Journal of Intelligent Information Systems (2019) 53:69–92
https://doi.org/10.1007/s10844-018-0541-4

Clustering of semantically enriched short texts

Marek Kozlowski[1] · Henryk Rybinski[1]

Received: 19 January 2018 / Revised: 23 October 2018 / Accepted: 12 December 2018 /
Published online: 18 December 2018

Abstract
The paper is devoted to the issue of clustering small sets of very short texts. Such texts are often incomplete and highly inconclusive, so establishing a notion of proximity between them is a challenging task. In order to cope with polysemy we adapt the SenseSearcher algorithm (SnS), by Kozlowski and Rybinski in Computational Intelligence 33(3): 335–367, 2017b. In addition, we test the possibilities of improving the quality of clustering ultra-short texts by means of enriching them semantically. We present two approaches, one based on neural-based distributional models, and the other based on external knowledge resources. The approaches are tested on SnSRC and other knowledge-poor algorithms.

Keywords Document clustering · Information retrieval · Semantic enrichment

1 Introduction

Over the last decade, short text clustering has become an active research area in many tasks of Natural Language Processing (NLP). The methods are especially important in grouping the results of information retrieval, mainly in order to reveal various meanings within groups of results. Generally, the methods are based on representing text as a bag-of-words (*bow*), and grouping texts on the basis of their lexical similarity. Most approaches are focused on documents containing at least the size of few tens of words (Ferragina and Scaiella 2012), or a web snippet (Di Marco and Navigli 2013), or a paragraph (Shrestha et al. 2012).[1]

There are various issues that impact clustering quality. Di Marco and Navigli (2013) indicate ambiguity of texts as one of the main problems. Another issue, addressed by Pinto et al. (2007), is the problem with clustering short texts with narrow domain characteristics. In any case, the length of texts seems to be the main issue; therefore the clustering of short

[1] In our experience texts with some 100–300 words can be considered medium, texts with less than 50 words are considered short, and texts with less than 30 words are very short.

✉ Marek Kozlowski
m.kozlowski@ii.pw.edu.pl

Henryk Rybinski
h.rybinski@ii.pw.edu.pl

1 Warsaw University of Technology, Warsaw, Poland

texts, such as snippets, micro-blogs, short texts in questionnaires, etc., has recently become an important challenge.

Actually, the difficulties with short text clustering result from text ambiguity and the lack of a significant number of features in document vectors. Solutions for this are usually based on semantics injection, i.e., by expanding features with additional external information or by assigning meaning to the textual layer of documents (see e.g. Metzler et al. (2007)).

The problem is even more difficult with small text collections containing very short texts (Kotlerman et al. 2015, 2017). One can identify various applications that require the clustering of small text collections composed of a few hundred very short texts. Typical examples include the clustering of transcripts of calls in call centers for the analysis of user interactions, or the clustering of incoming stream of short news articles.

Our problem results from practical needs. In particular, a company runs a business, which consists in organizing brain-storming seminars for the staff of their clients. During the seminars, participants answer a series of questions. A back-office application clusters responses so that one can identify problems in the client's company. The clusters of responses are presented to the audience in almost real time in order for strong ideas and weak signals easily to be identified. Participants can react to the results and create a dialogue. All the participants' responses are free text answers. The answers are short, often incomplete, and highly biased toward the questions, so establishing a measure of proximity between such texts is a difficult issue.

Although we begin with a specific task, we would like to analyze it in a more general way. Kozlowski and Rybinski (2017b) have shown interesting features of the SnS algorithm, which is a novel text mining approach to sense induction. SnS is a knowledge-poor algorithm, and to a large extent it is a language independent approach. It has been also shown that, based on SnS, it is possible to build an efficient text clustering algorithm, named SnSRC.

Unfortunately it turns out that for applications that work on small repositories of very short texts the clustering quality of SnSRC is not as good as for typical applications with medium to large texts clustering. In this paper we investigate how SnSRC can be modified so that it can be efficiently used to cluster short texts. In particular, the goal of this paper is to revise our approach so that SnS can be used for clustering very short texts generated *ad hoc* in a brainstorming session; this will mean that so that the manual work of ordering the above mentioned discussion material is minimized.

This paper is an extension of Kozlowski and Rybinski (2017a), presented on ISMIS '2017. Here, we provide a detailed description of the SnSRC algorithm, and present its improvements, as compared to the ISMIS version. Additionally, in this paper, we discuss various ways of expanding text features in more detail. We have verified a number of well-known short text clustering approaches, including data-centric, description-centric, and semantically-enhanced ones. Here, we provide more experiments with SnSRC and other clustering algorithms, combining the considered algorithms with data expansion methods, and showing how, in general, the semantic enrichment can influence the quality of clustering algorithms.

The paper is structured as follows: In Section 2 we discuss related work. Section 3 is devoted to presenting general concepts concerning SnSRC. Then in Section 4 we present our approach in more detail. Section 5 discusses the experiments and Section 6 presents the obtained results. The conclusions are in Section 7.

2 Related work

The clustering of short texts is an emerging field of research. Because of the shortness of the texts and the sparsity of the occurrences of terms, short texts suffer from a lack of contextual information, which may lead to their ambiguity. As most words occur only once in a short text, the representation of texts by *term frequency-inverse document frequency* (*tf-idf*) vectors cannot work well in this case. Without any contextual information and with only a small number of words available in the document, achieving semantic comparisons at an acceptable level is a challenge.

The problems with short texts can be analyzed from various points of view. An overview of the issues is presented by Shrestha et al. (2012). Here, the authors consider several classical data mining clustering algorithms (Complete Link, Single Link, Average Link Hierarchical clustering and Spectral clustering), and analyze various quality measures from the point of view of the usability of the clustering results, concluding that the values given by the evaluation methods do not always represent the readability of the clusters.

The issue of similarity measure is discussed by Sahami and Heilman (2006). The authors propose a novel method for measuring the similarity between short text snippets (even those without any overlapping terms) by leveraging web search results to provide a wider context for short texts. Then, a similarity kernel function is defined, so that it works with the use of initial texts and Google search engine.

The issue of similarity measure is also discussed by Metzler et al. (2007). The authors study the problem from an information retrieval perspective. As for short texts the standard text similarity measures perform poorly the authors analyze usefulness of purely lexical measures, stemming, and language modeling-based measures.

One of the approaches to short text clustering is to enrich the representations of texts with semantics. In order to address the problem, some research focuses on the semantic enrichment of the context of data from external knowledge resources (Hu et al. 2009; Hotho et al. 2003). These proposals are still far from helping to achieve high-level quality clustering; therefore we decided to conduct research on semantic enrichment with the use of new semantic-oriented approaches, namely the BabelNet eco-system,[2] and with neural-network based distributional semantic models.

In general, one can distinguish three main groups of methods: those based on surface representation; sense-enhanced ones; and those based on enriching texts with additional terms from external resources. Below we discuss them in more detail.

2.1 Surface representation methods

Surface Representation methods are usually based on collocation frequencies, and ignore the senses of words. The clustering algorithms of this group can be classified as data-centric, and description-centric.

Data-Centric Text Clustering The data-centric approach focuses more on the problem of data clustering, than on presenting the results to the user. The algorithm Scatter/Gather (Cutting et al. 1992) is an example of this approach. It divides the data set into a small number of clusters and, after the selection of a group, it performs clustering again and proceeds iteratively using the Buckshot-fractionation algorithm. Other data-centric methods use Bisecting k-means or hierarchical agglomerative clustering. The Bisecting k-means algorithm by Steinbach et al. (2000) starts from a single cluster that contains all points. Iteratively, it finds divisible clusters at the bottom level and bisects each of them using k-means, until there are k leaf clusters in total or until no leaf clusters are divisible. If bisecting all

[2]BabelNet is a multilingual encyclopedic dictionary and semantic network.

divisible clusters at the last level would give more than *k* leaf clusters, larger clusters receive higher priority.

Description-Centric Text Clustering Description-centric approaches are focused on describing the resulting clusters. They reveal diverse groups of semantically related documents associated with meaningful, comprehensible and compact text labels. Among the most popular and successful approaches are phrase-based ones that form clusters based on recurring phrases instead of the numerical frequencies of isolated terms. The Suffix Tree Clustering algorithm (STC), presented by Zamir and Oren (1998), employs frequently recurring phrases as both a document similarity feature and a final cluster description. Clustering with the STC algorithm is essentially finding groups of documents sharing a high ratio of frequent phrases.

