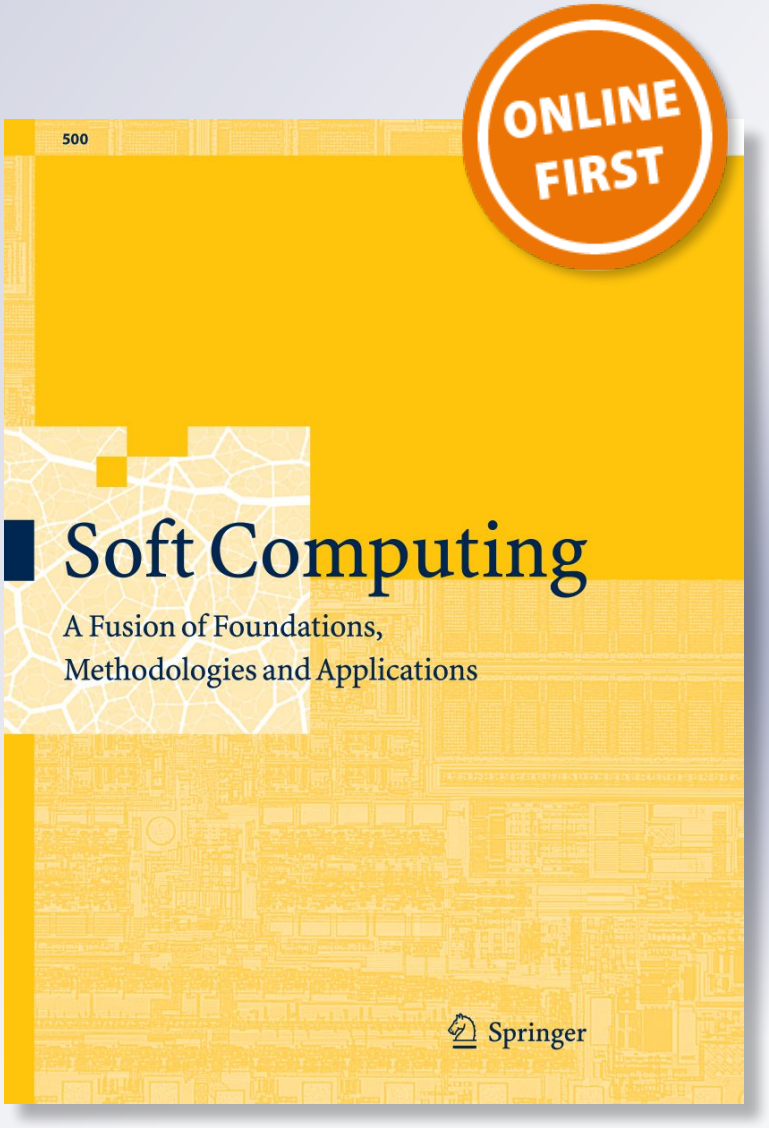
# *Exploiting extensible background knowledge for clustering-based automatic keyphrase extraction*

**Hassan Alrehamy & Coral Walker**

**Soft Computing**
A Fusion of Foundations, Methodologies and Applications

ISSN 1432-7643

Soft Comput
DOI 10.1007/s00500-018-3414-4




Springer

Soft Computing
https://doi.org/10.1007/s00500-018-3414-4

FOCUS

# Exploiting extensible background knowledge for clustering-based automatic keyphrase extraction

**Hassan Alrehamy**[1,2] · **Coral Walker**[1]



**Abstract**
Keyphrases are single- or multi-word phrases that are used to describe the essential content of a document. Utilizing an external knowledge source such as WordNet is often used in keyphrase extraction methods to obtain relation information about terms and thus improves the result, but the drawback is that a sole knowledge source is often limited. This problem is identified as the *coverage limitation* problem. In this paper, we introduce SemCluster, a clustering-based unsupervised keyphrase extraction method that addresses the coverage limitation problem by using an extensible approach that integrates an internal ontology (i.e., WordNet) with other knowledge sources to gain a wider background knowledge. SemCluster is evaluated against three unsupervised methods, TextRank, ExpandRank, and KeyCluster, and under the F1-measure metric. The evaluation results demonstrate that SemCluster has better accuracy and computational efficiency and is more robust when dealing with documents from different domains.

**Keywords** Natural language processing · Unsupervised keyphrase extraction · Clustering-based AKE · Knowledge-based AKE

## 1 Introduction

Keyphrases are single- or multi-word expressions that describe the essential content of a document. As a rich source of information about the theme of documents (Liu et al. 2009), high-quality keyphrases can benefit many text processing tasks, such as document summarization (Wan et al. 2007), classification (Androutsopoulos et al. 2000), clustering (Hammouda et al. 2005), and retrieval (Qiu et al. 2012). Assigning keyphrases to a free-text document is often done manually by the author of the document or a professional curator. It is laborious and time-consuming (Turney 2000) and could explain why, among the vast amount of textual data on the web such as news articles and blogs, few of them are associated with any keyphrases. In an explosive era of big data, automatic keyphrase extraction (AKE) is increasingly in demand. It analyzes the content of a free-text document using natural language processing (NLP) and automatically identifies and extracts keyphrases that represent the theme of the document.

The current AKE studies in the NLP literature are mostly divided into two lines of research (Hasan and Ng 2010; Hasan and Ng 2014; Siddiqi and Sharan 2015): supervised and unsupervised. A supervised AKE approach typically treats the extraction task as a classification problem, in which, a classifier is trained on a large corpus of documents annotated with "correct" keyphrases by human experts, and the result is a machine learning model that then can be used for discriminating keyphrases from non-keyphrases in unseen documents. Various text features and classification algorithms have been applied in supervised AKEs, for example, extractor (Turney 2000) employs a set of rule-based features and genetic algorithms to classify keyphrases, and KEA (Witten et al. 1999) uses a Naïve Bayesian classifier to identify keyphrases based on two features, namely the TF.IDF (term frequency-inverse document frequency) of each term in the document, and the distance of its first occurrence from the beginning of the text. Hulth (2003) suggests exploiting part-of-speech (POS) tag-

Communicated by F. Chao, Q. Zhang.

✉ Coral Walker
coral.walker@cardiff.ac.uk

Hassan Alrehamy
alrehamyhh@cardiff.ac.uk

[1] School of Computer Science and Informatics, Cardiff University, Cardiff, UK

[2] College of Information Technology, Babylon University, Babylon, Iraq

Published online: 16 August 2018

Canadian **Ben Johnson** left the **Olympics** today "in a complete state of shock," accused of cheating with drugs in the world's fastest **100-meter dash** and stripped of his **gold medal**. The prize went to American **Carl Lewis**. Many athletes accepted the accusation that Johnson used a muscle-building but dangerous and illegal anabolic steroid called **Stanozolol** as confirmation of what they said they know has been going on in track and field. Two tests of Johnson's urine sample proved positive and his denials of **drug use** were rejected today. "This is a blow for the **Olympic Games** and the Olympic movement," said International Olympic Committee President Juan Antonio Samaranch.

**Fig. 1** A text segment from news article #AP880927-0089 in DUC-2001 dataset (Wan and Xiao 2008). Manually assigned keyphrases are highlighted in bold

ging information as additional features during training and prediction. Empirical results of utilizing 56 POS patterns indicate that linguistic features can significantly improve AKE precision. More recent supervised approaches utilize advanced classification algorithms, such as decision trees (Sterckx et al. 2016), maximum entropy (Yih et al. 2006), conditional random fields (Zhang 2008), and deep recurrent neural networks (Zhang et al. 2016). Although these approaches perform AKE with promising results (Hassan et al. 2009; Kim et al. 2013), they often require a substantial amount of manually annotated training data, which is a very expensive requirement, and may lead to inconsistency in a heterogeneous processing environment that demands cross-domain tractability (Alrehamy and Walker 2015). For instance, a classifier trained on features of labeled keyphrases belonging to a particular domain (e.g., scientific papers) may exhibit poor performance when applied on documents in another domain (e.g., news articles).

Unsupervised AKE overcomes the critical challenges of training corpora and domain bias by casting the extraction task as a ranking problem. The typical workflow here is to select a particular set of terms from the input document, and by applying some ranking strategy, top-ranked terms are taken as keyphrases. Unsupervised AKE approaches can be either graph-based or statistics-based (Hasan and Ng 2014). In a graph-based approach, the input document is modeled as a graph, where a node represents a term, and an edge represents a relevance relation (e.g., co-occurrence) between two nodes. Subsequently, a centrality measure, such as PageRank (Page et al. 1999), is applied on the graph to rank each node based on its incoming and outgoing edges. Finally, the top-$k$ ranked nodes are selected as keyphrases. A statistics-based approach ranks terms based on their associated statistical information, such as TF, IDF, or statistical distances, and then selects the top-$k$ ranked term as either keyphrases or heuristics that are used to search for further candidates.

Compared with other NLP tasks, unsupervised approaches for performing AKE struggle to achieve a better result (Hasan and Ng 2014), because of the complexity of AKE tasks, which requires not only local statistical information about the terms contained in the document, but also extensive background knowledge to capture the relations between them (Hasan and Ng 2010). Many recent approaches suggest utilizing external knowledge sources, such as WordNet (Fellbaum 1998), to obtain rich relation information about terms during AKE (Wang et al. 2007; Martinez-Romo et al. 2016). Although these approaches demonstrate improved AKE performance in some cases, their utilized knowledge sources are not consistent enough to supply background information about terms in any arbitrary domain, and consequently, a term representative of the theme of the document may be disregarded simply because the knowledge source does not maintain information about it—this issue we refer to as *coverage limitation*. For example, in Fig. 1, in the text segment from the DUC-2001 news dataset (Wan and Xiao 2008) the term "Ben Johnson" is important because it refers to the name of the athlete that this news article is about and therefore is selected as a valid keyphrase by human curators; however, a WordNet-based unsupervised approach, such as SemGraph (Martinez-Romo et al. 2016), would disregard this term because WordNet has no entry matching "Ben Johnson."

Another issue in existing unsupervised approaches is their heavy reliance on statistical information to capture the statistical relations between terms, thereby failing to account for their latent semantic relations. In a graph-based approach, if two representative terms are not co-occurring within a predefined window, then no occurrence edge will be established between their donating nodes; thus, their rankings in the graph decrease. Similarly, statistics-based approaches treat terms solely as statistical elements; therefore, a term of high frequency is typically ranked higher than an infrequent but semantically important term. Due to this *semantics loss*, a term representative of the document theme is not guaranteed to be its top-ranking candidate if it occurs infrequently (Hassan et al. 2014), which also means a top-ranking candidate of statistical importance may not be suitable to be a keyphrase representative of the document (Liu et al. 2009). In our running example, most state-of-the-art, unsupervised AKE approaches fail to identify "dash" as a representative term, because of its distance from other co-occurring terms such as "Olympics" and its infrequent occurrence in the text, and hence, the phrase "100-m dash" is not regarded as a valid keyphrase. On the other hand, the term "Olympics" appears frequently, so it is not surprising that most AKE approaches select "Olympics," "Olympic Games," and "Olympic movement" as valid keyphrases,

without considering that "Olympic movement" is not identified by the human curators as a valid keyphrase because it has no immediate semantic relevance to the document theme. Many knowledge bases[1,2,3] map this term to the "Organization," not the "Sport," domain.

