Inf Retrieval J (2018) 21:273–306
https://doi.org/10.1007/s10791-017-9324-8

# Clustering small-sized collections of short texts

**Lili Kotlerman**[1] · **Ido Dagan**[1] · **Oren Kurland**[2]

Received: 30 October 2016 / Accepted: 21 November 2017 / Published online: 30 November 2017

**Abstract** The need to cluster small text corpora composed of a few hundreds of short texts rises in various applications; e.g., clustering top-retrieved documents based on their snippets. This clustering task is challenging due to the vocabulary mismatch between short texts and the insufficient corpus-based statistics (e.g., term co-occurrence statistics) due to the corpus size. We address this clustering challenge using a framework that utilizes a set of external knowledge resources that provide information about term relations. Specifically, we use information induced from the resources to estimate similarity between terms and produce term clusters. We also utilize the resources to expand the vocabulary used in the given corpus and thus enhance term clustering. We then project the texts in the corpus onto the term clusters to cluster the texts. We evaluate various instantiations of the proposed framework by varying the term clustering method used, the approach of projecting the texts onto the term clusters, and the way of applying external knowledge resources. Extensive empirical evaluation demonstrates the merits of our approach with respect to applying clustering algorithms directly on the text corpus, and using state-of-the-art co-clustering and topic modeling methods.

**Keywords** Clustering · Clustering short texts · Short text similarities

## 1 Introduction

There are various applications that require clustering of small text collections composed of a few hundreds of short texts. Typical examples include the clustering of transcripts of calls in call centers so as to analyze user interactions (Kotlerman et al. 2015b), clustering

✉ Lili Kotlerman
lili.dav@gmail.com

[1] Department of Computer Science, Bar-Ilan University, Ramat Gan, Israel

[2] Faculty of Industrial Engineering and Management, Technion-Israel Institute of Technology, Haifa, Israel

of incoming stream of short news articles for topic detection and tracking or summarization (Allan et al. 1998; Aslam et al. 2014), and clustering of top-retrieved documents based on their snippets, either for improving browsing (Hearst et al. 1995; Zamir and Etzioni 1998) or for automatic cluster-based re-ranking (Kurland 2009; Liu and Croft 2004).

There are two main challenges in clustering a relatively small corpus of short texts. First, estimating inter-text similarities is a difficult task due to insufficient information in the texts (Metzler et al. 2007)—i.e., the vocabulary mismatch problem. For example, the two tweets "*Whats wrong.. charging $$ for checking a/c*" and "*Now they want a monthly fee!*" do not share any single term, although they might discuss the same topic—monthly fee for checking accounts. Specifically, although the terms "*fee*" and "*charging*" are semantically related, this relation is not captured by surface-level similarity estimates. The second challenge rises from the fact that the corpus itself is small. Thus, relying on corpus-based statistics, such as term co-occurrence information, so as to improve inter-text similarity estimates [e.g., using translation models (Berger and Lafferty 1999; Karimzadehgan and Zhai 2010)], and more generally, perform spectral analysis [e.g., topic analysis (Blei et al. 2003)], can be somewhat ineffective.

We address the challenge of clustering a small corpus of short texts using a term-based clustering framework. As a first step we cluster the vocabulary used in the texts in the given corpus using information on term relatedness induced from a set of lexical resources. There are various lexical resources that model different types of inter-term relations; e.g., WordNet (Fellbaum 1998), CatVar (Habash and Dorr 2003), WikiRules! (Shnarch et al. 2009) and distributional similarity resources (Kotlerman et al. 2010; Lin 1998; Mikolov et al. 2013). Information induced from these relations also serves for an optional enrichment of the vocabulary with related terms. Then, texts are clustered by projecting them over the term clusters. Thus, our approach helps to address the vocabulary mismatch between short texts, and the insufficient information in the given text corpus, by utilizing available external knowledge for clustering the term space as a basis for clustering the texts.

Using an extensive array of experiments performed over three datasets, we study the effectiveness of our proposed clustering framework. We instantiate the framework by varying different aspects such as the term clustering algorithm used, the approach for projecting the given text corpus over the term clusters, and the way external knowledge resources are utilized.

Empirical evaluation demonstrates the merits of our approach. The performance substantially transcends that of applying clustering algorithms directly on the given text corpus, applying topic analysis of the corpus (Blei et al. 2003), and applying co-clustering methods (Dhillon et al. 2003) which iteratively cluster texts and their terms.

This paper extends our original short conference paper (Kotlerman et al. 2012a). The conceptual framework we present here, of using external knowledge resources for term clustering, and projecting the texts in the given corpus over the term clusters so as to cluster the texts, was proposed in our initial work (Kotlerman et al. 2012a). Yet, in our initial work we used a single instantiation of the framework. Here we study various additional instantiations of the framework and compare them with state-of-the-art topic modeling approaches (Blei et al. 2003) and co-clustering methods (Dhillon et al. 2003), which was not the case in our initial work. Furthermore, in this paper we experiment with term weighting schemes which were not considered in our initial work, both for direct clustering of texts and for projecting the text corpus over the term clusters; one such approach is shown to be more effective than those we applied in our initial work. While the evaluation in our initial work (Kotlerman et al. 2012a) was performed with two datasets, one of which was proprietary, here the evaluation is performed using three datasets that we make publicly available.

The main contributions of the paper are:

1. We propose a framework for clustering small collections of short texts. The framework allows to leverage rich and diverse external resources of semantic knowledge, which are commonly overlooked by work on traditional clustering methods.
2. We present three real-life datasets, created in collaboration with industrial partners.
3. We report an extensive evaluation of our proposed framework. We compare our approach with state-of-the-art methods.
4. We present error analysis, as well as exploratory analysis, which sheds light on our approach and the specific challenges in the clustering settings we address.

## 2 Related work

There are various approaches to measuring inter-text similarities that address the vocabulary mismatch between (short) texts. For example, the texts could be used as queries in a search engine and the similarity between the retrieved lists can then be utilized (Metzler et al. 2007; Sahami and Heilman 2006). Using word embeddings can help to bridge lexical gaps (Boom et al. 2016; Kenter and De Rijke 2015; Severyn and Moschitti 2015). Translation models (Berger and Lafferty 1999; Karimzadehgan and Zhai 2010) can also be used; however, in our setting, wherein the corpus is small, translation probabilities cannot be estimated reliably. Non-textual information such as hashtags in tweets was also used for clustering (Tsur et al. 2013), but such information is not available for general texts.

In contrast to the methods described above, we do not induce a direct inter-text similarity estimate. Rather, we create term clusters using external knowledge resources that provide information about inter-term relations, and project the given text corpus over these term clusters to cluster the texts. In Sect. 4 we show that our approach outperforms a clustering method that uses word embeddings. Furthermore, we note that various inter-text similarity measures, and their integration (Kenter and De Rijke 2015; Metzler et al. 2007; Raiber et al. 2015), can potentially be integrated with our approach of using term clusters to create clusters of texts. This interesting research venue is left for future work.

Using external knowledge resources, which provide information about inter-term relations, is a key aspect of our approach. Therefore, we now briefly describe some types of external knowledge resources. We then overview state-of-the-art clustering algorithms and methods for incorporating external knowledge in them.

### 2.1 Resources of external knowledge

We distinguish between two types of external knowledge sources which could be used for our task: external corpora and lexical resources.

*Large external corpora*, such as Reuters (Rose et al. 2002), UKWaC (Ferraresi et al. 2008), Wikipedia (Gabrilovich and Markovitch 2006) and the Web, can be used directly as resources of unstructured (or semi-structured) textual information. For example, as noted above, using a text as a query to a (Web) search engine can help in expanding the text representation (Metzler et al. 2007; Sahami and Heilman 2006). In a conceptually similar vein, the text could be represented using Wikipedia concepts (Gabrilovich and Markovitch

2006). Alternatively, such resources can serve as a basis for extracting structured knowledge. A case in point, various distributional similarity techniques (Kotlerman et al. 2010; Levy and Goldberg 2014; Lin 1998) and word embedding methods (Mikolov et al. 2013; Pennington et al. 2014) can be applied over corpora to induce semantic-relatedness relations between terms.

*Lexical resources* provide information about terms and semantic relations between them. These resources can be either hand-crafted or generated by some automatic techniques. For example, WordNet (Fellbaum 1998) allows to extract terms that are synonyms, antonyms and hyponyms of a given target term. Although WordNet is the main lexical resource commonly utilized for text clustering, we note that the range of available lexical resources of different types is quite wide and is constantly growing. For instance, CatVar (Habash and Dorr 2003) contains information about derivationally related word forms. WikiRules! (Shnarch et al. 2009) is a resource of about 8 million pairs of semantically related terms extracted from Wikipedia. Pivoted paraphrase resources, such as Meteor (Denkowski and Lavie 2010) and PPDB (Ganitkevitch et al. 2013), allow to extract paraphrases for a given term obtained from parallel corpora, etc.

Either induced from large corpora, or provided as is, lexical resources can be viewed as sets of term pairs. Specifically, if terms $u$ and $v$ are paired in the resource, they could be viewed as semantically related with a potential weight attesting to the strength of the relation. In our experiments we use WordNet and a lexical resource compiled from the UKWaC corpus (Ferraresi et al. 2008) using a distributional similarity technique (Kotlerman et al. 2010). Additional details are provided in Sect. 4.1.

## 2.2 Clustering algorithms

One characterization, among many, of clustering algorithms for texts which pertains to our work is whether the algorithm is applied directly to a term-based representation of the texts, or involves clustering in the term space.

### *2.2.1 Direct clustering of texts*

There are numerous clustering algorithms that can be applied directly to a term-based representation of a text. Examples of algorithms that are commonly used (e.g., Boros et al. 2001; Hotho et al. 2003; Naughton et al. 2006; Nomoto and Matsumoto 2001; Ye and Young 2006) include hierarchical agglomerative techniques (e.g., single-link and complete-link), divisive methods (e.g., bisecting K-means), and partitioning methods such as K-means and K-medoids, as well as graph-based algorithms (Aggarwal and Zhai 2012; Biemann 2006; Erkan and Radev 2004; Kaufman and Rousseeuw 1990; Steinbach et al. 2000; Ye and Young 2006). These algorithms utilize an inter-text similarity estimate which is often based on a *tf* or *tf-idf* vector-space representation of a text (Erkan and Radev 2004; Hotho et al. 2003; Hu et al. 2008, 2009; Sedding and Kazakov 2004).

To go beyond surface-level similarities, especially for corpora containing short texts, information induced from external knowledge resources can be used for expanding the representation of texts. A text could be augmented using a list of other texts (Metzler et al. 2007) or additional terms; e.g., WordNet synonyms and sometimes other semantically related terms such as hyponyms (Green 1999; Hotho et al. 2003; Sedding and Kazakov 2004; Shehata 2009). Assigning expansion terms with weights lower than those assigned to the original terms in the text is an important practice (Metzler et al. 2007) which we employ in our work as well.

We use a few of the clustering algorithms mentioned above in two different capacities. First, for clustering terms using some inter-term similarity measure. Second, as reference comparisons to our approach when applied directly to term-based representations of the texts. We then show, as noted earlier, that our approach which is based on term clustering outperforms the direct application of clustering algorithms to term-based representations of the texts.

### *2.2.2 Clustering texts using term clustering*

Our framework is based on clustering terms and using the term clusters to induce text clusters. As such, our framework could be viewed, in spirit, as applying a single iteration of *co-clustering*. Co-clustering algorithms [e.g., (Dhillon et al. 2003)], are based on iterative application of text clustering and term clustering, where texts are represented using terms they contain and terms are represented using the texts they appear in.[1] In contrast to our approach, external knowledge was not utilized in co-clustering algorithms; these solely rely on the given corpus although this is not a requirement; e.g., texts could be expanded using external knowledge resources and then co-clustering can be applied. We show that our approach outperforms a state-of-the-art co-clustering algorithm (Dhillon et al. 2003) applied to the texts without utilizing external knowledge, as well as applied to the texts expanded using knowledge resources (see Sect. 2.2.1). We note that, to the best of our knowledge, in these comparisons we are the first to apply co-clustering with expanded texts.

Topic modeling approaches such as pLSA (Hofmann 1999), LDA (Blei et al. 2003) and Pachinko allocation (Li and McCallum 2006), could be viewed as inducing simultaneously text and term clusters. As already noted, these methods are potentially less effective for clustering small corpora of short texts due to insufficient statistics about term co-occurrence in texts. To overcome this challenge, such methods can be applied over texts expanded with semantically related terms, which was not done before as far as we know. It is also possible to obtain a topic space from a large external corpus and use the induced topics for clustering the texts in the given small corpora (e.g., Phan et al. 2008). Using word embeddings to represent the texts is a somewhat conceptually reminiscent approach: a semantic vector space is induced from a large corpus and used to encode the terms; such encoding can then be used to encode texts (Mikolov et al. 2013). We demonstrate the merits of our proposed clustering approach with respect to applying LDA on texts in the given corpus—both directly and after expansion of the texts with related terms, as well as with respect to using LDA topics and word embeddings induced from a large external corpus to cluster the given corpus.

As noted above, in our original work (Kotlerman et al. 2012a) we presented the basic clustering framework presented here with a single instantiation. Later, Di Marco and Navigli (2013) experimented with a similar framework (although the technical details are different) for clustering and diversifying Web search results using document snippets. Here, we study many more instantiations of the framework, and study different considerations that affect these instantiations; e.g., while we use different types of clustering algorithms for terms, Di Marco and Navigli (2013) focused on graph-based approaches. In addition, we use external knowledge resources not used in their work, and compare with state-of-the-art co-clustering and topic modeling approaches which was not the case in Marco and Navigli (2013).

[1] We note that our approach of associating texts with term clusters is shown below to outperform in our setting a method which is commonly used in co-clustering algorithms.

### 2.2.3 Applications using term clustering

Our focus in this paper is on utilizing inter-term relations to induce term clusters and then use the term clusters to produce text clusters. We note that term clustering has also been used for other tasks and applications as a preliminary step: text classification (Baker and McCallum 1998), query expansion in ad hoc retrieval (Udupa et al. 2009) and interactive information retrieval (Tan et al. 2007).

## 3 Clustering via explicit term clusters

Our basic framework for using term clusters for clustering a collection of texts is depicted in Fig. 1. Below we describe a few alternatives of instantiating the framework and draw parallels between our approach and state-of-the-art clustering methods. In Sect. 3.1 we discuss the process of creating term clusters, and in Sect. 3.2 we detail our suggested approach of inducing text clusters based on term clustering.

### 3.1 Creating term clusters

We now turn to instantiate Step 1.2 from Fig. 1; that is, applying term clustering. We note that any term clustering approach can be used in our framework. In this section we describe the term clustering algorithms we applied in our experiments.

As stated above, we would like to utilize lexical resources to create clusters of terms in our term space. We note that creation of term clusters using lexical resources means that the term clustering is independent of the texts of a given collection, as opposed to the case of co-clustering. The only dependence is on the vocabulary which is being clustered.

