set is signed 3-D relationships obtained from users' feelings, represented by the sentimental keywords (either positive or negative) they use in their tweets on the issues. The preliminary version of this work is presented in an extended abstract in (Dawande et al., 2001), where the sketch of the tri-partite clustering method is given and tested on toy data sets.

One of the most well-known and general purpose co-clustering methods is spectral clustering, which could be applied to our signed 3-D data as well (Gao et al., 2005; Long et al., 2006). However, we have developed a new co-clustering algorithm customized to meet the requirements of our problem for 2 main reasons. These reasons are as follows:

1. Typical co-clustering algorithms, such as spectral clustering algorithms (Ng et al., 2001), include all the nodes of all the dimensions of the graph in the clusters generated for these dimensions. However, some of these nodes might have very few relationships or they may not have sufficiently common relationships with other nodes. Thus, we may want to exclude such nodes from the clusters generated. In other words, we don't need to include all the nodes in clusters formed from tripartite graph if they do not contribute to the clusters. The purpose of all clustering algorithms are to determine subgraphs with strong connections/relationships among nodes. However, when all the nodes are forced to be included in clusters, their connections weaken. Furthermore, we may want to define the minimum amount of connection in subgraphs as a constraint. This means, we have to construct clusters by excluding some of the nodes that violate these constraints. In this way, we can obtain denser and more meaningful clusters.
2. Again, typical co-clustering algorithms, like the spectral clustering method, place every node in exactly one cluster. In our data set, a node may be in more than one cluster. For example, we may detect a person which belongs to two different social groups. Thus, we have to allow overlaps of nodes (not hyperedges) in clusters.

We define this new form of clustering method as *triadic co-clustering*. A full definition of the method will be given in the next section. In this paper we present the following:

1. A full fledged system developed to crawl and collect tweets of selected users on selected issues, and then construct a 3-D relationship among

user issue-sentiment keyword triples by parsing these tweets and extracting the issue-sentiment keyword relationships.

2. A flexible tri-partite clustering algorithm (*triadic co-clustering*) with parameters to control the minimum density and the minimum size of clusters.
3. The clustering method has been applied on real data sets, more specifically on the tri-partite graph obtained from "Brexit" tweets, and its effectiveness has been shown both empirically and by the coverage metric defined.

For the dataset we studied in this paper, clusters cannot overlap. This means, for a triple (user u_i, issue i_j, sentiment keyword k_k), all of the following cannot happen at the same time: $u_i \in C_{ux}$, $u_i \in C_{uy}$; $i_j \in C_{ix}$, $i_j \in C_{iy}$, $k_k \in C_{kx}$, and $k_k \in C_{ky}$, where C is a cluster. However, one user, one issue, or one sentiment keyword may be in more than one cluster simultaneously, since, for example, a specific user's feelings toward some issues can be the same as with more than one different set of people (different people clusters), so he can be in both clusters at the same time. Again considering tweets, the same sentiment words may be used by people with different clusters corresponding to different political camps. They may even use the same keywords on the same issues. For the people dimension, a person who appears in more than one cluster may be interpreted as a person close to more than one different political camp.

The rest of the paper is organized as follows. Section II discusses current related literature. Section III introduces the Triadic Co-Clustering algorithm. Section IV presents experiments and section V concludes the paper.

2. Related Work

In a tri-partite graph, a hyperedge represents the relationships among the three nodes it connects. One of the well-known examples of the tri-partite network relationship from the literature is a social tagging system, which contains three types of nodes (users, tags, resources)(Lu et al., 2009). In this relationship, a hyperedge represents a user annotating a resource with a tag. In order to determine users with common interests on resources, tri-partite clustering may be applied on this hypergraph. As another example, in a biological system, a level of gene in a sample at a particular time can be represented as a tripartite hyperedge. On this hypergraph, tri-partite clusters

represent genes showing common characteristics in samples at common time slots (Zhao and Zaki, 2005).

It has already been shown that determining bi-clusters with maximum sizes from a bipartite graph is NP-hard (Dawande et al., 2001). Thus, finding maximum size tri-clusters from tri-partite graphs is also NP-hard, and, works in literature (Zhao and Zaki, 2005; Zhu et al., 2014; Lu et al., 2009) typically propose heuristics to determine clusters.

For example, in (Zhu et al., 2014), tri-clusters are constructed by first generating biclusters between each pair of three partitions, and then, by matching each bicluster with the two others in order to construct tri-clusters. However, this approach is very costly. Another work has proposed a faster method (Zhao and Zaki, 2005). In this approach, two partitions are selected, then biclusters of these bipartite graphs are constructed. In order to construct final tri-clusters, each of these biclusters are iterated on the third partition. Since the first two partitions are fixed, this approach has bias against the third partition. In the works of (Lin et al., 2009; Liu and Murata, 2010), tri-partite clusters correspond to one-to-one relationships among the nodes. On the other hand, as in social tagging system, a group of users may tag multiple sources with the same set of tags, which represents a many-to-many relationship.

The overlapping clustering concept has been studied as well. There are several works, such as (Gregory, 2010; Lancichinetti et al., 2009; Rosvall and Bergstrom, 2008; Blondel et al., 2008; Rhouma and Romdhane, 2014; Zhou et al., 2015; Li et al., 2017) dedicated on finding overlapping communities in large graphs. These methods are all heuristic based approaches and all of them approach the problem very similarly. However for higher dimensions, such as 2 or 3, the concept of overlapping is quite different. In the higher dimensional version of the clustering problem, where co-clusters are constructed for each dimension simultaneously by relating them with each other, we can define overlapping of clusters in a dimension as a more constrained concept. In this version of overlapping, co-clusters related with each other may have some overlaps with other clusters in their dimensions, however, all co-clusters cannot have common entities with another set of co-clusters at the same time. We define this version more formally in the following section.

There are also some works focused on partitioning of signed bipartite graphs (Gokalp et al., 2013; Omeroglu et al., 2013). The main difference between these works and our research is related to the density of the bi-partite

graphs. Both of these works (Gokalp et al., 2013; Omeroglu et al., 2013) aim to distribute the whole set of nodes of both partitions into clusters, since their main assumption is the bi-partite graph is complete and each node of one partition have edges to every node of the other partition with a negative or positive sign. If there are a small number of missing edges, then, they are replaced by unsigned edges.

Another paper uses ternary (3-dimensional) relationships also (Missaoui and Kwuida, 2011). However, rather than clustering, the aim of this paper is to find frequent triadic association rules.

To the best of our knowledge, the most similar work to our method is presented in Ignatov et al.'s paper (Ignatov et al., 2015). This paper introduces triadic clustering also. However, they do not have sign on hyperedges and clusters with overlapping nodes are not considered. They have also performed experiments on data sets such as IMDB and Bibsonomy. Most of these data sets have less than 30 objects at each dimension, or the total number of hyperedges are less than 4000. Example clusters obtained from IMDB dataset has also been presented in the paper.

In our algorithm, unlike most of the above mentioned methods, we address 3 dimensional input data with labeled (either positive or negative) hyperedges. Due to this characteristic of the input, the algorithm considers density of labeled hyperedges and it aims to construct tri-dimensional clusters with high density of positive edges while minimizing the density of negative ones in clusters. Secondly, our algorithm is not biased towards any dimension. This feature increases its applicability in different domains. Thirdly, the nodes can be shared among clusters as long as hyperedges were not repeated in clusters. In some applications, this node sharing helps us to discover more useful clusters.

3. The Triadic Co-Clustering Algorithm

In a typical co-clustering problem there are N dimensions (usually 2 or 3) and all the items of these N dimensions are clustered simultaneously by using the relationships among the items of different dimensions. For example, in the case of N=2, the dimensions correspond to the partitions of a bipartite graph, and the relationship used for clustering is defined by the edges between two partitions. This means, for the 2-cluster case, all the nodes of both partitions will be placed into some clusters which are formed simultaneously. However, this approach does not work well when the relationship

is too sparse. Consider a bipartite graph with a very few edges connecting two partitions. In this case most nodes from both partitions may have no edges incident to them. If our aim is to capture the clusters defined by the edges, then we should exclude the nodes with no or very little connections while forming the clusters. For example, consider a bipartite graph, where, one of the partitions with size 1 million represents users, the other partition with size 10 thousand represents movies, and the edges between two partitions define the relationship of people watching the movies. This bi-partite graph will most likely be very sparse. If we are trying to determine a group of people with similar taste in movies, they should have watched similar movies. When such a group of people have been determined, the common movies they have watched would also have been determined simultaneously, as two clusters from two different partitions. Notice that, we are talking about forming two clusters in two partitions simultaneously, such that the relationship between these two dimensions is expected to be dense. Also, in this formulation, a cluster of one partition will have a one-to-one relationship with a cluster from the other partition. We can assume that only a small portion of movies and people will have a chance to be in such clusters, thus, leaving most movies and most people without being in any cluster at all. Furthermore, some people may have common taste with more than one group of people, or, some movies might be in the watch list of more than one group of people, implying there can be overlap between clusters of a partition. However, we cannot have the same person and the same movie repeated in two clusters simultaneously.

In this work, we propose a feasible solution to this problem. Even though this problem can be defined for any dimension, we focus on the three dimensional case. Our modeling can easily be reduced to 2 dimensions, or can be extended to higher dimensions with a little additional effort. Below we define the 3-dimensional form formally.

Definition (Triadic Co-Clustering): Given 3 sets U_1, U_2, U_3 and 3D-relationships $U_1 \times U_2 \times U_3$ among the items of these 3 sets (i.e., hyperedges *hyperedge*$(x,\ y,\ z)$ such that $x \in U_1,\ y \in U_2,\ z \in U_3$), density and size thresholds as ϵ and λ_i for each dimension, *triadic co-clustering* determines k 3-clusters as $\{(C_{11},C_{21},C_{31}),\ (C_{12},C_{22},C_{32}),\cdots,\ (C_{1k},C_{2k},C_{3k})\}$ satisfying the following:

- Clusters satisfy size constraint. That is for each cluster C_{ij}, $size(C_{ij}) \geq \lambda_i$.

- Clusters satisfy density constraints. That is for each 3-cluster (C_{1j}, C_{2j},C_{3j}), $number\_of\_hyperedges(C_{1k} \times C_{2k} \times C_{3k}) \geq \epsilon$.