In this paper, we introduce SemCluster, a clustering-based method to extract high-quality keyphrases from free-text documents in any domain. SemCluster first extracts a particular set of terms from the input document and then performs clustering on them so that *similar* terms are grouped within the same cluster based on their latent lexical and semantic relations. Each resulting cluster may implicitly correspond to a topic in the document. Terms that are close to the centroids of *specific* clusters are selected as seeds and used to search for candidate phrases that are representative of the main theme of a document. Finally, candidate phrases are refined and the resulting candidates are chosen as *appropriate* keyphrases. SemCluster makes use of knowledge extensibility in order to address the aforementioned unsupervised AKE issues. SemCluster uses WordNet as its default knowledge source to obtain background semantic information about the terms within the document. The semantic coverage of WordNet can be flexibly extended by integrating any number of additional generic, specialized, or personalized knowledge sources, so that, when the semantic information of an arbitrary term is not present in WordNet, it is likely to be available in its integrated sources. For example, by integrating WordNet with DBPedia (Auer et al. 2007), SemCluster can obtain rich semantics about "Ben Johnson" from DBPedia, even though this term has no matching entry in WordNet. With the availability of rich semantics, SemCluster can readily capture the latent semantic relations between terms and adequately rank each term based on its semantic importance in the context of the document, and its relevance to the theme. In the example in Fig. 1, despite the infrequent occurrence of the representative term "dash" and its distance from statistically relevant terms, SemCluster assigns it a high rank due to its semantic *closeness* to "Olympics," "Ben Johnson," and "Carl Lewis."

The rest of the paper is organized as follows: a summary of related work is presented in Sect. 2, and SemCluster implementation details are provided in Sect. 3. In Sect. 4, we undertake the evaluation and analysis of SemCluster in terms of its performance, and, finally, we conclude the work and discuss future work directions in Sect. 5.

[1] http://www.babelnet.org.

[2] http://www.conceptnet.io.

[3] http://lookup.dbpedia.org/api.

## 2 Related work

Most early studies on unsupervised[4] methods for keyphrase extraction focus on utilizing the local information in the document. The simplest approach uses the word frequency criterion (Sparck 1972) to select keyphrases. More sophisticated methods incorporate additional statistical and linguistic information. Barker and Cornacchia (2000) suggest extracting noun phrases from a document and ranks them based on phrase length, frequency, and the frequency of its head noun. Munoz (1997) proposes an unsupervised algorithm based on adaptive resonance theory to identify bi-term keyphrases, although intuitively keyphrases vary in length. El-Beltagy and Rafea (2009) proposed KP-Miner, a state-of-the-art TF · IDF-based approach. KP-Miner operates on n-gram phrases, and only phrases that do not contain a stop word or punctuation mark, occur for the first time within the first $m$ words of the document, and have a frequency greater than a threshold determined by the document length, are selected as candidate phrases. Subsequently, candidates are ranked using a modified TF · IDF model that incorporates a boosting factor aimed at reducing the bias toward single-word candidates. KP-Miner suffers two main drawbacks: it treats phrases as solely statistical elements in the document, and secondly, it ignores the fact that, according to recent studies, in general, 15% of keyphrases contain stop words (Le et al. 2016).

Graph-based AKE is another major stream of AKE research (Beliga et al. 2015). Mihalcea (2004) proposes TextRank, the first approach to rank candidate keyphrases based on co-occurrence links between words. TextRank uses a sliding window technique to construct the word graph of an input document. The sliding window moves from the first word to the last word in the document, and words that co-occur within a window $m \geq 2$ are connected by an edge in the graph. The approach then applies PageRank on the graph to rank nodes through voting (Page et al. 1999). A node with more in- and out-edges has more probability of being top-ranked. However, because important words with low frequency are often ranked low, due to semantic loss, TextRank performs AKE with poor accuracy. To compensate for this issue, numerous methods have been proposed. Among them, Danesh et al. (2015) present a hybrid, statistics- and graph-based approach that computes an initial weight for each phrase based on its TF · IDF score and the position of its first occurrence in the document. Then, the phrases, together with their weights, are modeled as a graph and their weights are recomputed using a centrality measure to produce the final ranking of phrases. Wan and Xiao (2008) introduce an extension of TextRank that incorporates the co-occurrence

[4] In this work, we do not consider any AKE approaches that demand machine learning training to improve a term's ranking, such as word embedding based approaches.

information from a set of neighbor documents to weight the edges between words in the graphical representation of the input document. The algorithm uses the cosine similarity measure to retrieve documents from a large document corpus that are topically related to the input document. The documents retrieved contribute in identifying and ranking the phrases that correspond to the topics in the document. However, the retrieval of topic-related documents from large corpora is very expensive. Wang et al. (2007) extend TextRank by incorporating background semantic information form WordNet for weighting the nodes in the graph. Then PageRank is used to compute the top-k ranked nodes. Similarly, Martinez-Romo et al. (2016) use information from WordNet to enrich the semantic relationships between words in the word graph. Though the performance of the methods using WordNet has improved greatly, as indicated in the introduction section, WordNet is limited in terms of its semantic coverage and is not a panacea.

Clustering-based studies are another family of unsupervised AKE (Hasan and Ng 2014). Bracewell et al. (2005) present a method for extracting noun phrases from a document and grouping them into clusters based on their shared noun terms. The resulting clusters are ranked based on noun term frequencies, and the top-$k$ ranked clusters are selected as keyphrases. Liu et al. (2009) introduce a similar clustering-based algorithm called KeyCluster which extracts single noun words and groups them into clusters based on their semantic relatedness using a Wikipedia co-occurrence-based similarity measure (Cilibrasi and Vitanyi 2007). It then selects phrases that contain one or more cluster centroids and that follow a certain linguistic pattern as keyphrases. KeyCluster has been shown to outperform many prominent AKE methods; however, early clustering-based methods, in general, cannot guarantee that all generated clusters are sufficient to cover the document theme, and selecting the centroid of a topically unimportant cluster as a heuristic to identify and extract keyphrases yields erroneous outputs. Accordingly, more recent studies propose to incorporate topic analysis in the AKE task to ensure that output keyphrases have strong association with the document's main theme from a topical viewpoint. In the topical clustering-based method (Pasquier 2010; Liu et al. 2010; Danilevsky et al. 2014), terms are grouped into clusters using an appropriate clustering algorithm, and the method proceeds to conduct topic analysis using a probabilistic topical model, such as Latent Dirichlet Allocation (Blei et al. 2003), in order to extract all latent topics $\boldsymbol{T}$ in the document. The importance of each term is computed as the sum of its scores in each topic $T_i \in \boldsymbol{T}$, weighted by the probability of $T_i$. Hence, a term that belongs to an important topic $T_i$ is weighted more heavily than a term that belongs to a less important topic $T_j$. Although topical clustering-based methods improve significantly their AKE performance, they essentially suffer empirical challenges related to the topic analysis process. For instance, when applied to new domains, LDA and similar models induce high computational complexity and require hyperparameter (re)tuning, which is a non-trivial task in domain-agnostic text processing applications.

The proposed method is based on an extensive literature review and through learning the advantages and disadvantages of other approaches. It adopts an approach that extracts $n$-gram terms and named entities instead of single words (similar to KP-Miner) and relies greatly on background knowledge sources (similar to SGRank, SemGraph, ExpandRank, and KeyCluster). However, because of the coverage limitation problem that would arise if it were based on a sole knowledge source, SemCluster is designed to allow extensibility of its knowledge base by integrating with other knowledge sources. In addition, SemCluster, as a clustering-based method, ranks each term based on its latent semantic relations with other terms, as well as its frequency in the document, by integrating a term's lexical, semantic, and statistical information into an efficient clustering model. Furthermore, without requiring topic analysis, SemCluster can systematically identify and filter out thematically unimportant clusters from the clustering results, thus allowing only phrases representative of the document theme to qualify as candidate keyphrases.

## 3 SemCluster overview

Given an extensible background knowledge source that is modeled as ontology $O$, for an input free-text document $\boldsymbol{D}$, SemCluster performs the workflow depicted in Fig. 2 to extract a set of keyphrases that are most representative of the document's content. SemCluster workflow is explained in the following subsections.

### 3.1 Candidate terms selection

The first step in SemCluster is the selection of candidate terms, and it is aimed at extracting from the content of $D$ a general set of terms, where each term is associated with background semantic information. The step begins with preprocessing $\boldsymbol{D}$ by applying the following NLP tasks: tokenization, sentence boundary detection, part-of-speech (POS) tagging, and shallow parsing (chunking). Penn Treebank notion (Clark et al. 2013) is adopted for POS tagging and chunking. The aim of chunking is to group words into chunks based on their discrete grammatical meanings. Many NLP studies have shown that almost all keyphrases assigned by expert curators are typically embedded in noun phrases (i.e., NP chunks) (Barker and Cornacchia 2000; Hulth 2003; Bracewell et al. 2005; Liu et al. 2009). Accordingly, SemCluster considers only NP chunks to find keyphrases and

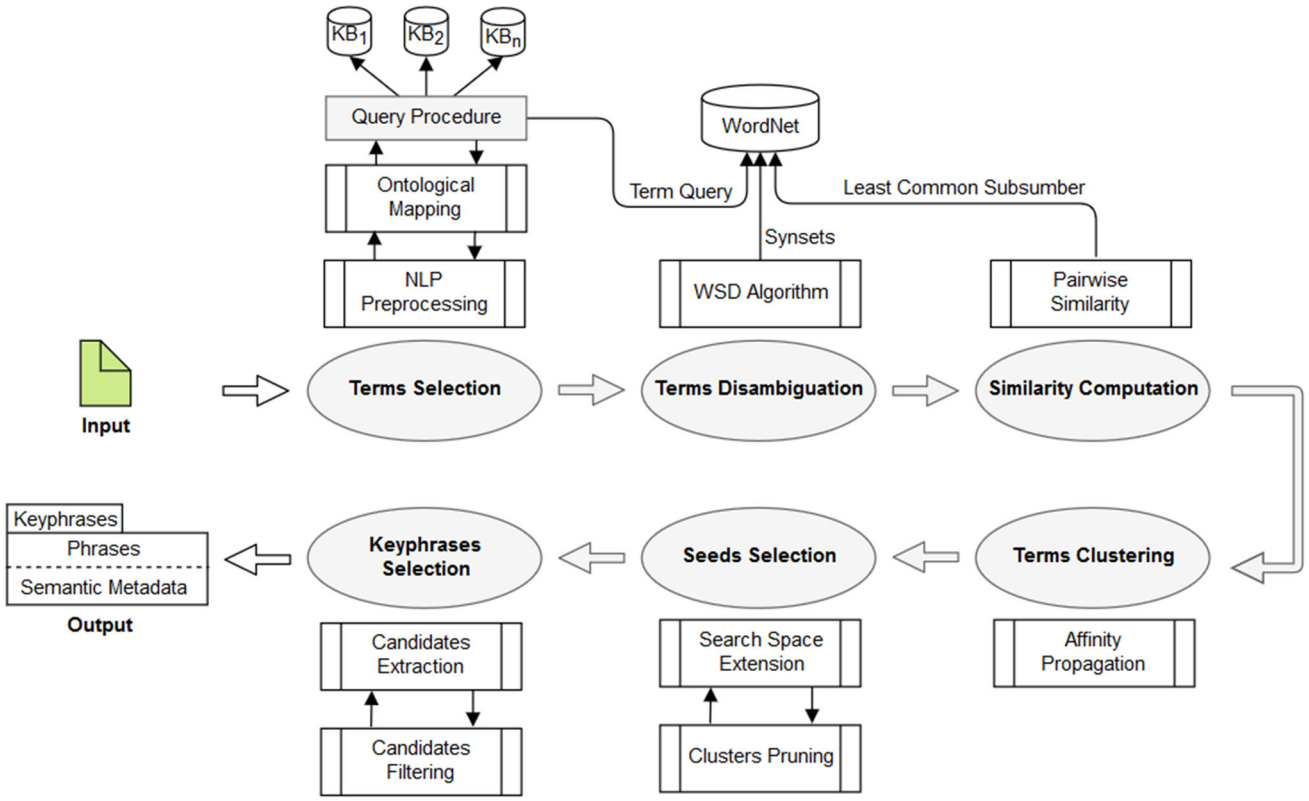



**Fig. 2** SemCluster workflow overview

**Table 1** POS patterns for extracting n-gram terms from NP chunks, with examples from Fig. 1

| Pattern | Example |
|---|---|
| $\mathcal{N} = (NN\|\text{NNS})$ | Dash/NN, prize/NN, drugs/NNS |
| $\mathcal{C} = (JJ) * (NN\|\text{NNS})+$ | Anabolic/JJ steroid/NN, gold/NN medal/NN |
| $\text{E} = (\text{NNP}\|\text{NNPS}) * (\mathcal{S}) * (\text{NNP}\|\text{NNPS})+$ | Stanozolol/NNP, Ben/NNP Johnson/NNP, Olympics/NNPS |