As noted in Sect. 2.1, lexical resources, e.g. WordNet, can be viewed as sets of semantically-related term pairs, and as such they can provide information on similarity (distance) between terms. We consider two terms $v$ and $u$ related according to a resource $r$ if the resource contains either the pair ($v$, $u$) or the pair ($u$, $v$). For example, the terms (*apple*, *fruit*) will be considered related according to a resource constructed from WordNet hypernyms, since *fruit* is a hypernym of *apple*.

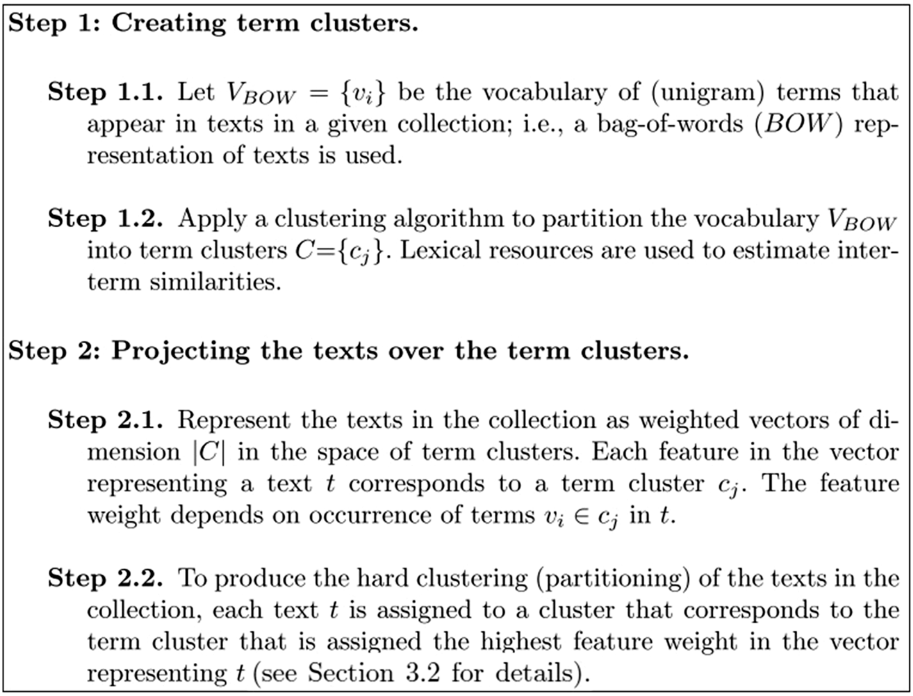
We create term clusters using lexical resources as described in Fig. 2. We set the similarity between terms $v$ and $u$ based on a set of lexical resources $R = \{r_i\}$ as follows:

$$sim(v,u) = sim(u,v)c_{ub} \cdot \frac{|r_i \in R : (v,u) \in r_i \vee (u,v) \in r_i|}{|R|}, \tag{1}$$

where the coefficient $c_{ub}$ is used to set the upper bound of the similarity values as will be explained in Sect. 4.1. The similarity is thus equal to the proportion of resources according to which the terms $u$ and $v$ are considered related in any direction regardless of whether $u$ is related to $v$ or the other way around; the similarity value is upper bounded by $c_{ub}$ which is a parameter. We use $c_{ub} \leq 1$; thus, the similarity values are in the [0, 1] interval.

Thus, applying the procedure in Fig. 2 to the vocabulary $V_{BOW}$ results in a clustering of the original terms, i.e. the terms that occur in the texts of the given text collection. The lexical resources are used in this case to provide information about semantic relatedness between the original terms.

In Fig. 3 we illustrate the algorithm from Fig. 2. At Step 1 of Fig. 2, an empty similarity matrix is created. Then, at Step 2 inter-term similarity scores are assigned. We visualize the



**Step 1: Creating term clusters.**

**Step 1.1.** Let $V_{BOW} = \{v_i\}$ be the vocabulary of (unigram) terms that appear in texts in a given collection; i.e., a bag-of-words ($BOW$) representation of texts is used.

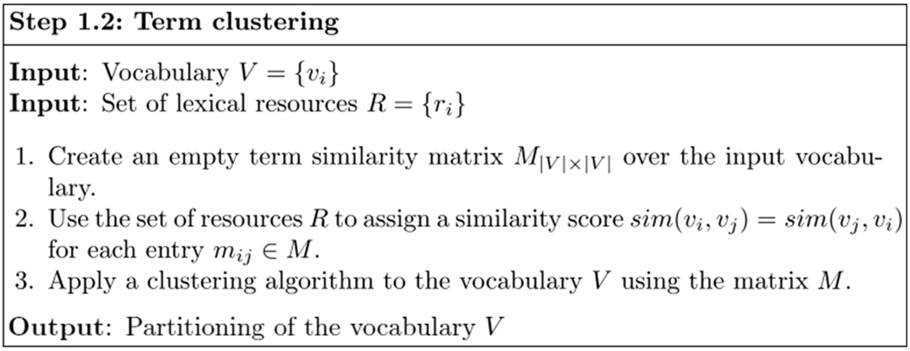
**Step 1.2.** Apply a clustering algorithm to partition the vocabulary $V_{BOW}$ into term clusters $C = \{c_j\}$. Lexical resources are used to estimate inter-term similarities.

**Step 2: Projecting the texts over the term clusters.**

**Step 2.1.** Represent the texts in the collection as weighted vectors of dimension $|C|$ in the space of term clusters. Each feature in the vector representing a text $t$ corresponds to a term cluster $c_j$. The feature weight depends on occurrence of terms $v_i \in c_j$ in $t$.

**Step 2.2.** To produce the hard clustering (partitioning) of the texts in the collection, each text $t$ is assigned to a cluster that corresponds to the term cluster that is assigned the highest feature weight in the vector representing $t$ (see Section 3.2 for details).

**Fig. 1** Text clustering via explicit term clusters: general framework



**Step 1.2: Term clustering**

**Input**: Vocabulary $V = \{v_i\}$
**Input**: Set of lexical resources $R = \{r_i\}$

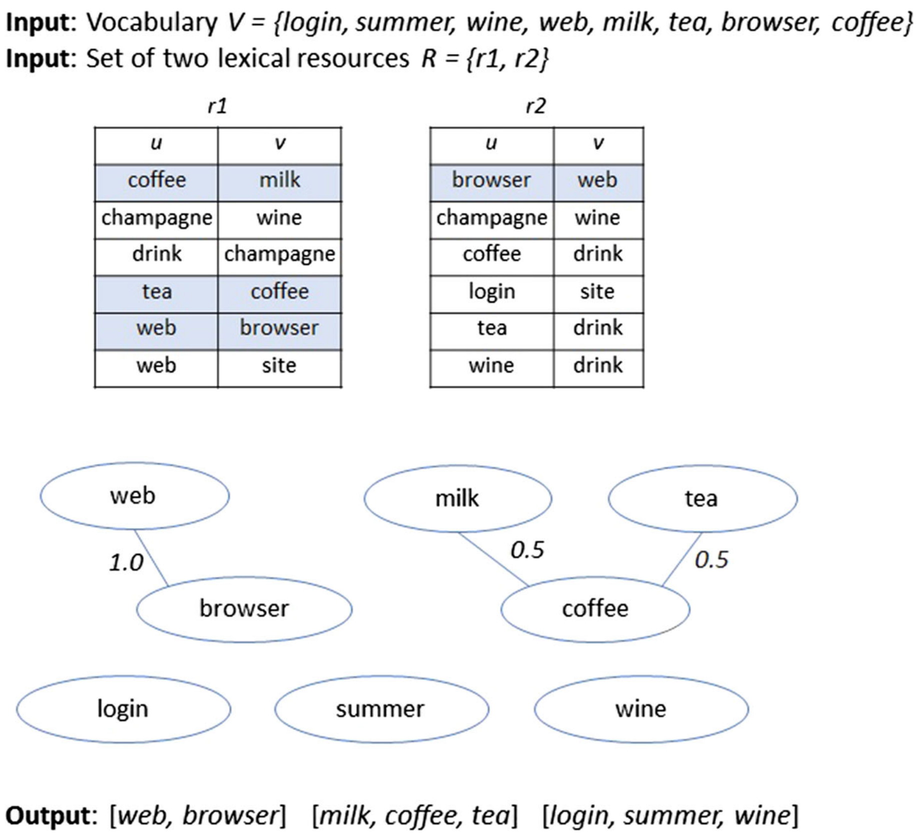
1. Create an empty term similarity matrix $M_{|V|\times|V|}$ over the input vocabulary.
2. Use the set of resources $R$ to assign a similarity score $sim(v_i, v_j) = sim(v_j, v_i)$ for each entry $m_{ij} \in M$.
3. Apply a clustering algorithm to the vocabulary $V$ using the matrix $M$.

**Output**: Partitioning of the vocabulary $V$

**Fig. 2** Term clustering procedure

resulting matrix as a graph with vocabulary terms as its nodes. The nodes are connected with edges, where the weight of an edge ($v$, $u$) is $sim(v, u) = sim(u, v)$ as set by Eq. 1 with $c_{ub} = 1$. The similarity $sim(web, browser) = 1$ since the terms are related according to both given resources. The similarity *sim*(*coffee*, *tea*) and *sim*(*milk*, *coffee*) are equal to 0.5 since each pair is present only in one resource out of two. The resulting partitioning depends on the exact clustering algorithm applied in Step 3 of Fig. 2. We provide one possible partitioning as an example.

In past work on clustering of texts (Sect. 2.2) lexical resources were used in a different role—as a source of lexical expansions. The resources were used prior to text clustering to



**Input**: Vocabulary *V = {login, summer, wine, web, milk, tea, browser, coffee}*

**Input**: Set of two lexical resources *R = {r1, r2}*

*r1*

| *u* | *v* |
|---|---|
| coffee | milk |
| champagne | wine |
| drink | champagne |
| tea | coffee |
| web | browser |
| web | site |

*r2*

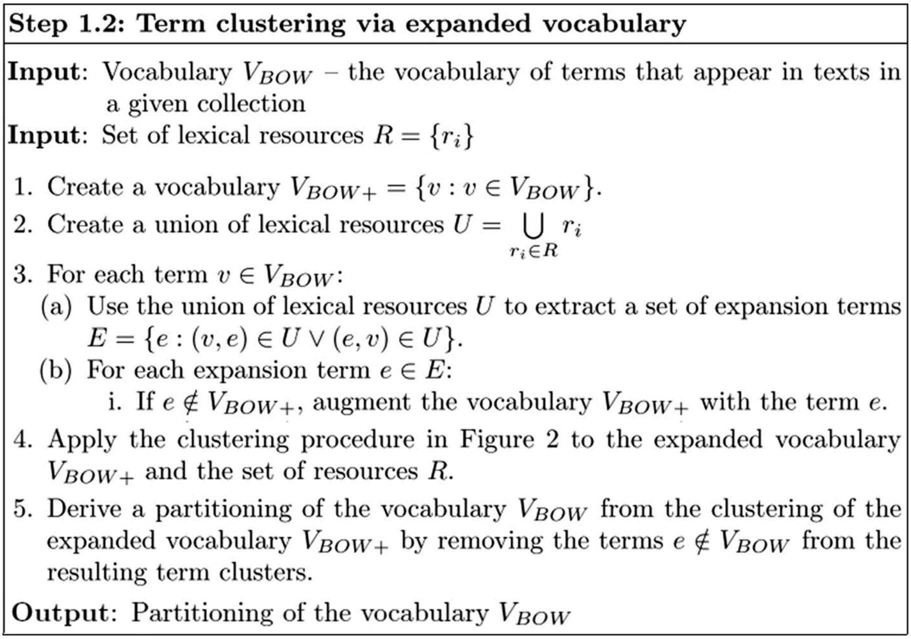
| *u* | *v* |
|---|---|
| browser | web |
| champagne | wine |
| coffee | drink |
| login | site |
| tea | drink |
| wine | drink |

**Output**: [*web, browser*] [*milk, coffee, tea*] [*login, summer, wine*]

**Fig. 3** Example of applying the term clustering procedure from Fig. 2. Term pairs (*u*, *v*) in which both *u* and *v* belong to the input vocabulary are highlighted in the lexical resources

expand the texts with semantically related terms. In our approach such expansion terms can be utilized as well. To this end, terms semantically related to those from the $V_{BOW}$ vocabulary are extracted from the lexical resources. Then, the vocabulary $V_{BOW}$ is augmented with the extracted terms to create an expanded vocabulary $V_{BOW+}$, which is further used as input for the term clustering procedure in Fig. 2. The rationale behind expanding the vocabulary is that clustering a larger set of terms is likely to be more stable and accurate. The application of term clustering to the expanded vocabulary is detailed in Fig. 4. In Sect. 4.2 we empirically investigate the influence of vocabulary expansion and show that it helps to improve the performance of our method.

In Fig. 5 we illustrate the algorithm from Fig. 4 using the same input vocabulary and lexical resources as in the example in Fig. 3. At Steps 1–3 of Fig. 4 an expanded vocabulary is created by augmenting the input vocabulary with related terms from the lexical resources. We present these terms as dotted nodes. Then, at Step 4, the algorithm in Fig. 2 is applied to the extended vocabulary. The algorithm returns a partitioning of the extended vocabulary, from which the expanding terms are removed to obtain the final output. In our example, the output of Step 4 can be as follows: [*web, browser, login, site*], [*milk, coffee, tea, wine, champagne, drink*], [*summer*]. Then, after removing the expansion terms, the final partitioning presented in Fig. 5 is obtained.