- No common hyperedges. That is, if $x \in U_1$, $y \in U_2$, $z \in U_3$, then, it is not possible to generate two different 3-clusters as (C_{1i},C_{2i},C_{3i}) and (C_{1j},C_{2j},C_{3j}), such that, $x \in C_{i1}$, $y \in C_{i2}$, $z \in C_{i3}$ and at the same time $x \in C_{j1}$, $y \in C_{j2}$, $z \in C_{j3}$.

The first item in the above definition enforces forming dense clusters with strong relationships among the sets from three different dimensions. The second item prevents forming trivial and very small clusters satisfying the density constraints. Finally, the third item allows node-overlapping among the clusters of the same dimension while disallowing hyperedge-overlapping among 3-clusters.

In this work, we extend this problem with one more feature by defining signs (positive or negative) for hyperedges. Therefore, in this signed version, there will be two density parameters to maximize the number of positive edges while minimizing the number of negative edges (or vice versa) in clusters.

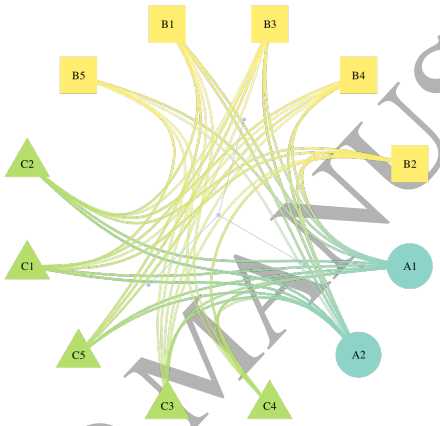
The input parameters, together with related cluster features and the definitions of the size and the ratio constraints, are given in Table 1. Since, the positive sign of the hyperedge means three nodes forming this edge agree, and the negative sign means they disagree, the idea is to form co-clusters on three dimensions using this agreement relationship and maximizing the agreement while minimizing the disagreement among the nodes of co-clusters. Furthermore, we want to generate the largest possible clusters. So, we have two conflicting objectives. For a given tri-partite graph, we would like to construct the largest possible tri-clusters, as well as generate clusters with maximum density of positive signed hyperedges and minimum density of negative signed hyperedges. There is a trade-off between these two objectives. By trying various values of the parameters in Table 1, clusters with different properties/qualities may be generated. As expected, only very small size or trivial perfect clusters can be generated with full positive labeled edges. In order to generate more meaningful and useful clusters with larger sizes, we may tolerate some negative labeled edges in clusters and reduce the minimum positive edge ratio requirement.

To simplify the process, we have generated a fully connected tri-partite graph by adding hyperedges with no sign between all 3 nodes from 3 different dimensions, if they are not already connected in the original graph with

a_1	b_1	b_2	b_3	b_4	b_5
c_1		−		+	−
c_2	−	+	+	+	
c_3	+				+
c_4		−			+
c_5	+		−	+	

a_2	b_1	b_2	b_3	b_4	b_5
c_1		+	−		+
c_2	+	+		+	
c_3	+	+	−		−
c_4	−	+	+		+
c_5	+				

(a) Matrix Representation



(b) Graph Representation

Figure 1: Input Data

negative or positive signed edges. We have developed a simple and efficient greedy heuristic in order to generate clusters satisfying the above mentioned constraints. The general idea is to start with the whole graph as a cluster, then trim less effective nodes until ratio constraints are satisfied, or it cannot satisfy the size constraint. Effectiveness of the nodes is also defined using simple formulas which will be discussed below. Our method works as follows:

1. Start with a single co-cluster of the whole graph and trim the least effective node from it in order to increase its positive density and decrease its negative density as much as possible. Notice that this trimming operation reduces the size of the cluster.
2. Repeat this trimming operation until a cluster is obtained satisfying both the density constraints and the cluster size constraints, or until

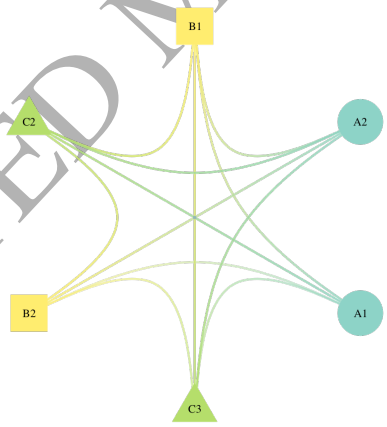
one of the minimum cluster size constraints are violated.

- If the obtained cluster satisfies all the constraints, it is added to the cluster list. Then, in order to remove the hyperedges used in this cluster, least effective nodes incident to each hyperedge are removed from the graph. The process then repeats itself from the beginning using the remaining graph.
- If the obtained cluster violates one of the size constraints, all node removals from this iteration were backtracked, and only one hyperedge with negative sign is selected from the graph, and one of the nodes incident to that edge is removed from the graph. Following this, the process repeats itself from the beginning using the remaining graph.

a_1	b_1	b_2
c_2	−	+
c_3	+	

a_2	b_1	b_2
c_2	+	+
c_3	+	+

(a) Matrix Representation



(b) Graph Representation

Figure 2: Output Cluster, when ϵ_p=0.75, ϵ_n=0.15

In this paper, we use the notations given in Table 1. Triadic Co-Clustering Algorithm takes a set of hyperedges, Γ as an input, such that each hyperedge h connects three nodes from three different types $U =$

Table 1: Cluster Control Parameters and Constraints

Symbol	Meaning
Input Parameters	
ϵ_p	minimum h_+ edges ratio parameter in a cluster
ϵ_n	maximum h_- edges ratio parameter in a cluster
λ_i	minimum size of dimension i parameter in a cluster
Cluster Properties	
L_i	number of nodes for type i in a cluster (size of type i)
C_p	number of h_+ edges in a cluster
C_n	number of h_- edges in a cluster
Constraints	
$\epsilon_p \leq \frac{C_p}{L_1 \times L_2 \times L_3}$	ratio of h_+ edges constraint in a cluster
$\epsilon_n \geq \frac{C_n}{L_1 \times L_2 \times L_3}$	ratio of h_- edges constraint in a cluster
$L_i \geq \lambda_i$	size for dimension i constraint in a cluster

(U_1, U_2, U_3). Figure 1 illustrates hyperedges given as 3D matrix. These hyperedges have either positive or negative labels, represented by positive and negative signs respectively in Figure 1. Remaining entries (white cells) correspond to node triples without connecting hyperedges. In the example, nodes are $\{\{a_1, a_2\}, \{b_1, b_2, b_3, b_4, b_5\}, \{c_1, c_2, c_3, c_4, c_5\}\}$ from types U_1, U_2, U_3 respectively. An example cluster obtained from the input graph given in Figure 1 is depicted in Figure 2.

The aim of Triadic Co-Clustering is to find tripartite clusters of hyperedges with highly positive labels. To be a valid tripartite cluster, it has to satisfy threshold values for both density and size. The density threshold values are ϵ_p and ϵ_n, such that $0 \leq \epsilon_p, \epsilon_n \leq 1$, $(\epsilon_p + \epsilon_n) \leq 1$. The former one represents the minimum ratio density of positive hyperedges (h_+) among all possible hyperedges (i.e., there may be $L_1 \times L_2 \times L_3$ number of possible hyperedges for a cluster with size (L_1, L_2, L_3), where L_i is number of nodes with U_i type in the cluster). If C_p is the number of h_+, then:

$$\epsilon_p \leq \frac{C_p}{L_1 \times L_2 \times L_3}, \tag{1}$$

If $\epsilon_p = 1$, generated tripartite clusters become tripartite cliques as well. ϵ_n is the value to control the density of negatively signed hyperedges (h_-). If C_n represents the number of h_-, then:

$$\epsilon_n \geq \frac{C_n}{L_1 \times L_2 \times L_3}, \tag{2}$$

shows the maximum allowed tolerance of h_- in a cluster if $\epsilon_n \neq 0$.

In order to prevent constructing very small clusters, λ_i is defined, such that:

$$L_i \geq \lambda_i, \tag{3}$$

for $1 \leq i \leq 3$, and this constraint should also be satisfied by every cluster.

The input parameters presented in Table 1 have different effects on the structures and the qualities of the clusters constructed by the algorithm. By adjusting them, different clusters may be constructed. Therefore, how these parameters are going to be adjusted can be very important. To do this adjustment well, both the structure of the input data and what is expected from the clusters must be known in advance. These metrics are: density of positively and negatively labeled edges, percentage of sparseness, and sizes of each dimension. For densely connected input data, density parameters can be chosen as high, or depending on the requirements, they can also be set to a lower value as well. If it is known that positive connections are dominant, then ϵ_p can be set to a higher value. On the other hand, for a sparse data set, the algorithm may behave more severely on eliminating potential clusters. If ϵ_p is not low enough or λ_i is high, then no clusters may be found. In order to be able to find the best settings of these input parameters, some exploration of potential values may be needed.

Since in our problem, we have both positive and negative labelled edges, without some apriori knowledge about their densities, arbitrary settings of the relevant density parameters may produce uninteresting clusters. We assume that interesting clusters will have high density of positive and low density of negative edges. For example, if the input graph is very sparse, we may obtain trivial clusters with a very few nodes only, if the (positive) density parameter is set to a high value. On the other hand, for dense graphs, a small (positive) density parameter may generate a giant single cluster including almost all the nodes. The difference between the densities of positive and negative edges in the graph is also very important. For example, if the negative edge density is close to the positive edge density in

the graph, relatively high negative density ratio may produce clusters with so many negative edges, maybe very close to the number of positive edges. Typically, such clusters are not considered as interesting.

In addition to the density, the dimension sizes of the input graphs are also very important. Since each dimension size can be different, their corresponding size constraint parameters may also be set to different values. While we can choose a small size constraint for clusters for a dimension with small size, we can set the size constraint of larger dimension to a higher value.

Since dimension size parameters and density parameters conflict with each other, some kind of exploration of these parameters are needed in order to be able to find more interesting clusters. As we increase the size parameters it becomes more difficult to find high (positive) density clusters. In general, we are not interested in very small clusters even if they have high densities. Therefore, a simple exploration process may start with high density and large size parameters and reduce them to values that produce clusters with acceptable densities and non-trivial sizes.