detects and extracts terms in each NP chunk based on their POS annotations. We allow the selection of $n$-gram terms (where $0 < n \leq 5$) using the POS patterns listed in Table 1. $\mathcal{N}$ denotes *Noun*, a word tagged as a singular noun ($NN$) or plural noun (NNS). $\mathcal{C}$ denotes *Compound Noun*, a sequence of words starting either with an adjective ($JJ$) or noun (both $NN$ and NNS). E denotes an *Entity*, a sequence of words of singular proper nouns (NNP) or plural proper nouns (NNPS) with at most one stop word ($\mathcal{S}$): *the* at the beginning, or *of* in the middle. Each term extracted using these patterns is mapped into SemCluster's ontology $O$, and depending on the mapping result, a term is regarded either as a *candidate term* or *miscellaneous*. When a term does not map to any entries in the ontology, it is decomposed into smaller constituents to be mapped again. The terms that fail to find matches even after being reduced to smaller constituents are discarded.

SemCluster uses WordNet as its ontology $O$. WordNet is a widely used lexical database. It comprises four lexical networks (Fellbaum 1998): Nouns, Verbs, Adjectives, and Adverbs. In SemCluster, we use only the Nouns network. WordNet groups nouns of equivalent meanings into *synsets*. A synset consists of a list of synonyms and a short definition called a *gloss*. Synsets are connected to other semantically relevant synsets by means of semantic relations. Noun synsets are organized using hyponym/hypernym (Is-A) and meronym/holonym (part-of) relationships, providing a hierarchical tree-like structure that can be directly modeled as an ontology.

#### 3.1.1 Background knowledge extensibility

In practice, no knowledge base is comprehensive, and neither is WordNet. The Nouns network contains a large but limited number of English nouns collected nearly two decades ago, and therefore, WordNet does not support newly emerging nouns, or new meanings of already existing nouns. Relying

**Input:**
$t_i \in D, KB_x \in \{KB\}$
**Output:**
$H = \{(e,s)_1, (e,s)_2, \ldots, (e,s)_n\}$ : The set of entries matching $t_i$ and their corresponding WordNet synsets.
**Procedure:**
If $t_i$ found in $KB_x$ then
  Retrieve all entries $E$ matching $t_i$, $E = \{e_1, e_2, \ldots, e_n\}, E \subset KB_x$
  For each entry $e_j$ in $E$:
    Retrieve all type classes $C$ of $e_j$, $C = \{c_1, c_2, \ldots, c_m\}$
    For each $c_h$ in $C$:
      If $c_h$ is the deepest type class in $KB_x$ schema ontology then
        Select $c_h$ as hypernym of $e_j$
        Find the equivalent synset $s_i$ of $c_h$
        Assign $s_i$ as hypernym of $e_j$
        Add the pair $(e_i, s_j)$ to $H$
  Return $H$
Else
  Return ∅

**Fig. 3** Extensible background knowledge querying procedure

solely on WordNet as the only background knowledge source leads to the background knowledge coverage, limitation as discussed earlier. To overcome this, we design a novel procedure for extending WordNet coverage by integrating external knowledge bases that use ontology-based schemas for structuring their knowledge, such as DBPedia, BabelNet (Navigli and Ponzetto 2012), Yago (Hoffart et al. 2013), and any ad hoc (specialized or personalized) knowledge bases.

The workflow of our proposed procedure is as follows: given an external knowledge base, donated as $\boldsymbol{KB_x}$, its schema is modeled as an ontology, and each entry in $\boldsymbol{KB_x}$ is assigned one or more ontological concepts called a *type class*. To perform a meaningful integration, the schema ontology of $\boldsymbol{KB_x}$ is horizontally aligned with $O$ by mapping each type class to its semantically equivalent synset. To prevent conceptual ambiguity, ontological alignments are performed as one-to-one mappings, such that, each type class in the $\boldsymbol{KB_x}$ schema ontology is mapped to exactly one synset in $O$. During the selection of candidate terms, an $n$-gram that is extracted from the preprocessed content of $\boldsymbol{D}$ and that cannot be mapped to WordNet is queried against the integrated knowledge base(s), denoted as $\{\boldsymbol{KB}\}$, using the procedure depicted in Fig. 3. Given the knowledge base $\boldsymbol{KB_x} \in \{\boldsymbol{KB}\}$, whose schema ontology is properly aligned with $O$, SemCluster queries $\boldsymbol{KB_x}$ with the $n$-gram $t_i$. If there are entries in $\boldsymbol{KB}_x$ matching $t_i$, then each matched entry is retrieved from $\boldsymbol{KB}_x$ and is considered as an external contextual meaning (or *sense*) of $t_i$. All the type classes associated with external senses of $t_i$ in $KB_x$ are mapped into their corresponding synsets in $O$ and are considered as hypernyms of $t_i$. The synset that corresponds to the deepest type class in the schema ontology of $\boldsymbol{KB}_x$ is considered the correct hypernym of the external sense. With this construct, we allow SemCluster to dynamically generate appropriate senses for the terms that are absent in WordNet, or even expand the set of synsets for an existing term. To illustrate with a real-world example, we consider extending $O$ with DBPedia (i.e., $\boldsymbol{KB_{DBPedia}}$) and aligning the type classes in its schema ontology[5] with their equivalent WordNet synsets. For example, the type class "*dbo:Athlete*" in $\boldsymbol{KB_{DBPedia}}$ is directly mapped to "*wn:Athlete#n1*" in WordNet, while "*dbo:MusicFestival*" is mapped to its equivalent synset "*wn:Fete#n2*." Revisiting the news article example depicted in Fig. 1, we see that the term "*Ben Johnson*" has no entries in WordNet but five entries in $\boldsymbol{KB_{DBPedia}}$. Accordingly, SemCluster generates five new external senses for "*Ben Johnson,*" each reflecting one entry in $KB_{\text{DBPedia}}$. The third sense in particular, "*Ben Johnson (Sprinter),*"[6] is associated with four type classes as depicted in Fig. 4: "*owl:Thing,*" "*dbo:Agent,*" "*sc:Person,*" "*dbo:Athlete.*" According to the querying algorithm, the deepest among the four classes, "*dbo:Athlete,*" becomes the hypernym of the third sense and is referred to as "*wn:Athlete#n1.*"

After mapping each extracted term against the extended ontology $O^{\dagger}$, only a subset of the terms are selected as candidate terms. We denote the set of the candidate terms as $\boldsymbol{T_D}$. Due to the pattern-based method of term extraction, especially when $\boldsymbol{D}$ contains informal text, $\boldsymbol{T_D}$ may contain

[5] DBPedia schema ontology is available at http://mappings.dbpedia.org/server/ontology/classes/.

[6] http://dbpedia.org/page/Ben_Johnson_(sprinter).

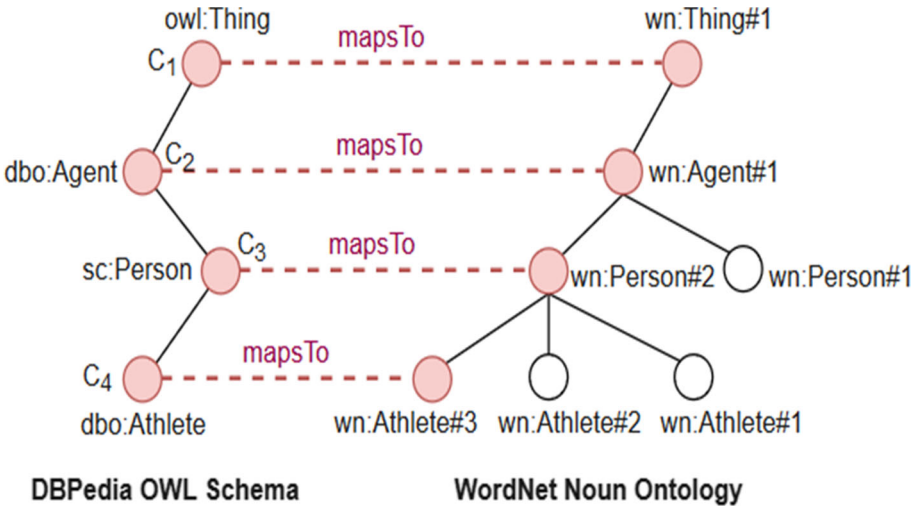



**Fig. 4** A fragment of the ontological alignment between WordNet's noun ontology and DBPedia's schema ontology

noisy terms that can adversely affect similarity computation and clustering performance. Noisy terms are nouns with no semantic value (e.g., "*one,*" "*someone*"). To identify and remove noisy terms, SemCluster maps each term in $\boldsymbol{T_D}$ to an internal list that contains the most frequent noisy terms in the English language, and any term found in the list is removed from $\boldsymbol{T_D}$.

### 3.2 Candidate terms disambiguation

A consequence of obtaining semantic background information about candidate terms is that each term in $\boldsymbol{T_D}$ may be associated with one or more contextual meanings (or *senses*), whether local or external. Prior to semantic similarity computation, SemCluster must identify the correct sense of each term in $\boldsymbol{T_D}$. Word sense disambiguation (WSD) is an NLP task that gives machines the ability to computationally determine which sense of a term is activated by its use in a particular context. WSD approaches are generally divided into three categories (Navigli 2009): supervised, unsupervised, and knowledge based. SemCluster employs the SenseRelate-TargetWord method (Patwardhan et al. 2005) for term sense disambiguation. The algorithm is WordNet-based and is implemented in WordNet::Similarity,[7] a widely used package in computational linguistics. The SenseRelate-TargetWord method takes one *target* candidate term as input and outputs a WordNet synset as the disambiguated sense of the target candidate term, based on information about the target as well as a few other candidate terms surrounding the target. The surrounding candidate terms are called the *context window*. Let $t_i$ be a target candidate term,$t_i \in T_D$, and the context window size be $N$, and the set of surrounding candidate terms in the context window be $\boldsymbol{W}$, $\boldsymbol{W} = \{w_1, w_2, \ldots, w_N\}$, where if $|\boldsymbol{W}| < \boldsymbol{N}$, then $\boldsymbol{N} = |\boldsymbol{W}|$. Since $t_i$ is deemed to be associated with a set of one or more senses, we denote this set by $Sense(t_i) = \{s_{i1}, s_{i2}, \ldots, s_{im}\}$. For each sense $s_{ij}$, we obtain not only its synonyms list and gloss from WordNet, but also the synonyms lists and glosses of other synsets that are related to $s_{ij}$ via the following set of semantic relations: {Hypernym, Hyponym, Meronym, Holonym}. The goal of the SenseRelate-TargetWord algorithm is to find the synset responsible for $s_{ij}$ whose synonyms and gloss content maximize the string-based overlap score with each $w_k$ in the context window.