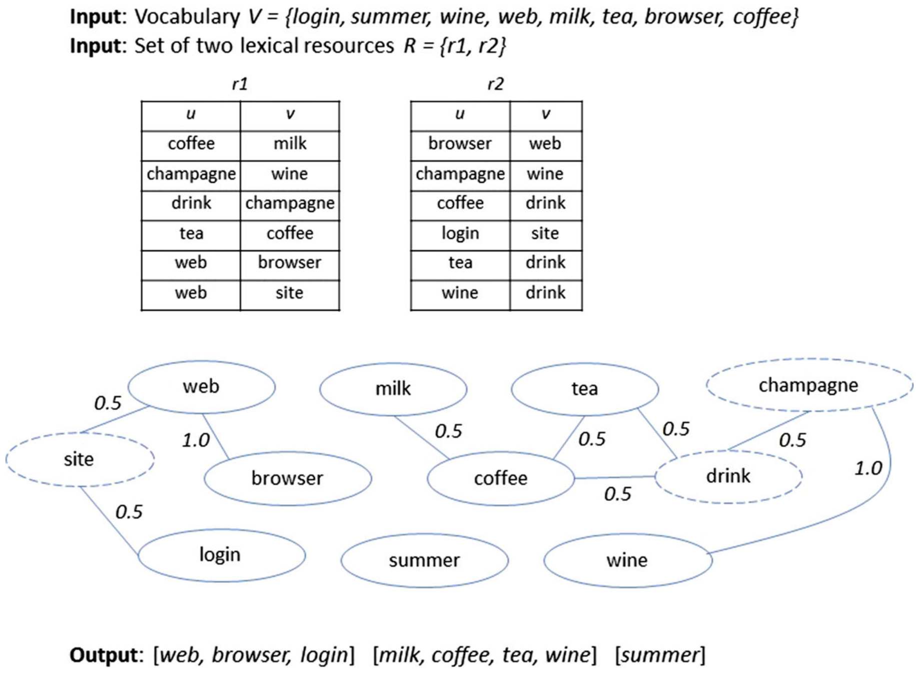


**Step 1.2: Term clustering via expanded vocabulary**

```
Input: Vocabulary $V_{BOW}$ – the vocabulary of terms that appear in texts in
       a given collection
Input: Set of lexical resources $R = \{r_i\}$

1. Create a vocabulary $V_{BOW+} = \{v : v \in V_{BOW}\}$.
2. Create a union of lexical resources $U = \bigcup_{r_i \in R} r_i$
3. For each term $v \in V_{BOW}$:
   (a) Use the union of lexical resources $U$ to extract a set of expansion terms
       $E = \{e : (v, e) \in U \vee (e, v) \in U\}$.
   (b) For each expansion term $e \in E$:
       i. If $e \notin V_{BOW+}$, augment the vocabulary $V_{BOW+}$ with the term $e$.
4. Apply the clustering procedure in Figure 2 to the expanded vocabulary
   $V_{BOW+}$ and the set of resources $R$.
5. Derive a partitioning of the vocabulary $V_{BOW}$ from the clustering of the
   expanded vocabulary $V_{BOW+}$ by removing the terms $e \notin V_{BOW}$ from the
   resulting term clusters.
Output: Partitioning of the vocabulary $V_{BOW}$
```

**Fig. 4** Term clustering with vocabulary expansion (alternative instantiation of Fig. 2)



**Input**: Vocabulary *V = {login, summer, wine, web, milk, tea, browser, coffee}*
**Input**: Set of two lexical resources *R = {r1, r2}*

*r1*

| *u* | *v* |
|---|---|
| coffee | milk |
| champagne | wine |
| drink | champagne |
| tea | coffee |
| web | browser |
| web | site |

*r2*

| *u* | *v* |
|---|---|
| browser | web |
| champagne | wine |
| coffee | drink |
| login | site |
| tea | drink |
| wine | drink |

**Output**: [*web, browser, login*] [*milk, coffee, tea, wine*] [*summer*]

**Fig. 5** Example of applying the term clustering procedure from Fig. 4. Terms added to the vocabulary $V_{BOW+}$ from the lexical resources are presented as dotted nodes.

To summarize, to instantiate Step 1.2 from Fig. 1, a term similarity matrix is created either for the original vocabulary $V_{BOW}$ or for the expanded vocabulary $V_{BOW+}$. Then, a clustering algorithm is applied using the matrix.

For proper comparison of our method with the baselines, in this work we used for term clustering algorithms that are commonly used for clustering texts. Specifically, we used algorithms which can be applied to a similarity matrix and do not require a vector representation of terms, namely: (1) Hierarchical Agglomerative Clustering using complete-link, (2) Partitioning Around Medoids (PAM) algorithm (Kaufman and Rousseeuw 1990) and (3) the K-medoids algorithm of Ye and Young (2006). In addition, we experimented with utilizing term clusters produced by co-clustering and topic modeling methods, as well as by a graph clustering algorithm.

The K-medoids algorithm of Ye and Young (2006) was originally suggested for clustering of short texts. This algorithm, henceforth referred to as *KMY*, is not widely known. However, it turned out to be one of the best-performing term clustering algorithms in our framework. Thus, below we first provide the details of this algorithm (Sect. 3.1.1) and then describe the other term clustering algorithms we used in our experiments (Sect. 3.1.2). We note that our goal is not to engage in excessive optimization of the term clustering algorithms used to instantiate our framework, but rather demonstrate the effectiveness of the framework using several commonly used algorithms.

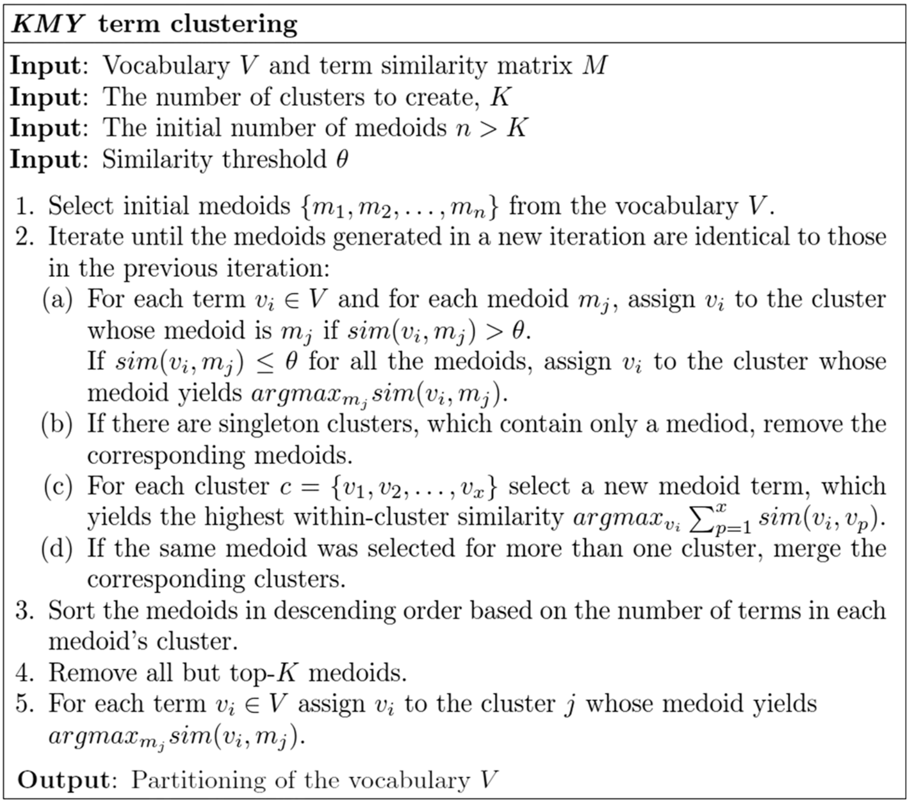
### *3.1.1 KMY algorithm for term clustering*

We used the KMY variant of the K-medoids algorithm (Ye and Young 2006) as one of the term clustering methods to instantiate our framework, in Step 3 of the term clustering procedure (Fig. 2). The algorithm is detailed in Fig. 6. The algorithm starts with more than $K$ medoids, each of which defines a single cluster. At every iteration, each term is assigned to all the clusters whose medoids are similar enough to the term, as defined by a threshold. We note that such assignment induces a soft clustering. When all the terms are assigned, new medoids are calculated, until convergence. Then, only the top-$K$ largest clusters are retained, where $K$ is the required number of clusters which is an input to the K-medoids algorithm, and the terms are re-assigned to a single cluster each, thus forming the final hard partitioning.

As explained above, we apply term clustering either to the original vocabulary $V_{BOW}$, which contains terms occurring in the texts of a given collection, or to the expanded vocabulary $V_{BOW+}$, augmented with semantically related terms of the terms $v_i \in V_{BOW}$. For both cases, in Step 1 in Fig. 6 we initially use each term from the original vocabulary $v_i \in V_{BOW}$ as a medoid. This gives each such term a “chance” to form a cluster. We set the threshold $\theta = 0$. If in Step 5 a term from the original vocabulary $v_i \in V_{BOW}$ is not assigned to any cluster due to zero similarity with all medoids, we perform new term clustering into $K - 1$ clusters. Then, an additional *miscellaneous* cluster is created to hold all the out-of-cluster terms from the original vocabulary $V_{BOW}$, thus generating a partitioning into $K$ term clusters. In Sect. 3.2 we explain how this *miscellaneous* cluster is treated when projecting the texts over term clusters.

In addition, we experimented with the following variations of the KMY algorithm:

- *Soft* term clustering: in Step 5 of the algorithm (Fig. 6) each term can be associated with multiple medoids rather than with the single one with the highest similarity as in Figure 6 (which results in *Hard* term clustering). Here, we use a *Soft* clustering scheme where each term is associated with all the clusters for which it has a non-zero similarity with their medoids.



```
KMY term clustering
Input: Vocabulary $V$ and term similarity matrix $M$
Input: The number of clusters to create, $K$
Input: The initial number of medoids $n > K$
Input: Similarity threshold $\theta$
1. Select initial medoids $\{m_1, m_2, \ldots, m_n\}$ from the vocabulary $V$.
2. Iterate until the medoids generated in a new iteration are identical to those
   in the previous iteration:
   (a) For each term $v_i \in V$ and for each medoid $m_j$, assign $v_i$ to the cluster
       whose medoid is $m_j$ if $sim(v_i, m_j) > \theta$.
       If $sim(v_i, m_j) \leq \theta$ for all the medoids, assign $v_i$ to the cluster whose
       medoid yields $argmax_{m_j} sim(v_i, m_j)$.
   (b) If there are singleton clusters, which contain only a mediod, remove the
       corresponding medoids.
   (c) For each cluster $c = \{v_1, v_2, \ldots, v_x\}$ select a new medoid term, which
       yields the highest within-cluster similarity $argmax_{v_i} \sum_{p=1}^{x} sim(v_i, v_p)$.
   (d) If the same medoid was selected for more than one cluster, merge the
       corresponding clusters.
3. Sort the medoids in descending order based on the number of terms in each
   medoid's cluster.
4. Remove all but top-$K$ medoids.
5. For each term $v_i \in V$ assign $v_i$ to the cluster $j$ whose medoid yields
   $argmax_{m_j} sim(v_i, m_j)$.
Output: Partitioning of the vocabulary $V$
```

**Fig. 6** KMY term clustering algorithm (Ye and Young 2006) of the K-medoids family

- $K_{auto}$: instead of removing the medoids to retain only the $K$ largest clusters in Step 4, all the medoids can be retained without pruning. We use $K_{auto}$ to denote this variant, since in this case the value of $K$ is not predefined but is rather automatically determined by the algorithm.

To summarize, to instantiate our framework we use the following variants of the KMY term clustering algorithm for both the original $V_{BOW}$ and the expanded $V_{BOW+}$ vocabulary: *Hard-K* (the variant presented in Fig. 6), *Hard-*$K_{auto}$, *Soft-K* and *Soft-*$K_{auto}$.

### *3.1.2 Additional methods for term clustering*

Below we list additional methods we employed for term clustering in our experiments.

The *Hierarchical agglomerative clustering with complete-link* and *Partitioning Around Medoids (PAM)* algorithms were applied over exactly the same input similarity matrices as that of the KMY algorithm detailed in Sect. 3.1.1.

The *Chinese Whispers graph clustering* algorithm of Biemann (2006),[2] which was reported in Biemann (2006) to outperform other algorithms for several Natural Language Processing tasks, such as acquisition of syntactic word classes and word sense

[2] Implementation available at https://marketplace.gephi.org/plugin/chinese-whispers-clustering/

disambiguation. The algorithm was applied for term clustering by Di Marco and Navigli (2013) (see Sect. 2.2.2) and resulted in performance on par with that of other graph clustering methods. We applied it over a term connectivity graph based on the matrix of either $V_{BOW}$ or $V_{BOW+}$ vocabulary, with edge weights assigned according to Eq. 1. The algorithm is parameter-free and it automatically determines the number of clusters to be produced. It can return a clustering where some of the input terms are not included. As was the case for KMY clustering (see Sect. 3.1.1), here we also created an additional *miscellaneous* cluster to hold all the out-of-cluster terms from the original vocabulary ($V_{BOW}$).

*Co-clustering* (Dhillon et al. 2003) and *LDA* (Blei et al. 2003). While these algorithms were applied in our experiments as baselines for the task of text clustering (see Sect. 4.1), we also used them as term-clustering algorithms to instantiate our framework.

We use the underlying term clusters generated by co-clustering in its final iteration. We used term clusters generated either when co-clustering the original texts of a given collection, or when co-clustering the texts augmented with semantically related terms (see Sect. 2.2).

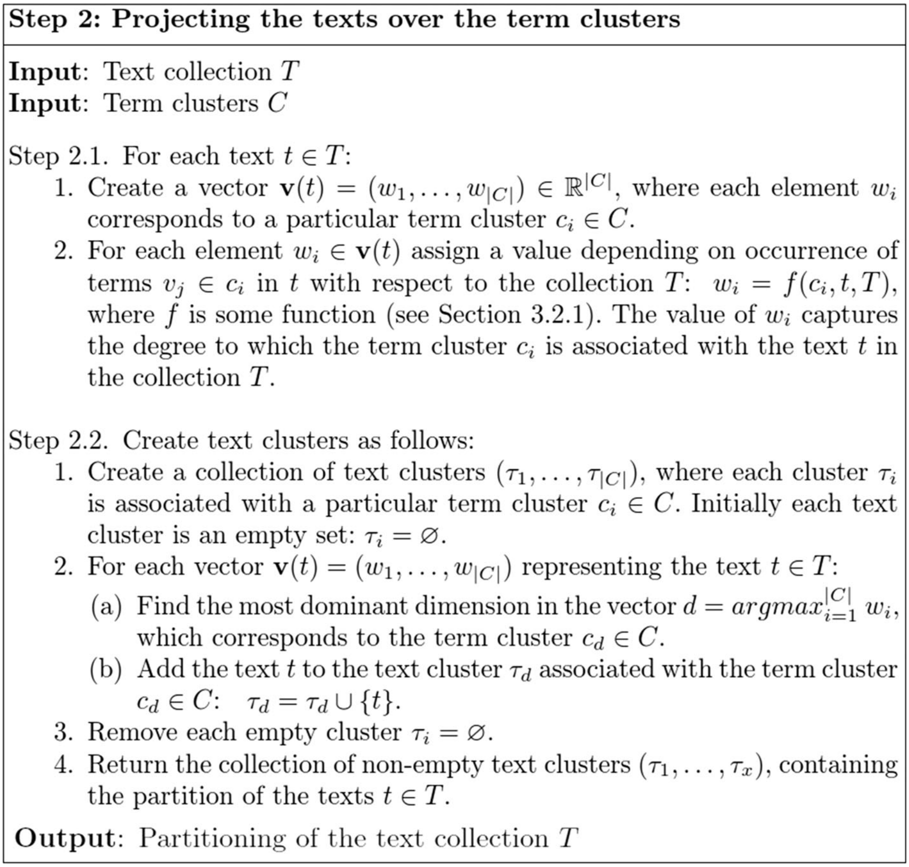
Similarly to utilizing term clusters generated by co-clustering, one can view LDA topics as soft term clusters. We thus experimented with using as term clusters the topics created by each of the following baselines: LDA trained over a given text collection with (1) original texts and (2) texts augmented with semantically related terms, as well as (3) LDA trained over the UKWaC corpus.