Triadic Co-Clustering algorithm (Algorithm 1) starts by generating a potential cluster α which contains all hyperedges in Γ. The main loop (lines from 3 to 23 in Algorithm 1) iteratively constructs clusters satisfying both size and density constraints.

It begins with the remaining nodes of the graph and removes least effective nodes (through while loop in lines 5 to 7), until either all constraints are satisfied (conditions of the if statement at lines 9 and 10), or until the graph becomes too small to satisfy minimum size constraints due to these node removals.

If both the density and size constraints are satisfied, the remaining nodes of the graph form a cluster which is added to the cluster list (at line 11). In addition, its edges (all signed edges) are removed from the graph (by adding them to invalid hyperedge list at lines 12 and 13), so they cannot be used in the construction of a new cluster. This way its guaranteed clusters will not share edges, but can still share nodes.

As mentioned above, due to trimming of the graph by removing its least effective nodes to satisfy density constraints, the graph may become too small and unable to satisfy the size constraints any more, and thus, the else branch of the if statement between the lines 15 and 18 of the algorithm is executed. A very simple heuristic is done here. A negative edge is randomly selected from the graph and removed (adding it to an invalid edge set). This way, the next round of iteration of the main loop starts with a graph with one less

Algorithm 1 Triadic Co-Clustering Algorithm

Input: Γ : all hyperedges
Input: ϵ_p : lowest ratio value of h_+ in a cluster
Input: ϵ_n : highest ratio value of h_- in a cluster
Input: λ_i : minimum size value for type i in a cluster
Output: $\Re$: list of clusters

procedure TriadicCluster($\Gamma, \epsilon_p, \epsilon_n, \lambda_1, \lambda_2, \lambda_3$)
  $\Gamma' \leftarrow \Gamma$
  $\Re \leftarrow \emptyset$
  loop
    $\alpha \leftarrow \Gamma'$
    while (SizeCheck($\alpha, \lambda_1, \lambda_2, \lambda_3$)) **and**
      not DensityCheck($\alpha, \epsilon_p, \epsilon_n$) **do**
        RemLeastEffNode(α)
    end while
    if DensityCheck($\alpha, \epsilon_p, \epsilon_n$) **and**
      SizeCheck($\alpha, \lambda_1, \lambda_2, \lambda_3$) **then**
      $\Re \leftarrow \Re \bigcup \{\alpha\}$
      for each $h_{-/+} \in \alpha$ **do**
        $\Gamma_{inval} \leftarrow \Gamma_{inval} \bigcup \{h_{-/+}\}$
      end for
    else
      $h_- =$ RandomNegativeEdge(Γ)
      $\Gamma_{inval} \leftarrow h_-$
    end if
    CleanInvalids($\Gamma, \Gamma_{inval}, \Gamma'$)
    if not SizeCheck($\Gamma', \lambda_1, \lambda_2, \lambda_3$) **then**
      return $\Re$
    end if
  end loop
end procedure

negative edge, and the chance for generating higher positive density clusters increases.

After this main if statement (from lines 9 to 18 in the algorithm) either one cluster is generated and its edges are added to the invalid edge list, or just one negative edge is added to the invalid edge list. Then, these invalid edges must be removed from the graph. This is done by removing one of the nodes incident to these edges in order to be able to reduce the graph size as well. The procedure at line 19 removes one node for each of the invalid edges from the original graph, then the remaining graph (Γ') is used in the next iteration to discover another cluster from it. However, the remaining graph may be too small, and if it does not satisfy the size constraints (checked at line 20), the process ends and clusters obtained so far are returned (at line 21).

3.1. Density Check

DensityCheck is a sub-procedure with 3 input parameters. They are a potential cluster α and two constraints values ϵ_p and ϵ_n. The purpose of this sub-procedure is controlling the quality of a given cluster due to constraints 1 and 2. If both constraints are satisfied, then this sub-procedure returns **true**. Otherwise, it returns **false** to point out the fact that removing a node is necessary.

3.2. Size Check

SizeCheck is a sub-procedure with 4 input parameters. The first one is a potential cluster α, the others are size constraints for each of 3 types. This sub-procedure checks the size of a given cluster. If the size of each dimension is not below its corresponding limit (constraint 3), then this sub-procedure returns **true**. Otherwise, it returns **false**.

3.3. Remove Least Effective Node

RemLeastEffNode sub-procedure takes one input parameter which is a potential cluster α. It removes a node from α.

Heuristic calculations are used to determine which node to remove. There is a simple approach behind this. If a node is connected by high number of h_+, it should be less likely to be removed. If a node is highly linked by negatively signed hyperedges or it is loosely connected, its probability of being removed is high. Due to the statement of the problem, positively labeled hyperedges are valuable. Therefore, it is not desired to extract them from a cluster. In

a_1	b_1	b_2	b_3	b_4	b_5
c_1		−		+	−
c_2	−	+	+	+	
c_3	+				+
c_4		−			+
c_5	+		−	+	

a_2	b_1	b_2	b_3	b_4	b_5
c_1		+	−		+
c_2	+	+		+	
c_3	+	+	−		−
c_4	−	+	+		+
c_5	+				

Figure 3: Removing Node b_3

order to keep them inside, they are valued by a positive number. In contrast, a negative number is given the ones carrying negative labels (Equation 4).

$$val(h) = \begin{cases} 1 & \text{if } h \text{ has positive label} \\ -1 & \text{if } h \text{ has negative label.} \end{cases} \tag{4}$$

With the help of it, effectiveness of each node is determined. For this purpose, formula 5 is used.

$$E_{ir} = \frac{1}{S_i} \times \sum_{h \in \alpha} \begin{cases} val(h) & \text{if } r \in h \\ 0 & \text{otherwise} \end{cases} . \tag{5}$$

E_{ir} is the effectiveness value of node r from type i. It is a density calculation of hyperedges which connect a node to the cluster. S_i refers to the maximum number of hyperedges which can be linked to that node. For each type, S_i value differs (Table 2). The lowest effectiveness value indicates the node to remove. An example is given in Figure 3, where the least effective node is b_3.

Table 2: Maximum possible number of hyperedges for each type

Type	Value
S_1	$L_2 \times L_3$
S_2	$L_1 \times L_3$
S_3	$L_1 \times L_2$

3.4. Random Negative Edge

RandomNegativeEdge is a sub-procedure taking 1 input parameter, which is a potential cluster α. This sub-procedure helps to invalidate a hyperedge. It finds and returns one which is not significant whether it is covered or not. Predictably, it is the negatively signed one. The sub-procedure traverses all hyperedges and returns the first (h_-) it finds.

3.5. Clean Invalids

Invalid hyperedges are not desired to be part of future clusters. The hyperedges of all tripartite clusters previously found are invalid. Moreover, hyperedges returned by RandomNegativeEdge sub-procedure are added to the list of invalids. CleanInvalids sub-procedure aims to remove all invalid hyperedges (Γ_{inval}) from (Γ'). Γ is copied as Γ' at the beginning of this sub-procedure. As a result, (Γ') does not contain any hyperedge in Γ_{inval}, and this sub-procedure terminates. Then, the algorithm searches a valid cluster inside Γ'.

In order to remove a hyperedge, one of the nodes linked by this hyperedge should be removed. In this manner, to remove an invalid hyperedge, CleanInvalids sub-procedure looks for a node, then removes it. It repeats the same removing action until no invalid hyperedge is left in Γ'. While selecting a node, it uses a heuristic that reduces side effects of removing nodes on the cluster size as much as possible.

On the heuristic calculation, first, the number of invalid hyperedges connected to each node is counted. This value is then divided by S_i where i refers to the type of that node (Equation 6). The final value (θ_{ir}) is the density of invalids for that node. If (θ_{ir}) is high, the node r is more likely to be removed.

$$Q_{ir} = \frac{1}{S_i} \times \sum_{h \in \alpha} \begin{cases} 1 & \text{if } r \in h \wedge h \text{ is invalid} \\ 0 & \text{otherwise} \end{cases} . \tag{6}$$

Next, the effectiveness of each node in Γ' for all the valid hyperedges is calculated. This calculation is a modified version of equation (5). Additionally it checks if hyperedges are valid. If not, they are not counted in the calculation (Equation 7). The final value (E^v_{ir}) is the effectiveness value for node r. A node with high (E^v_{ir}) will likely not be extracted.

$$E^v_{ir} = \frac{1}{S_i} \times \sum_{h \in \alpha} \begin{cases} 1 + val(h) & \text{if } r \in h \wedge h \text{ is valid} \\ 0 & \text{otherwise} \end{cases} . \tag{7}$$

(θ_{ir}) is directly proportional with selecting a node to remove, as (E^v_{ir}) value is inversely proportional. Therefore, the final calculation is performed as in Equation 8. c is a constant value which is generally set to 1.

$$\gamma_{ir} = \theta_{ir} \times \frac{1}{c + E^v_{ir}}. \tag{8}$$

Among all nodes, the one with the highest γ value is removed. The removing node is performed iteratively until there is no invalid hyperedge inside Γ'.

For the input data in Figure 1, Triadic Co-Clustering Algorithm finds the cluster in Figure 2 in the first iteration. Then, hyperedges of this newly generated cluster are labeled as invalid. In the next iteration, a new potential cluster Γ' (Figure 4a) is generated from Γ. But Γ' contains some invalid hyperedges (colored with purple in Figure 4a). Therefore, Γ' is passed to CleanInvalids sub-procedure which cleans Γ' from invalid hyperedges. First, node c_3 is removed since γ_{2c_3} is $3 \div 0.3 = 30$, is the maximum among γ values. Then, nodes b_2 and b_1 are selected and removed respectively (Figure 4b, 4c). This will result a clean Γ' (Figure 4d) and the sub-procedure terminates.

3.6. Complexity Analysis

Let us assume that the size of each dimension is equal to L. Thus, $S_i = L^2$ and $L^3 = n$, where n represents the maximum potential number of hyperedges. In this algorithm, the main calculations are (E_{ir}) and (γ_{ir}). (E_{ir}) is calculated for each node from each dimension. Therefore, the total number of calculations are $(L \times S + L \times S + L \times S) = 3 \times n$. (γ_{ir}) value includes (θ_{ir}) and (E^v_{ir}). This makes total calculation number for (γ_{ir}) $6 \times n$. Thus, the asymptotic complexity of (E_{ir}) and (γ_{ir}) is $\mathcal{O}(n)$.