[7] http://wn-similarity.sourceforge.net.

### 3.3 Candidate terms similarity computation

After disambiguating all the candidate terms in $\boldsymbol{T_D}$, each $t_i \in \boldsymbol{T_D}$ becomes associated with the following information: POS tag, position in the document, and a pointer linking $t_i$ with its correctly disambiguated WordNet synset $s_i$. In this step, SemCluster computes the pairwise semantic similarity between each pair of terms in $\boldsymbol{T_D}$ based on their synset pointers. There exist many measures to quantify the similarity between two synsets, and these measures are broadly divided into three main categories (Meng et al. 2013): path length based, information content based, and feature based. Unlike the other two, path length measures offer greater flexibility in computing the similarity between synsets based on SemCluster's extensible ontology. The WuPalmer measure (Wu and Palmer 1994) is a prominent path length measure to compute semantic similarity between two synsets $s_i, s_j$ by finding the shortest path between them relative to the deepest common parent synset, i.e., the *Least Common Subsumer* (LCS). The similarity $S(s_i, s_j)$ is quantified by counting the nodes in the shortest path between each synset and the LCS in $O$. The measure is defined as follows:

$$\mathrm{S}(s_i, s_j) = \frac{2d}{L_{si} + L_{sj} + 2d} \tag{1}$$

where $d$ is the depth of *LCS* from the root node, $L_{si}$ is the path length from $s_i$ to LCS, and $L_{sj}$ is the path length from $s_j$ to LCS. In this work, we modify the WuPalmer metric to capture extra semantic similarity between $s_i$ and $s_j$. Path length measures in general, and WuPalmer in particular, focus on measuring the semantic similarity between a pair of synsets $s_i$ and $s_j$ by exploiting the explicit semantic relations existing between them. However, WordNet does not cover all possible relations that may exist between synsets. For example, there is no direct link between "*wn:Bush#n4*" and "*wn:President#n2,*" although they are clearly related if they co-occur in a document. To capture explicit, as well as implicit, semantic similarities using WuPalmer, we extend its mathematical notion as follows:

$$\mathrm{S}(s_i, s_j) = \frac{2d + \mathrm{Overlap}(C(s_i), C(s_j))}{L_{si} + L_{sj} + 2d + \mathrm{Overlap}(C(s_i), C(s_j))} \tag{2}$$

where $C(s_i), C(s_j)$ are functions that retrieve $s_i$ and $s_j$ information from WordNet in string format, and $\mathrm{Overlap}(C(s_i), C(s_j))$ is a function that measures the string-based overlap between $C(s_i)$ and $C(s_j)$. Let $\mathrm{Synonyms}(s_i)$ be a function that retrieves all the words in the synonyms list of the synset $s_i$, $Gloss(s_i)$ be a function that retrieves the definition of $s_i$, $Related(s_i)$ be a function that retrieves the synonyms and glosses of all synsets connected directly to $s_i$ via the relation set {Hypernym, Hyponym, Meronym, Holonym}, then $\boldsymbol{C(s_i)}$ is defined as follows:

$$C(s_i) = \mathrm{Synonyms}(s_i) \cup \mathrm{Gloss}(s_i) \cup \mathrm{Related}(s_i) \tag{3}$$

where $\cup$ is the string concatenation function. $\mathrm{Overlap}(\boldsymbol{C(s_i)}, \boldsymbol{C(s_j)})$ finds the maximum number of words shared in the output of $\boldsymbol{C(s_i)}$ and $\boldsymbol{C(s_j)}$ normalized by the natural logarithm to prevent too much effect of implicit semantic similarity on the WuPalmer explicit semantic similarity measurement. Thus, we define *overlap* as follows:

$$\mathrm{Overlap}(C(s_i), C(s_j)) = \log(C(s_i) \cap C(s_j) + 1). \tag{4}$$

The extended WuPalmer measure is used to compute the pairwise similarities between each pair of terms in $\boldsymbol{T_D}$, and the result is a complete adjacency similarity matrix of size $|\boldsymbol{T_D}|^2$ denoted as $\mathcal{A}$. Once we have produced $\mathcal{A}$, we move on to the next step—clustering $\boldsymbol{T_D}$ based on $\mathcal{A}$.

### 3.4 Candidate terms clustering

There are many state-of-the-art clustering algorithms to efficiently cluster the adjacency matrix $\mathcal{A}$ resulting from the previous step. Affinity propagation (AP) (Frey and Dueck 2007) has been proposed as a powerful technique for exemplar learning by passing messages between nodes. It is reported to find clusters with much lower error compared with other algorithms (Guan et al. 2011). In addition, AP does not require specifying the number of desirable clusters in advance. Both these advantages are extremely important for SemCluster to support fully automatic keyphrase extraction, and hence, AP is adopted as the clustering algorithm in SemCluster. The input to AP is the matrix $\mathcal{A}$. The set $\boldsymbol{T_D}$ is modeled as a graph. An edge exists between two candidate terms $t_i$ and $t_j, t_i, t_j \in \boldsymbol{T_D}$, if $\mathrm{S}(t_i, t_j) > 0,$, and the weight of the edge is given by the cell $\mathcal{A}[i][j]$. Initially, all the nodes are viewed as exemplars, and after a large number of real-valued information messages have been transmitted along the edges of the graph, a relevant set of exemplars and corresponding clusters are identified. In AP terms, the similarity metric $\mathbf{S}(t_i, t_j)$ indicates how much $t_j$ is suitable as an exemplar of $t_i$. In SemCluster, $\mathbf{S}(t_i, t_j) = \mathcal{A}[i][j], i \neq j$. If there is no heuristic knowledge, self-similarities are called *preferences* and are taken as constant values. The preference $\boldsymbol{P}(t_i) = \mathbf{S}(t_i, t_i)$ represents the a priori suitability of the term $t_i$ to serve as an exemplar. In SemCluster, preferences are computed using the *median*. AP computes two kinds of messages exchanged between nodes: *responsibility* and *availability*. A responsibility message, denoted by $\mathrm{r}(t_i, t_j)$, is sent from node $t_i$ to node $t_j$ and reflects the accumulated evidence for how well-suited $t_j$ is to serve as the exemplar of $t_i$. An availability message, denoted as $\mathrm{a}(t_i, t_j)$, is sent from $t_j$ to $t_i$ and reflects the accumulated evidence for how well-suited it would be for $t_i$ to choose $t_j$ as its exemplar. At the beginning, all availabilities are initialized to zero, i.e., for each $\mathrm{a}(t_i, t_j) = 0$; then responsibility and availability messages are updated using Eqs. (5, 6) Frey and Dueck 2007).

$$\mathrm{r}(t_i, t_j) = s(t_i, t_j) - \max_{j' \neq j} \left\{ a\left(t_i, t_{j'}\right) + s\left(t_i, t_{j'}\right) \right\} \tag{5}$$

$$\mathrm{a}(t_i, t_j) = \begin{cases} \min\left\{0, r\left(t_j, t_j\right) + \sum\limits_{i' \neq i, j} \max\{0, r\left(t_{i'}, t_j\right)\}\right\}, & i \neq j \\ \sum\limits_{i' \neq i} \max\{0, r\left(t_{i'}, t_j\right)\}, & i = j \end{cases}. \tag{6}$$

The responsibility and availability messages are updated iteratively for $m$ iterations, and a dumping factor, denoted by $\lambda, \lambda \in [0, 1]$, is added to both types of messages in order to avoid *numerical oscillations* (Frey and Dueck 2007), as depicted in Eqs. (7, 8).

$$R_{m+1} = (1 - \lambda) R_m + \lambda R_{m-1} \tag{7}$$

$$A_{m+1} = (1 - \lambda) Y_m + \lambda A_{m-1} \tag{8}$$

where $\boldsymbol{R}$ is the responsibility matrix, $\boldsymbol{R} = \left[r\left(t_i, t_j\right)\right]$ and $A$ is the availability matrix, $\boldsymbol{A} = \left[a\left(t_i, t_j\right)\right]$. AP continues updating $\mathrm{r}(t_i, t_j)$ and $\mathrm{a}(t_i, t_j)$ until they remain constant for a specified number of iterations, and then both types of mes-

sages are combined to discover the exemplar candidate terms in $\boldsymbol{T_D}$, specified as follows:

$$\varepsilon_j \leftarrow \arg_{1\le j\le N} \max\left[r(t_i,t_j)+a(t_i,t_j)\right], \quad \text{where } N=|T_D| \tag{9}$$

where $\varepsilon_j$ is a term in $\boldsymbol{T_D}$ and is regarded as an exemplar of term $t_i$. Eventually, every term in $\boldsymbol{T_D}$ is annotated with its exemplar. The number of clusters, and other clustering information, are directly obtained by grouping terms based on their shared exemplars. At start-up, we allow the set $\boldsymbol{T_D}$ to be *redundant* in order to incorporate not only the semantic and lexical information of each term $\boldsymbol{T_D}$ but also the influence of its frequency information on the clustering results, such that, if the term $t_i$ is highly frequent in the document, its frequency can be a reason to qualify as an exemplar on the condition that $t_i$ is always allocated the same WordNet synset $s_{ij}$ in all its occurrences in $\boldsymbol{D}$.