### 3.2 Projecting the texts over the term clusters

The last step required for instantiating our framework is projecting the texts over the term clusters so as to cluster the texts. Once the term clusters are generated, we represent the texts in the given corpus as weighted vectors in the term-cluster space (Step 2.1 of the framework in Fig. 1). Then we associate each text in the given collection with a cluster that corresponds to the term cluster which is the most dominant in the text's vector representation; that is, the term cluster whose corresponding feature in the vector has the highest weight (Step 2.2, Fig. 1). The procedure of projecting the texts over the term clusters is formalized in more detail in Fig. 7. In Sect. 3.2.1 we elaborate on the feature weighting schemes used to instantiate our framework.

If a term clustering includes a *miscellaneous* cluster $c_{misc} = \{v_i\}$ with terms from the original vocabulary $V_{BOW}$ (see Sect. 3.1.1), we split $c_{misc}$ and create a singleton term-cluster for each term $v_i$. In addition, we applied a simple heuristic splitting each term cluster to singletons if its size exceeded 20% of the original vocabulary ($V_{BOW}$), since we noticed that the term clustering algorithms used in our work tend to produce several large clusters with unwarranted similarities. In our experiments there were typically one or two such term clusters. The algorithms that produce a *miscellaneous* cluster did not produce very large clusters except for the *miscellaneous* cluster itself, whose size typically ranged from 20 to 40% of the original vocabulary. We emphasize again that for proper comparison of our method with the baselines, we used for term clustering algorithms that are commonly used for clustering texts, rather than using algorithms geared specifically for term clustering. Since clustering assumes that a cluster contains items similar to each other, we employ this heuristic to split *miscellaneous* clusters with out-of-class terms, as well as large clusters which have high potential to include unwarranted similarities.

Thus, there were often more than $K$ term-cluster features in our vectors. In such cases there is a chance that more than $K$ text clusters will be created as output by our method. In that event we do the following: (1) retain only the $K - 1$ largest clusters and (2) assign all



```
Step 2: Projecting the texts over the term clusters

Input: Text collection $T$
Input: Term clusters $C$

Step 2.1. For each text $t \in T$:
    1. Create a vector $\mathbf{v}(t) = (w_1, \ldots, w_{|C|}) \in \mathbb{R}^{|C|}$, where each element $w_i$ corresponds to a particular term cluster $c_i \in C$.
    2. For each element $w_i \in \mathbf{v}(t)$ assign a value depending on occurrence of terms $v_j \in c_i$ in $t$ with respect to the collection $T$: $w_i = f(c_i, t, T)$, where $f$ is some function (see Section 3.2.1). The value of $w_i$ captures the degree to which the term cluster $c_i$ is associated with the text $t$ in the collection $T$.

Step 2.2. Create text clusters as follows:
    1. Create a collection of text clusters $(\tau_1, \ldots, \tau_{|C|})$, where each cluster $\tau_i$ is associated with a particular term cluster $c_i \in C$. Initially each text cluster is an empty set: $\tau_i = \varnothing$.
    2. For each vector $\mathbf{v}(t) = (w_1, \ldots, w_{|C|})$ representing the text $t \in T$:
        (a) Find the most dominant dimension in the vector $d = argmax_{i=1}^{|C|} w_i$, which corresponds to the term cluster $c_d \in C$.
        (b) Add the text $t$ to the text cluster $\tau_d$ associated with the term cluster $c_d \in C$: $\tau_d = \tau_d \cup \{t\}$.
    3. Remove each empty cluster $\tau_i = \varnothing$.
    4. Return the collection of non-empty text clusters $(\tau_1, \ldots, \tau_x)$, containing the partition of the texts $t \in T$.

Output: Partitioning of the text collection $T$
```

**Fig. 7** The projection procedure

the texts not associated with any cluster to an additional *miscellaneous* cluster to produce $K$ clusters.

We note that our term-cluster-based vector representation of texts is conceptually similar to that used by co-clustering algorithms and LDA, which are essentially using simultaneously the term and text spaces. Co-clustering algorithms iteratively perform term clustering based on term co-occurrence in the text clusters that have been created in the previous iteration. The texts are then represented in the term-cluster space and re-clustered. LDA generates term clusters (topics) based on term co-occurrence in the clustered texts (or in an external corpus). Each text is then represented as a mixture of topics, which can be viewed as a weighted vector of term clusters.

In order to create text clusters we associate each text with a cluster that corresponds to the term cluster with the highest weight in the vector representing the text (Step 2.2, Figs. 1 and 7). This is similar to the practice of using LDA to induce hard text clusters, where each text is assigned to the topic with which it has the strongest association. An alternative approach is to apply a clustering algorithm over the texts represented as vectors in the term-cluster space. Such practice resembles in spirit a single iteration of co-clustering. We experimented with this approach by applying the Complete Link, PAM and

KMY algorithms to the term-clusters-based vectors. The resultant performance, reported in Sect. 4, was inferior to that of our suggested approach from above. The main reason is that short texts in the setting of our scope usually contain one or just few concrete terms indicating to which cluster a text should be assigned. We further discuss these findings in Sect. 4.3.

In Fig. 8 we illustrate our projection procedure with a toy example. The term clusters are presented to the right. On the left, the texts from the input collection are shown. Each text is accompanied by the clusters in its vector representation. The cluster with the highest weight is boldfaced. Arrows depict the association of texts with term clusters.

Thus, the main differences between our approach, co-clustering and LDA-based clustering are: (1) we create term clusters by utilizing various types of external knowledge rather than relying solely on term co-occurrence in the given collection or in an external corpus, and (2) unlike in LDA, term clusters in our framework are explicit, which allows for flexible selection of the weighting scheme used to derive term-clusters-based text representations as we show below.

### 3.2.1 Weighting schemes

Feature weighting is a key component of our method. We adapt the standard *term frequency* (*tf*) and *term frequency-inverse document frequency* (*tf-idf*) weighting schemes to create feature weights, as well as propose a novel *term frequency-document frequency* (*tf-df*) weighting approach, which we anticipate to be preferable in our setting as advocated below.

The rationale behind using *tf* weighting is to promote the features corresponding to terms which occur more frequently in a text, since such terms are likely to be more important for the given text. In our case, where each feature is a cluster of terms, we define *tf* as follows:

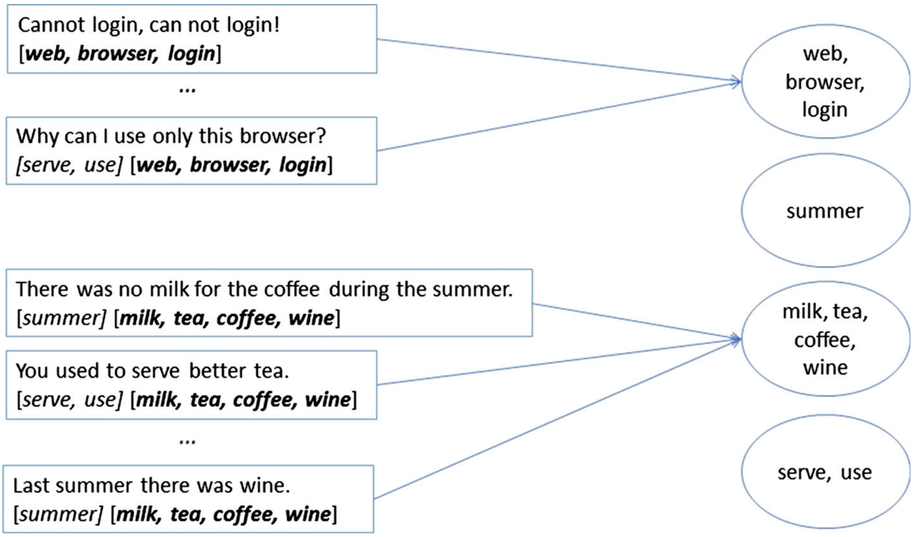

Cannot login, can not login!
[***web, browser, login***]
...
Why can I use only this browser?
*[serve, use]* [***web, browser, login***]
There was no milk for the coffee during the summer.
*[summer]* [***milk, tea, coffee, wine***]
You used to serve better tea.
*[serve, use]* [***milk, tea, coffee, wine***]
...
Last summer there was wine.
*[summer]* [***milk, tea, coffee, wine***]
web, browser, login
summer
milk, tea, coffee, wine
serve, use

**Fig. 8** Example of applying the projection procedure

*Term frequency tf* of a term cluster $c = \{v_i\}$ with respect to text $t$ is the number of times that the terms $v_i \in c$ occur in $t$:

$$tf(c,t) \overset{\text{def}}{=} \sum_{v_i \in c} count(v_i, t).$$

The *tf-idf* weighting scheme balances the *term frequency* component with *inverse document frequency* (*idf*) to decrease the weight of terms which generally tend to occur frequently in the given text collection. For our cluster-based representation the *idf* component is defined as follows:

*Inverse document frequency idf* of a term cluster $c = \{v_i\}$ with respect to the text collection $T = \{t_1, \ldots, t_N\}$ is defined as the logarithmically scaled fraction of the texts that contain at least one term from the cluster:

$$idf(c,T) \overset{\text{def}}{=} log \frac{N}{|t_i \in T : \exists v_i \in t_i \text{``}such\text{''} \text{``}that\text{''} \; v_i \in c|}.$$

Accordingly, the *tf-idf* weight of a term-cluster feature $c$ in a vector representing text $t$ is obtained by multiplying the corresponding components:

$$tf.idf(c,t,T) \overset{\text{def}}{=} tf(c,t) \times idf(c,T).$$

It was found that for clustering large text collections, based on a term-based representation of texts, using *tf* alone is often more effective than using *tf-idf* (Whissell and Clarke 2011). When clustering short texts, we believe *tf* alone to be insufficient, because terms (term-cluster features) will rarely occur more than once in a text. We hypothesize that in our setting, where the text collections are quite small, using *idf* would not be effective, as the terms (term-cluster features) with high *idf* are sparse and are not good signals for general weighting. In fact, we expect that terms which occur in many texts in such collections are likely to be more representative of the given domain, similarly to the case of frequent terms in a single text. We further presume that clusters formed by semantically related terms occurring in many texts of the collection are likely to refer to the main topics of a given domain, while clusters whose terms occur sporadically would correspond to less prominent topics. Following this rationale, we suggest using a *document frequency* component (*df*) instead of *inverse document frequency* (*idf*), yielding a *tf-df* weighting scheme. In Sect. 4 we empirically investigate the impact of different weighting schemes and show that *tf-df* weighting helps to improve the performance of our method, as well as of several reference comparison methods.

We note that *df* weighting is likely to promote stop-words, since they have high occurrence in almost every text. We employ two mechanisms that prevent such words from having an adverse effect. First, we presume that stop-word filtering is performed as part of text processing. In Sect. 4.1.2 we provide details on the stop-word filtering we used in our experiments. In addition, the use of external resources to measure similarity between terms (see Eq. 1) reduces the probability to create a stop-word cluster feature, since most available knowledge resources perform some kind of stop-word filtering and rarely include stop-word term pairs.

We define the *document frequency df* of a term-cluster feature $c$ in a text collection $T$ as follows:

$$df(c,T) \overset{\text{def}}{=} \frac{1}{idf(c,T)}.$$

Accordingly, the *tf-df* weight of a term-cluster feature *c* is obtained by multiplying the corresponding components:

$$tf.df(c,t,T) \overset{\text{def}}{=} tf(c,t) \times df(c,T).$$

# 4 Evaluation

## 4.1 Experimental setting

### *4.1.1 Datasets*

For our experiments we collected real-life industrial data from different domains and source channels, and annotated the datasets in collaboration with domain experts. Table 1 summarizes the information about the data.[3]

The datasets contain anonymized texts expressing reasons of customer dissatisfaction. For the railway domain the texts are manually extracted from customer e-mails, in which the customers provide feedback to a railway company. For the banking domain the texts are automatically crawled from Twitter via an industrial query-based system targeting tweets with criticism towards a bank. For the airline domain the texts are manually extracted from transcripts of call center interactions of an airline company. Anonimization includes changing the name of the bank in the Bank dataset, as well as changing all personal names and geographic names.

The texts in each dataset were assigned to clusters, allowing assignment of a text to multiple clusters if needed. The annotation of each dataset was performed by an individual domain expert, placing two texts in the same cluster if the texts express the same reason for dissatisfaction. Random samples of 50 texts from each dataset were annotated by an additional annotator to evaluate the inter-annotator agreement. The kappa values are presented in Table 1.[4]

**Table 1** Dataset statistics

| Domain | Source | Number of texts | Avg text length (tokens) | Number of clusters | Avg cluster size | Agreement (kappa) |
|---|---|---|---|---|---|---|
| Railway | E-mail | 321 | 6.29 | 29 | 11.40 | 0.87 |
| Bank | Twitter | 190 | 17.25 | 17 | 12.35 | 0.76 |
| Airline | Speech | 90 | 10.07 | 12 | 7.50 | 0.84 |

[3] The datasets are available at http://u.cs.biu.ac.il/~davidol/.

[4] To account for multi-cluster assignment, we used the adaptation suggested by Rosenberg and Binkowski (2004), according to which each text has partial membership in each of its multiple clusters.

**Table 2** Examples of (anonymized) texts and gold-standard clusters

| Domain | Text examples | Tokens | Cluster(s) |
|---|---|---|---|
| Railway | In Standard Magnum go back to the type and quality of meals you offered until about a year ago | 19 | Food |
| | Introduce a season ticket for weekly travellers | 7 | Frequent travellers |
| | Have kid-friendly coaches | 3 | Child friendly |
| Bank | Stupid ntuc do not accept quasibank visa now I have to redraw $150 my stuff as I did not bring my DBS card .... Argh waste my time | 28 | Credit card |
| | I have been trying to login since today evening .. #quasibank #fail | 12 | Internet services |
| | Quasibank ad blitz annoying me | 6 | Advertising |
| Airline | I was just told it's gonna cost me a hundred fifty dollars a ticket to change my tickets | 18 | Fees & charges, changes |
| | I've managed to get three mechanical delays on two flights | 10 | Delays |
| | Customer service is terrible | 4 | Customer service |

Table 2 shows examples of texts of different length with their corresponding gold-standard clusters.

### *4.1.2 Preprocessing and text representation*

We tokenized the input texts, lemmatized the tokens and converted them to lowercase. Then stopword tokens and punctuation were removed.[5] We did not remove hashtags from the Twitter-based dataset. We followed the common-practice heuristic and further removed tokens which occurred in less than $x$ percent or in more than $100 - x$ percent of the texts. We set $x = 2$ to only omit the most critically low and high occurrence words, since in our initial examination we observed that the more common threshold of 5 percent leads to filtering out about 70% of the vocabulary. In addition, for the Bank dataset we removed the terms *quasibank* and *quasi*. These terms are used in the dataset for anonomity, to replace the name of the bank which was used to retrieve the tweets for this dataset. The remaining lemmas were used to form the bag-of-words (*BOW*) representation of the texts.

In some past work on clustering of texts (Sect. 2), the bag-of-words vectors representing the original texts in the given collection were augmented with semantically related terms to go beyond surface-level similarities and obtain an expanded representation of the texts. To evaluate these prior art methods, we automatically augmented the input texts with semantically related terms and thus created the *expanded bag-of-words* representation of the texts (*BOW+*). These additional terms were assigned with lower weights than the terms originally present in the texts, as we detail below. Lexical resources were thus used to induce semantic relations between terms and extract semantically related terms for the *BOW+* vectors. The same resources were used in our method to obtain similarity scores used by the term clustering procedure and to extract expansion terms so as to expand the vocabulary.