Worst case scenario for Triadic Co-clustering algorithm happens when no tripartite cluster is generated. In this case, RemLeastEffNode is called until SizeCheck returns **false** in line 6. This loop is iterated at most $(3 \times L)$ times. Time complexity of the loop between lines 6 and 9 is $\mathcal{O}(L \times n)$. CleanInvalids sub-procedure has same time complexity as well. The outer loop (lines between 4-24) eliminates a single node in each iteration in worst case. Therefore, until it ends, it iterates $(3 \times L)$ times. As a result, the worst case asymptotic complexity of Triadic Co-clustering algorithm is $\mathcal{O}(L^2 \times n) = \mathcal{O}(n^{\frac{5}{3}})$.

a_1	b_1	b_2	b_3	b_4	b_5
c_1		$-$		$+$	$-$
c_2	$-$	$+$	$+$	$+$	
c_3	$+$				$+$
c_4		$-$			$+$
c_5	$+$		$-$	$+$	

a_2	b_1	b_2	b_3	b_4	b_5
c_1		$+$	$-$		$+$
c_2	$+$	$+$		$+$	
c_3	$+$	$+$	$-$		$-$
c_4	$-$	$+$	$+$		$+$
c_5	$+$				

(a) Removing First Node

a_1	b_1	b_2	b_3	b_4	b_5
c_1		$-$		$+$	$-$
c_2	$-$	$+$	$+$	$+$	
c_4		$-$			$+$
c_5	$+$		$-$	$+$	

a_2	b_1	b_2	b_3	b_4	b_5
c_1		$+$			$+$
c_2	$+$	$+$		$+$	
c_4	$-$	$+$	$+$		$+$
c_5	$+$				

(b) Removing Second Node

a_1	b_1	b_3	b_4	b_5
c_1			$+$	$-$
c_2	$-$	$+$	$+$	
c_4				$+$
c_5	$+$	$-$	$+$	

a_2	b_1	b_3	b_4	b_5
c_1		$-$		$+$
c_2	$+$		$+$	
c_4	$-$	$+$		$+$
c_5	$+$			

(c) Removing Third Node

a_1	b_3	b_4	b_5
c_1		$+$	$-$
c_2	$+$	$+$	
c_4			$+$
c_5	$-$	$+$	

a_2	b_3	b_4	b_5
c_1	$-$		$+$
c_2		$+$	
c_4	$+$		$+$
c_5			

(d) Clean Γ'

Figure 4: Cleaning Invalids

In the best case, the whole input is a complete triadic co-cluster. If the input satisfies density and size constraints (line 6-7), it will be added into $\Re$ (line 12). RemLeastEffNode sub-procedure will not be called. Then, all nodes inside the input will be invalid. Therefore, they will be removed in CleanInvalids (line 20), and Γ' will be empty. Thus, the algorithm will conclude (line 22). As a result, asymptotic running time complexity of Triadic Co-clustering algorithm in best case will be $\mathcal{O}(n)$.

The algorithm concludes when Γ' does not satisfy the size constraint. Therefore, values for λ_i, which are determined by the user may effect the number of iterations, and thus, the running time complexity. In these calculations, we assume $(\lambda_1, \lambda_2, \lambda_3)$ are all set to 1. Furthermore, the running time of the algorithm is highly dependent on the quality of the input. If positivity of the input data is high, large clusters are found in early iterations, and the algorithm quickly converges. If the negativity or the sparseness is high, then RemLeastEffNode operation is called more often. Also, negativity will decrease the chance of finding a cluster in line 10. This will increase the number of iterations of the outer loop (lines between 4-24). In this case, the algorithm will behave closely to the worst case.

4. Experiments

4.1. Experiment Metrics

We have performed two different kinds of experiments. We have generated synthetic data sets in order to analyze the behavior of our method, and we have also tested our algorithm with a real data set in order to illustrate the applicability of our method. Synthetic data is denser. Therefore, in these experiments, ϵ_p and ϵ_n parameters are set to relatively high values. Size constraints are kept at minimum in order to not eliminate small clusters. In the end, many clusters are found. Real world data set is much sparser. When we have tried to find larger clusters, the number of clusters obtained became very small. All experiments are performed on MacBook Pro Mid 2015 (Intel i7 2,5 GHz, 16GB memory).

In our experiments, we have used internal evaluation measures. Namely, to evaluate the baseline and proposed methods, we have used well known clustering quality metrics such as coverage, density, and purity. The definitions of the first two are as follows:

$$Coverage(\Re) = \sum_{\alpha \in \Re} L_1 \times L_2 \times L_3, \tag{9}$$

where $\Re$ is a cluster list. α is a cluster inside $\Re$ and (L_1, L_2, L_3) are sizes of dimensions of α.

$$DensityP(\alpha) = \frac{C_p}{L_1 \times L_2 \times L_3}, \tag{10}$$

$$DensityN(\alpha) = \frac{C_n}{L_1 \times L_2 \times L_3}, \tag{11}$$

where α is a cluster. C_p is number of positive hyperedges and C_n is number of negative hyperedges inside α. (L_1, L_2, L_3) are sizes of dimensions of the given cluster.

Purity (Manning et al., 2008) is formally defined as;

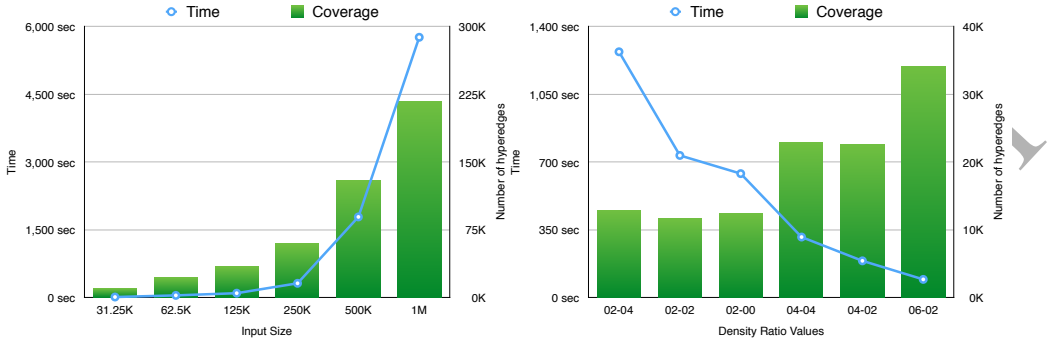
$$Purity(\Re) = \frac{1}{n} \sum_{i=1}^{k} max_j |C_i \cap l_j| \tag{12}$$

where k is the number of clusters that model produces, n is the number of instances, l_j is the set of instances which belong to the ground-truth cluster j, and C_i is the set of instances that are members of cluster i of the model's output.

4.2. Experiments on Synthetic Data

This experiment is conducted to determine the effectiveness and the scalability of our proposed method. In order to evaluate Triadic Co-Clustering algorithm, we generate datasets with varying sizes.

In the first set of experiments, we fix h_+ and h_- density ratios while changing input sizes. We generate 6 sample datasets. Each set contains positive hyperedges with 60%, negative hyperedges with 20%, and 20% empty. $(L_1 \times L_2 \times L_3)$ values for the samples are (31.25K, 62.5K, 125K, 250K, 500K, 1M), respectively. Figure 5a presents the results. Input parameters are fixed as $\epsilon_p = 0.75$, $\epsilon_n = 0.10$, $\lambda_i = (2,2,2)$. ϵ_p value has been chosen larger than the positivity ratio of the input data. If it was smaller, the whole input would have been accepted as one large cluster. Similarly, ϵ_n value is chosen smaller than the negativity ratio of the input data. Therefore, elimination of negative hyperedges was possible. Additionally, we set the size constraints (λ_i) small in order to find as many clusters as possible. The way, we have



(a) Varying Input Size (b) Varying Density Ratios in Input Data

Figure 5: Performance Graphs of Experiments on Synthetic Data

managed to cover as many hyperedges as possible by these clusters. Test results are plotted in Figure 5a.

In the second test, we have generated 6 datasets. In this test, we fix the total size as $(L_1 \times L_2 \times L_3) = 125K$ and we have performed tests with varying density ratios for (h_+, h_-) pairs as {(0.2,0.4), (0.2,0.2), (0.2,0.0), (0.4,0.2), (0.4,0.4), (0.6,0.2)}. We did not change any of input parameters for test 2 as well. Chosen ϵ_p value is larger than positivity ratios of all inputs while ϵ_n value is relatively small (which is 0.1). λ_i values are chosen to be small in order to increase the number of clusters found even with small sizes. The results are shown in Figure 5b. Negativity of an input data effects the execution time of the algorithm poorly. However, if positivity is high, the algorithm finds large clusters and terminates more quickly.

The figures show both execution times and the number of hyperedges included in the constructed clusters. We prefer most (positive) hyperedges to be included in clusters while clusters being non-trivial. The results show that we achieve very high coverage in that sense, since constructed clusters include almost half of the positive signed hyperedges.

4.3. Experiments on Brexit Data

Since the problem that we define is new, previous datasets made publicly available by other scholars do not meet our constraints. Given the scarcity of publicly available datasets containing hypergraphs with signed tri-partite hyperedges, we decide to exploit well-known social media data source Twitter,

Table 3: Keywords and number of tweets they occur in the dataset.

Keyword	Count	Keyword	Count
eu	12,206	#leave	157
tax	3,278	hospitals	150
#brexit	2362	gas	147
nhs	2312	daesh	126
#strongerin	2249	muslims	125
#voteleave	1,802	worker	123
workers	1,250	healthcare	114
immigration	1,066	tuition	86
#remain	807	immigrants	72
#takecontrol	460	visas	50
debt	456	abortion	49
hospital	411	deport	45
brussels	396	immigrant	34
wage	364	islam	33
borders	343	citizenship	32
border	271	deportation	22
isis	262	vaccination	11
wages	259	same-sex	9
nuclear	247	lgbtq	7
lgbt	232	monarchy	6
muslim	223	#remainin	3
medical	204	samesex	2
oil	200	al-qaeda	1
taxes	166	lbtq+	1
Total Number of tweets:		26,624	

thanks to its generous APIs providing researchers social data freely over the last decade. We choose to focus on recent Brexit referendum in the United Kingdom to be able to generate signed tri-partite edges from a social media data.