## 3.5 Selection of seeds

Typically, clustering-based AKE approaches use the centroids of clusters as seeds (Barker and Cornacchia 2000; Liu et al. 2009; Hasan and Ng 2014), and any phrase in $\boldsymbol{D}$ containing one or more centroids is chosen as a keyphrase. From our empirical observation, we suggest that direct selection of centroids resulting from the adopted clustering algorithm may lead to poor keyphrase extraction recall and/or precision, due to the following reasons:

### 3.5.1 Theme-independent seed selection

Clustering-based methods assign equal importance to all cluster centroids (Hasan and Ng 2014; Liu et al. 2010). Thus, a phrase containing a centroid of an unimportant cluster is ranked exactly equivalent to a phrase containing a centroid of an extremely important cluster relative to the document theme (Liu et al. 2010). Consequently, there is no guarantee that the extracted keyphrases are the best representative phrases. Our solution to this is to discard irrelevant or marginally related clusters and keep the most relevant ones. The solution is largely based on the observation that clusters that sufficiently cover the document theme tend to be semantically more related to each other than irrelevant or marginally related clusters. Regarding AP, the exemplar is the best representative of its cluster's semantics. Therefore, we assess the average of semantic relatedness strength of each exemplar against all other exemplars, and any cluster whose exemplar exhibits *weak* semantic relatedness is removed. Let $\boldsymbol{C_D}$ be the set of clusters resulting from clustering $\boldsymbol{T_D}$, $\boldsymbol{C_D}=\{C_1,C_2,\ldots,C_N\}$, where$\boldsymbol{N}=|\boldsymbol{C_D}|$. For each cluster $C_i$, we compute its exemplar's average semantic relatedness, $Ave(\varepsilon_i)$, as follows:

$$\text{Ave}(\varepsilon_i)=\frac{\sum_{i\neq j} SR(\varepsilon_i,\varepsilon_j)}{N-1}, \quad N>1 \tag{10}$$

Here $\boldsymbol{SR}(\varepsilon_i,\varepsilon_j)$ is a metric to quantify the semantic relatedness between the exemplars of two clusters $\boldsymbol{C}_i,\boldsymbol{C}_j$. Each cluster $\boldsymbol{C}_i$ is ranked based on its exemplar average score and is removed from $\boldsymbol{C_D}$ if its average score, $Ave(\varepsilon_i)$, is below the average of all clusters. $\boldsymbol{SR}(\varepsilon_i,\varepsilon_j)$ is concerned with measuring the relatedness between $\varepsilon_i$ and $\varepsilon_j$ rather than their latent semantic similarity. For instance, the terms "*drug*" and "*Olympics*" are not similar, but, because of their tendency to co-occur together ("*drug use*" appears frequently in "*Olympics*" themes), they are judged semantically related. To quantify such relatedness in an unsupervised cross-domain environment, we expand SemCluster to take advantage of *Wikipedia*, the largest and fastest growing knowledge base. There are a number of approaches that measure semantic relatedness by exploiting Wikipedia. *Explicit Semantic Analysis* (Gabrilovich and Markovitch 2007) is one of the most accurate Wikipedia-based measures that, to an extent, comes close to the accuracy of a human (Witten and Milne 2008), and, hence, is employed by SemCluster to compute the relatedness of exemplars.

### 3.5.2 Search space restriction

Relying solely on the centroids of clusters may lead to restricting the search space for finding the best representative phrases in a given document and, consequently, may result in degrading keyphrase extraction recall and/or precision. Suppose we have a valid keyphrase containing a term $t_j$ that is semantically close to a centroid term $\varepsilon_i$. The phrase will not be selected as a candidate keyphrase simply because $t_j$ is not a centroid. This may explain why the spectral clustering algorithm outperforms AP in KeyCluster experiments—the former allows multiple terms close to a cluster centroid to be chosen as seeds and accordingly, extends the keyphrase search space. Taking advantage of this observation, SemCluster expands the selection of seeds from AP clustering in a fashion similar to that of spectral clustering. Let $\boldsymbol{C}'_{\boldsymbol{D}}$ be the final set of clusters resulting from clustering $\boldsymbol{T_D}$ using AP after centroid relatedness average ranking, where $\boldsymbol{C}'_{\boldsymbol{D}}\subseteq\boldsymbol{C_D}$, $\boldsymbol{C}'_{\boldsymbol{D}}=\{\boldsymbol{C}_1,\boldsymbol{C}_2,\ldots,\boldsymbol{C}_k\}$. For each cluster $\boldsymbol{C_i}$, $i\le k$, we select its exemplar $\varepsilon_i$ as a seed. We regard each member $t_j$ in $\boldsymbol{C_i}$ ($t_j\neq\varepsilon_i$) as an additional seed if $S(\varepsilon_i,t_j)\ge\tau$, where $S(\varepsilon_i,t_j)$ is the computed score stored in $\mathcal{A}$ from the previous step (see Sect. 3.3), and $\tau$ is a predefined distance threshold specifying how semantically close $t_j$ should be to the centroid $\varepsilon_i$ in order to qualify as a seed.

We repeat this procedure for all the clusters in $\boldsymbol{C}_{\boldsymbol{D}}^{'}$ to obtain a set of appropriate seeds from the extended search space.

### 3.6 Candidate phrase extraction and keyphrase selection

After selection of the seeds, each chunk $NP_i$ in $\boldsymbol{D}$ is scanned by SemCluster. Any sequence of words in $NP_i$ is regarded as a *candidate phrase* if it satisfies the following conditions: i) it contains a seed and ii) it matches any of the following POS-based extraction rules:

- $NP_i$ contains a seed extracted using an E-pattern.
- If $NP_i$ contains a seed extracted using a $\mathcal{C}$-pattern, then two cases are considered: if the seed starts with $(JJ)$, then the sequence matching the pattern $(\mathcal{C}) * (NN|\text{NNS})+$ is extracted from $NP_i$; if the seed starts with (NN), then the sequence matching pattern $(JJ) * (\mathcal{C})+$ is extracted from $NP_i$).
- If $NP_i$ contains a seed extracted using a $\mathcal{N}$-pattern, then the sequence matches pattern $(JJ) * (\mathcal{N}) +$ is extracted from $NP_i$.

Once all NP chunks have been scanned and processed, the step proceeds to the next phase—refining the set of extracted candidate phrases. The refining phase starts by pruning redundant candidate phrases. Two or more candidate phrases may be semantically equivalent but exist in different forms. They may be synonymous phrases, for example, in Fig. 1, both "*Olympics*" and "*Olympic Games*" belong to the synonyms list of the same WordNet synset, or adjective synonymous phrases. For example, in the Wikipedia article about "*Bernard Madoff,*"[8] there are many candidate phrases which share the same representative seed "*fraud,*" such as "*financial fraud,*" "*gigantic fraud,*" "*massive fraud,*" and in this case, we keep the first occurring candidate phrase and remove the others. There is also the case of subphrases, as in the example of "*Johnson*" and "*Ben Johnson.*" Both phrases contain "*Johnson,*" so we keep the longer phrase, which is more specific and discard the shorter one.

By default, refined candidate phrases are selected as appropriate keyphrases for the input document $D$. However, for documents with moderate content size, the set of output keyphrases may be relatively large, which would affect the algorithm's performance. To overcome this drawback, we adopt an empirically effective heuristic from (El-Beltagy and Rafea 2009), where the position that a given candidate term first occurs in a lengthy document in significant in two ways: (i) it is likely that the keyphrase of more importance appears "*sooner*" in the document than others, and (ii) after a certain location in the document, candidate phrases that appear for the first time are highly unlikely to be keyphrases. Based on such an empirical heuristic, for a lengthy input document, we predefine a window of size $k$, and any candidate phrase that occurs beyond a window starting from the first word up to the $k$th word is disregarded.

[8] https://en.wikipedia.org/wiki/Bernard_Madoff.

**Table 2** Keyphrase evaluation datasets

| Name | Domain | $\#D$ | $W/D$ | $KP/D$ | $eKP/D$ |
|---|---|---|---|---|---|
| Inspec | Scientific | 500 | 121.824 | 9.826 | 7.726 |
| DUC | News | 308 | 740 | 8.080 | – |

$D$ document, $W$ word, $KP$ manually assigned keyphrase, $eKP$ $KP$ exists in the text

## 4 Evaluation

To evaluate SemCluster, two experiments are conducted using two evaluation datasets, and the results are reported in this section. In the first experiment, we examine the impact of SemCluster parameter settings on the keyphrase extraction performance, and we also provide guidelines for parameter setting in two popular domains. In the second experiment, SemCluster is compared with multiple AKE methods in terms of precision, recall, and F-measure of the reported keyphrases.

### 4.1 Datasets and evaluation metrics

Two frequently used datasets in AKE literature are chosen as the evaluation datasets: Inspec[9] (Hulth 2003) and DUC-2001.[10] Both datasets consist of free-text documents with manually assigned keyphrases and differ in length and domain (see Table 2) and, therefore, are appropriate to test the robustness of SemCluster AKE performance over documents that belong to different domains.

The Inspec dataset is a collection of abstracts of scientific papers from the Inspec database, consisting of 2000 abstracts. Each abstract is represented by three files: *.abstr, .contr,* and *.uncontr.* The file *.abstr* contains the abstract content; *.contr* contains keyphrases restricted to a specific dictionary; and *.uncontr* contains keyphrases freely assigned according to the personal judgements of human curators. In Hulth's work (2003), the evaluated AKE method was supervised, and the dataset was split into three partitions: 1000 abstracts for training, 500 for validation, and 500 for testing. TextRank and KeyCluster are unsupervised methods, and thus only the test partition was used in their evaluations. Since SemCluster is also unsupervised, we adopt a similar approach and

[9] https://github.com/alrehamy/SemCluster/data/inspec.

[10] http://www-nlpir.nist.gov/projects/duc/guidelines/2001.html.

use only the test partition to provide a precise comparison with the other AKE methods mentioned. As listed in Table 1, the average length of each abstract ($\boldsymbol{W/D}$) is 121.824, and the average number of keyphrases assigned to each abstract ($\boldsymbol{KP/D}$) is 9.826. However, since the manual assignment of keyphrases is uncontrolled, not all the keyphrases in a particular .uncontr file necessarily occur in the corresponding .abstr file. Instead, any phrases regarded by the human curators as suitable are stored in .uncontr as valid keyphrases. In our evaluation, we programmatically[11] scan each .uncontr file and filter out any keyphrases that do not occur in the corresponding .abstr file. A similar preprocessing practice has been applied to the dataset during the experimental evaluations of TextRank, ExpandRank (Wan and Xiao 2008), and KeyCluster as well as many others. After processing the dataset, the average number of assigned keyphrases ($\boldsymbol{eKP/D}$) drops to 7.726.

The DUC-2001 dataset is a collection of news articles retrieved from TREC-9, originally consisting of 309 articles with one duplicate (i.e., d05a\FBIS-41815 and d05a\FBIS-41815~). The dataset was originally published as a benchmark for a document summarization task, and (Wan and Xiao 2008) have used human curators to manually annotate each article with 10 keyphrases in order to evaluate the ExpandRank algorithm. The Kappa statistic of inter-agreement between the curators regarding manual keyphrase assignments was 0.7, and assignment conflicts were resolved by discussions, and therefore, the $\boldsymbol{KP/D}$ dropped to 8.08. Each article is represented as a.txt file and consists of multiple HTML tags. In our evaluation, we only consider the textual content in text tags (i.e., <text>…</text>).

As mentioned earlier, the metrics used for all SemCluster evaluations are *precision (P)*, *recall (R)*, and *F-measure (F)*, which are defined as follows:

$$P = \frac{KP_{\text{correct}}}{KP_{\text{extracted}}}, \quad R = \frac{KP_{\text{correct}}}{KP_{\text{gold}}}, \quad F = 2 \times \frac{P \times R}{P + R} \tag{11}$$

where $KP_{\text{correct}}$ is the number of correct keyphrases extracted by SemCluster, $KP_{\text{extract}}$ is the total number of keyphrases extracted, and $KP_{\text{gold}}$ is the total number of keyphrases manually assigned by human curators, which in our case are considered the gold standard. An output phrase extracted from a given input document is regarded as a valid keyphrase if it is identical to, semantically equivalent to, or is a subphrase of, a gold standard keyphrase manually assigned to the document in any given dataset.

[11] Datasets statistics are calculated using the code at https://github.com/alrehamy/SemCluster/data/stats.