[5] We publish our list of stopwords along with the datasets.

We used two resources representing the two types of external knowledge mentioned in Sect. 2.1:

- **WordNet**—a resource, which is given as-is and does not provide access to any underlying data. Our WordNet-based resource for an input text collection was built by extracting for each of the terms $v_i \in V_{BOW}$ its synonyms, derivations, hyponyms and hypernyms. As often done, we limited the resource to only the first (most frequent) sense of the terms; for hyponyms and hypernyms we limited the distance from the expanded term $v_i$ to two steps.
- **UKWaC**-based distributional similarity resource,[6] obtained from the corpus as described in Kotlerman et al. (2010). For this resource, underlying textual data (the UKWaC corpus) is available. The similarity measure used for construction of this resource is reported to produce more accurate output than a range of other measures (Kotlerman et al. 2010). The resource contains pairs of (potentially) semantically related terms $(v, u)$ supplemented with confidence scores. Thus, given a term $v$ its related terms $u_j$ can be extracted and sorted by confidence. In our evaluation we limited the resource to the top-5 most confident related terms for each term $v_i \in V_{BOW}$ of the corresponding text collection.

### *4.1.3 Baselines*

We compare our method to the following state-of-the-art clustering algorithms[7] summarized in Table 3:

- K-medoids algorithms (PAM and KMY) and Complete Link with cosine as the vector similarity measure. We applied these algorithms to the original texts of a given text collection, as well as to the texts augmented with semantically related terms (see Sect. 2.2).
- The information-theoretic co-clustering algorithm of Dhillon et al. (2003), applied to the original texts as well as to the texts augmented with semantically related terms.
- LDA-based clustering. To obtain the partitioning we did not use thresholds, but rather assigned each text to the cluster associated with the highest-scoring topic in its distribution.
    - With "*local*" topics, where the texts of a given text collection are first used to induce the topic space and are then partitioned based on the created LDA topics; i.e. the topics are induced *locally*. We induce local topics using either the original texts of the collection or the texts expanded with semantically related terms.
    - With "*external*" topics, where topics are obtained from a large *external* UKWaC corpus. These topics are then applied to cluster the input text collections using either the original (*BOW*) or the expanded (*BOW*+) representation of the texts.

[6] Available for download from https://github.com/hltfbk/EOP-1.2.0/wiki/English-Knowledge-Resources.

[7] We used the following tools: LingPipe http://alias-i.com/lingpipe/index.html for Complete Link, R implementation of the PAM algorithm https://cran.r-project.org/web/packages/cluster/cluster.pdf, http://www.cs.utexas.edu/users/dml/Software/cocluster.html for co-clustering and http://mallet.cs.umass.edu for LDA. For the KMY baseline we applied for text clustering the algorithm in Fig. 6 with $\theta = 0.7$ as suggested in the original report (Ye and Young 2006). We experimented with additional thresholds (0 through 1 with the step of 0.1) and found the threshold of 0.7 to be one of the best for our setting. The algorithm is not very sensitive to threshold values around 0.7, although much higher and lower thresholds from the range of [0, 1] result in degraded performance.

**Table 3** Application of the baseline methods for clustering in our experiments

| Baseline method | Input for topic modeling | Input for clustering | |
|---|---|---|---|
| | | Original texts (*BOW*) | Expanded texts (*BOW*+) |
| Complete Link | n/a | *V* | *V* |
| PAM | n/a | *V* | *V* |
| KMY | n/a | *V* | *V* |
| Co-clustering | n/a | *V* | *V* |
| LDA with “*local*” topics | Original texts | *V* | – |
| | Expanded texts | – | *V* |
| LDA with “*external*” topics | UKWaC corpus | *V* | *V* |

In addition, we evaluate a word-embedding-based clustering. Namely, we apply Complete Link with cosine as vector similarity measure to 200-dimensional skip-gram-based word embeddings induced from Wikipedia.[8] To represent a text, we used the centroid of the word-embedding vectors of the terms in the text's *BOW* representation. We note that LDA with topics induced from the UKWaC corpus and Complete Link with word-embeddings are the only reference comparison methods that directly use an external corpus. The other methods use external corpus indirectly when applied over the expanded *BOW*+ vectors.

These baseline algorithms require the number of clusters $K$ to be given as a parameter. For co-clustering and LDA, as well as for our method, we set the number of underlying term clusters (topics) $|C| = K$. Clustering with LDA topics was determined by calculating the topic distribution for each text and assigning the text to the topic with the highest probability. For the KMY algorithm we start with each text as a separate medoid,[9] iterate until convergence, retain the medoids of the $K$ biggest clusters and re-assign the texts to obtain the final clustering. Similarly to our method, if some of the texts had zero similarity with each of the $K$ converged medoids, we re-assigned the texts to $K - 1$ medoids and assigned the unclassified texts to an additional *miscellaneous* cluster; the goal was to ensure that $K$ clusters are produced as required. We note that this policy was applied as part of the algorithm so as to allow for a fair comparison between the methods. It was usually employed for low values of $K$ (see Sect. 4.1.5) and was not frequent overall.

### *4.1.4 Weighting schemes*

We compare all the methods over *tf-idf*, *tf* and *tf-df* weighting schemes. We note that our approach uses each weighting scheme for term-cluster features in a text, as defined in Sect. 3.2.1, while the reference comparison methods use the scheme for weights of terms in a text. To apply different term weighting schemes for the LDA and co-clustering

[8] We used the word2vec software accompanying (Mikolov et al. 2013) with context size of 5, the negative-training approach with 15 negative samples (NEG-15), and sub-sampling of frequent words with a parameter of $10^{-5}$. The parameter settings follow Mikolov et al. (2013).

[9] We experimented also with $K$ randomly selected initial medoids, but having each text as an initial medoid showed better results. Since our text collections are small this is not computationally expensive.

methods, we rounded each weight up to the closest integer and duplicated the corresponding term accordingly in the given input text. Below we detail the calculation of *tf*, *tf-idf* and *tf-df* weights used in our evaluation for the baseline methods for the *BOW* and *BOW+* text representations:

- Term frequency *tf* of a term $v$ in a text $t$ is the number of times that term $v$ occurs in $t$. In order to prefer original terms over external terms in the spirit of Metzler et al. (2007), when calculating the weights of expansion terms we used the following pseudo counts:

$$count(v_{exp}, t) \overset{\text{def}}{=} \frac{1}{2} \max_{v_i : v_i \in t \wedge v_i \in V_{BOW}} \frac{|r_i \in R : (v_i, v_{exp}) \in r_i \vee (v_{exp}, v_i) \in r_i|}{|R|},$$

  where $R = \{r_i\}$ is the set of resources used to extract the expansion terms. The count is thus equal to the proportion of resources according to which the term $v_{exp}$ is considered related to any of the original terms $v_i$ in the text $t$; the value is upper bounded by $\frac{1}{2}$. Note that this is equivalent to the similarity values in Eq. 1 with $c_{ub} = 0.5$. For term clustering in our method (Sect. 3.1) the value of $c_{ub}$ in Eq. 1 was set to 0.5 for compatibility with the term weighting scheme applied in the baselines. One exception is the Chinese Whispers algorithm (Sect. 3.1.2), for which we set $c_{ub} = |R|$ to account for the fact that the clustering tool only accepts integer weights.
- Inverse document frequency *idf* of a term $v$ with respect to the text collection $T = \{t_1, \ldots, t_N\}$ is the logarithmically scaled fraction of the documents (either original or expanded with semantically related terms) that contain this term. Accordingly, the *tf-idf* weights were calculated as follows:

$$tf.idf(v, t, T) \overset{\text{def}}{=} tf(v, t) \times log \frac{N}{|t_i \in T : v \in t_i|}.$$

- Similarly to our term-cluster features (Sect. 3.2.1), the *tf-df* weight of a term $v$ with respect to text $t$ in collection $T$ was calculated as follows:

$$tf.df(v, t, T) \overset{\text{def}}{=} tf(v, t) \times \frac{1}{idf(v, t, T)}.$$

### *4.1.5 Evaluation measures*

We use parwise $F1$ and Rand Index as evaluation measures. For F1 we view the clustering as a series of decision one for each of the $N(N-1)/2$ pairs of texts in the collection.

The evaluated clustering algorithms require the number of clusters $K$ to be provided as a parameter. For each algorithm we produce five outputs for $K = \{10, 15, 20, 25, 30\}$, taking into account the range of ground truth cluster numbers, and report $F1$ and Rand Index values macro-averaged over the choices of $K$. We examine statistical significance of the results using McNemar's test (Dietterich 1998) with alpha of 0.05.

## 4.2 Results

In Table 4 we report our main evaluation results, providing comparison of the best instantiation of our method with the baselines. The instantiation used in Table 4 is the *KMY Hard-K* clustering via the expanded vocabulary $V_{BOW+}$ (Sect. 3.1.1). Below we show that this instantiation is the best-performing one for our method.

**Table 4** Main evaluation results in terms of F1(%)

| Method | Railway dataset | | | Bank dataset | | | Airline dataset | | |
|---|---|---|---|---|---|---|---|---|---|
| | *tf-idf* | *tf* | *tf-df* | *tf-idf* | *tf* | *tf-df* | *tf-idf* | *tf* | *tf-df* |
| $CL_{BOW}$ | 33.3 | 34.1 | 40.8 | 19.4 | 24.6 | 32.0 | 36.7 | 38.4 | 37.4 |
| $CL^{emb}_{BOW}$ | 35.7 | 26.6 | 28.2 | 24.3 | 21.1 | 21.5 | 28.1 | 24.9 | 24.6 |
| $CL_{BOW+}$ | 44.6 | 40.8 | 43.0 | 31.3 | 29.4 | 33.2 | 39.8 | 34.9 | 39.9 |
| $KMY_{BOW}$ | 54.6 | **60.2** | 60.8 | 26.7 | 25.8 | 28.4 | 31.6 | 34.0 | 35.2 |
| $KMY_{BOW+}$ | 55.6 | 60.0 | 60.8 | 26.7 | 26.2 | 28.4 | 29.3 | 33.7 | 35.3 |
| $PAM_{BOW}$ | 40.6 | 45.2 | 50.5 | 23.6 | 23.0 | 23.3 | 34.1 | 37.4 | 37.4 |
| $PAM_{BOW+}$ | 40.6 | 45.2 | 50.5 | 23.6 | 23.0 | 23.3 | 34.1 | 37.4 | 37.4 |
| $Cocl_{BOW}$ | 25.7 | 28.7 | 31.0 | 16.6 | 16.0 | 16.2 | 19.8 | 21.7 | 19.1 |
| $Cocl_{BOW+}$ | 35.6 | 36.1 | 42.6 | 18.8 | 19.1 | 21.7 | 24.4 | 26.2 | 28.0 |
| $LDA^{loc}_{BOW}$ | 22.0 | 24.0 | 22.1 | 21.5 | 26.7 | 21.6 | 36.6 | **40.8** | 37.2 |
| $LDA^{loc}_{BOW+}$ | **58.9** | 59.7 | 60.2 | **33.5** | **32.9** | 31.2 | **40.9** | 39.1 | 41.4 |
| $LDA^{ext}_{BOW}$ | 21.0 | 21.7 | 24.3 | 17.1 | 19.5 | 19.5 | 22.5 | 23.4 | 25.4 |
| $LDA^{ext}_{BOW+}$ | 42.6 | 46.9 | 47.6 | 20.6 | 21.3 | 22.8 | 27.7 | 27.3 | 31.5 |
| Our method | 30.4 | 37.4 | **78.0***| 10.8 | 21.9 | **37.3*** | 14.8 | 30.1 | **43.2*** |

The best result in each column is boldfaced. The best result per dataset is underlined. The results marked with a star (*) are statistically significantly better than those of each of the baselines across all weighting schemes. *CL*, Complete Link; $CL^{emb}$, Complete Link applied to word-embedding representation of the texts; *Cocl*, co-clustering; $LDA^{loc}$, LDA with topics obtained locally from the input text collection; $LDA^{ext}$, LDA with topics induced from the external UKWaC corpus. The subscript shows whether the corresponding clustering method was applied to the original texts (*BOW*) or to the expanded texts (*BOW*+). For our method we use the instantiation with *KMY Hard-K* term clustering via the expanded vocabulary $V_{BOW+}$

Table 4 shows consistent and significant advantage of our method with *tf-df* weighting, which we advocated in Sect. 3.2.1. We also note that, as shown by Table 4, in our setting the *tf-df* weighting scheme turns out to be beneficial for many reference comparison methods. Further throughout this section we show *tf-df* numbers for our method as it was shown to be the best approach. In addition, in Table 5 we report the evaluation in terms of Rand Index, which also shows the advantage of our method.

In Fig. 9 we show the performance of different term clustering algorithms within our suggested framework, as detailed in Sect. 3.1. The *KMY Hard-K* method over the *BOW*+ vocabulary (the leftmost bar in each chart) is the term clustering algorithm reported in Table 4, which is overall our method's best-performing instantiation. Figure 9 allows comparing the performance of different instantiations of our method to the best-performing baseline, which is different for each dataset ($KMY_{BOW}$ and $KMY_{BOW+}$ with *tf-df* weighting for the Railway dataset, $LDA^{loc}_{BOW+}$ with *tf-idf* weighting for the Bank dataset, and $LDA^{loc}_{BOW+}$ with *tf-df* weighting for the Airline dataset).

For further comparison of our suggested framework with the baseline methods it is interesting to see whether the performance improves if we apply a clustering algorithm, e.g. *PAM*, for term clustering within our framework rather than directly use it to partition the input texts. In Table 6 we report the gain in performance achieved when applying each reference comparison algorithm within our framework versus its application to cluster the

**Fig. 9** Performance of different term clustering methods within our framework using the *tf-df* weighting scheme. For the *KMY*, *PAM*, Complete Link (*CL*) and Chinese Whispers (*CW*) methods, using the *BOW* vocabulary refers to term clustering in Fig. 2 for the original vocabulary $V_{BOW}$, while using the *BOW+* vocabulary refers to term clustering via expanded vocabulary (Fig. 4). For co-clustering (*Coclust*) and *LDA local* we applied the corresponding method to the original (*BOW*) or the expanded (*BOW+*) texts, and used the term clusters generated by the method. For *LDA ext* topic modeling was applied over the external UKWaC corpus

input texts directly. For each reference comparison method we use its highest $F1$ from Table 4 or Rand Index from Table 5, i.e. we select the best-performing weighting scheme and text representation (vocabulary). For our framework we use the performance of each instantiation with *tf-df* weighting (as shown in Fig. 9) and, if not explicitly specified, term clustering via expanded vocabulary $V_{BOW+}$ (Fig. 4). We can see in Table 6 that for the majority of the methods applying them for term clustering within our suggested framework is beneficial with respect to our settings.