A different idea of 'label-driven' clustering appears in clustering with committees algorithm (Pantel and Dekang 2002). The Description-Comes-First (DCF) approach reverses the traditional order of cluster discovery. This is a special case within the description-centric approach. Instead of calculating the proximity between documents and then labeling the revealed groups, DCF first attempts to find good, conceptually varied cluster labels and then assigns documents to the labels to form groups.

A good example of the DCF approach is the algorithm called Lingo (Osinski et al. 2004; Osinski and Weiss 2005). Lingo combines common phrase discovery with latent semantic indexing techniques to separate search results into meaningful groups. Lingo uses singular value decomposition of the term-document matrix to select good cluster labels among the candidates extracted from the text (frequent phrases). The algorithm was designed to cluster results from web search engines (short snippets and fragmented descriptions of original documents), and has proven to provide diverse meaningful cluster labels. Lingo bridges existing phrase-based methods and numerical cluster analysis to form readable and diverse cluster descriptions.

2.2 Sense-enhanced text clustering

Phrase-based methods reveal some problems when some senses are dominated, or texts contain various words with the same meaning. Di Marco and Navigli (2011, 2013); Navigli and Crisafulli (2010) present a novel approach for clustering snippets, based on automatically uncovering word senses from raw text. Word Sense Induction (WSI) is performed in order to dynamically acquire an inventory of senses of the input set of texts. Instead of clustering texts based on a surface similarity of the snippets, induced word senses are used to group snippets. The acquisition of senses is done by a graph-based clustering algorithm that exploits cycles in the co-occurrence graph of the query. Then, the results are clustered on the basis of their semantic similarity to the induced senses. Methods of this kind usually need representative external corpora in order to build relevant sense representations. They are called sense-enhanced clustering algorithms.

Another approach to sense-enhanced clustering is presented by Kozlowski and Rybinski (2014, 2017b). Here an algorithm named SnSRC is proposed; it also induces word senses, but does not use any external corpora. Instead, it induces word senses solely from the corpus. As shown, SnSRC is well suited to clustering short to medium texts. More information about SnSRC is provided in Sections 3 and 4.

2.3 Data expansion based clustering

Several ways of using external information resources have been proposed to resolve the problem of insufficient text representation. We categorize them into the following two groups: (1) the ones using knowledge resources; and (2) the ones applying distributional semantic models. Both approaches are tested in our paper in order to verify the impact of text enrichment on clustering quality. In general, *data expansion* is defined as inducing new features and integrating them with original data. In the case of short texts, data expansion consists in producing terms and phrases that do not appear explicitly in the original text. Such new terms extend the original text features, so that expanded data become input for clustering methods. Data expansion is usually achieved in two phases: (1) automatic summarization (obtained by applying either of the two methods mentioned above); and (2) given the summarization results, the retrieval of additional information by means of knowledge resources or distributional models. Automatic summarization can be done by modeling some dense representation or through keyword extraction. Using distributional semantic models (e.g. neural-based distributional models proposed by Mikolov et al. (2013a, b) we can compute a vector derived from words embeddings for a whole text, and this way we obtain summarized texts.

Keyword extraction methods may be categorized by the type of technique used to identify important words. In most cases they are based on the linguistic approach (Justeson and Katz 1995), the statistical approach (Andrade and Valencia 1998), machine learning (Hulth 2003), or knowledge resources (Milne and Witten 2013). Retrieving additional information is based on expanding native features resulting from the summarization phase. In the case of distributional semantic models for texts represented as vectors, top words that are semantically similar (by means of cosine measure) are retrieved.

Once keyphrases are extracted from a short text, methods based on well-structured knowledge resources (like Wikipedia or WordNet) perform a search for an entry in the resource repository that has title or labels equal to, or are at least fuzzy-similar to the extracted keyphrase, and then enrich the text with acquired labels, synonyms, categories, glosses[3] associated with the concepts.

2.3.1 Extending texts using knowledge resources

The idea of using external knowledge resources for various tasks of NLP emerged alongside the dynamic development of very large semantic tools, such as WordNet or later Wikipedia. A lot of research has been done with these resources, for example, WordNet was used by Hotho et al. (2003) for the improvement of clustering by extending short texts with synsets,[4] Wikipedia was used by Gabrilovich and Markovitch (2005, 2009) for improving the semantics of texts. Wikipedia was also used by Ferragina and Scaiella (2012) in order to improve clustering by resolving polysemy and synonymy in short texts by referencing to Wikipedia articles.

In general, Wikipedia can be exploited in two ways – term oriented and text-oriented. The former describes a given term with other words that come from Wikipedia content, whereas the latter approach consists in extracting keywords from a text using a dictionary of terms (defined as Wikipedia article's titles/labels). Milne et al. (2006) use Wikipedia in order to expand data in two steps: (1) given a term extracted from a processed document, it searches for an article with the title or labels equal to the term, or at least containing it; (2) from the found article its labels, senses, translation, or glosses are extracted.

[3]Usually the first paragraphs of an article's definition.

[4]In ontologies dictionaries and thesauri, *synset* is a group of data elements that are considered semantically equivalent for the purposes of information.

Among the text clustering methods one can also distinguish between those that apply well-formed knowledge resources and those that use them as large text repositories. A good example of the first approach is presented by Hotho et al. (2003), where the authors use WordNet as background knowledge, and integrate it into the process of clustering text documents. WordNet is treated here as a general purpose ontology. The original text representations are extended by applying the following strategies: (1) by adding concepts; (2) by replacing terms by concepts – in particular, terms that appear in WordNet are only counted at the concept level, but terms that do not appear in WordNet are not discarded; (3) terms that do not appear in WordNet are discarded, and others are replaced by concepts signatures. Having extended the texts, the documents are subject to clustering by a standard partitional algorithm. This approach is similar to one of the enrichment methods tested in this paper (presented in Section 5.1.3). There are, however, differences: (1) we use BabelNet as a knowledge resource, which is semantically much richer than WordNet as it integrates WordNet with Wikipedia; and (2) the authors experiment on texts from Reuters news, whereas we test our algorithm on much shorter texts.

A method that uses knowledge repository as an external text resource is presented by Banerjee et al. (2007). In this method the conventional bag-of-words representation of text items is augmented with the titles of selected Wikipedia articles. The titles of the retrieved Wikipedia articles serve as additional features for clustering. The experiments were performed on Google News with the use of agglomerative bisections algorithms.

Hu et al. (2009) also propose Wikipedia as an additional generator of features. The authors propose a framework combining internal and external semantics to prepare new term-features that extend the original ones. In their proposed framework, internal semantics represents the features from the original text by applying techniques specific for NLP, such as segmentation or Part of Speech tagging (PoS-tagging), while external semantics represent the features derived from external knowledge bases, which in this case are Wikipedia and WordNet, treated here as text repositories. To evaluate the methods, two clustering algorithms, k-means and EM, are employed on two test collections: Reuters-21578 and 10-category Web Dataset with 1000 texts. The use of external resources improves the original clustering algorithms by some 4%.

All the methods discussed above have been tested on sets containing at least several hundred documents, each composed of at least one paragraph, and of at least 50 words. Our sets, on the other hand, contain texts composed of on average 8 words. by means of statistical co-occurrences models.

The work on applying Wikipedia as a knowledge base is further developed by Flati and Navigli (2014), Krause et al. (2016), Bovi and Navigli (2017), and widespread in the form of BabelNet. BabelNet is a multilingual encyclopedic dictionary and semantic network, and it currently covers 284 languages. It provides both lexicographic and encyclopedic knowledge thanks to the seamless integration of WordNet, Wikipedia, Wiktionary, OmegaWiki, Wikidata and Open Multilingual WordNet. BabelNet encodes knowledge within a labeled directed graph $G = (V, E)$, where V is a set of nodes (concepts) and E is a set of edges connecting pairs of concepts. Each edge is labeled with a semantic relation. Each node contains a set of lexicalizations of the concepts for different languages. The multilingually lexicalized concepts are Babel synsets. At its core, concepts and relations in BabelNet are harvested from WordNet, and Wikipedia.

Babelfy (Bovi and Navigli 2017) is a unified graph-based approach that leverages BabelNet to perform both Word Sense Disambiguation (WSD) and entity linking in any language covered by BabelNet. The WSD algorithm is based on a loose identification of candidate meanings coupled with a densest subgraph heuristic. As shown by Moro et al. (2014), the evaluation of the Babelfy WSD outperforms many state-of-the-art supervised systems in various applications. It is therefore one of our goals to test how the new semantic tools based on Wikipedia can improve clustering of very short texts with the use of SnSRC and other typical text-oriented clustering algorithms.