4.3.1. Data acquisition

We collect tweets from 411 politicians[1] from 5 major political parties in the United Kingdom. Twitter Search API is utilized to get the latest 3,200 tweets of each politician. For preprocessing, tweets dated before January 1, 2016 are removed. Then, tweets that do not contain the words shown in Table 3 are eliminated. Number of tweets that contain relevant keywords after preprocessing can be seen in Table 3.

To represent each politician's stance towards the issue in binary format, we utilize an off-the-shelf sentiment analysis tool called SentiStrength[2]. We assume that overall sentiment score of the tweet implies the opinion of the tweet towards the issue word the tweet contains in this work. Whether the assumption holds or not is not the focus of this study, yet some insights based

[1]Users' Twitter id lists can be obtained from http://mlg.ucd.ie/aggregation/index.html
[2]http://sentistrength.wlv.ac.uk/

on our experiments are discussed in section 4.3.2. To build input tensor Γ, sentiment scores of the tweets of the i'th politician with j'th issue using k'th sentiment-expressing words are summed up and put into the i'th row, j'th column and k'th slice entry.

The major 5 political party viewpoints can be summarized as below (Hobolt, 2016);

- Labour: Overwhelming majority of Labour Party members campaign for staying in European Union although there were raising concerns about the structure and function of the European Union.
- Conservatives: The leader of the Conservative Party, David Cameron offered the referendum and started the campaign for remaining in EU. There was a clear leaning towards leaving the EU despite the Cameron's efforts.
- Libdem: Liberal Democrats campaigned for staying in the EU.
- UKIP: UK Independence Party was a prominent figure in the referendum campaign. They passionately advocated to leave the EU. Blocking the refugees from entering the country, opposing international and EU-wide trade agreements, defending UK-born workers' rights over immigrants' rights were standing out as motivating factors in their campaign.
- SNP: Scottish National Party campaigns to stay in EU.

The input data has 411 users and 48 different issues. The data also contains 6,776 keywords. Keywords are derived from a state-of-the-art sentiment word list [3]. Occurrence of each keyword is counted. Then, the most frequent 1000 keywords are selected, by keeping number of users and issues stable.

This results in a maximum of $411 \times 48 \times 1000 = 19,728,000$ possible hyperedges in the input data. This is the size of 3D matrix constructed from nodes. The total number of hyperedges in the input data is 26,624. The rest of it is sparse. Thus, the density of hyperedges in the input is $\sim$0.001. More than 60% of the hyperedges have a negative label. Since the negative hyperedges dominate the positives and the original algorithm mines clusters with high positive density, the labels of hyperedges are negated in order to

[3] http://www.cs.uic.edu/$\sim$liub/FBS/opinion-lexicon-English.rar

utilize negative feelings rather than the positive feelings in determining the clusters.

4.3.2. Discussions of Experiment Results

Tucker Decomposition Results

As a baseline method, we apply Tucker decomposition to Γ to find user, issue and sentiment word clusters in the Brexit data set. To that end, we utilize the Tucker decomposition component (Kolda and Sun, 2008) of MATLAB Tensor Toolbox Version 2.6 of (Bader and Kolda, 2007) which has the following objective function;

$$\begin{aligned} \min_{\mathbf{C},\mathbf{U},\mathbf{I},\mathbf{K}} & ||\mathbf{\Gamma} - \mathbf{C} \times_1 \mathbf{U} \times_2 \mathbf{I} \times_3 \mathbf{K}||_F^2 \\ \text{s.t.} \quad & \mathbf{\Gamma} \in \mathbb{R}^{u \times i \times k}, \mathbf{C} \in \mathbb{R}^{c_1 \times c_2 \times c_3}, \\ & \mathbf{U} \in \mathbb{R}^{u \times c_1}, \mathbf{I} \in \mathbb{R}^{i \times c_2}, \mathbf{K} \in \mathbb{R}^{k \times c_3} \end{aligned} \tag{13}$$

where u is the number of users, i is the number of issues, and k is the number of keywords.

Our experiment setup for measuring the performance of Tucker decomposition spans different values of c_1, c_2 and c_3 between 3 and 40. For brevity, we set all parameters of c_1, c_2 and c_3 equal to each other.

To evaluate the purity of clusters, we simply check cluster assignments in matrix $\mathbf{U}$. Purity measure of user clusters are reported for different cs (where $c = c_1 = c_2 = c_3$) between 3 and 40.

To be able to evaluate the tucker decomposition against density and coverage metrics, we form hyper-clusters of users, issues and keywords according to the core tensor $\mathbf{C}$. A simple heuristic is followed when choosing which user, issue and keyword clusters to form a hyper-cluster. Top c (where $c = c_1 = c_2 = c_3$) entries of core tensor $\mathbf{C}$ with highest absolute values are picked in a way where none of the picks share any row, column or slice index. Therefore, each cluster of user, issue and keyword represented in $\mathbf{U}, \mathbf{I}, \mathbf{K}$ is assigned to a hyper-cluster. Density and coverage measures of hyper-clusters are reported for different c's between 3 and 40.

Purity of the clusters found tend to increase with increasing numbers of clusters. It oscillates between 0.43 and 0.52 for different c's. Coverage of hyper-clusters tend to decrease dramatically with increasing number of clusters. This signals the fact that splitting hyper-clusters to further smaller hyper-clusters does not help to find dense sub-clusters. Density of the clusters tend to increase with increasing number of clusters, yet not with any

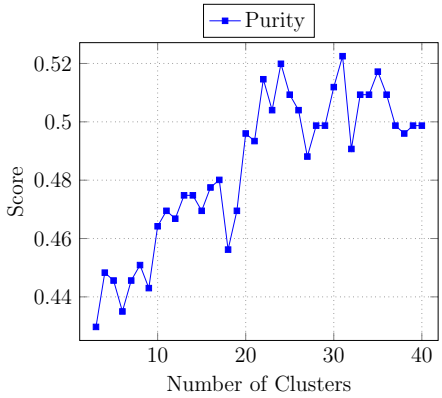
significance. It supports the claim we make about tucker decomposition's inability to find dense sub-clusters with increasing number of clusters.

For qualitative analysis of the Tucker decomposition's performance, with the expectation to determine *remain* and *leave* camps, we focus on the case of 2 clusters for the user dimension ($c_1 = 2$), and 3 clusters for the issues and sentiment words dimensions ($c_2 = c_3 = 3$), respectively. When two opposing camps exist, it is also likely that the other dimensions will have reflections of these two camps as two clusters, and there may be a third cluster for the remaining issues and the sentiment words. Since spectral clustering includes all the nodes of all these dimensions, we obtain very large clusters with very low edge densities as seen in Figure 6. Notable results that we observed from this experiment are as follows:

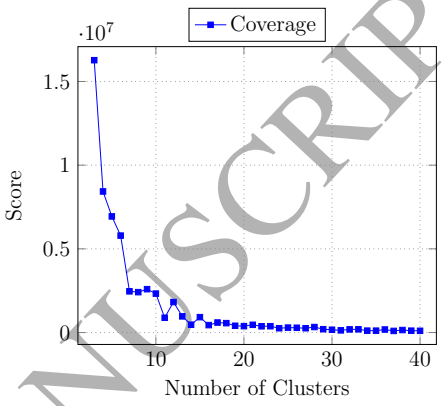
- Issues are unevenly distributed to 3 clusters. The first one contains three of the most popular issues, namely "citizenship", "brussels" and "worker". There is a strong negative reaction towards these issues from the first user cluster. The next cluster contains 11 issues, and towards those issues, there is less negative reaction from the second user cluster. This issue cluster contains issues such as "humanrights", "tuition", "eu". The reaction is not very clear on remaining clusters. Thus, it is not possible to obtain any useful information from issue dimension.

- Sentiment keywords are also unevenly distributed across three clusters. Even the smallest cluster contains 105 sentiment words. Positive and negative sentiment words are also distributed through the clusters. There is no useful result that can be obtained from these clusters either.

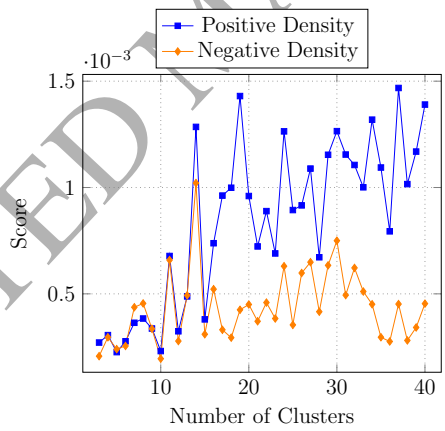
- The user clusters obtained from this method are also not very informative. The first cluster contains 133 politicians from a variety of parties. The largest group in this cluster is Labours with 62 members, which is followed by 44 members of the Conservatives. It also contains 8 SNP, 8 Liberal, and 5 UKIP members, as well as 6 members from other parties. The second cluster contains 239 politicians. Labours increase to 95, Conservatives almost double to 90, and other parties also increase, Liberals to 25, UKIP to 11, and SNP to 15. So, these clusters do not give any information about the party membership vs issue relationship either.



(a) Effect of number of clusters ($c = c_1 = c_2 = c_3$) on purity of user clusters



(b) Effect of number of clusters ($c = c_1 = c_2 = c_3$) on coverage of hyper-clusters



(c) Effect of number of clusters ($c = c_1 = c_2 = c_3$) on density of hyper-clusters

Figure 6: Experimental Results for Tucker Decomposition

Triadic Co-Clustering Results

When we applied our method on the Brexit dataset, we tried different density and cluster size constraints in order to be able to explore different clustering structures. Four different runs are performed on various values for λ_i, ϵ_p and ϵ_n. The values are listed on the Table 4 with respect to test numbers. λ_i values are for users, issues, and keywords, respectively. The sizes of clusters obtained from these tests are given in Table 5. Cluster sizes are written in the form of (Users × Issues × Keywords).

Since most tweets have negative sentiments, we swap positive and negative signs of the edges in order to obtain correlation between users, issues and negative sentiment words they use on these issues.