### 4.2 SemCluster prerequisites

In the first step of SemCluster (see Sect. 3.1), we adopt OpenNLP,[12] an open source and publicly available NLP library, for text preprocessing. The content of an input document is tokenized using a rule-based tokenizer, whereas sentence boundary detection, POS tagging, and chunking are performed using a maximum entropy sequence labeling algorithm that utilizes large machine learning models trained on corpora in multiple domains. The default background knowledge source of SemCluster is WordNet[13] v.3.1. We perform slight modifications on the *data.noun* and *index.noun* files to accommodate our needs, including re-indexing the original byte-based synsets' indices for faster access, POS tagging the tokens in each synset's gloss, filtering out any token that is not tagged as a noun or adjective, and lemmatizing the result gloss to improve string-based operations during disambiguation similarity computations of terms (see Sect. 3.3).

To support SemCluster with rich and tractable background information, we adopt two external knowledge bases to reinforce the semantic coverage of WordNet: DBPedia, and BabelNet. For DBPedia integration, its schema ontology is aligned with the WordNet ontology using the alignment procedure described in Sect. 3.1, and the alignment results are made publicly available.[14] For computational efficiency, we adopt a lookup-server[15] that allows DBPedia to be run in a local mode. BabelNet is a lexicalized semantic network that combines and interlinks knowledge facts extracted from many online resources (Navigli and Ponzetto 2012), providing unified access to them.[16] Similar to the structure of WordNet, a noun phrase in BabelNet may have one or more synsets, with each synset consisting of a short definition that is often extracted from Wikipedia, and a list of one or more type classes that are expressed as concepts and linked with the noun phrase using an Is-A relationship. Unlike DBPedia, BabelNet utilizes WordNet directly as its schema ontology, which makes its integration in SemCluster a straightforward undertaking.

Finally, we use EasyESA[17] as a local server for measuring the relatedness between cluster centroids using the Wikipedia-based ESA metric (see Eq. 10).

[12] http://opennlp.apache.org.

[13] WordNet v.3.1 is available at https://wordnet.princeton.edu/wordnet/download.

[14] https://github.com/alrehamy/SemCluster/extensions/dbpedia/alignment.

[15] At https://github.com/dbpedia/lookup.

[16] http://babelnet.org/download.

[17] http://treo.deri.ie/easyesa.

### 4.3 Comparative methods and parameter settings

Three unsupervised AKE methods relevant to the SemCluster workflow are selected for comparative evaluation: TextRank, ExpandRank and KeyCluster. TextRank is a graph-based method that computes the importance scores of candidate words using only *local* structure information embedded in the word graph of the document; ExpandRank is also a graph-based method that exploits an *external* textual neighborhood in addition to the local structure information of the document's word graph to enhance co-occurrence relations between graph nodes; KeyCluster is a cluster-based method that exploits Wikipedia as an external background knowledge source to capture the semantic relations between candidate terms and compute their pairwise similarities. KeyCluster is implemented using three different clustering algorithms: hierarchal clustering (HC), spectral clustering (SC), and affinity propagation (AP). Due to the poor performance of HC reported in (Liu et al. 2009), we evaluate KeyCluster based on only SC and AP implementations.

During the test, only the best results under the best possible parameter settings, if any, for a given method are considered. As shown in Table 2, the $\boldsymbol{eKP/D}$ of each dataset is less than 10; therefore, we set the co-occurrence window in ExpandRank to 10, whereas for TextRank, the co-occurrence window size is set to 2 for Inspec, and 5 for DUC-2001. The PageRank dumping factor is a constant value that is used to balance the probability of a random walk from a given node to a random node in the graph. Setting this factor to 0.85 has been shown to be the best empirical setting not only in web surfing (Page et al. 1999), but also in keyphrase extraction (Mihalcea 2004). For ExpandRank, we set the number of neighbor documents to 5, because for this setting ExpandRank obtains the highest F score. The setting for KeyCluster-SC is that, $m$, the predefined number of clusters, is $m = \frac{2}{3}n$, where $n = |\boldsymbol{D}|$. For KeyCluster-AP, the maximum number of iterations is set to 1000, the propagation damping factor is set to 0.9, and the clustering preference is computed using *mean*, which has been shown to outperform other preference functions in KeyCluster experiments.

Although SemCluster performs AKE in fully automatic mode, it requires general tuning for a set of parameters, which are: (1) WSD context windows size $N$ (see Sect. 3.2), (2) AP algorithm parameters (see Sect. 3.4), (3) distance threshold $\tau$ (see Sect. 3.5), and (4) window size $k$ (see Sect. 3.6). From empirical observation, SemCluster performs the best possible WSD when $N$ =10. However, when $N$ <10, WSD performance degrades, whereas $N$ >10 has no discernible influence on the task. The default tuning of AP parameters is as follows: $m$ is set to 500, and $\lambda$ is set to 0.9 similar to that of KeyCluster. As indicated earlier, AP iteratively computes responsibilities and availabilities, and the execution terminates only if decisions for the exemplars and the cluster boundaries are unchanged for *convit* iterations. For computational efficiency, we set *convit* to 50. The custom tuning of AP parameters has no influence on the clustering results regardless of the dataset used during evaluation or its domain, because the input similarities are always positive and in the range [0,1]. Unlike KeyCluster, we choose the *median* function as SemCluster's clustering preference, to ensure that SemCluster performs clustering with higher granularity (i.e., a larger number of clusters) so that unimportant terms with weak inter-cluster relations can be automatically allocated in unimportant clusters and hence easily identified and pruned from the clustering results using Eq. (10). As shown in Table 2, the $\boldsymbol{W/D}$ of Inspec abstracts is very low, and therefore, we set $k = |\boldsymbol{D}|$. Conversely, the $\boldsymbol{W/D}$ of DUC-2001 articles is relatively high, and therefore, we set $k = 400$ (El-Beltagy and Rafea 2009), and, if $k > |\boldsymbol{D}|$ then $k = |\boldsymbol{D}|$.

The distance threshold $\tau$ has a direct influence on SemCluster's performance, such that, when $\tau$ =1, only centroids of clusters are chosen as seeds to identify and extract candidate keyphrases; when $\tau = 0$, all the terms in $T_D$ (except those belonging to the pruned clusters) are selected as seeds, and hence most NP chunks in $\boldsymbol{D}$ are chosen as keyphrases. Given that the pairwise semantic similarity score 0.5 is the least extent to which two terms can be judged similar on scale from 0 (dissimilarity) to 1 (identicality) (Tversky 1977), then $\tau$ can be assigned any value in the range $0.5 \leq \tau < 1$. Estimating the optimal value of $\tau$ is a hyperparameter optimization problem that can be readily solved either by multiple trials or by employing a dedicated optimization search algorithm such as random search (Bergstra and Bengio 2012). In this work, we design a sampling-based procedure to infer the best $\tau$ setting: from each evaluation dataset we select 100 random documents as inputs to SemCluster, select different $\tau$ settings starting from $\tau = 0.99$ and gradually decrease it in the series $\tau_{i+1} = \tau_i - 0.01$, testing the precision and recall of SemCluster's output from each run using the value $\tau_{i+1}$. The results of our sampling-based trials are plotted in Fig. 5 for both datasets. As depicted in Fig. 5a, the precision and recall scores are very low when $\tau > 0.8$, and this is because a relatively large number of important candidate terms are not close enough to their cluster centroids in order to qualify as seeds, and consequently, many valid keyphrases are not identified by SemCluster as construed in Sect. 3.5. However, when $\tau < 0.8$, the performance gradually improves as semantically important terms start being qualified as seeds, which contributes toward improving the total number and the quality of extracted keyphrases. SemCluster's best performance ($P = 0.401, R = 0.742$) is achieved when $\tau = 0.665$. Similarly, Fig. 5b depicts SemCluster's performance for the DUC-2001 dataset using different $\tau$ settings. A prominent performance improvement is achieved when $\tau < 0.8$ and continues to gradually improve until

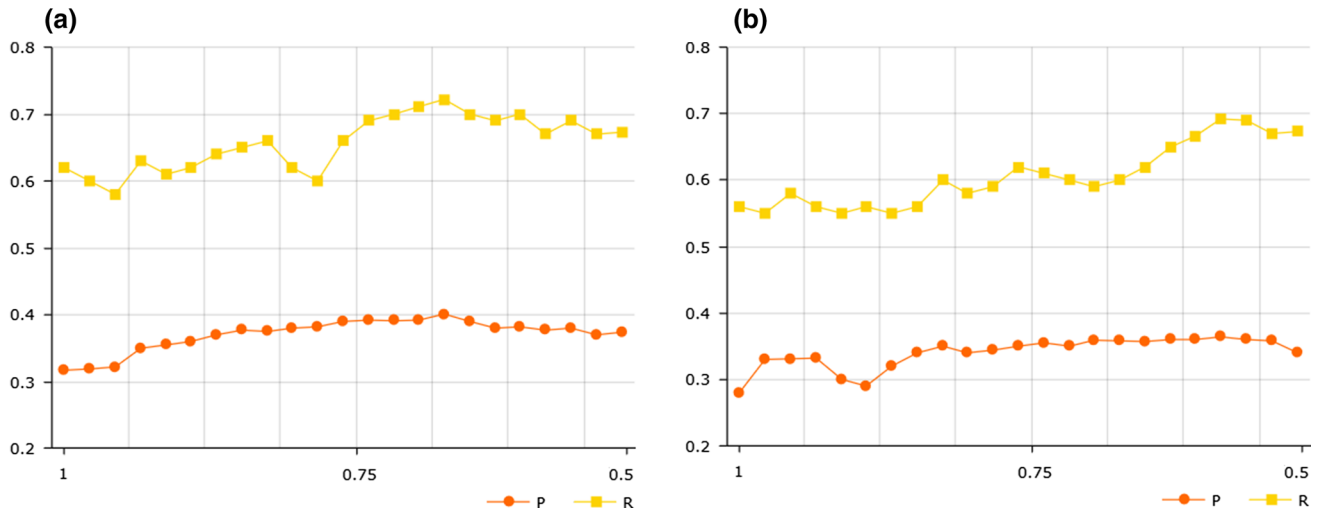



**Fig. 5** Impact of $\tau$ on SemCluster performance using various settings in the range $0.5 \leq \tau < 1$. **a** *P*/*R* of runs on Inspec dataset. **b** *P*/*R* of runs on DUC-2001 dataset

**Table 3** Comparison of SemCluster against other algorithms

| Methods | Inpec | | | DUC-2001 | | |
|---|---|---|---|---|---|---|
| | *P* | *R* | *F* | *P* | *R* | *F* |
| TextRank | 0.312 | 0.431 | 0.362 | 0.189 | 0.391 | 0.127 |
| ExpandRank | 0.344 | 0.471 | 0.398 | 0.288 | 0.354 | 0.317 |
| $\text{KeyClsuter}_{\text{SC}}$ | 0.350 | 0.660 | 0.457 | 0.256 | 0.529 | 0.345 |
| $\text{KeyCluster}_{\text{AP}}$ | 0.330 | 0.697 | 0.448 | 0.239 | 0.538 | 0.331 |
| SemCluster | **0.401** | **0.742** | **0.520** | **0.364** | **0.692** | **0.477** |

Bold values indicate the best result for each dataset

$\tau = 0.59$, where the best performance ($P = 0.364$, $R = 0.692$) is realized.