2.3.2 Neural network based distributional semantic models

Distributional Semantic Models have recently received increased attention alongside the rise of neural architectures for the scalable training of dense vector embeddings. One of the strongest trends in NLP is the use of word embeddings (Huang et al. 2012) by means of vectors, whose relative similarities correlate with semantic similarity. Distributional semantics that has been modeled on neural networks has received an especially substantial increase in attention. The main reason for this is the very promising approach of employing neural network language models trained on large corpora to learn distributional vectors for words (Mikolov et al. 2013b). Mikolov et al. (2013a); Le and Mikolov (2014) introduced the Skip-gram and Continuous Bag-of-Words models, which are efficient methods for learning high-quality vector representations of words from large amounts of unstructured text data.

The word representations computed with neural networks are very interesting as the learned vectors explicitly encode many linguistic regularities and patterns. In their paper, Le and Mikolov (2014) propose the Paragraph Vector, which is an unsupervised framework that learns continuous distributed vector representations for pieces of texts. Taghipour and Ng (2015) show that the performance of conventional supervised WSD systems can be increased by taking advantage of word embeddings as new features. The authors show how word embeddings alone can provide significant improvement over a state-of-the-art WSD system. Iacobacci et al. (2016) take the real-valued word embeddings as new features This approach inspired us to enrich semantically short texts in a similar way.

The most well-known tool in this area is now Word2vec,[5] which allows fast training on huge amount of raw linguistic data. Word2vec takes as its input a large corpus and builds a vector space, typically of a few hundred dimensions, with each unique word in the corpus represented by a corresponding vector.

Bearing in mind the novel approaches in the area of knowledge resources, and neural network based distributional semantic models, we decided to evaluate new tools for enriching very short texts and to investigate which of the approaches can improve SnSRC for clustering small corpora of very short texts. We also evaluate the impact of the new approaches to text-oriented knowledge-poor clustering algorithms, such as STC and Lingo. In particular we focus on enriching short texts by applying:

1. BabelNet/Babelfy, which are well-structured knowledge resources, integrating Wikipiedia and WordNet;
2. neural network based distributional semantic models, built with Word2vec.

To the best of our knowledge, no experiments were done with so small repositories containing very short texts (usually several words), enriching texts on the feature level by the above mentioned approaches.

[5] https://code.google.com/p/word2vec/

3 Basic concepts

In many text-oriented research areas it is common practice to classify words on the basis of their co-occurrence with other words. In fact, this idea is behind the well known distributional hypothesis, which has been explored thoroughly in the field of research in distributional semantics. It says that words with similar contexts will have similar meanings (*"you shall know a word by the company it keeps"* (Firth 1957)). The related techniques vary, starting from basic word co-occurrence matrices, where a vector associated with a given word is created by counting the words that appear in its immediate context (determined by a window of a certain size), and concluding with latent semantic indexing.

Our approach is also grounded on this hypothesis. So, if a term appears in various texts in a similar context, it probably refers to the same meaning. Additionally, we presume that there is only one meaning of the term in one text, even if the term occurs more than once in the text. This assumption has two implications:

1. at each level at which we search for meanings of a term within a given context, once we identify all the texts containing the term and this context (i.e., the given term has a specific meaning), we can drill-down within these texts for submeanings of the given meaning, which are associated with the term and the context,[6]
2. the texts that have been identified as the ones that contain contexts specific for a given meaning at a given hierarchy level cannot be (re)used to look for other nonrelated meanings at the same levels; this means that they can be eliminated from the repository while other meanings are sought.

Further on, we assume that contexts are expressed by closed frequent sets (Pasquier et al. 1999). The set of all closed sets is the lossless representation of all frequent sets in the sense that it uniquely determines the set of all frequent termsets and their exact frequencies. At the same time, the number of closed frequent sets extracted from a corpus can be orders of magnitude smaller than all frequent sets (i.e., all contexts), which means that the set of all closed sets is a concise representation of all possible contexts. Actually, the idea of *closed frequent termset* is similar to the notion of *concept*, originated from the elegant mathematical framework of Formal Concept Analysis (Ganter and Wille 1999). In the area of data mining, closed frequent termsets have (by definition) the feature by which none of their super-sets has the same frequency. So if X and Y are closed termsets, such that $X \subset Y$, we assume that X (more frequent in the corpus than Y) represents a more general meaning than Y. As a result, we are able to discover hierarchies of senses, instead of a flat list of senses.

Let us define the main notions in a formal way. By R we denote a repository, which is composed of a set of texts $\mathcal{P}$. Given the repository R, we can extract a dictionary $\mathcal{D} = \{t_1, t_2, \ldots, t_m\}$ composed of terms – either *simple terms* (one-word terms) or *compound terms*, usually n-grams. In our experiments dictionary $\mathcal{D}$ will be used for building contexts of a term for which we look for senses. We consider two cases: (a) we limit the dictionary only to nouns and noun phrases (usually multi-word terms denoting concepts, definite named entities, etc.), and (b) we build the dictionary from all the words except stop-words. The dictionary may contain both, the terms for which we search for senses, and terms that are used for building contexts. Any set $X \subseteq \mathcal{D}$ of terms will be called *termset*.

[6]So, in the texts containing *apple* in the context indicating that the texts are about the IT company we can drill down for topics related to *iphone, macintosh, apple software* etc.

By $\mathcal{P}(t) \subseteq \mathcal{P}$ we denote a set of all texts containing term t. In other words $\mathcal{P}(t)$ is a set of texts retrieved by a query t. For termsets X and Y we write XY to denote $X \cup Y$. Similarly, for $\{t\} \cup X$ we just write tX. The statistical significance of a termset X in the text repository R is measured by the number of texts, in which all the terms $t \in X$ are present. The number of such texts is called *support*, and it is denoted by $sup(X)$. Termset X is called frequent if it occurs in more than ε texts in R, where ε is a user-defined support threshold. Additionally, X is called *closed frequent termset* iff

$$\forall Y : Y \supset X \Rightarrow \text{sup}(X) > \text{sup}(Y) \tag{1}$$

So, X is a closed termset if any of its (proper) supersets has lower support. We say that X is a context of t if $sup(tX) > 0$ in R.

Definition 1 If a given text $P \in \mathcal{P}(t)$ supports X and there is no other set Y, $Y \supset X$ such that P supports Y, we say that X is a *complete text context* of t. Given term t and text $P \in \mathcal{P}(t)$ we denote the complete text context by $C(P, t)$. A family of complete text contexts describing term t will be denoted as $\mathcal{C}(t)$, and is defined as $\mathcal{C}(t) = \{C(P, t) : P \in \mathcal{P}(t)\}$.

Example 1 Given snippets retrieved for the term $jaguar$, we illustrate below the sample contexts:

1. $C(P_1, jaguar)$= {car, luxury, vehicle, brand}
2. $C(P_2, jaguar)$= {car, manufacturer, English}
3. $C(P_3, jaguar)$= {wild, cat, panther}
4. $C(P_4, jaguar)$= {species, cat, animal}
5. $C(P_5, jaguar)$= {cat, animal, Americas, native}

Let us set the support threshold $\epsilon = 2$, then the family of significant contextual patterns for $jaguar$, retrieved from these contexts are $\{cat\}$ with $\text{sup} = 3$; $\{car\}$ with $\text{sup} = 2$; and $\{cat, animal\}$ with $\text{sup} = 2$.

Now we define a notion of contextual patterns for t. By $F(t)$ we denote a set of all frequent termsets in $\mathcal{C}(t)$. In addition, by $CF(t)$ we denote closed frequent termsets in $\mathcal{C}(t)$.

Definition 2 Given a term t, we define the set of contextual patterns as:

$$\mathcal{CP}(t) = \{X : X \in CF(t)\} \tag{2}$$

Let us define now the notion of *sense frame*. Sense frame is a hierarchical structure organizing contextual patterns, with unlimited a priori number of levels. The root of a sense frame is assigned to the context with a representative label for the senses, and its context is typically discriminative against contexts of other sense frames. The sub-trees are assigned to the contexts, which are super-sets of the main contextual pattern, the patterns of their eventual children are then the super-set of the parents, and so on. As a result, the sense frame is a multi-level sense tree. Given t and $X \in \mathcal{CP}(t)$, we define a set of *direct sub-sense contexts*, denoted by $DSC(X, t)$, as follows:

$$DSC(X, t) = \{Y \in \mathcal{CP}(t) : Y \supset X \wedge \neg\exists Z \in \mathcal{CP}(t) : Z \supset X \wedge Z \subset Y)\} \tag{3}$$

Formally, for a given t and pattern $X \in \mathcal{CP}(t) \cup \emptyset$, a sub-sense context tree, denoted as *s-tree*(X, t), is defined recursively as a sequence of *s*-trees:

$$s\text{-}tree(X, t) = \langle (Y_1, s\text{-}tree(Y_1, t)), \ldots, (Y_n, s\text{-}tree(Y_n, t)) \rangle \tag{4}$$

such that
$n = |DSC(X, t)|$, $Y_i \in DSC(X, t), i = 1...n$, and $\sup(Y_j) \geq \sup(Y_{j+1})$, $j = 1 \dots n-1$. If $X = \emptyset$ then Y_i are called the *main contextual patterns* for t.