Finally, we evaluate the alternative assignment of texts to clusters within our framework, as mentioned in Sect. 3.2. That is, we apply a clustering algorithm over the texts represented as vectors in the term-cluster space rather than associate each text with a cluster that corresponds to the term cluster with the highest weight in the vector representing the text. In Table 7 we report the change in performance following this alternative assignment process as compared to the results of our method from Tables 4 and 5. We see

**Table 5** Main evaluation results in terms of Rand Index

| Method | Railway dataset | | | Bank dataset | | | Airline dataset | | |
|---|---|---|---|---|---|---|---|---|---|
| | *tf-idf* | *tf* | *tf-df* | *tf-idf* | *tf* | *tf-df* | *tf-idf* | *tf* | *tf-df* |
| $CL_{BOW}$ | .463 | .481 | .494 | .560 | .609 | .663 | .723 | .741 | .740 |
| $CL^{emb}_{BOW}$ | .358 | .303 | .302 | .789 | .787 | .801 | .824 | .820 | .820 |
| $CL_{BOW+}$ | .503 | .458 | .471 | .706 | .729 | .727 | .813 | .780 | .828 |
| $KMY_{BOW}$ | .841 | **.854** | .855 | .852 | .851 | .853 | .842 | .844 | .843 |
| $KMY_{BOW+}$ | **.843** | .853 | .856 | .851 | .846 | .850 | .838 | .845 | .844 |
| $PAM_{BOW}$ | .735 | .749 | .755 | .850 | .846 | .847 | .803 | .813 | .813 |
| $PAM_{BOW+}$ | .735 | .749 | .755 | .850 | .846 | .847 | .803 | .813 | .813 |
| $Cocl_{BOW}$ | .778 | .790 | .786 | .847 | .850 | .849 | .823 | .828 | .821 |
| $Cocl_{BOW+}$ | .808 | .798 | .806 | .847 | .846 | .840 | .825 | .803 | .824 |
| $LDA^{loc}_{BOW}$ | .750 | .755 | .749 | .855 | .860 | .852 | **.852** | **.855** | .861 |
| $LDA^{loc}_{BOW+}$ | .826 | .829 | .825 | .850 | .854 | .846 | .845 | .845 | .842 |
| $LDA^{ext}_{BOW}$ | .621 | .554 | .423 | .771 | .701 | .514 | .689 | .562 | .399 |
| $LDA^{ext}_{BOW+}$ | .735 | .748 | .757 | .810 | .811 | .802 | .800 | .791 | .795 |
| Our method | .803 | .811 | **.908*** | **.870** | **.865** | **.875*** | .835 | .853 | **.870*** |

The best result in each column is boldfaced. The best result per dataset is underlined. The results marked with a star (*) are statistically significantly better than those of each of the baselines across all weighting schemes. *CL* Complete Link; $CL^{emb}$, Complete Link applied to word-embedding representation of the texts; *Cocl*, co-clustering; $LDA^{loc}$, LDA with topics obtained locally from the input text collection; $LDA^{ext}$, LDA with topics induced from the external UKWaC corpus. The subscript shows whether the corresponding clustering method was applied to the original texts (*BOW*) or to the expanded texts (*BOW+*). For our method we use the same instantiation as in Table 4, with *KMY Hard-K* term clustering via the expanded vocabulary $V_{BOW+}$

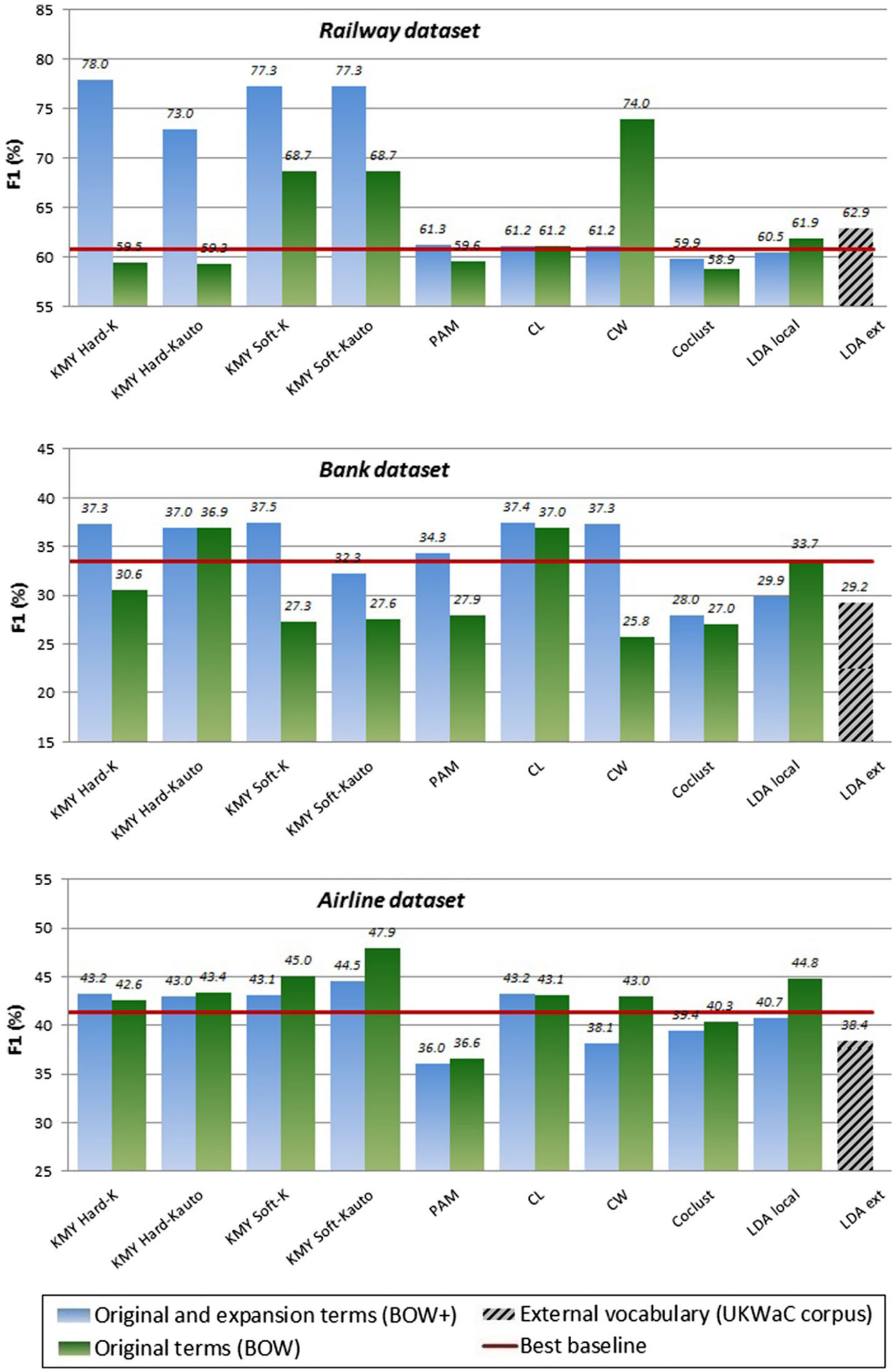

Railway dataset
F1 (%)
85
80
75
70
65
60
55
78.0
59.5
73.0
59.3
77.3
68.7
77.3
68.7
61.3
59.6
61.2
61.2
61.2
74.0
59.9
58.9
60.5
61.9
62.9
KMY Hard-K
KMY Hard-Kauto
KMY Soft-K
KMY Soft-Kauto
PAM
CL
CW
Coclust
LDA local
LDA ext
Bank dataset
F1 (%)
45
40
35
30
25
20
15
37.3
30.6
37.0
36.9
37.5
27.3
32.3
27.6
34.3
27.9
37.4
37.0
37.3
25.8
28.0
27.0
29.9
33.7
29.2
KMY Hard-K
KMY Hard-Kauto
KMY Soft-K
KMY Soft-Kauto
PAM
CL
CW
Coclust
LDA local
LDA ext
Airline dataset
F1 (%)
55
50
45
40
35
30
25
43.2
42.6
43.0
43.4
43.1
45.0
44.5
47.9
36.0
36.6
43.2
43.1
38.1
43.0
39.4
40.3
40.7
44.8
38.4
KMY Hard-K
KMY Hard-Kauto
KMY Soft-K
KMY Soft-Kauto
PAM
CL
CW
Coclust
LDA local
LDA ext
Original and expansion terms (BOW+)
Original terms (BOW)
External vocabulary (UKWaC corpus)
Best baseline

**Table 6** Performance achieved by applying each clustering algorithm within our framework rather than for clustering the text collections directly

| Method | Railway dataset | | Bank dataset | | Airline dataset | |
|---|---|---|---|---|---|---|
| | F1, % | Rand Index | F1, % | Rand Index | F1, % | Rand Index |
| *CL* | 61.2 (+ 16.6)* | .860 (+ .357)* | 34.3 (+ 4.2)* | .869 (+ .140)* | 43.2 (+ 3.9)* | .869 (+ .041)* |
| *KMY* | 78.0 (+ 17.2)* | .908 (+ .052)* | 37.3 (+ 8.9)* | .875 (+ .022)* | 43.2 (+ 7.9)* | .870 (+ .025)* |
| *PAM* | 61.3 (+ 1.8)* | .856 (+ .101)* | 37.4 (+ 1.7) | .858 (+ .008) | 36.0 (-1.4) | .826 (+ .013) |
| $Cocl_{BOW}$ | 58.9 (+ 27.9)* | .845 (+ .054)* | 27.0 (+ 10.4)* | .807 (-.043)* | 40.3 (+ 18.6)* | .844 (+ .016)* |
| $Cocl_{BOW+}$ | 59.9 (+ 17.3)* | .851 (+ .043)* | 28.0 (+ 6.3)* | .821 (-.026)* | 39.4 (+ 11.4)* | .828 (+ .002)* |
| $LDA^{loc}_{BOW}$ | 61.9 (+ 37.9)* | .854 (+ .098)* | 33.7 (+ 7.0)* | .847 (-.013)* | 44.8 (+ 4.0) | .864 (+ .003) |
| $LDA^{loc}_{BOW+}$ | 60.5 (+ 0.3) | .857 (+ .029) | 29.9 (-3.6)* | .837 (-.017)* | 40.7 (-0.2) | .844 (-.002) |
| $LDA^{ext}$ | 62.9 (+ 15.3)* | .865 (+ .107)* | 29.2 (+ 6.4)* | .849 (+ .038)* | 38.4 (+ 6.9)* | .855 (+ .055)* |

Change in performance is given in parenthesis. *CL*: Complete Link, *Cocl*, co-clustering; $LDA^{loc}$, LDA with topics obtained locally from the input text collection; $LDA^{ext}$, LDA with topics induced from the external UKWaC corpus. The results marked with a star (*) are statistically significant

**Table 7** Change in performance when applying clustering methods to term-cluster vectors rather than associating each text with a cluster that corresponds to the term-cluster feature with the highest weight

| Method | Railway dataset | | Bank dataset | | Airline dataset | |
|---|---|---|---|---|---|---|
| | F1, % | Rand Index | F1, % | Rand Index | F1, % | Rand Index |
| *CL* | − 30.1* | − 0.339* | − 9.9* | − 0.134* | − 1.6* | − 0.158* |
| *KMY* | − 0.4 | − 0.003 | − 8.9* | − 0.020* | − 5.9 | − 0.022 |
| *PAM* | − 19.2* | − 0.057* | − 13.6* | − 0.029* | − 6.4 | − 0.019 |

*CL*, Complete Link. Term-cluster vectors are formed by applying the *KMY Hard-K* term clustering via expanded vocabulary (Fig. 4), with *tf-df* weighting of the term-cluster features. The results marked with a star (*) are statistically significant

that this alternative approach overall yields decreased performance. We thus conclude that our suggested approach of associating each text with a cluster that corresponds to the term-cluster feature with the highest weight is preferable in our setting.

## 4.3 Exploratory analysis

In this section we report the results of exploratory analysis we performed so at to better understand the behavior of clustering methods in our setting. The results of error analysis are reported in Sect. 4.4.

### *4.3.1 Cluster identifiers in texts*

Most of the texts in the datasets we consider seem to contain one or just few concrete terms indicating to which cluster a text should be assigned. For example, in the text "*In Standard Magnum go back to the type and quality of meals you offered until about a year ago*" the term "*meal*" implies that this text should be assigned to the "*Food*" cluster, while other terms convey no semantics related to this cluster. Terms which constitute concrete references in text to a specific topic of interest serve as *lexical references* to target clusters (Glickman et al. 2006; Barak et al. 2009; Liebeskind et al. 2015). We further refer to such terms as *cluster identifiers*.

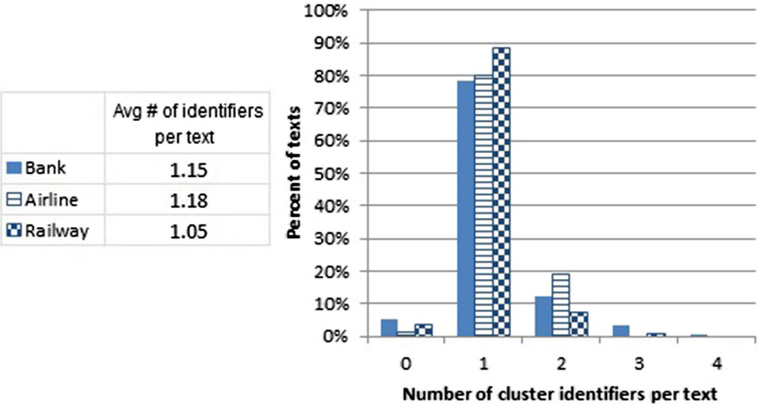
In order to better understand the performance of different methods for our datasets, we manually annotated cluster identifiers in the texts and analyzed the resulting annotation. The decision whether a term in a text is a cluster identifier was made with respect to the gold-standard clustering: given a text and a target cluster, all the terms referring to the topic of the target cluster were annotated as identifiers of this cluster. In Table 8 we show examples of the annotated cluster identifiers from the three datasets. The annotation was performed by one of the authors. A random sample of 150 text-cluster pairs (50 from each dataset) was also annotated by an additional annotator to evaluate the inter-annotator agreement, resulting in Cohen's kappa of 0.87.