Table 4: Variables for tests with real data

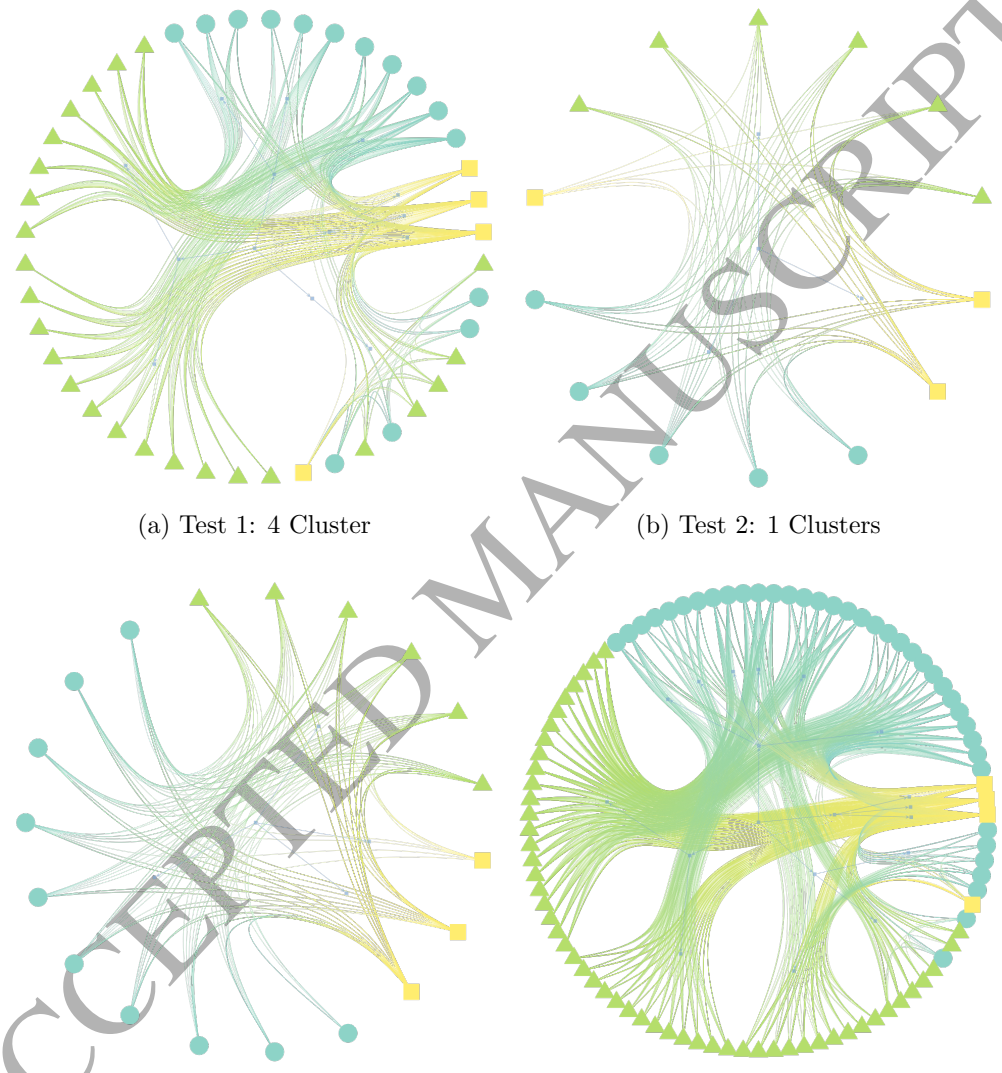
Test No	λ_i	ϵ_p	ϵ_n
1	(5,3,5)	0.4	0.08
2	(5,3,5)	0.5	0.05
3	(10,3,5)	0.4	0.08
4	(10,3,5)	0.25	0.1

Table 5: Sizes of clusters found in tests

Num	Test No 1	Test No 2	Test No 3	Test No 4
1	8×3×8	5×3×6	10×3×6	24×3×22
2	5×3×6			12×3×13
3	5×3×6			10×3×10
4	5×3×5			10×3×7

On Figure 7, circles refers to users, while triangles for keywords and squares for issues. Bundles of edges are expressing the diversity. Nodes with similar connections are adjacent. Coverage of clusters can be seen on Figure 8b. Average density of clusters in each test are plotted in Figure 8c.

A summary of observations are as follows:



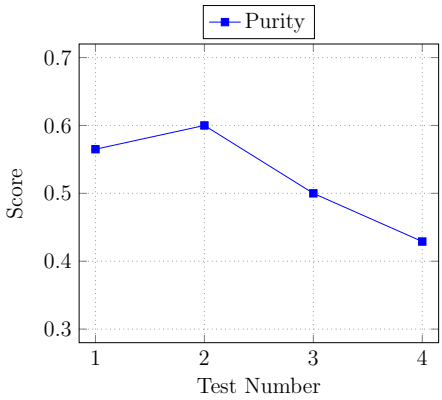
(a) Test 1: 4 Cluster

(b) Test 2: 1 Clusters

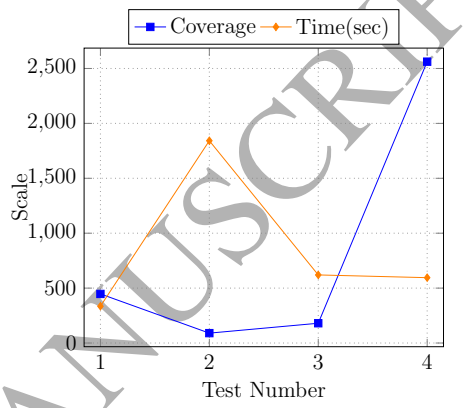
(c) Test 3: 1 Clusters

(d) Test 4: 4 Clusters

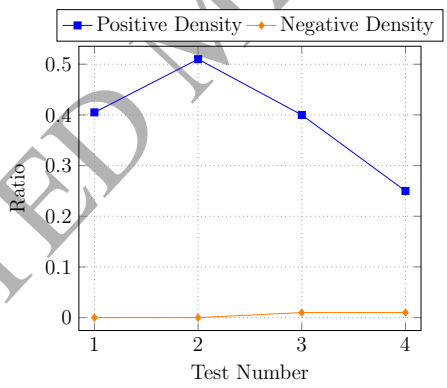
Figure 7: Test Results on Brexit Data



(a) Purity score of clusters



(b) Quality of clusters



(c) Plot of Average Density Ratios of Clusters in Test Cases

Figure 8: Result Plots of Test Cases

- Purity score shows the homogeneity of clusters. High purity score together with high coverage for results are highly desired. For 4 test cases, purity scores are plotted in Figure 8a. It can be seen that homogeneity of clusters is around 0.5. We observed that generally half of the tweets are from same division. Others are mixed.

- Cluster sizes are much smaller since we only constructed clusters with high relationships among the nodes of 3 dimensions. Most nodes are not included in these clusters. The largest cluster we obtain with the lowest density requirement, as 25% for minimum negative signed edges, and up to 10% for maximum positive signed edges, is with 24 users on 3 issues, using 22 different negative sentiment words (test number 4). For this cluster and other clusters, with these density constraints, we obtain the resulting user clusters containing members from different parties. The visual representation of clusters obtained from these experiments are given in Figure 7.

- As we tighten the density constraints we obtain much smaller and a lower number of clusters from the experiments. With high density constraint and/or larger minimum size constraint, it is not possible to obtain many clusters. Therefore, we obtain only a single cluster from tests number 2 and 3. As these constraints are slightly relaxed, it became possible to generate more clusters, as it can be seen from test number 1. Moreover, we clearly see that 3 issues stand above almost all other issues, namely "EU", "tax", and "Brexit". Since our method allows node sharing among clusters, we see that when more than one cluster is obtained from an experiment, either all, or at least two out of these three issues are in the issue dimension of these clusters. We also observe that users are distributed to different clusters due to sentiment words they choose to use as they express their feelings against these issues.

- In the experiments with higher negative density requirements, the users are mostly from Labour and Conservative parties. Even though these users are from different parties, when we look at their tweets we see they express their negative feelings with similar sentiment words against common issues. Since the number of users using common sentiment words are small, the cluster sizes turn out to be small as well.

The details of all the clusters generated by all of the above discussed experiments are given in Appendix A.

In addition to these general observations, we also investigated the details of the clusters formed by our approach. Consider the first clusters generated from the first experiment, which contains 8 politicians, 3 issues, and 8 keywords. In order to understand this cluster, we need to look into the tweets written by these politicians on these issues including these keywords, which are given in Appendix B.

When we look these tweets we see the following:

- The 8 politicians are distributed to 3 parties as 4 UKIP members, 3 Conservatives, and 1 Labour. 7 out of these 8 have strong feelings against the EU and are for the Leave campaign.

- The three issues, namely, "EU", "tax", and "Brexit", are also very common almost for all clusters formed in our experiments. This is not surprising since these tweets are collected just before Brexit referendum. As it can also be seen from Table 3, these are the top three issues of the collected tweets. The issues with smaller counts did not have enough tweets to form clusters with required densities.

- As it can also be seen from Appendix B these tweets include the same set of negative sentiment words, such as "crisis", "risk", "bad", "worse", etc. Since these 8 politicians have tweets on the same set of issues and they use the same sentiment words, our method clustered the politicians, issues, and the sentiment words as three clusters of these three dimensions, respectively.

- When the tweets of a Labour member has been investigated, it can easily be seen that he does not have the same political view as the other 7 politicians. Unfortunately, since we have not studied natural language semantic implications, we cannot capture such outlier behaviors. This politician has used the same negative sentiment words in his tweets on the same issues, just like the other seven, but, in a completely reverse way. Since currently we only syntactically analyze tweets, we cannot capture these kind of anomalies.

As it can be seen from the above observations, our approach had been quite successful in finding strong small clusters from large, very sparse data

sets. Considering the nature of politicians, each one trying to be different from others, even if they want to say exactly the same thing, they would probably try to be different by using different terminology, focusing on different aspects etc. That is why forming large clusters, even among the same party members, are unlikely. We have demonstrated that there are at least some small sets of politicians with very similar views.