Definition 3 Given a term $t \in \mathcal{D}$, such that $\mathcal{CP}(t)$ is not empty, we define a sense frame for t, denoted by $\mathcal{SF}(t)$, as an *s-tree*

$$\mathcal{SF}(t) = \textit{s-tree}(\emptyset, t) \tag{5}$$

Example 2 Let us refer to patterns form Example 1. The family of sense frames for a term $jaguar$ is presented below:

$$\mathcal{SF}(jaguar) = \textit{s-tree}(\emptyset, jaguar) =$$

$$\langle(\{car\}, \textit{s-tree}(\{car\}, jaguar)), (\{cat\}, \textit{s-tree}(\{cat\}, jaguar))\rangle$$

In cases when the corpus is large and representative for the discovered senses, i.e., all the senses of given terms are represented in a sufficient number of texts, sense frames correspond to distinctive senses. In the cases where the corpus is limited, two or more sense frames may refer to the same sense. Therefore, there is a need to compare frames of singletons or less representative senses, and identify similar frames in order to group them into sense clusters. The main idea for merging two or more sense frames is that there are no terms discriminating one sense from the other. Such discriminating terms will be called $discriminants$. Formally, the notion of $discriminant$ is defined below:

Definition 4 Given t, such that $\mathcal{CP}(t)$ is a nonempty set of contextual patterns, and a pattern $Y \in \mathcal{CP}(t)$ we define the set of all discriminants for t and the pattern Y as follows:

$$PD(t, Y) = \{Z \in \mathcal{CP}(t) : relSup(t, Y, Z) \leq \gamma\} \tag{6}$$

where γ is a user-defined threshold, and $relSup$ denotes *relative support*, which is defined as follows:

$$relSup(t, Y, Z) = \frac{\sup(tYZ)}{\min\{\sup(tY), \sup(tZ)\}} \tag{7}$$

By analogy, for t such that $\mathcal{CP}(t) \neq \emptyset$ we can define the complementary set of nondiscriminating contextual pattern $Y \in \mathcal{CP}(t)$:

$$PND(t, Y) = \{Z \in \mathcal{CP}(t) : relSup(t, Y, Z) > \gamma\} \tag{8}$$

We call such patterns *contextual patterns compatible with tY*. The compatible contextual patterns can be clustered in one sense representation.

The sense representation S, also called *sense cluster*, is a set of similar sense frames. Let $root(x)$ denote the root of a sense frame x (the main contextual pattern in a sense frame); and $seed(S)$ denote a sense frame with the highest position in $\mathcal{SF}(t)$ (having the root with the highest support) compared to the other sense frames in the sense cluster S. Finally, the sense is defined as a group of similar sense frames, such that either each clustered sense frame x has $root(x) \in PND(t, root(seed(S)))$, or it fulfills the expression $senseFrameSim(seed(S), x) > \sigma$, where σ is a user-defined threshold, and given sense frames $x, y \in \mathcal{SF}(t)$ the function $senseFrameSim$ is defined as cosine similarity:[7]

$$senseFrameSim(x, y) = \cos(t2v(x), t2v(y)) \tag{9}$$

where $t2v(x)$ and $t2v(y)$ denote the mean vectors of the word embeddings from two $bows$, each obtained as the union of all content words of contextual patterns building sense frames x and y respectively.

4 SnSRC revisited

As mentioned above, SnSRC is a clustering algorithm using the SnS method, which is knowledge-poor. On the basis of the discovered sense frames, SnSRC maps texts to senses, giving, as a result, final clusters. Below we describe the sense induction algorithm SnS and based on it the clustering algorithm SnSRC.

4.1 SnS - word sense induction method

The SnS algorithm consists of five phases, as follows:

Phase I. In this initial stage a full-text search index $L(Corp)$ is built using the corpus containing short texts as input. Technically, the process is performed by Lucene,[8] which at the same time reduces each word to its canonical form; i.e., it removes punctuation and transforms it to lowercase. Additionally, given the repository $Corp$ we extract a dictionary $\mathcal{D} = \{t_1, t_2, \ldots, t_m\}$. Optionally, for building the dictionary we use the NLTK POS-Tagger[9] and a dictionary of compound terms from Wikipedia. Given the inverted index, the algorithm works in a pipeline, i.e., in order to find senses for term t it goes consecutively through phases II to V:

Phase II. Given a term t, the set of texts $\mathcal{P}(t)$ is retrieved with the index $L(Corp)$. Then the texts are iteratively processed using the dictionary $\mathcal{D}$ and stop-word remover, so that finally each paragraph is transformed into a context. The output of this stage is a family of contexts $\mathcal{C}$, passed to Phase III.

Phase III. With the contexts $\mathcal{C}$, as generated in the previous step, contextual patterns $\mathcal{CP}$ are discovered. The contexts are treated as termsets. To identify contextual patterns we look for the closed frequent termsets. The most suitable algorithms for mining closed frequent termsets are CHARM (Zaki and Hsiao 2002) and CLOSET+ (Wang et al. 2003); of the two, we have decided to use CHARM, which turned out to be computationally efficient, and can be easily adapted to text processing.[10] In order to improve CHARM's performance we have modified it by implementing multi-threading and the bitSet representation of covered contexts. Additionally, an inverted index is built simultaneously on closed frequent termsets as they are mined in a given set of contexts. Each new discovered contextual pattern cp is split into particular terms and then they are added to the index.

[7] In Kozlowski and Rybinski (2017b) the Jackard similarity between frames was used. Here, sense clustering is based on the cosine measure (9), which turns out to be better.

[8] Lucene is an open source information retrieval software (http://lucene.apache.org/).

[9] See http://www.nltk.org/ and http://www.nltk.org/book/

[10] Aggarwal and Han (2014) indicate CHARM as the one, which outperforms other solutions. It works especially well with very large data sets, and is still efficient with a low value support threshold, which is very useful for mining text data.

Phase IV. In this phase, sense frames $\mathcal{SF}(t)$ are built from $\mathcal{CP}(t)$. Let us recall that the sense frame is a hierarchical structure containing contextual patterns. Each main contextual pattern has an assigned $s\text{-}tree$, which in turn contains sub-sense contextual patterns. Sub-sense contextual patterns are super-sets of the main contextual pattern, which results in building a tree structure. For each contextual pattern from the ordered list of $\mathcal{CP}(t)$ two constraints are checked: (1) the current contextual pattern cp is verified as to whether it is contained in a previously created sense frame ($cp \notin LC$) and (2) cp is verified as to whether it contains any terms already included in the previously built sense frames ($cp \cap LT = \emptyset$). If both constraints are satisfied, the given contextual pattern is treated as a main pattern, and after having found all its super-set patterns, they are organized in a multi-level tree in the sense frame.

Phase V. In this final stage of SnS, the sense frames $\mathcal{SF}(t)$ are clustered in order to find tight senses $\mathcal{S}(t)$. $\mathcal{SF}(t)$ is a family of tuples containing the main contextual patterns and a set of its sub-patterns represented as trees. SnS compares sense frames, and the similar frames are grouped. The similarity is measured according to the formula $senseFrameSim$, as defined by (9) (Section 3). Additionally, the nondiscriminating contextual patterns can be used in order to cluster sense frames. Two sense frames are clustered if one of them has root mp_i belonging to $PND(t, mp_j)$, where mp_j is a root of another sense frame.

4.2 Clustering

As already mentioned, SnSRC is strictly based on the sense induction algorithm SnS:

1. First, SnS is run; when Phase V is reached a clustering procedure is performed to cluster the frames that are semantically close, so that the representations of senses are created;
2. each generated sense represents a set of frequent termsets derived from the texts, and supporting the given sense; to this end, for each sense the SnSRC algorithm generates a final cluster of these short texts that support the given sense; as a result, for each sense one cluster is created;
3. when this phase is completed there may still remain texts, which are not assigned to any sense, as they do not support any termset qualified as frequent; this means there are single text clusters (singletons); therefore, for each singleton additional steps are performed to find the closest meaning among the existing clusters, and then to add it to such cluster; if nothing is found, the singleton is added to an extra cluster named *others*.