Figure 10 shows the statistics of the occurrence of cluster identifiers in texts. We see that most texts have a single cluster identifier, while some have no identifier terms, some have two identifiers, rarely three and never more than four. This observation explains the advantage of assigning texts to clusters by associating each text with the single most highly weighted term-cluster feature in its vector representation (see Sect. 3.2 for details). It also explains the overall relatively low results of co-clustering and LDA-based clustering of the original texts ($Cocl_{BOW}$ and $LDA^{loc}_{BOW}$ in Table 4). Co-clustering and LDA rely on term co-occurrence in the input texts to create term clusters (topics) so as to enhance text clustering. Terms co-occurring with each other in the texts are likely to be assigned to the same term cluster (topic). Figure 10 shows that cluster identifiers only co-occur in about 10–20% of the texts. When cluster identifiers co-occur with each other so rarely within the given texts,

**Table 8** Examples of cluster identifiers in texts

| Text | Identifiers | Cluster |
|---|---|---|
| It seems like it's indefinite when it's going to leave | 0 | Delays (Airline domain) |
| It would be better if there were staff walking around selling **snacks** | 1 | Food (Railway domain) |
| The departure date is the wrong date than what i had **booked** | 1 | Booking (Airline domain) |
| Would be ready to pay higher price for full **dinner** with hot **meal** | 2 | Food (Railway domain) |
| Great , I **call** my American bank and I get a **call center** in Indian , with the worst **phone** connection ever ! ! ! | 3 | Phone services (Bank domain) |
| **quasibank.co.in** does not allow you to **login** again , immediately after **logout**, unless you close your **browser** and restart it . #quasibank #fail | 4 | Website (Bank domain) |

The identifier terms are boldfaced



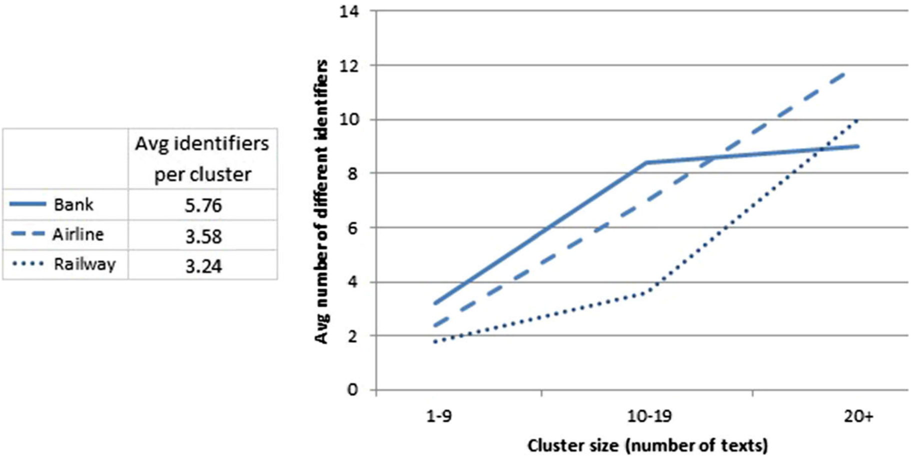
| | Avg # of identifiers per text |
|---|---|
| ■ Bank | 1.15 |
| ▤ Airline | 1.18 |
| ▦ Railway | 1.05 |

**Fig. 10** Occurrence of cluster identifiers in texts

Co-clustering and LDA are less likely to generate a partitioning where the identifier terms of each cluster are grouped together. And, therefore, the subsequent document clustering yields modest results.

### *4.3.2 Cluster identifiers and language variability*

It is quite obvious that due to language variability the same target cluster might have different identifier terms in different texts as can be seen in the examples in Table 8. Figure 11 shows the results of a quantitative analysis we performed for our annotated cluster identifiers to better understand this issue. We see that the average number of different identifiers per cluster ranges from 3 to almost 6 in our datasets. Figure 11 also shows that larger clusters, which are more interesting for most application settings, tend to have more identifiers than small ones. Table 9 provides some examples of cluster identifiers' variability. This supports using term-cluster features rather that single-term features for the representation of texts.



| | Avg identifiers per cluster |
|---|---|
| — Bank | 5.76 |
| – – Airline | 3.58 |
| ···· Railway | 3.24 |

**Fig. 11** Occurrence of different cluster identifiers in clusters

**Table 9** Examples of cluster identifiers in different clusters

| Cluster label | Cluster size | Number of distinct identifiers | Avg # of identifiers per text in a cluster | Identifiers |
|---|---|---|---|---|
| Personnel (Bank domain) | 9 | 11 | 1.11 | chief, employee, staff, officer, personnel, pol, sco, incompetent, support person, upper management |
| Drinks (Railway domain) | 20 | 8 | 1.15 | tea, coffee, drink, espresso, milk, nespresso, champagne, wine |
| Fees & charges (Airline domain) | 26 | 11 | 1.50 | charge, cost, dollar, fare, fee, pay, penalize, penalty, price, spend, surcharge |

We analyzed the term clusters automatically generated by the best instantiation of our method (*KMY Hard-K* clustering via the expanded vocabulary; see Sect. 4.2 for evaluation). Apparently, we detected considerable differences between the automatically generated clusters and the term groups from our manual annotation. In order to assess the influence of these differences we performed an oracle experiment reported in Table 10. In this experiment we used our manually annotated groups of cluster identifiers to replace the automatic term-cluster features. That is, as input for Step 2 of our framework we used either the term clusters automatically generated by the best instantiation of our method (denoted as *Automatic* in Table 10) or the manually annotated term clusters (denoted as *Oracle* in Table 10). Then we automatically performed the projection of the texts over the input term clusters as detailed in Sect. 3.2 so as to produce text clustering. Table 10 shows that, as expected, the performance of our method considerably improves with the oracle term clusters. We note that our gold-standard clustering allows multiple classes per text, while the projection of the texts over the term clusters produces a hard partitioning. Thus, some "errors" are unavoidable when evaluating the hard clustering results. We see a significant improvement in terms of Recall due to the reduced number of false negative decisions, when two texts which discuss the same topic are assigned to different clusters. This means that our current automatic term clustering does not manage to group the terms in such a way that all the identifiers of each cluster would form a single term cluster. When the identifiers of a cluster appear in more than one term cluster, the texts with the corresponding terms end up in different clusters. The improvement in terms of Precision is due to the reduced number of false positive errors, when two texts, which do not discuss the same topic, are assigned to the same cluster. This means that our current automatic term clusters mix up identifiers of different clusters. We thus assume that improving the underlying term clustering, which is the core of our suggested method, would yield an additional boost in performance. We further analyze the reasons of false positive and false negative errors in Sect. 4.4.

### *4.3.3 Distribution of identifiers*

Although each cluster might have quite many identifiers, each individual identifier is likely to occur in more than one text, especially for large clusters. For example, the cluster

**Table 10** Evaluation of our method with automatically generated vs. oracle term clusters

| Term clusters | Railway dataset | | | Bank dataset | | | Airline dataset | | |
|---|---|---|---|---|---|---|---|---|---|
| | R | P | F1 | R | P | F1 | R | P | F1 |
| Automatic | 81 | 75 | 78 | 33 | 44 | 37 | 31 | 70 | 43 |
| Oracle | 97 | 87 | 92 | 69 | 64 | 66 | 83 | 95 | 88 |

"*Fees & charges*" has 11 identifiers for 26 texts, and the identifier "*charge*", for instance, occurs in seven of them. We counted the number of different texts in which each manually-annotated cluster identifier occurs, as well as the number of texts in which each non-identifier term occurs, and summarized the results in Fig. 12. The figure shows that 40% to over 60% of the identifiers occur in 2 and more texts, and that about 10–20% of the identifiers occur in 6 texts or more. Non-identifier terms, on the contrary, tend to occur in a single text and no more than 20% of them occur in 2 texts or more. This behavior accounts for the superior performance of the *tf-df* weighting for many baseline methods and especially for our suggested framework.

## 4.4 Error analysis

For error analysis we use the results of our method from Table 4 with $K = 20$ (the average of all $K$ values used in our experiments). We randomly sampled 50 false positive and 50 false negative errors across the three datasets. Each false positive error is a pair of texts, which in the gold standard annotation are assigned to different clusters, while our method placed them in the same cluster. Each false negative error is a pair of texts, which in the gold-standard annotation belong to the same cluster, while our method failed to assign them together. We note that our sampling does not reflect the underlying error distribution. In this analysis our goal was not analyzing the ratio of false positive versus false negative errors, but rather to gain some insights about the nature of each error type.

### 4.4.1 *False positive errors*

The results of the analysis of the false positive examples are presented in Fig. 13. We assigned each error to one of the following categories:

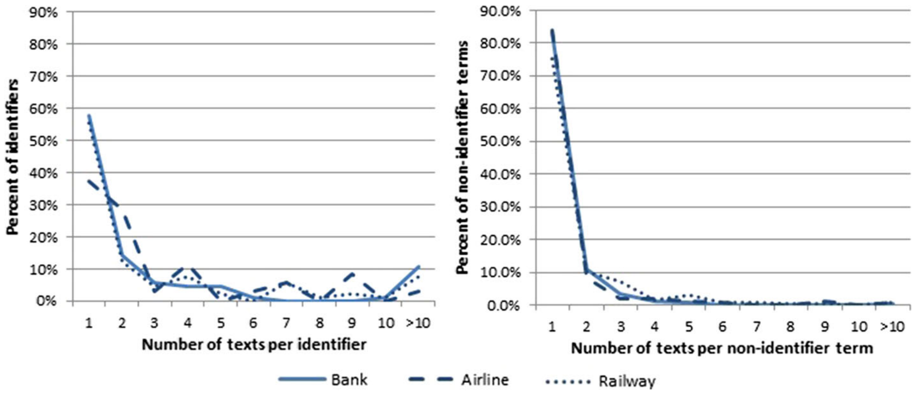


**Fig. 12** Document frequency of cluster identifiers and non-identifier terms

- *Meaningful alternative* This category denotes pairs of texts which would not be considered errors in an alternative gold-standard annotation. For example, in our gold-standard Bank dataset the text "*is disappointed to find out that even in 2010 #Quasibank treats wives as second-class customers right under their husbands*" is placed in a "*Chauvinism*" cluster. Our method placed it under the "*Customer service*" cluster, together with the text "*Ver bad customer response by #Quasibank #fail ! ! ! !*" One could say that assigning this pair of texts to the same cluster is meaningful and is not erroneous.
- *Sentiment* This category denotes the cases where the texts are assigned together due to their sentiment rather than their topic. Filtering the terms that convey the author's sentiment, either automatically or via manually constructed sentiment term lists, can be used to reduce the frequency of such errors.
- *General domain cluster* Texts in this category were assigned to irrelevant general-domain clusters, such as "*Bank*" for the Bank dataset. Unlike the cases in the *Meaningful alternative* category, here the texts are too diverse to belong together. For example, although the texts "*new updated train would be nice*" and "*the train was overbooked*" can potentially belong to a cluster about trains, it seems that such dimension is not relevant and is too general for the Railway domain.
- *False relatedness* In this category we list text pairs such as "*seats need to be more comfortable*" and "*more organization in the lounge areas*". The texts were clustered together since our underlying term clustering created a cluster with the terms "*seat*" and "*lounge*". The terms are indeed related, especially given the ambiguity of the term "*lounge*", for which the first WordNet sense is "*an upholstered seat for more than one person*". Yet, we decided to distinguish such cases from the *Meaningful alternative* ones, since the relatedness of the given texts is questionable. Using more lexical resources, including domain-specific ones geared for the target domain of the given text collection, might be helpful in eliminating such errors.
- *Misleading feature* Under this category we place pairs of texts which were clustered together due to an irrelevant common feature. For example, the texts "*quasibank charges a monthly fee on us poor people*" and "*quasibank are ninjas at catching fraudulent charges.*" belong to the clusters "*Fees and charges*" and "*Fraud detection*", but are clustered together due to selection of an inappropriate feature for the second text, a feature which in the current context does not indicate the right

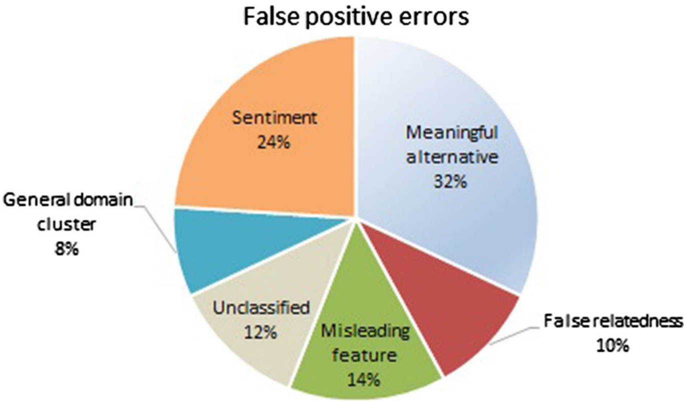


**Fig. 13** Analysis of false positive errors

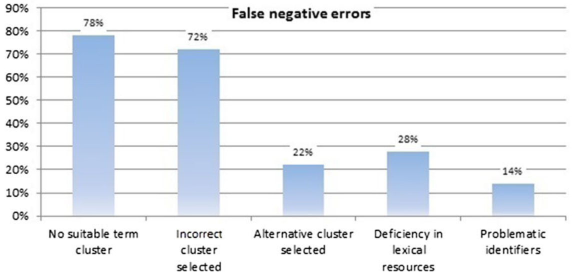
cluster. To avoid such errors, context-sensitive feature weighting can be potentially developed and more complex inter-text similarity measures can be integrated for projecting the texts over term clusters.

- *Unclassified* Here we list the examples in which texts had zero overlap with the underlying term clusters and thus could not be assigned. As explained in Sect. 3.2, when not all the texts could be partitioned into the requested $K$ clusters, we performed new term clustering into $K - 1$ clusters and assigned all the unclassified texts into an additional *miscellaneous* cluster, thus creating false positive errors. For example, the text "*quasibank site is getting on my nerves*" remained unclassified, since no term-cluster feature for "*site, website, internet, etc.*" was created during term clustering. Improving the term clustering by integrating more lexical resources and employing algorithms geared specifically for term clustering can potentially reduce this type of errors.

### 4.4.2 False negative errors

The results of the analysis of the false negative examples are summarized in Fig. 14. Per a pair of texts, there were usually several reasons due to which our method did not succeed in placing the two texts under the same cluster. We identified the following main reasons:

- *No suitable term cluster* As already mentioned in the analysis of false positive errors, there are cases when texts cannot be assigned by our method to their correct gold-standard cluster, since no term-cluster feature was created for the corresponding topic. For example, the texts "*the further ahead you plan the worse your situation is*" and "*it seems to me you're paying penalty for planning ahead*" belong to the gold-standard "*Planning*" cluster. But since there was no term cluster for "*plan, planning, ahead, etc.*" the texts were assigned to different clusters by our method.
- *Incorrect cluster selected* Sometimes two texts are not assigned together since for both of the texts or for one of them an incorrect cluster was selected by our method. For example, the texts "*better rewards for frequent travellers*" and "*Introduce a season ticket for weekly travellers*" belong to the gold-standard "*Frequent travellers*" cluster, but the "*Introduce a season ticket for weekly travellers*" text was assigned by our method to the "*Tickets*" cluster, which does not exist in the gold standard. Incorrect cluster is usually selected when no suitable term cluster is created during the term-clustering step (see *No suitable term cluster* error type above). Less frequently a



| | No suitable term cluster | Incorrect cluster selected | Alternative cluster selected | Deficiency in lexical resources | Problematic identifiers |
|---|---|---|---|---|---|
| No suitable term cluster | 39 | - | - | - | - |
| Incorrect cluster selected | 31 | 36 | - | - | - |
| Alternative cluster selected | 9 | 2 | 11 | - | - |
| Deficiency in lexical resources | 10 | 11 | 3 | 14 | - |
| Problematic identifiers | 1 | 0 | 2 | 1 | 7 |

**Fig. 14** Analysis of (possibly overlapping) false negative error types. Left: Percent of each error type in our sample. Right: Confusion matrix

suitable term-cluster feature is created, but an incorrect term-cluster feature is given a higher weight in the vector representation of a text, resulting in the assignment of the text to an incorrect cluster. As noted above for the false positive errors, improved term clustering is likely to be helpful for such cases. The pre-processing policies are also influential here. For example, using multi-word expressions would potentially allow revealing the relatedness between "*weekly travellers*", "*frequent travellers*" and "*season ticket*", especially if a domain-specific lexical resource would be used.