## 4.4 Performance comparison and results

Using Inspec and DUC-2001, we compare SemCluster's performance with the methods described in the previous section. Table 3 presents the evaluation results of each evaluation dataset in terms of the precision, recall, and F-measure of the extracted keyphrases. The results show that, for both datasets, SemCluster outperforms the compared methods on the recall of correct keyphrases and the precision of the extracted keyphrases. Comparing with KeyCluster-SC, which has the second-best performance, SemCluster achieves F-measure improvements of ~ 14 and ~ 38%, respectively. Although both SemCluster and KeyCluster-AP utilize the same clustering algorithm, the former outperforms the latter with F-measure improvements of ~ 16 and ~ 44%, respectively. To the best of our knowledge, SemCluster's F-measure scores of 0.520 and 0.477 are the highest among current state-of-the-art unsupervised cluster-based methods.

The main contributors to the significant improvements in the F-measure in SemCluster can be summarized as follows:

1. Given sufficient background knowledge, we extract n-grams from the input document's content as potential candidates, including successfully mapped noun phrases and proper named entities (see Table 1), while other state-of-the-art approaches typically extract single words only, causing many potentially important candidates to be either eliminated early or to become semantically inadequate during the selection of terms. For example, instead of selecting "third world" as a candidate term (which is a compound noun manually assigned as a keyphrase for the article *AP880926-0203*/DUC-2001), all comparative methods extract the words "third" and "world" separately. The drawback of n-gram terms selection, however, is that it may lead to keyphrase overgeneration (Hasan and Ng 2014). SemCluster overcomes this issue by eliminating semantically irrelevant candidates during cluster pruning, as discussed in Sect. 3.5, thus boosting SemCluster's recall.
2. Although the background knowledge obtained from relevant documents used in ExpandRank, and the vector representation of terms based on Wikipedia articles used in KeyCluster, contribute to enhancing their F scores compared with TextRank, SemCluster's exten-

sible background knowledge is more effective. This because SemCluster clusters candidate terms based on their latent semantic relations rather than frequency and co-occurrence statistics, and also obtains thematically representative seeds even if they occur infrequently in the input document to improve the keyphrase extraction precision.

3. We observe that expanding seeds with $\tau$ equal to 0.665 and 0.59 for Inspec and DUC-2001, respectively, allows SemCluster to extract keyphrases that match the gold standard keyphrases, while KeyCluster fails to identify them because their corresponding seeds often do not qualify as cluster centroids and are thus eliminated from the clustering results. This accounts for the significant improvements in the recall and precision of SemCluster, compared with both implementations of KeyCluster.

It is also noteworthy that SemCluster is more computationally efficient than the other methods, especially KeyCluster. Due to its reliance on WordNet, SemCluster loads the WordNet ontology and any related ontology alignments into its physical memory (WordNet noun files and external ontology mapping files require ~ 22 MB) so that accessing the semantics of a term in $\boldsymbol{D}$ requires O(1) time. Because of this, our method performs AKE with significant improvements in computational complexity compared with other methods. For example, KeyCluster requires ~ 5 M Wikipedia articles to be crawled in order to construct the Wikipedia-based conceptual vector for each term in $\boldsymbol{D}$ during the pairwise similarity computation of terms. Furthermore, Wikipedia crawling is correlated with the length of the input document, whereas SemCluster accesses Wikipedia only for computing the relatedness averaging for cluster centroids, which, based on we have observed, often requires less than 15 centroids in the evaluation.

One of the main contributions of SemCluster is the way that background knowledge extensibility is leveraged to overcome knowledge and semantic losses. To evaluate the impact of knowledge extensibility on SemCluster performance, we produced two implementations of SemCluster. In the first implementation, denoted as $\text{SemCluster}_{\text{DBP}}$, we extend WordNet using DBPedia only, and in the second version, denoted as $\text{SemCluster}_{\text{DBP,BN}}$, WordNet is extended with DBPedia as well as BabelNet. Table 4 presents a performance comparison between these implementations using the same evaluation datasets and settings described above. The results indicate that $\text{SemCluster}_{\text{DBP,BN}}$ outperforms $\text{SemCluster}_{\text{DBP}}$ in all the metrics except for the precision metric on DUC-2001. Although the improvements in $\text{SemCluster}_{\text{DBP,BN}}$ performance are not significant, they provide empirical evidence that background knowledge extensibility can enhance the AKE performance of the unsupervised clustering-based method.

As depicted in Fig. 6, it can be readily seen that both SemCluster implementations perform AKE more efficiently on Inspec than DUC-2001. This performance aspect is also shared with all the comparator methods as presented in Table 3. In SemCluster, this may be explained as follows: (1) as presented in Table 2, Inspec documents are shorter than their DUC-2001 counterparts, such that, for any given document $\boldsymbol{D}$, $k = |\boldsymbol{D}|$, whereas for DUC-2001 documents, $k = 400$, and accordingly, valid keyphrases that occur

**Table 4** Comparison of $\text{SemCluster}_{\text{DBP}}$ and $\text{SemCluster}_{\text{DBP,WN}}$

| Versions | Inspec | | | DUC-2001 | | |
|---|---|---|---|---|---|---|
| | *P* | *R* | *F* | *P* | *R* | *F* |
| $\text{SemCluster}_{\text{DBP}}$ | 0.392 | 0.721 | 0.507 | **0.371** | 0.639 | 0.475 |
| $\text{SemCluster}_{\text{DBP,BN}}$ | **0.401** | **0.742** | **0.520** | 0.364 | **0.692** | **0.477** |

Bold values indicate the best result for each dataset

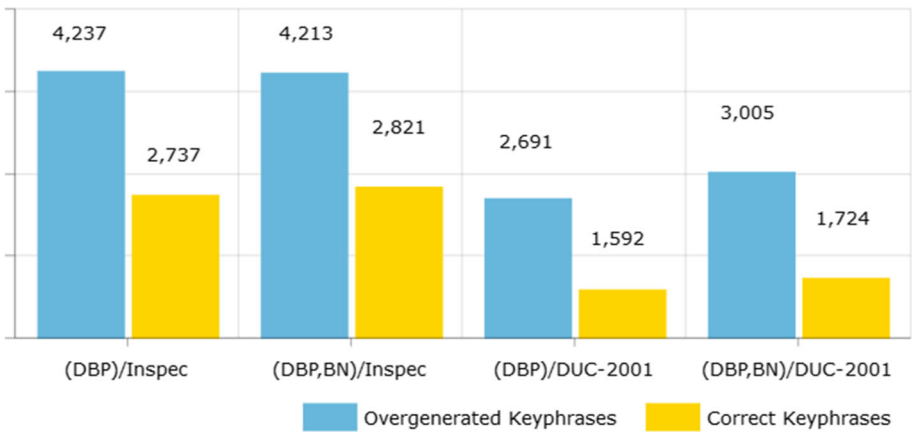



**Fig. 6** Influence of background knowledge on the keyphrase extraction result

**Table 5** SemCluster extraction results from the article sample depicted in Fig. 1

| Candidate phrase | Centroid | Ontological tagging | Valid |
|---|---|---|---|
| NNP/Ben NNP/Johnson | wn:athlete#1 | wn:#9820263 | Yes |
| NNPS/Olympics | wn:olympics#1 | wn:#7457126 | Yes |
| JJ/100-meter NN/dash | wn:prize#1 | wn:#7469043 | Yes |
| NN/gold NN/medal | wn:prize#1 | wn:#3444942 | Yes |
| NNP/Carl NNP/Lewis | wn:athlete#1 | wn:#11131135 | Yes |
| NNP/Johnson | wn:athlete#1 | wn:#9820263 | No |
| JJ/anabolic NN/steroid | wn:drug#1 | wn:#15111116 | Yes |
| JJ/urine NN/sample | wn:olympics#1 | wn:#6026635 | Yes |
| NNP/Stanozolol | wn:drug#1 | wn:#3247620 | Yes |
| NN/drug NN/use | wn:drug#1 | wn:#3247620 | Yes |
| JJ/Olympic NNPS/Games | wn:olympics#1 | wn:#7457126 | No |

outside the window $k$ are eliminated in the early steps of the SemCluster workflow, leading to degraded recall; (2) Inspec documents contain more scientific and technical noun phrases than DUC-2001, many of which have matching entries in BabelNet, and therefore are picked by SemCluster as valid keyphrases, thus boosting both implementation's F scores. In contrast, a news article in DUC-2001 often contains more named entities that tend to have high semantic similarities due to infrequent topic shifting or changing (Allan 2012), consequently leading to the over generation of keyphrases and degraded recall.

Tomokiyo and Hurst (2003) propose that an *appropriate* keyphrase should be a semantically and syntactically correct phrase without any unnecessary words and suggest a measure to quantify the appropriateness of keyphrases, which is called *phraseness*. Similarly, Liu et al. (2009) suggest that keyphrases should be *understandable* to humans in order to qualify as appropriate keyphrases. They give an example that "machine learning" is appropriate, while "machine learned" is not. Based on our empirical evaluation, we observe that SemCluster always extracts appropriate keyphrases because candidate phrases are extracted from the input document using a set of NLP patterns (see Sect. 3.6) that encode generally accepted linguistic knowledge/feature assumptions (Hulth 2003). Table 5 lists the result of applying SemCluster to the news article depicted in Fig. 1. It can be readily seen that all candidate phrases are human readable and without any unnecessary words. Furthermore, semantically duplicated candidates (marked with No) are automatically identified and filtered out from the final results in the last step of SemCluster. For example, the candidate phrase "Olympic Games" is removed because it is semantically identical with "Olympics" (i.e., both belong to the same synonyms list of the synset "wn:#7457126"), and likewise, "Johnson" is removed as it is a substring of "Ben Johnson."

Unlike other state-of-the-art AKE algorithms, SemCluster outputs keyphrases that are automatically associated with two types of machine-readable metadata as presented in Table 5, which are: (1) the WordNet synset of the cluster's centroid to which the keyphrase belongs and (2) the WordNet synset of the seed embedded in the keyphrase. Such metadata are valuable semantic information that allows related keyphrases within and without the document to be unambiguously linked. For example, keyphrases like "anabolic steroid," "Stanozolol," and "drug use" can be grouped together because they share the same ontological annotation concept, i.e., "wn:drug#1"; similarly, "Ben Johnson" and "Carl Lewis" can be grouped together based on their shared concept "wn:athlete#1." Likewise, the document can be grouped with other semantically similar documents if they share the same keyphrases, or with topically similar documents if they share the same annotation concepts. The machine-readable metadata of keyphrases are equivalent to the metadata generated using traditional semantic annotation tools such as (Erdmann et al. 2000; Giannopoulos et al. 2010); thus, they can be utilized to efficiently support many semantics-based data management tasks such as semantic search (Bontcheva and Rout 2014), content aggregation and recommendation (Yang et al. 2017; Nguyen et al. 2016), and automatic relationship discovery (Xu et al. 2015).

## 5 Conclusions and future work

In this paper, we have introduced SemCluster, a clustering-based unsupervised keyphrase extraction method. By incorporating extensible background knowledge, SemCluster identifies and extracts semantically important terms from a given document, clusters them, and identifies thematically important seeds that are then used to search for representative phrases and from which appropriate keyphrases are selected. We conducted two experiments over two datasets of various document lengths and domains to study multiple aspects of SemCluster performance. The results show that SemCluster outperforms the compared methods and thus verifies the findings of Liu et al. (2009) that unsupervised clustering-based AKE methods can be effective and robust, even across multiple domains.