Clustering of the singletons consists in using the *bow* representation and a similarity measure between the snippet representations $b_i \in B$ and sense clusters $\{S_1, \ldots, S_m\}$. Given a snippet r_i, the sense cluster closest to its *bow* representation b_i will be selected as the most likely meaning:

$$Sense(r_i) = \begin{cases} \arg\max\limits_{j=1,..,m} sim(b_i, S_j) & \texttt{if } sim(b_i, S_j) > \sigma \\ m+1 \ \ (Group \ \ Others) & \texttt{otherwise} \end{cases} \tag{10}$$

where $sim(b_i, S_j)$ is a generic similarity value between b_i and S_j, $\sigma > 0$ is a user-defined threshold; the cluster $Others$ denotes that none of the induced senses is assigned to the result snippet. As a result, we obtain a clustering of repository R into $\mathcal{C}$, $\mathcal{C} = (C_1, \ldots, C_m, C_{m+1})$ such that:

$$C_j = \{r_i \in R : Sense(r_i) = j\}$$

i.e., C_j contains the snippets classified with the j-th induced sense of the clustered set.

In SnSRC the similarity measure between the snippet b_i and the previously detected sense cluster S_j is the cosine similarity measure, denoted by $bowSim(b_i, S_j)$, between the vectors $t2v(b_i)$ and $t2v(bow(S_j))$:

$$bowSim(b_i, S_j) = \cos(t2v(b_i), t2v(bow(S_j))) \tag{11}$$

where $t2v(b_i)$ is a simple mean vector of the Word2vec embeddings for the bag of words of i-th snippet b_i and $t2v(bow(S_j))$ is a simple mean vector of the Word2vec embeddings for bag-of-words $bow(S_j)$ representing the j-th sense cluster. Let us recall that the bag $bow(S_j)$ is obtained as union of all content words of contextual patterns building sense frames clustered within sense S_j.

5 Experiments

One of our first tasks was to find the best knowledge-poor clustering method for texts provided by the company. Hence, we tested four algorithms: data-centric (Bisecting k-means), two description-centric algorithms (Lingo by Osinski and Weiss (2005), and STC by Zamir and Oren (1998)), and our solution SnSRC (a sense-enhanced algorithm). Then, we performed experiments aiming at verifying how semantic enrichment can improve short text clustering. For the semantic enrichment of the texts we have used two approaches: (1) Babel eco-system (Babelnet, Babelfy), and (2) neural-network based distributional models, for which we used the Word2vec tool.

5.1 Testing environment

In order to perform our experiments we built an environment for investigating various clustering algorithms applied to semantically enriched text. In the environment, the text data are processed in a pipeline, starting with

1. preprocessing of the input data (language dependent), followed by
2. semantic enrichment of the texts (using the BabelNet resources and Word2vec, respectively), and then
3. clustering by one of the tested algorithms.

Some experiments aimed to find the best algorithms for short texts without enriching them. For such cases step (2) was skipped. In our experiments, we used one general-purpose clustering algorithm (Bisecting k-means), and three text-oriented ones (STC, Lingo and SnSRC).

5.1.1 Preprocessing

Our main goal was to test the algorithms on the seminars data. Additionally, in order to make sure that for such small sets to be clustered the results were not biased towards some characteristics of the data set, we tested the clustering algorithms using additional data sets of web search snippets, namely AMBIENT (Ambiguous Entries)[11] and MORESQUE (More Sense-tagged Queries).[12]

[11] http://credo.fub.it/ambient/

[12] http://lcl.uniroma1.it/moresque/

For each data set the first preprocessing step was to process the texts into *bow* representations. The texts were processed by sentence segmentation, word tokenization, stop-words and non-alphanumeric cleaning, and spell-correction; after this, tokens were stemmed (Snowball Stemmer).[13]

For the English data sets we used dedicated PoS-tagging (NLTK PoS tagger, English part) in order to extract nouns, adjectives, adverbs and verbs, as they are the most informative ones.

The French texts were very often misspelled, therefore in the case of French data, special importance was additionally given to spell correction in this pipeline. This was applied before stemming. We customized the Norvig spell corrector.[14] We also removed from the pipeline the PoS-tagger, because the texts are very short and any word in the text was crucial for clustering.

Finally, independently of which data set was preprocessed, the texts were transformed into the vector space model, and stored in a term-document matrix.

5.1.2 Semantic enrichment by BabelNet

Babelfy (Bovi and Navigli 2017) provides a possibility to identify candidate meanings. In order to examine how BabelNet and Babelfy can influence clustering we used them for enhancing texts with semantic features, such as synsets,[15] glosses, hypernyms and categories describing them.[16] We used the API provided by both resources.

Firstly, Babelfy was used to disambiguate the texts by finding relevant synsets for words. This is performed in three key steps. The preliminary step, called *semantic signatures building*, consists in finding in the input text concepts or named entities in the BabelNet graph (being a lexicalized semantic network), labeling the terms in the text with the corresponding *semantic signatures*, built of a set of related vertices in BabelNet. Then, the unconstrained identification of candidate meanings for all possible textual fragments is performed by extracting all the linked fragments from this text and, for each of them, listing the possible meanings according to the semantic network. The final matching to a proper sense results from the high-coherence densest subgraph algorithm. Then, with the disambiguated words represented by synsets we used BabelNet API in order to access the corresponding semantic entities. In order to perform the experiments we prepared the following text enrichments with the BabelNet API:

1. Babel-synsets – the text is disambiguated using Babelfy, and the retrieved synsets are added, as additional tokens, to the text;
2. Babel-categories – the text is disambiguated using Babelfy, and the retrieved synsets are processed with BabelNet in order to obtain the corresponding categories, which are then added to the text as additional tokens;
3. Babel-glosses – the text is disambiguated using Babelfy, and the retrieved synsets are processed with BabelNet in order to get the corresponding glosses, which are added to the text as additional phrases;

[13]http://snowball.tartarus.org/texts/introduction.html

[14]http://norvig.com/spell-correct.html

[15]https://babelfy.io/v1/disambiguate

[16]https://babelnet.io/v3/getSynset

4. Babel-hypernyms – the text is disambiguated using Babelfy, and the retrieved synsets are processed with BabelNet in order to get the corresponding hypernyms, which are then added as additional tokens to the text.

5.1.3 Semantic enrichment by Word2vec

To investigate how the neural network based distributional model can improve clustering quality in short textual answers, firstly we trained the model with continuous bag of words (*cbow*) and negative sampling[17] using raw textual data. In the *cbow* model the context is represented by multiple words for a given target word. For example, we could use *sweet, red* and *fruit* as context words for *strawberry*. In the *cbow* model the hidden layer output is the average of word vectors corresponding to context words at input.

After building the word embeddings for each word in the vocabulary for each text, we compute cosine similarity between a simple mean vector of the embeddings for the text's words and the vectors for each word in the distributional model. Then, the retrieved words with top semantic similarity are added as additional term features to the initial bow text representation.

We tested Word2vec models for three various dimensions, reducing the original vector space to 100, 300 and 500 dimensions, and verified experimentally that the latter size provides the most relevant words. Therefore, in our experiments we apply space dimensionality equal to 500.

5.2 Data

As mentioned above, for the experiments we used sets of custom French texts from seminars, as well as additional data sets – web search snippets.

5.2.1 French data

The gold standard data set used for the experiments was given as a set of XML files, each one in the form representing answer texts to a query, grouped into final clusters. Each cluster was manually assigned a label expressing the meaning of the cluster. The obtained gold standard set contained 60 files (each representing answers to a query). An example of one answer in the XML form is provided in Fig. 1. The tag *answer* contains a short text which is one of the answers to a seminar question with *id*=203. The answer texts are members of a set to be clustered; the parameters *id, lv*, and *origin* denote the organizational data concerning the seminar, participants, and answers.

Here are some details about this data set (called UltraShortAnswers):

1. the number of answers for each question was between 21 to 97 short texts, on average 49.28 answers per question;
2. the total number of answers in all files of the gold standard was 2957;
3. the length of each answer varied from 3 to 20 words (on average 8), after preprocessing this was reduced to 5 words on average;
4. on average, there were 7.85 manually created clusters for a question.

[17]This is a training method of Word2vec models that approaches the maximization problem by minimizing the log-likelihood of sampled negative instances (Mikolov et al. 2013a).

```
<group id="203" lv="0">
  <question id=21>
      <answer id="1049" origin="30">
              Ouverture: partager une veille
              commerciale ,
              technologique , manageriale ,
              innovation , performance
      </answer>
       ... ... ...
      <answer id="1050" origin="33">
           ... ...
      </answer>
  </question>
</group>
```

Fig. 1 Example of XML with answers

The problem is not only with the length of the texts but also with the number of texts to be clustered. Hence, in order to make sure that for such small sets to be clustered SnSRC is not biased towards some characteristics of the data set, we decided to test the algorithms using additional data sets, namely sets of web search snippets; the additional data sets are described below.