In terms of running times, as expected the spectral clustering method is much faster than our method and determines the clusters in almost real-time. However, spectral clustering does not solve exactly the same problem. In our spectral clustering experiments we have ignored negative signed edges and all there dimensions of clusters are distinct from each other, but, as we have mentioned finding overlapping clusters with comon objects in some of their dimensions is the main novality of our problem. Moreover, since this clustering problem does not require a real-time response, and the quality of the clusters obtained is more important, we aimed to improve the quailty while sacrificing execution time. For the above mentioned experiments, our method took less than an hour to complete.

5. Conclusions

In this paper, a new version of co-clustering problem is defined. In this problem, from a given signed 3-partite graph, clusters with high positive labeled edge, and low negative labeled edge density are determined which potentially can have common nodes. We have proposed a greedy heuristic solution for this problem. The effectiveness of our method has been shown using both synthetic and real data sets. Similar to many data mining applications, the success of our proposed method also depends on the adjustments of several parameters. Thus, apriori knowledge about the structure of the data set, such as its density, dimension sizes etc., are needed in adjusting these parameters properly.

In order to evaluate the quality of the clusters generated from our real-world experiment, we mainly used internal evaluation metrics. As a future work, we plan to use an external evaluation method by obtaining comments from the domain experts, such as political scientists, about the clusters generated.

Furthermore, we are also planning to expand our Brexit experiments using more tweets in order to increase the density of input graphs. Even with relatively small experiments that we have presented in this paper, we were

able to obtain very encouraging results by obtaining highly dense clusters. As the input graph gets denser, we are expecting to produce much larger and more meaningful clusters. Moreover, we also plan to use different kinds of social network data sets with 3 dimensions and signed relationships, such as tagged (as like and unlike) images, comments on movies etc. in order to apply our method.

6. Acknowledgments

This research was supported partially by USAF Grant FA9550-15-1-0004.

References

Angiulli, F., Pizzuti, C., 2005. Gene expression biclustering using random walk strategies. In: International Conference on Data Warehousing and Knowledge Discovery. Springer, pp. 509–519.

Bader, B. W., Kolda, T. G., 2007. Efficient matlab computations with sparse and factored tensors. SIAM Journal on Scientific Computing 30 (1), 205–231.

Blondel, V. D., Guillaume, J.-L., Lambiotte, R., Lefebvre, E., 2008. Fast unfolding of communities in large networks. Journal of statistical mechanics: theory and experiment 2008 (10), P10008.

Dawande, M., Keskinocak, P., Swaminathan, J. M., Tayur, S., 2001. On bipartite and multipartite clique problems. Journal of Algorithms 41 (2), 388–403.

Dhillon, I. S., 2001. Co-clustering documents and words using bipartite spectral graph partitioning. In: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp. 269–274.

Gao, B., Liu, T.-Y., Zheng, X., Cheng, Q.-S., Ma, W.-Y., 2005. Consistent bipartite graph co-partitioning for star-structured high-order heterogeneous data co-clustering. In: Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining. ACM, pp. 41–50.

Giannakidou, E., Koutsonikola, V., Vakali, A., Kompatsiaris, Y., 2008. Co-clustering tags and social data sources. In: Web-Age Information Management, 2008. WAIM'08. The Ninth International Conference on. IEEE, pp. 317–324.

Gokalp, S., Temkit, M., Davulcu, H., Toroslu, I. H., 2013. Partitioning and scaling signed bipartite graphs for polarized political blogosphere. In: Social Computing (SocialCom), 2013 International Conference on. IEEE, pp. 168–173.

Gregory, S., 2010. Finding overlapping communities in networks by label propagation. New Journal of Physics 12 (10), 103018.

Hobolt, S. B., 2016. The brexit vote: a divided nation, a divided continent. Journal of European Public Policy 23 (9), 1259–1277.

Ignatov, D. I., Gnatyshak, D. V., Kuznetsov, S. O., Mirkin, B. G., Oct 2015. Triadic formal concept analysis and triclustering: searching for optimal patterns. Machine Learning 101 (1), 271–302.
URL https://doi.org/10.1007/s10994-015-5487-y

Kolda, T. G., Sun, J., 2008. Scalable tensor decompositions for multi-aspect data mining. In: Data Mining, 2008. ICDM'08. Eighth IEEE International Conference on. IEEE, pp. 363–372.

Lancichinetti, A., Fortunato, S., Kertész, J., 2009. Detecting the overlapping and hierarchical community structure in complex networks. New Journal of Physics 11 (3), 033015.

Li, W., Xie, J., Xin, M., Mo, J., 09 2017. An overlapping network community partition algorithm based on semi-supervised matrix factorization and random walk.

Lin, Y.-R., Sun, J., Castro, P., Konuru, R., Sundaram, H., Kelliher, A., 2009. Metafac: community discovery via relational hypergraph factorization. In: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, pp. 527–536.

Liu, X., Murata, T., 2010. Detecting communities in tripartite hypergraphs. arXiv preprint arXiv:1011.1043.

Long, B., Zhang, Z. M., Wu, X., Yu, P. S., 2006. Spectral clustering for multi-type relational data. In: Proceedings of the 23rd international conference on Machine learning. ACM, pp. 585–592.

Lu, C., Chen, X., Park, E., 2009. Exploit the tripartite network of social tagging for web clustering. In: Proceedings of the 18th ACM conference on Information and knowledge management. ACM, pp. 1545–1548.

Manning, C. D., Raghavan, P., Schütze, H., 2008. Introduction to Information Retrieval. Cambridge University Press, New York, NY, USA, pp. 356–360.

Missaoui, R., Kwuida, L., 2011. Mining Triadic Association Rules from Ternary Relations. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 204–218.
URL https://doi.org/10.1007/978-3-642-20514-96_16

Ng, A. Y., Jordan, M. I., Weiss, Y., et al., 2001. On spectral clustering: Analysis and an algorithm. In: NIPS. Vol. 14. pp. 849–856.

Omeroglu, N. B., Toroslu, I. H., Gokalp, S., Davulcu, H., 2013. K-partitioning of signed or weighted bipartite graphs. In: Social Computing (SocialCom), 2013 International Conference on. IEEE, pp. 815–820.

Rhouma, D., Romdhane, L. B., 2014. An efficient algorithm for community mining with overlap in social networks. Expert Systems with Applications 41 (9), 4309–4321.

Rosvall, M., Bergstrom, C. T., 2008. Maps of random walks on complex networks reveal community structure. Proceedings of the National Academy of Sciences 105 (4), 1118–1123.

Zha, H., He, X., Ding, C., Simon, H., Gu, M., 2001. Bipartite graph partitioning and data clustering. In: Proceedings of the tenth international conference on Information and knowledge management. ACM, pp. 25–32.

Zhao, L., Zaki, M. J., 2005. Tricluster: an effective algorithm for mining coherent clusters in 3d microarray data. In: Proceedings of the 2005 ACM SIGMOD international conference on Management of data. ACM, pp. 694–705.

Zhou, L., Lü, K., Yang, P., Wang, L., Kong, B., 2015. An approach for overlapping and hierarchical community detection in social networks based on coalition formation game theory. Expert Systems with Applications 42 (24), 9634–9646.

Zhu, L., Galstyan, A., Cheng, J., Lerman, K., 2014. Tripartite graph clustering for dynamic sentiment analysis on social media. In: Proceedings of the 2014 ACM SIGMOD international conference on Management of data. ACM, pp. 1531–1542.

Appendix A. Tweets of one cluster

Table A.6: Tweets of 1^{st} cluster in Test 1

User	Issue	Keyword	Tweet
15157283 : conservative	EU	bad	RT @Vote_LeaveMedia: Professor David Blake gives 10 reasons why staying in the EU ranges from "pretty bad to very, very bad" for pensions h
		crisis	RT @Telegraph: Leaving the EU will halve net migration, boost pay and help to solve housing crisis, according to a new study https://t.co/a
		risk	RT @TelegraphNews: Theresa May warns Tory Brexit rebels that they risk 'incentivising' EU to offer UK a bad deal https://t.co/Y3ekbmMraw ht
		risks	RT @bbckamal: Carney says there are greater risks for the continent than for the UK during the #Brexit process. EU wd face financial capaci
		threat	RT @suttonnick: Friday's Telegraph front page: Britain can fight terror threat better outside EU #tomorrowspaperstoday #euref https://t.co/
		threaten	RT @MustBeRead: If leaving the EU would really be so ruinous, asks @PaulGoodmanCH, why did @David_Cameron threaten to do so? https://t.co/Y
		worse	RT @Change_Britain: Jacob Rees Mogg MP reminds people how wrong economists were about the EU Referendum https://t.co/qnXwqi7ULa
	tax		
	#brexit	threat	RT @AndrewRosindell: The biggest threat to #Brexit now is complacency. We need to fight this campaign with every spare minute of the day. W
		worse	RT @BrexitTheMovie: EU protectionism raises the price of goods from the rest of the world and makes us all worse off #brexit https://t.co/6
107254637 : labor	EU	bad	bad : 746810051512393729: RT @TradingJeremy: EU warning that clearing houses should not be in London needs more public airing and explanation. Catastrophically bad f
		crisis	Greek financial crisis hasn't gone away: deal needs to be done by end of May or post-UK referendum on EU membership https://t.co/3sWHXK62LK
		risk	RT @pdacosta: #Brexit talks must be quick, City of London at risk of losing 'EU passport': @ECB's Villeroy https://t.co/XELL6JG7Jo
		risks	Couldn't be clearer from @bankofengland MPC today: leaving EU risks lower UK growth & investment, higher unemployment & prices #StrongerIn
	tax	worse	RT @resfoundation: PT worker on UC cld be 28 a wk worse off compared to tax credits. They'd need to work 10 hrs more to offset losses http
	#brexit	bad	IMF's Lagarde: consequences of #Brexit range from "pretty bad to very, very bad" #EURef https://t.co/PlARumMOaU
		risk	Lots emerged from @IMFNews report on UK economy: 1: echoes dire warnings on #Brexit risk to growth, prices, funding current account deficit
		risks	July @ECB monetary policy minutes out: spots stronger growth risks including #Brexit but no policy loosening for now https://t.co/LFcrMgBlTh
		worse	RT @EuropeElects: UK: Especially young voters are concerned that a #Brexit would leave them financially worse off (TNS poll). #EURef https:

User	Issue	Keyword	Tweet
478679663 :ukip	EU	bad	@FredLitten I understand this article bland claims UK entered eu at bad time and did well. The lag in UK simply caught up whilst EU declined
		crisis	Just posted: When will EU and the world recognise that #migration crisis is a @UN issue too https://t.co/KqIRT2WXIu https://t.co/0IpY2iVNXR
		risks	Breaking: Europol reporting that 800,000 migrants in Libya alone waiting to come to EU. Huge security risks & we have no control of borders. (+)
		threat	Read: My article for @heatstreet. The threat to European Peace is the EU itself NOT #Brexit. Hope will beat fear. https://t.co/EsDtqYVNVy
		threaten	Don't be scared and threatened of the EU powerbrokers. The power and size of our market will do the talking.
		worse	Mr Cameron, EU's expansionist foreign policy agenda, provoking Russia has made Europe less safe. Will only get worse as EU grows.
		wrong	Britain will do well outside of the EU: Project fear is wrong says top economist Roger Bottle https://t.co/6HXokYgLj1
	tax	risks	For the City of London, the risks of a regulatory body based in Frankfurt & EU financial transaction tax are far greater than Brexit.
	#brexit	threat	Must Read: EU's own Frontex force recognises terrorist porous borders threat https://t.co/aJOSqD8M3w #Brexit https://t.co/MbNk26VESp
		threaten	Great morning #Brexit bill passed but the Lords threaten to demolish it. Well here's brilliant @SuellaFernandes dem https://t.co/5HGoTgKzCO
		wrong	Major wrong on what #Brexit is about. Brexit is not about isolationism, it's about globalism and mutual cooperation.
1668992125 : ukip	EU	bad	RT @Nigel_Farage: I don't just think the EU has been bad for Britain, I think it's been disastrous for the whole of Europe. https://t.co/yi...
		crisis	RT @V_of_Europe: Juncker: No Matter How Bad Migrant Crisis, Terrorism Gets, We'll Never Give Up Open Borders https://t.co/EdxqwxV67i https:...
		risks	RT @hermannkelly: Farage: Staying In EU Risks More Cologne -like Sex Attacks .@UKLabour @MumsnetTowers #EURef https://t.co/DRqY80eGoi http...
		threat	RT @BBCRealityCheck: Criminal record not enough to reject EU citizens, must pose current threat https://t.co/B4tnV6nK7N #BBCDebate https://...
		threaten	RT @MikeHookemMEP: Blundering Corbyn vows to veto TTIP trade deal threatening NHS - but EU rules say he CAN'T https://t.co/6hztGhlGFh
	tax	wrong	RT @richardcalhoun: Petition: Abolish Inheritance Tax. It is wrong to tax assets that have already been taxed please RT https://t.co/7wJprH...
	#brexit	bad	Remember the 'good day to bury bad news'? Well every I'll in the world now due to #Brexit ... Apparently! https://t.co/8678egiedB
		crisis	RT @LeaveEUOfficial: Another Greek crisis is coming. The Eurocrats are trying to hush it up before #Brexit vote: https://t.co/rHZG5t6c0A ht...
121171051 : ukip	EU	crisis	RT @KulganofCrydee: EU migrant crisis 'is colossal': British borders face threat from terrorists and smugglers https://t.co/3PaoM2zKJK
		risk	RT @oflynnmep: What does Mr Carney think about risk of UK being in EU at next €zone crisis? Predecessor Lord King thinks €will lurch from...
		risks	RT @oflynnmep: Janet Daley right to wish Leave would emphasise more the economic risks of remaining in the EU. #VoteLeave
		threat	RT @KulganofCrydee: EU migrant crisis 'is colossal': British borders face threat from terrorists and smugglers https://t.co/3PaoM2zKJK
		threaten	RT @ConHome: If leaving the EU would really be so ruinous, why did @David_Cameron threaten to do so only recently? https://t.co/MCtydB0w1N
		worse	RT @andrealeadsom: EU is making the migrant crisis so much worse, says Defence Minister @PennyMordauntMP https://t.co/pVWlQ1nZRX @vote_leav...
		wrong	RT @montie: Tim Farron wrong. This isn't about Little Britain but about Little Europe. The EU is a 28 nation logrowth zone that is obsesse...
	tax	threat	RT @terencehooson: Chancellor plotting 'punishment' Budget with threat to hike income tax https://t.co/iMldSIkexm via @MailOnline

User	Issue	Keyword	Tweet
	#brexit	bad	RT @Charlton_UKIP: #BREXIT not so bad: UK employment hits record high: fundamentals of the British economy are strong https://t.co/r81UEb4E...
		risk	RT @DavidJo52951945: With open borders to 500m & ISIS terrorists flooding the EU,the EU is a risk to UK security.We're wide open #Brexit ht...
		risks	RT @minefornothing: The Spanish PM is warning Britain about the risks of leaving the EU. Youth unemployment in Spain is 50%!!!! #Brexit
		threat	Fury at PMs EU pensions threat: Vindictive PM tries Brexit blackmail https://t.co/hgp7etDuIC Shameful behaviour have your say #Brexit
		wrong	@Renegade_Inc Brit PM wrong Brits who support #Brexit put their people and country first!
14758838 : conservative	EU	bad	Leaving the EU & copying Canada's trade deal would be a bad deal for Britain. Here's why: https://t.co/51QRQ15FcZ https://t.co/APqkRlctRo
		risk	Manufacturing jobs are at risk if we leave the EU. Here's why: https://t.co/BwNDgkE4io #StrongerIN https://t.co/5f69NeJgUu
		risks	RT @David_Cameron: The IMF is right - leaving the EU would pose major risks for the UK economy. We are stronger, safer and better off in th...
		threat	Suggestion UK's vital intelligence 'five-eyes' relationship is under any sort of threat from the EU is wrong. Membership makes us safer.
		worse	RT @George_Osborne: Britain will be worse off by over 6% of GDP, to the tune of £4,300 per household if we vote to leave the EU on June 23
		wrong	Suggestion UK's vital intelligence 'five-eyes' relationship is under any sort of threat from the EU is wrong. Membership makes us safer.
	#brexit	risk	Years of "gruelling negotiations" would follow a #Brexit vote - hurting jobs and investment. Its not worth the risk. https://t.co/VbDhj63NK9
		risks	
		threat	New figures show that the threat of a #Brexit is hitting business investment https://t.co/GWRGDR36l6 @ConservativesIN
		worse	Comprehensive @hmtreasury report shows #Brexit would make British families £4,300 worse off https://t.co/zNUQBfCzEk https://t.co/3wFfkx2ALr
118984824 : conservative	EU	bad	RT @montie: if things are so bad in Norway, outside of the EU, why do 18% want to join but 70% want to stay out? #marr https://t.co/ILoXjRS...
		crisis	@grahamstuart You mean two richest countries in Europe outside the EU? Remain have to explain risk of Greek default & migrant/Euro crisis
		risk	So just today: Heseltine says we'll be forced to join Euro, Turkey opening EU negotiations, Juncker says no new deal for UK.Why risk Remain?
		risks	RT @NadineDorriesMP: Time for Cameron and Osborne to debate live with Gove and Johnson the risks of staying in the EU People need to know!...
		threat	RT @suttonnick: Monday's Daily Express: Fury at PM's EU pension threat #tomorrowspaperstoday #bbcpapers https://t.co/wCOserkLPa
		threaten	RT @Vote_LeaveMedia: If leaving the EU would really be so ruinous, why did Cameron threaten to do so only recently? https://t.co/omOhNWZkwP...
		worse	RT @vote_leave: .@wdjstraw just admitted live on #r4today that 'we will be worse off if we stay in the EU'
		wrong	RT @David_Cameron: The Leave campaign is wrong to say there'll be a 2nd referendum if we vote to remain in the EU. This is a referendum and...
	tax	risk	RT @MrRBourne: So following Cameron logic, he'd have been willing to risk war to secure minor changes to tax credits.
	#brexit	risk	RT @BeeAHoney_: Hey @David_Cameron please explain 'in detail': what changed from Nov 2015, to NOW? YOU implied #BREXIT not a RISK?! https:/...
		threat	So John Major's scaremongering over "threat" to peace in Northern Ireland of #Brexit is rejected by a whopping 61%-21% in IpsosMori poll
		wrong	RT @ajcdeane: Look at this rogues' gallery. Wrong then, wrong now. #Brexit https://t.co/WWFzERiUrc

User	Issue	Keyword	Tweet
2673995912 : ukip	EU	crisis	Germany's largest bank's profits drop 98% as EU banking crisis spreads https://t.co/TSo2biDhAc
		risk	Biggest canard of Remainians is staying in EU is risk free. In fact, it's a massive gamble we escape federalist tide https://t.co/CwMBc8MKHn
		threat	RT @UKIP: "Despite clear evidence the EU is under threat from the migration crisis, our PM will not secure our borders" @DianeJamesMEP #UKI...
		threaten	Here's the deal. EU gives Turkey €3B to house refugees and Erdogan threatens to send millions more migrants https://t.co/HnjuI1U2q5 #Cushty!
	#brexit	bad	If things are going to be as bad after #BREXIT as the #scaremongering #Remainians suggest, surely we will need to be even more creative!
		crisis	If you would like to know what is imperiling the global economy its not #Brexit it is the European banking crisis https://t.co/g5oXXtrFsX
		risks	Another example of risks we face if #Remain in a federalist EU. Only way to make lawmakers accountable #Brexit https://t.co/b791dDCC6u
		wrong	Cameron's #ProjectFear speech Straw Man 3: #Leave equals uncertainty. Wrong. Staying in EU SuperState means Decades of uncertainty #Brexit

Table A.7: Clusters of Test 1

Cluster No	Users	Issues	Keywords
1	conservative	eu	bad
	labour	tax	crisis
	ukip	#brexit	risk
	ukip		risks
	ukip		threat
	conservative		threaten
	conservative		worse
	ukip		wrong
2	ukip	eu	desperate
	conservative	tax	failing
	ukip	#brexit	fails
	ukip		fear
	conservative		sorry
			threatening
3	labour	eu	fears
	ukip	tax	risky
	labour	#brexit	rival
	ukip		shock
	conservative		threats
			worried
4	labour	nhs	benefit
	labour	eu	top
	labour	tax	attacks
	labour		lose
	ukip		worst