- *Alternative cluster selected* As opposed to selecting an incorrect cluster, in this case one or both texts in a pair are assigned to their alternative gold-standard cluster. For example, the pair of texts "*A trolley serving tea would be a welcome addition to the economy section*" and "*lounge: coffee machines not always working*" belong to the gold-standard cluster "*Drinks*". But the text "*lounge : coffee machines not always working*" also belongs to the "*Lounge*" cluster in the gold-standard annotation, and is assigned to a cluster about lounge by our method. Since our gold-standard annotation allows several clusters per text, while our automatic method performs hard clustering, such errors are inevitable. We note that we only use this error type when the alternative cluster is one of the clusters to which a text is assigned in our gold-standard annotation.
- *Deficiency in lexical resources* Here we list errors caused by problems incurred by the lexical resources, such as ambiguity, vague similarities and lack of similarity between terms. This reason is in most cases coupled with *Incorrect cluster selected* or with *No suitable term cluster*. For example, the texts "*Quasibank, your student loan site is awful.*" and "*There is one benefit to Quasibank's TERRIBLE website*" were not assigned to the "*Internet services*" cluster, since no term cluster for "*site, website, internet, etc.*" was created. Analysis of our lexical resources showed that the terms "*site*" and "*website*" were not listed as semantically related in our UKWaC-based resource. In our WordNet resource, which was limited to the first sense of the terms, the term "*site*" occurred only in its "*land site*" sense and thus was not linked to terms like "*website*", "*internet*" etc. Integrating various resources, as well as performing domain- and context-sensitive selection of terms from the resources can be used to improve the performance.
- *Problematic identifiers* Sometimes the texts cannot be assigned to their correct clusters because they either have no concrete identifiers or the identifiers are not explicit. For example, the text "*good idea! quasibank have been the worst blood suckers for me, and I will be done with them real soon!*" belongs to the gold-standard "*Fees and charges*" cluster, which is not obvious even for a human reader.

## 5 Conclusions and future work

In this paper we presented a framework for clustering small-sized collections of short texts by first clustering the term space and further projecting the texts in the collection onto the emerged term clusters. The framework utilizes external knowledge resources to address the vocabulary mismatch between short texts and the insufficient information on term co-occurrence in the given text corpus. We evaluated various instantiations of the proposed framework and demonstrated the merits of our approach via an extensive empirical evaluation. The analysis presented in the paper suggests that improved term clustering is likely to yield additional gain in performance. Among the directions for further improvement we could name integration of more knowledge resources, domain-specific and context-sensitive selection of terms from the resources, use of clustering algorithms developed specifically for term clustering, as well as integration of various inter-text similarity measures for projecting the texts over term clusters. Finally, evaluating our approach with additional datasets of short texts annotated by domain experts is also a venue we intend to pursue.

**Acknowledgements** This work was partially supported by the MAGNETON Grant No. 43834 of the Israel Ministry of Industry, Trade and Labor, the Israel Ministry of Science and Technology, the Israel Science Foundation Grant 1112/08 and Grant 1136/17 the PASCAL-2 Network of Excellence of the European Community FP7-ICT-2007-1-216886 and the European Communitys Seventh Framework Programme (FP7/2007-2013) under Grant Agreement No. 287923 (EXCITEMENT). We would like to thank NICE Systems and especially Maya Gorodetsky, Gennadi Lembersky and Ezra Daya for help in creating the datasets. Finally, we thank the anonymous reviewers for their useful comments and suggestions.

## References

Aggarwal, C.C., & Zhai, C. (2012). A survey of text clustering algorithms. In *Mining text data* (pp. 77–128). Springer.

Allan, J., Papka, R., & Lavrenko, V. (1998). On-line new event detection and tracking. In *Proceedings of SIGIR* (pp. 37–45).

Aslam, J. A., Ekstrand-Abueg, M., Pavlu, V., Diaz, F., McCreadie, R., & Sakai, T. (2014). TREC 2014 temporal summarization track overview. In *Proceedings of TREC.*

Baker, L. D., & McCallum, A. K. (1998). Distributional clustering of words for text classification. In *Proceedings of the 21st annual international ACM SIGIR conference on research and development in information retrieval* (pp. 96–103). ACM (1998)

Barak, L., Dagan, I., & Shnarch, E. (2009). Text categorization from category name via lexical reference. In *Proceedings of human language technologies: the 2009 annual conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers, NAACL-Short '09* (pp. 33–36), Association for Computational Linguistics, Stroudsburg, PA, USA. http://dl.acm.org/citation.cfm?id=1620853.1620864

Berger, A.L., & Lafferty, J.D. (1999). Information retrieval as statistical translation. In *Proceedings of SIGIR* (pp. 222–229).

Biemann, C. (2006). 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). Association for Computational Linguistics.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent dirichlet allocation. the. *Journal of machine Learning research*, *3*, 993–1022.

Boros, E., Kantor, P. B., & Neu, D. J. (2001). A clustering based approach to creating multi-document summaries. In *Proceedings of the 24th annual international ACM SIGIR conference on research and development in information retrieval.*

De Boom, C., Van Canneyt, S., Demeester, T., & Dhoedt, B. (2016). Representation learning for very short texts using weighted word embedding aggregation. *Pattern Recognition Letters*, *80*, 150–156.

Denkowski, M., & Lavie, A. (2010). Meteor-next and the meteor paraphrase tables: Improved evaluation support for five target languages. In *Proceedings of the joint fifth workshop on statistical machine translation and MetricsMATR*, (pp. 339–342). Association for Computational Linguistics.

Dhillon, I.S., Mallela, S., & Modha, D.S. (2003). Information-theoretic co-clustering. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 89–98). ACM.

Di Marco, A., & Navigli, R. (2013). Clustering and diversifying web search results with graph-based word sense induction. *Computational Linguistics*, *39*(3), 709–754.

Dietterich, T. G. (1998). Approximate statistical tests for comparing supervised classification learning algorithms. *Neural computation*, *10*(7), 1895–1923.

Erkan, G., & Radev, D. R. (2004). Lexrank: graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, *22*, 457–479.

Fellbaum, C. (Ed.). (1998). *WordNet: An electronic lexical database*. Cambridge: MIT Press.

Ferraresi, A., Zanchetta, E., Baroni, M., & Bernardini, S. (2008). Introducing and evaluating ukwac, a very large web-derived corpus of english. In *Proceedings of the 4th web as corpus workshop (WAC-4) can we beat Google* (pp. 47–54).

Gabrilovich, E., & Markovitch, S. (2006). Overcoming the brittleness bottleneck using Wikipedia: Enhancing text categorization with encyclopedic knowledge. *Proceedings of AAAI*, *6*, 1301–1306.

Ganitkevitch, J., Van Durme, B., & Callison-Burch, C. (2013). Ppdb: The paraphrase database. In *HLT-NAACL* (pp. 758–764).

Glickman, O., Shnarch, E., & Dagan, I. (2006). Lexical reference: A semantic matching subtask. In *Proceedings of the 2006 conference on empirical methods in natural language processing, EMNLP '06* (pp. 172–179). Association for Computational Linguistics, Stroudsburg, PA, USA. http://dl.acm.org/citation.cfm?id=1610075.1610103

Green, S. J. (1999). Building hypertext links by computing semantic similarity. *IEEE Transactions on Knowledge and Data Engineering*, *11*(5), 713–730.

Habash, N., & Dorr, B. (2003). Catvar: A database of categorial variations for english. In *Proceedings of the MT summit* (pp. 471–474).

Hearst, M.A., Karger, D.R., & Pedersen, J.O. (1995). Scatter/gather as a tool for the navigation of retrieval results. In *Working Notes of the 1995 AAAI fall symposium on AI applications in knowledge navigation and retrieval*.

Hofmann, T. (1999). Probabilistic latent semantic indexing. In *Proceedings of SIGIR* (pp. 50–57).

Hotho, A., Staab, S., Stumme, G. (2003). Ontologies improve text document clustering. In *Third IEEE international conference on data mining, 2003. ICDM 2003* (pp. 541–544). IEEE.

Hu, J., Fang, L., Cao, Y., Zeng, H.J., Li, H., et al. (2008). Enhancing text clustering by leveraging wikipedia semantics. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 179–186). ACM.

Hu, X., Zhang, X., Lu, C., Park, E.K., & Zhou, X. (2009). Exploiting wikipedia as external knowledge for document clustering. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 389–396). ACM.

Karimzadehgan, M., & Zhai, C. (2010). Estimation of statistical translation models based on mutual information for ad hoc information retrieval. In *Proceedings of the 33rd international ACM SIGIR conference on research and development in information retrieval* (pp. 323–330). ACM.

Kaufman, L., & Rousseeuw, P. (1990). *Finding groups in data: an introduction to cluster analysis*. Applied probability and statistics section (EUA): Wiley series in probability and mathematical statistics.

Kenter, T., & De Rijke, M. (2015). Short text similarity with word embeddings. In *Proceedings of the 24th ACM international on conference on information and knowledge management* (pp. 1411–1420). ACM.

Kotlerman, L., Dagan, I., Gorodetsky, M., & Daya, E. (2012a). Sentence clustering via projection over term clusters. In: **SEM 2012: The first joint conference on lexical and computational semantics—Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the sixth international workshop on semantic evaluation (SemEval 2012)* (pp. 38–43). Association for Computational Linguistics, Montréal, Canada (2012). http://www.aclweb.org/anthology/S12-1005

Kotlerman, L., Dagan, I., Magnini, B., & Bentivogli, L. (2015b). Textual entailment graphs. *Natural Language Engineering*, *21*(5), 699–724.

Kotlerman, L., Dagan, I., Szpektor, I., & Zhitomirsky-Geffet, M. (2010). Directional distributional similarity for lexical inference. *Natural Language Engineering*, *16*(4), 359–389. http://dblp.uni-trier.de/db/journals/nle/nle16.html#KotlermanDSZ10

Kurland, O. (2009). Re-ranking search results using language models of query-specific clusters. *Information Retrieval*, *12*(4), 437–460.

Levy, O., & Goldberg, Y. (2014). Dependencybased word embeddings. In *Proceedings of the 52nd annual meeting of the association for computational linguistics* (Vol. 2, pp. 302–308).

Li, W., & McCallum, A. (2006). Pachinko allocation: Dag-structured mixture models of topic correlations. In *Proceedings of ICML* (pp. 577–584).

Liebeskind, C., Kotlerman, L., Dagan, I. (2015). *Text categorization from category name in an industry-motivated scenario. Language resources and evaluation* (pp. 1–35).

Lin, D. (1998). Automatic retrieval and clustering of similar words. In *COLING-ACL* (pp. 768–774).

Liu, X., & Croft, W.B. (2004). Cluster-based retrieval using language models. In *Proceedings of SIGIR* (pp. 186–193).

Metzler, D., Dumais, S., & Meek, C. (2007). *Similarity measures for short segments of text*. Berlin: Springer.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems* (pp. 3111–3119).

Naughton, M., Kushmerick, N., & Carthy, J. (2006). Clustering sentences for discovering events in news articles. In *Advances in information retrieval* (pp. 535–538). Springer.

Nomoto, T., & Matsumoto, Y. (2001). A new approach to unsupervised text summarization. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 26–34). ACM.

Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. *EMNLP*, *14*, 1532–1543.

Phan, X.H., Nguyen, L.M., & Horiguchi, S.: Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In *Proceedings of the 17th international conference on world wide web*, (pp. 91–100). ACM.

Raiber, F., Kurland, O., Radlinski, F., & Shokouhi, M. (2015). Learning asymmetric co-relevance. In *Proceedings of ICTIR* (pp. 281–290).

Rose, T., Stevenson, M., & Whitehead, M. (2002). The reuters corpus volume 1-from yesterday's news to tomorrow's language resources. *LREC*, *2*, 827–832.

Rosenberg, A., & Binkowski, E. (2004). Augmenting the kappa statistic to determine interannotator reliability for multiply labeled data points. In: *Proceedings of HLT-NAACL 2004: short papers* (pp. 77–80). Association for Computational Linguistics.

Sahami, M., Heilman, T.D.: A web-based kernel function for measuring the similarity of short text snippets. In *Proceedings of WWW* (pp. 377–386).

Sedding, J., & Kazakov, D. (2004). Wordnet-based text document clustering. In *Proceedings of the 3rd workshop on robust methods in analysis of natural language data* (pp. 104–113). Association for Computational Linguistics.

Severyn, A., & Moschitti, A. (2015). Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval* (pp. 373–382). ACM.

Shehata, S. (2009). A wordnet-based semantic model for enhancing text clustering. In *IEEE international conference on data mining workshops, 2009. ICDMW'09*, (pp. 477–482). IEEE.

Shnarch, E., Barak, L., & Dagan, I. Extracting lexical reference rules from wikipedia. In *Proceedings of the joint conference of the 47th annual meeting of the ACL and the 4th international joint conference on natural language processing of the AFNLP: Volume 1-Volume 1, ACL '09* (pp. 450–458). Association for Computational Linguistics, Stroudsburg, PA, USA. http://dl.acm.org/citation.cfm?id=1687878.1687942

Steinbach, M., Karypis, G., Kumar, V., et al. (2000). A comparison of document clustering techniques. In *KDD workshop on text mining* (Vol. 400, pp. 525–526). Boston

Tan, B., Velivelli, A., Fang, H., & Zhai, C. (2007). Term feedback for information retrieval with language models. In *SIGIR 2007: proceedings of the 30th annual international ACM SIGIR conference on research and development in information retrieval* (pp. 263–270), Amsterdam, The Netherlands, July 23–27, 2007. https://doi.org/10.1145/1277741.1277788.

Tsur, O., Littman, A., & Rappoport, A. (2013). Efficient clustering of short messages into general domains. In *Proceedings of ICWSM*.

Udupa, R., Bhole, A., & Bhattacharyya, P. (2009). "A term is known by the company it keeps": On selecting a good expansion set in pseudo-relevance feedback. In *Proceedings of second international conference on the theory of information retrieval, advances in information retrieval theory, ICTIR 2009* (pp. 104–115), Cambridge, UK, September 10–12, 2009. https://doi.org/10.1007/978-3-642-04417-5

Whissell, J. S., & Clarke, C. L. (2011). Improving document clustering using okapi bm25 feature weighting. *Information retrieval*, *14*(5), 466–487.

Ye, H., & Young, S. (2006). A clustering approach to semantic decoding. In: *Ninth international conference on spoken language processing*.

Zamir, O., & Etzioni, O. (1998). Web document clustering: A feasibility demonstration. In *Proceedings of SIGIR* (pp. 46–54).