5.2.2 Web search snippets

For testing the clustering algorithms with other data sets we used AMBIENT and MORESQUE. The sets contain snippets, returned by a search engine. The snippets are short texts, often incomplete, and highly biased toward the query, but usually consist of 2-4 sentences (giving the size of the bag-of-words 3–5 times longer when compared to UltraShortAnswers). Below some details about AMBIENT and MORESQUE are provided:

1. AMBIENT consists of 44 topics, each with a set of subtopics and a list of 100 ranked documents (Carpineto and Romano 2008).
2. MORESQUE consists of 114 topics, each with a set of subtopics and a list of 100 top-ranking documents. The dataset has been developed by Navigli and Crisafulli (2010) as a complement for AMBIENT.

5.3 Scoring

In the literature one can find many measures for evaluating clustering quality. In our experiments we have considered Rand Index (RI) and Adjusted Rand Index (ARI). Given a clustering gold standard, say $\mathcal{G}$, the Rand Index of clustering $\mathcal{C}$ is a measure of accordance between the clusterings $\mathcal{C}$ and $\mathcal{G}$, based on text pairs evaluated across the clusterings. RI is calculated as follows:

$$RI(\mathcal{C},\mathcal{G}) = \frac{TP+TN}{TP+TN+FP+FN}$$

where TP is the number of true positives (i.e., pairs of texts) which are in the same cluster both in $\mathcal{C}$ and $\mathcal{G}$, TN is the number of true negatives, i.e., number of pairs which are in different clusters in both clusterings, and FP and FN are, respectively, the number of false positives and false negatives. RI ranges between 0 and 1, where 1 indicates the perfect correspondence.

The main weakness of RI is that it does not take into account the probabilistic nature of clustering, so if the clustering $\mathcal{C}$ were obtained by a random creation of clusters, the measure for such a random clustering algorithm should be set to a reference value. This problem is solved by *ARI*, which sets the expected index value for random clustering to 0 (Manning et al. 2008). It corrects the RI and makes it vary according to expected value for random clustering:

$$ARI(\mathcal{C}, \mathcal{G}) = \frac{RI(\mathcal{C}, \mathcal{G}) - E(RI(\mathcal{C}_{\mathcal{R}}, \mathcal{G}))}{\max RI(\mathcal{C}, \mathcal{G}) - E(RI(\mathcal{C}_{\mathcal{R}}, \mathcal{G}))}$$

where $E(RI(\mathcal{C}_{\mathcal{R}}, \mathcal{G}))$ is the expected value of the RI. The ARI measure ranges between -1 and $+1$ and is 0 when the index is equal to its expected value. Given two clusterings $\mathcal{C} = \{C_1, \ldots, C_m\}$ and $\mathcal{G} = \{G_1, G_2, \ldots, G_g\}$, firstly we quantify the degree of overlapping between $\mathcal{C}$ and $\mathcal{G}$ using the contingency table.

As the company was using entropy-based measures, namely Homogeneity, Completeness, and V-measure (harmonic mean of homogeneity and completeness), we also used these measures in our preliminary experiments. However, these measures are not normalized with regard to random labeling. This problem can safely be ignored, when the number of samples is higher than a thousand and the number of clusters is less than 10. Therefore, let us recall that in our experiments we worked with sets containing from 20 to 100 answers, which means that the entropy-based measures may be misleading.

In order to demonstrate the problem with the entropy-based measures and to show how they behave, we decided to experiment with Lingo and a data set with a small number of short texts. It turned out to be quite interesting, and two paradoxes in particular were revealed. They are illustrated in Table 1. In the clustering experiments, numbered in the table from 1 to 8, we measured homogeneity and completeness of the clustering results, increasing the number of clusters in each consecutive experiment. As can be noted:

1. with an increasing number of clusters, the entropy-based completeness also grows, despite the fact that more elements should be missing in each cluster;
2. even more interestingly, both the homogeneity and completeness increase in line with an increase in the number of clusters – so with almost 2.5 times more clusters than in the gold standard, the entropy-based parameters are both better than for clustering giving similar results to gold standard (as a rule, the gains in precision should cause loss of completeness).
3. At a given point (experiment 7) the ARI measure starts decreasing, though both homogeneity and completeness still increase.

The paradoxes result from data sets that are too small for the entropy-based measures. Hence, in the our experiments presented here, we used only ARI.

6 Results

As a first step of our experiments, we compared one general-purpose clustering algorithm (Bisecting k-means), and three text-oriented ones (STC, Lingo and SnSRC), all of them run on the company data, and without semantic enrichment. Let us note that Bisecting k-means requires defining a priori the number of resulting clusters, whereas the other algorithms automatically determine the number of clusters. Therefore in this phase we performed two

Table 1 The paradox of using the entropy-based measures

Measure	1	2	3	4	5	6	7	8
ARI	7.98	11.11	11.26	12.62	13.34	15.00	14.85	14.22
Homogeneity	28.79	35.18	36.33	43.72	55.74	64.28	66.48	67.12
Completeness	38.39	39.99	40.44	41.92	42.01	43.27	44.49	45.21
C	6.48	7.85	8.45	10.32	13.02	16.95	19.25	20.01
GC	7.85	7.85	7.85	7.85	7.85	7.85	7.85	7.85

C – the average number of obtained clusters; GC – the average number of clusters in the gold standard

series of experiments: the first one with a limited number of clusters,[18] and the second one with Bisecting k-means run with a predefined limitation k, while Lingo, STC and SnSRC were used in a way where k was defined automatically by the algorithms. Clearly, in the case where the clustering algorithm automatically determines the number of clusters during its run, it is important how much the number of clusters obtained differs from the number of clusters in the gold standard. Therefore, in the results of our experiments we provide both the average number of clusters and the average number of clusters in gold standards.

The results of experiments from this phase are provided in Table 2a and b. Table 2a presents the following results: for Bisecting k-means the number of clusters to be obtained was set to the optimal value k. In the case of STC, Lingo and SnsRC, the evaluations were performed without defining $a\ priori$ any limitation for the number of resulting clusters. Table 2b shows the results for the case where STC, Lingo, and SnSRC also had a defined limited number of clusters k. As one can see, for French ultra short texts, only text-oriented clustering methods give reasonable results, whereas the clustering quality of Bisecting k-means in these experiments is very low. In addition, SnSRC outperforms STC and Lingo in both cases.

The second series of experiments was performed in order to verify how the text-oriented clustering algorithms work with other data. We applied the algorithms for clustering the AMBIENT and MORESQUE data sets. Let us recall that both sets contain longer texts that are in English (yahoo snippets). In addition, the MORESQUE data set is semantically very heterogeneous, which results in knowledge-poor algorithms that provide much worse results. To this end we tested the algorithms for two combinations of data; namely (a) the AMBIENT + MORESQUE data sets, and (b) the AMBIENT data set only. Table 3a shows the clustering measures for these algorithms for AMBIENT + MORESQUE, while Table 3b presents the measures for AMBIENT data only. This experiment confirms a very low quality of Lingo and STC for heterogeneous texts. SnSRC also reveals lower quality, but it is still much better than Lingo and STC. In addition, even for more advantageous data the SnSRC algorithm outperforms the other tested algorithms.

In the last series of experiments we focused on analyzing how semantic enrichment of texts influences the clustering quality. Here, we experimented with Lingo and SnSRC. The experiments were run on the AMBIENT data (Table 4), and for UltraShortAnswers data (Table 5). For semantic enrichment we used the BabelNet tools, and Word2vec, as described in Sections 5.1.2 and 5.1.3 respectively. In both tables the first row presents the baseline, which corresponds to the algorithm without the semantic enrichment of texts. The next

[18]In order to make this possible the algorithms SnSRC, STC and Lingo were specially adapted by grid search parameters optimization to generate an a priori given number of clusters.