SemCluster is a part of a larger project specializing in big data processing called the personal data lake (PDL) (Alrehamy and Walker 2015). Though SemCluster is a general

tool for extracting keyphrases from textual data, PDL is the main motivation behind the development of SemCluster. PDL requires a domain-agnostic AKE tool to process free-text documents ingested from heterogeneous data providers and extract from them keyphrases that are automatically annotated with ontological concepts. This allows machine-readable metadata for documents to be generated and with the best possible accuracy and computational efficiency and thus to interrelate them at a micro-semantic level.

Although SemCluster exhibits better performance than other approaches, there is still room for improvement. In an experiment on a collection of mixed documents from Inspec and DUC-2001, we replaced the WuPalmer measure with the Jiang-Conrath metric (Jiang and Conrath 1997) and used Babelfy (Moro 2014) for the WSD task. An improvement in F1-measure compared with that of SemCluster$_{DBP,BN}$ was observed; however, its computational efficiency significantly decreased because Babelfy is available only as an online service. This suggests a potential enhancement to SemCluster, particularly by improving its semantic similarity metric and WSD algorithm. We are also interested in extending WordNet with more personalized knowledge sources and study their impact on performance using personal documents with greater length and domain variance (e.g., emails, health records, microblogs) than the currently used datasets.

**Acknowledgements** The authors declare that they have received no research grants for the research discussed in this paper.

## Compliance with ethical standards

**Conflict of interest** The authors declare that they have no conflict of interest.

**Ethical standards** The authors declare that the research does not contain any studies with human participants or animal parts performed by the authors.



## References

Allan J (2012) Topic detection and tracking: event-based information organization, vol 12. Springer, Berlin

Alrehamy H, Walker C (2015) Personal data lake with data gravity pull. In: Proceedings of the 2015 IEEE fifth international conference on big data and cloud computing, pp 160–167

Androutsopoulos I, Koutsias J, Chandrinos K, Spyropoulos C (2000) An experimental comparison of naive Bayesian and keyword-based anti-spam filtering with personal e-mail messages. In: Proceedings of the 23rd annual international ACM SIGIR conference on research and development in information retrieval, pp 160–167

Auer S, Bizer C, Kobilarov G, Lehmann J, Cyganiak R, Ives Z (2007) Dbpedia: a nucleus for a web of open data. The semantic web, pp 722–735

Barker K, Cornacchia N (2000) Using noun phrase heads to extract document keyphrases. Advances in Artificial Intelligence

Beliga S, Mestrovic A, Martincic-Ipsic S (2015) An overview of graph-based keyword extraction methods and approaches. J Inf Organ Sci 39(1):1–20

Bergstra J, Bengio Y (2012) Random search for hyper-parameter optimization. J Mach Learn Res 13(2):281–305

Blei D, Ng Y, Jordan M (2003) Latent dirichlet allocation. J Mach Learn Res 3(1):993–1022

Bontcheva K, Rout D (2014) Making sense of social media streams through semantics: a survey. Semant Web 5(5):373–403

Bracewell D, Ren F, Kuroiwa S (2005) Multilingual single document keyword extraction for information retrieval. In: Proceedings of the 2005 IEEE international conference on natural language processing and knowledge engineering. Wuhan, pp 517–522

Cilibrasi R, Vitanyi P (2007) The google similarity distance. IEEE Trans Knowl Data Eng 19(3):370–383

Clark A, Fox C, Lappin S (2013) The handbook of computational linguistics and natural language processing. Wiley, Hoboken

Danesh S, Sumner T, Martin J (2015) SGRank: combining statistical and graphical methods to improve the state of the art in unsupervised keyphrase extraction. In: Proceedings of the fourth joint conference on lexical and computational semantics, pp 117–126

Danilevsky M, Wang C, Desai N, Ren X, Guo J, Han J (2014) Automatic construction and ranking of topical keyphrases on collections of short documents. In: Proceedings of the 2014 SIAM international conference on data mining, SIAM, pp 398–406

El-Beltagy S, Rafea A (2009) KP-Miner: a keyphrase extraction system for English and Arabic documents. Inf Syst 34(1):132–144

Erdmann M, Maedche A, Schnurr H, Staab S (2000) From manual to semi-automatic semantic annotation: about ontology-based text annotation tools. In: Proceedings of the COLING-2000 workshop on semantic annotation and intelligent content. ACL, pp 79–85

Fellbaum C (1998) WordNet. Wiley, Hoboken

Frey B, Dueck D (2007) Clustering by passing messages between data points. Science 315(5814):972–976

Gabrilovich E, Markovitch S (2007) Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In: Proceedings of the 20th international joint conference on artificial intelligence, pp 1606–1611

Giannopoulos G, Bikakis N, Dalamagas T, Sellis T (2010) GoNTogle: a tool for semantic annotation and search. In: extended semantic web conference. Springer, pp 376–380

Guan R, Shi X, Marchese M, Yang C, Liang Y (2011) Text clustering with seeds affinity propagation. IEEE Trans Knowl Data Eng 23(4):627–637

Hammouda K, Matute D, Kamel M (2005) Corephrase: keyphrase extraction for document clustering. MLDM 2005:265–274

Hasan K, Ng V (2010) Conundrums in unsupervised keyphrase extraction: making sense of the state-of-the-art. In: Proceedings of the 23rd international conference on computational linguistics: posters. ACL, pp 365–373

Hasan K, Ng N (2014) Automatic keyphrase extraction: a survey of the state of the art. In: ACL (1), pp 1262–1273

Hoffart J, Suchanek F, Berberich K, Weikum G (2013) YAGO2: a spatially and temporally enhanced knowledge base from Wikipedia. Artif Intell 194:28–61

Hulth A (2003) Improved automatic keyword extraction given more linguistic knowledge. In: Proceedings of the 2003 conference

on empirical methods in natural language processing. ACL, pp 216–223
Jiang J, Conrath D (1997) Semantic similarity based on corpus statistics and lexical taxonomy. arXiv: preprint cmp-lg/9709008
Kim S, Medelyan O, Kan M, Baldwin T (2013) Automatic keyphrase extraction from scientific articles. Lang Resour Eval Springer 47(3):723–742
Le T, Le N, Shimazu A (2016) Unsupervised Keyphrase extraction: introducing new kinds of words to keyphrases. In: Australasian joint conference on artificial intelligence, Springer International Publishing, pp 665–671
Liu Z, Li P, Zheng, Y, Sun M (2009) Clustering to find exemplar terms for keyphrase extraction. In: Proceedings of the 2009 conference on empirical methods in natural language processing, pp 257–266
Liu Z, Huang W, Zheng Y, and Sun M (2010) Automatic keyphrase extraction via topic decomposition. In: Proceedings of the 2010 conference on empirical methods in natural language processing. ACL, pp 366–376
Martinez-Romo J, Araujo L, Fernandez A (2016) SemGraph: extracting keyphrases following a novel semantic graph-based approach. J Assoc Inf Sci Technol 67(1):71–82
Meng L, Huang R, Gu J (2013) A review of semantic similarity measures in WordNet. Int J Hybrid Inf Technol 6(1):1–12
Mihalcea, R, Tarau P (2004) TextRank: bringing order into texts. In: Proceedings of EMNLP. ACL, pp 404–411
Moro, A, Cecconi F, Navigli R (2014) Multilingual word sense disambiguation and entity linking for everybody. In: Proceedings of the 2014 international conference on posters & demonstrations track-vol 1272, pp 25–28
Munoz A (1997) Compound key word generation from document databases using a hierarchical clustering ART model. Intell Data Anal 1(4):25–48
Navigli R (2009) Word sense disambiguation: a survey. ACM Comput Surv 41(2):1–69
Navigli R, Ponzetto S (2012) BabelNet: the automatic construction, evaluation and application of a wide-coverage multilingual semantic network. Artif Intell 193:217–250
Nguyen Q, Nguyen T, Cao T (2016) Semantic-based recommendation method for sport news aggregation system. In: International conference on research and practical issues of enterprise information systems, Springer, pp 32–47
Page, L, Brin S, Motwani, R, Winograd T (1999) The PageRank citation ranking: bringing order to the web. Technical report, Stanford InfoLab
Pasquier C (2010) Task 5: single document keyphrase extraction using sentence clustering and latent dirichlet allocation. In: Proceedings of the 5th international workshop on semantic evaluation. ACL, pp 154–157
Patwardhan S, Banerjee S, Pedersen T (2005) SenseRelate::Targetword: a generalized framework for word sense disambiguation. In: Proceedings of the ACL 2005 on Interactive Poster and Demonstration Sessions, pp 73–76
Qiu M, Li, Y, Jiang J (2012) Query-oriented keyphrase extraction. In: Information retrieval technology, pp 64–75
Siddiqi S, Sharan A (2015) Keyword and keyphrase extraction techniques: a literature review. Int J Comput Appl 109(2):15
Sparck K (1972) A statistical interpretation of term specificity and its application in retrieval. J Doc 28(1):11–21
Sterckx L, Demeester T, Develder C, Caragea C (2016) Supervised keyphrase extraction as positive unlabeled learning. In: EMNLP2016, the conference on empirical methods in natural language processing, pp 1–6
Tomokiyo T, Hurst M (2003) A language model approach to keyphrase extraction. In: Proceedings of the ACL 2003 workshop on multiword expressions: analysis, acquisition and treatment. ACL, vol 18, pp 33–40
Turney P (2000) Learning algorithms for keyphrase extraction. Inf Retr 2(4):303–336
Tversky A (1977) Features of similarity. Psychol Rev 84(4):327–352
Wan X, Xiao J (2008) Single document keyphrase extraction using neighborhood knowledge. AAAI 8:855–860
Wan X, Yang J, Xiao J (2007) Towards an iterative reinforcement approach for simultaneous document summarization and keyword extraction. ACL
Wang J, Liu J, Wang C (2007) Keyword extraction based on pagerank. In: Advances in knowledge discovery and data mining. Springer, pp 857–864
Witten I, Milne D (2008) An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. In: Proceeding of AAAI workshop on wikipedia and artificial intelligence: an evolving synergy. AAAI Press, pp 25–30
Witten I, Paynter G, Frank E, Gutwin C, Nevill-Manning C (1999) KEA: practical automatic keyphrase extraction. In: Proceedings of the fourth ACM conference on digital libraries. ACM, pp 254–255
Wu Z, Palmer M (1994) Verbs semantics and lexical selection. In: Proceedings of the 32nd annual meeting on association for computational linguistics. ACL, pp 133–138
Xu Z, Wei X, Luo X, Liu Y, Mei L, Hu C, Chen L (2015) Knowle: a semantic link network based system for organizing large scale online news events. Future Gener Comput Syst 43:40–50
Yang S, Lu W, Yang D, Li X, Wu C, Wei B (2017) KeyphraseDS: automatic generation of survey by exploiting keyphrase information. Neurocomputing 224(2017):58–70
Yih T, Goodman J, Carvalho V (2006) Finding advertising keywords on web pages. In: Proceedings of the 15th international conference on World Wide Web. ACM, pp 213–222
Zhang C (2008) Automatic keyword extraction from documents using conditional random fields. J Comput Inf Syst 4(3):1169–1180
Zhang Q, Wang Y, Gong Y, Huang X (2016) Keyphrase Extraction using deep recurrent neural networks on Twitter. In: EMNLP, pp 836–845