Table 2 The quality evaluation of the clustering methods measured by ARI (percentages) for the UltraShort-Answers data set with and without a constraint on the number of clusters

Method	ARI	C	GC
(a)			
k-means	3.71	9.45	7.85
STC	11.35	8.61	7.85
Lingo	12.62	10.32	7.85
SnSRC	**16.18**	10.55	7.85
(b)			
k-means	4.61	7.85	7.85
STC	11.21	7.85	7.85
Lingo	11.11	7.85	7.85
SnSRC	**12.93**	7.85	7.85

C – the average number of clusters obtained; GC – the average number of clusters in gold standard

(a) – without a predefined constraint on the number of clusters

(b) – with a predefined constraint on the number of clusters

The bold values present the highest values of the measures

rows present the results obtained with the evaluated types of semantic enrichment of texts – specifically the first four rows with BabelNet tools, and the last two rows with Word2vec:

1. Babel-synsets
2. Babel-categories
3. Babel-glosses
4. Babel-hypernyms
5. Word2Vec-1 presents the results of the neural network based model trained on texts taken from French and English Wikipedia;
6. Word2Vec-2 presents the results of the neural network based model trained on the test corpus (company corpus of manually created clusters of seminar answers, or test corpus AMBIENT).

Table 4 presents the results for Lingo (left table) and SnSRC (right table). As one can see, for both tested algorithms an improvement can be obtained through the approach labeled as Babel-synsets, i.e., consisting in disambiguating texts with the Babelfy tools and adding

Table 3 A comparison of text clustering algorithms (percentages)

Algorithm	ARI	C	GC
(a)			
Lingo	–0.53	2.0	6.16
STC	–7.90	2.0	6.16
SnSRC	**14.24**	14.5	6.16
(b)			
Lingo	18.09	11.15	7.93
STC	23.05	10.05	7.93
SnSRC	**28.24**	9.65	7.93

(a) – on the AMBIENT +MORESQUE data set;

(b) – on the AMBIENT data only

The bold values present the highest values of the measures

Table 4 ARI scores (percentages) obtained by Lingo (left) and SnSRC (right) for English AMBIENT texts, enriched by BabelNet and Word2vec

Method	ARI	C	GC	Method	ARI	C	GC
Baseline - Lingo	18.09	11.15	7.93	Baseline - SnSRC	28.24	9.65	7.93
Babel-synsets	**18.61**	10.01	7.93	Babel-synsets	**29.90**	9.02	7.93
Babel-categories	17.01	9.55	7.93	Babel-categories	27.30	8.53	7.93
Babel-glosses	12.27	8.33	7.93	Babel-glosses	23.77	7.73	7.93
Babel-hypernyms	16.35	9.65	7.93	Babel-hypernyms	26.15	9.95	7.93
Word2Vec-1	18.05	9.88	7.93	Word2Vec-1	28.01	9.28	7.93
Word2Vec-2	**19.12**	9.38	7.93	Word2Vec-2	**29.92**	9.08	7.93

The bold values present the highest values of the measures

appropriate synsets to the texts. The relative improvement for Lingo and SnSRC applied on the AMBIENT data sets is some 2,87% and 5,87% respectively. On the other hand, the approaches labeled Babel-categories, Babel-glosses, and Babel-hypernyms do not improve either of the tested algorithms.

Let us also note that both semantic enrichment approaches based on Word2vec improve the original clustering algorithms. What is interesting here is that in both cases the model trained on the corpus (Word2Vec-2) gives essentially better results than the model trained on Wikipedia (i.e., Word2Vec-1).

Finally, we performed tests with SnSRC for the French seminar data. Table 5 shows the results obtained by SnSRC for the French seminar data. Let us note that with the Babel-synset approach the quality of the SnSRC clustering for the UltraShort data set is better, however the improvement is not so meaningful as for AMBIENT. This may result from too short texts in the UltraShort data set. As in the case of AMBIENT + SnSRC and AMBIENT + Lingo, the other BabelNet approaches do not improve the French data set clustering either.

In the case of French data, i.e., the data containing ultra short texts, the experiments also showed that the best clustering results are obtained for texts enriched by Word2Vec-2. The relative improvement against the baseline reached 8,89% in this case.

As can be noted, in all experiments for both Lingo and SnSRC Word2Vec-2 outperforms Word2Vec-1. It seems to result from the fact that the texts enrichments coming from the

Table 5 ARI scores (percentages) obtained by SnSRC for French seminar texts, enriched by BabelNet and Word2vec

Method	ARI	C	GC
Baseline - SnSRC	16.18	10.55	7.85
Babel-synsets	**16.23**	10.04	7.85
Babel-categories	15.13	10.55	7.85
Babel-glosses	14.03	9.33	7.85
Babel-hypernyms	15.21	9.56	7.85
Word2Vec-1	16.42	10.38	7.85
Word2Vec-2	**17.62**	9.98	7.85

The bold values present the highest values of the measures models built on the domain-oriented texts are more relevant than those built on a general purpose repository, like Wikipedia.[19]

We used some tools for French language, namely PoS tagging and spell-checker. Still, some more advanced language-oriented tools could be probably applied, so one may expect an additional improvement with such tools. Our experiments show the value of SnSRC as a knowledge-poor approach, especially suitable for the cases where no advanced language means are available.

7 Conclusions

In the paper we have presented an approach to clustering small to medium text corpora containing very short texts, which is based on the semantic enrichment of the texts. The semantic enrichment in the preprocessing step is a general-purpose approach. It expands the initial texts with additional features (tokens representing categories, synsets, glosses, hypernyms, or similar words), and does not influence the text clustering methods. The clustering methods still receive texts as an input, which can be interpreted as bag of words, so this methodology can be very easily deployed.

We tested four text enrichment methods that are based on the BabelNet/Babelfy resources, and two methods that are based on neural network based distributional models. The methods were applied for three knowledge-poor text clustering algorithms, namely STC, Lingo, and SnSRC. They represent two various text clustering approaches, namely description centric text clustering (STC and Lingo), and sense-enhanced text clustering (SnSRC).

The first phase of our experiments was devoted to the evaluation of the three text-oriented clustering algorithms, i.e., STC, Lingo and SnSRC, and Bisecting k-means, which is a general purpose clustering algorithm. The algorithms were tested with the ultra-short French texts from brain-storming seminars. It was shown that only text-oriented clustering methods give reasonable results for French ultra short texts, whereas the clustering quality of Bisecting k-means in these experiments is very low. In these experiments we also showed that SnSRC outperforms the other two text-oriented algorithms, especially in the case when the number of clusters is not predefined.

The second phase of experiments was devoted to verifying how the text-oriented clustering algorithms work with other data. In the experiments the algorithms were tested for two cases: (1) the combination of data sets AMBIENT + MORESQUE , and (2) AMBIENT only. Both data sets contained longer snippets (in English). The texts in these experiments are 2-3 times longer than in the case of French data set. Additionally, MORESQUE is the data set that introduces ambiguous texts. The first experiment showed that with the introduced ambiguity the algorithms Lingo and STC do not work properly, whereas SnSRC provides reasonable results. The experiments with the data set limited to AMBIENT showed much better results, with SnSRC still outperforming Lingo and STC.

The last series of experiments were devoted to testing the idea of semantic enrichment. Based on the previous experiments, for this series we chose SnSRC and Lingo. The two algorithms were applied to the AMBIENT data set. In particular, we tested how enhancing short texts with semantic features from BabelNet/Babelfy influences the quality of

[19]Let us note that the texts from brainstorming sessions were full of corporation jargon.

clustering. Then we performed experiments with the neural network based distributional model.

The experiments concerning semantic enrichment for both data sets showed that among the approaches using BabelNet tools, only the Babel-synsets approach provided better results, whereas the other approaches did not improve the clustering quality. In particular, the enrichment of texts by corresponding categories, hypernyms and glosses reduced the clustering quality for both algorithms when compared to the baseline solutions. On the other hand, the semantic enrichment by Babel-synsets increased the quality of both algorithms. Also, in the case of the AMBIENT data set the results were much better than for the ultra-short French data. This confirms that the results of semantic enrichment are better for longer texts.

The experiments with the neural network based models (implemented by means of Word2vec) showed much better results than other semantic enrichment methods for both algorithms and for both data sets. In particular, the models trained with the texts from the repository to be clustered outperform the models trained on Wikipedia. In the case of data from seminars enriching the texts by Word2Vec-2 increases ARI by 1.44% when compared to the baseline; this is a relative improvement of 8.89% over the baseline. The experiments show that in the case of very short texts and small corpora the SnSRC algorithm outperforms the classical data-centric and description-centric algorithms. Moreover, the semantic enrichment of texts by applying neural networks can additionally improve the clustering quality, especially in the case of a specific domain language.

In the future we plan improving the core steps in the proposed method, namely (1) sense induction method, and (2) automatic summarization method. Recently high attention is given to the possibilities of using deep learning in the context of text processing. For example, Xu et al. (2017) propose a flexible self-taught convolutional neural network framework for short text clustering, which can learn non-biased deep text representation in an unsupervised manner. Generally, the key improvement of deep learning approaches is an essential reduction of data sparsity and the compactness of the lexical representations. This comes, unfortunately, at the expense of flattening information by merging the meanings of ambiguous words into a single vector representation. Therefore, we plan to investigate deep learning based methods, capable of capturing lexical semantics and syntactic relations, coping with the imbalanced semantic distribution. More importantly, a real challenge concerning neural approaches is their explainability. We plan to address this problem, as well.

Acknowledgments We would like to thank three anonymous referees for their valuable and constructive comments, which helped us to improve the quality of this article.

References

Aggarwal, C.C., & Han, J. (2014). *Frequent pattern mining*. Berlin: Springer.

Andrade, M.A., & Valencia, A. (1998). Automatic extraction of keywords from scientific text: application to the knowledge domain of protein families. *Bioinformatics (Oxford)*, *14*(7), 600–607.

Banerjee, S., Ramanathan, K., Gupta, A. (2007). Clustering short texts using Wikipedia. In *Proc. 30th ann. int'l ACM SIGIR conf.* (pp. 787–788).

Bovi, C.D., & Navigli, R. (2017). Multilingual semantic dictionaries for natural language processing: the case of BabelNet. *Encyclopedia with Semantic Computing and Robotic Intelligence*, *1*(01), 1630015.

Carpineto, C., & Romano, G. (2008). Ambient dataset. http://credo.fub.it/ambient.

Cutting, D., Karger, D., Pedersen, J., Tukey, J. (1992). Scatter/gather: a cluster-based approach to browsing large document collections. In *Proc. 15th ACM SIGIR* (Vol. 51, pp. 318–329).

Di Marco, A., & Navigli, R. (2011). Clustering web search results with maximum spanning trees. In *Proc. 12th congress of the Italian association for AI* (pp. 201–212).

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

Ferragina, P., & Scaiella, U. (2012). Fast and accurate annotation of short texts with Wikipedia pages. *IEEE Software*, *29*(1), 70–75.

Firth, J.R. (1957). A synopsis of linguistic theory, 1930–1955. *Studies in Linguistic Analysis*, 1–32.

Flati, T., & Navigli, R. (2014). Three birds (in the LLOD cloud) with one stone: BabelNet, Babelfy and the Wikipedia bitaxonomy. In *Proc. of 10th international conference on semantic systems, SEMANTiCS* (pp. 10–13).

Gabrilovich, E., & Markovitch, S. (2005). Feature generation for text categorization using world knowledge. In *IJCAI* (Vol. 5, pp. 1048–1053).

Gabrilovich, E., & Markovitch, S. (2009). Wikipedia-based semantic interpretation for natural language processing. *Journal of Artificial Intelligence Research*, *34*, 443–498.

Ganter, B., & Wille, R. (1999). *Formal concept analysis: mathematical foundations*. Berlin: Springer.

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

Hu, X., Sun, N., Zhang, C., Chua, T. (2009). Exploiting internal and external semantics for the clustering of short texts using world knowledge. In *Proc. 18th ACM conf. on inf. and knowledge management* (pp. 919–928).

Huang, E.H., Socher, R., Manning, C.D., Ng, A.Y. (2012). Improving word representations via global context and multiple word prototypes. In *Proc. of the 50th ann. meeting of the ACL: long papers-volume 1* (pp. 873–882). ACL.

Hulth, A. (2003). Improved automatic keyword extraction given more linguistic knowledge. In *Proc. of the 2003 conf. on empirical methods in nat. lang. processing* (pp. 216–223).

Iacobacci, I., Pilehvar, M.T., Navigli, R. (2016). Embeddings for word sense disambiguation: an evaluation study. In *Proceedings of the 54th annual meeting of the association for computational linguistics (volume 1: long papers)* (Vol. 1, pp. 897–907).

Justeson, J., & Katz, S. (1995). Technical terminology: some linguistic properties and an algorithm for identification in text. *Natural Language Engineering*, *1*(1), 9–27.

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

Kotlerman, L., Dagan, I., Kurland, O. (2017). Clustering small-sized collections of short texts. *Information Retrieval Journal*, *21*(4), 1–34.

Kozlowski, M., & Rybinski, H. (2014). SNS: a novel word sense induction method. In *Proc. of rough sets and intelligent systems paradigms: 2nd int'l conf.* (pp. 258–268).

Kozlowski, M., & Rybinski, H. (2017a). Semantic enriched short text clustering. In I*nternational symposium on methodologies for intelligent systems* (pp. 435–445). Springer.

Kozlowski, M., & Rybinski, H. (2017b). Word sense induction with closed frequent termsets. *Computational Intelligence*, *33*(3), 335–367.

Krause, S., Hennig, L., Moro, A., Weissenborn, D., Xu, F., Uszkoreit, H., Navigli, R. (2016). Sar-graphs: a language resource connecting linguistic knowledge with semantic relations from knowledge graphs. *Web Semantics: Science, Services and Agents on the World Wide Web*, *37*, 112–131.

Le, Q., & Mikolov, T. (2014). Distributed representations of sentences and documents. In *Proc. 31st int'l conf. on machine learning* (pp. 1188–1196).

Manning, C.D., Raghavan, P., Schütze, H., et al. (2008). *Introduction to information retrieval* Vol. 39. Cambridge: Cambridge University Press.

Metzler, D., Dumais, S., Meek, C. (2007). Similarity measures for short segments of text. In *European conference on information retrieval* (pp. 16–27). Springer.

Mikolov, T., Chen, K., Corrado, G., Dean, J. (2013a). Efficient estimation of word representations in vector space. arXiv:13013781.

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

Milne, D., & Witten, I.H. (2013). An open-source toolkit for mining Wikipedia. *Artificial Intelligence*, *194*, 222–239.

Milne, D., Medelyan, O., Witten, I. (2006). Mining domain-specific thesauri from Wikipedia: a case study. In *Proc. IEEE/WIC/ACM int'l conf. on web intelligence* (pp. 442–448).

Moro, A., Raganato, A., Navigli, R. (2014). Entity linking meets word sense disambiguation: a unified approach. *Trans. of the Assoc. for Comp. Ling.*, *2*, 231–244.

Navigli, R., & Crisafulli, G. (2010). Inducing word senses to improve web search result clustering. In *Proceedings of EMNLP 2010* (Vol. 2010, pp. 116–126).

Osinski, S., & Weiss, D. (2005). A concept-driven algorithm for clustering search results. *IEEE Intelligent Systems*, *20*(3), 48–54.

Osinski, S., Stefanowski, J., Weiss, D. (2004). Lingo: Search results clustering algorithm based on singular value decomposition. In *Proc. of the int'l IIS: IIPWM'04 conf.* (pp. 359–368).

Pantel, P., & Dekang, L. (2002). Discovering word senses from text. In *Proc. of 8th ACM SIGKDD int'l conf. on knowledge discovery and data mining* (pp. 613–619).

Pasquier, N., Bastide, Y., Taouil, R., Lakhal, L. (1999). Discovering frequent closed itemsets for association rules. In *Proc. of 7th intl. conf. on database theory* (pp. 398–416).

Pinto, D., Benedí, J.M., Rosso, P. (2007). Clustering narrow-domain short texts by using the kullback-leibler distance. In *Int'l conf. on intell. text processing and comp. ling.* (pp. 611–622). Springer.

Sahami, M., & Heilman, T. (2006). A web-based kernel function for measuring the similarity of short text snippets. In *Proc. 15th int'l conf. on world wide web* (pp. 377–386).

Shrestha, P., Jacquin, C., Daille, B. (2012). Clustering short text and its evaluation. In *Proc. of int'l conf. on intelligent text processing and computational linguistics* (pp. 169–180).

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

Taghipour, K., & Ng, H.T. (2015). Semi-supervised word sense disambiguation using word embeddings in general and specific domains. In *Proceedings of the 2015 conference of the North American chapter of the association for computational linguistics: human language technologies* (pp. 314–323).

Wang, J., Han, J., Pei, J. (2003). Closet+: Searching for the best strategies for mining frequent closed itemsets. In *Proceedings of the 9-th ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 236–245).

Xu, J., Xu, B., Wang, P., Zheng, S., Tian, G., Zhao, J. (2017). Self-taught convolutional neural networks for short text clustering. *Neural Networks*, *88*, 22–31.

Zaki, M., & Hsiao, C. (2002). Charm: an efficient algorithm for closed itemset mining. In *Proc. 2002 SIAM int'l. conf. data mining* (pp. 457–473).

Zamir, O., & Oren, E. (1998). Web document clustering: a feasibility demonstration. In *Proc. 21st ann. int'l ACM SIGIR conf.* (pp. 46–54).