Proceedings of the 52nd Annual Meeting of the Association for
Computational Linguistics (ACL), Baltimore, MD, June 2014, pp. 1262--1273.

Automatic Keyphrase Extraction: A Survey of the State of the Art

Kazi Saidul Hasan and **Vincent Ng**
Human Language Technology Research Institute
University of Texas at Dallas
Richardson, TX 75083-0688
{saidul,vince}@hlt.utdallas.edu

Abstract

While automatic keyphrase extraction has been examined extensively, state-of-the-art performance on this task is still much lower than that on many core natural language processing tasks. We present a survey of the state of the art in automatic keyphrase extraction, examining the major sources of errors made by existing systems and discussing the challenges ahead.

1 Introduction

Automatic keyphrase extraction concerns "the automatic selection of important and topical phrases from the body of a document" (Turney, 2000). In other words, its goal is to extract a set of phrases that are related to the main topics discussed in a given document (Tomokiyo and Hurst, 2003; Liu et al., 2009b; Ding et al., 2011; Zhao et al., 2011).

Document keyphrases have enabled fast and accurate searching for a given document from a large text collection, and have exhibited their potential in improving many natural language processing (NLP) and information retrieval (IR) tasks, such as text summarization (Zhang et al., 2004), text categorization (Hulth and Megyesi, 2006), opinion mining (Berend, 2011), and document indexing (Gutwin et al., 1999).

Owing to its importance, automatic keyphrase extraction has received a lot of attention. However, the task is far from being solved: state-of-the-art performance on keyphrase extraction is still much lower than that on many core NLP tasks (Liu et al., 2010). Our goal in this paper is to survey the state of the art in keyphrase extraction, examining the major sources of errors made by existing systems and discussing the challenges ahead.

2 Corpora

Automatic keyphrase extraction systems have been evaluated on corpora from a variety of sources ranging from long scientific publications to short paper abstracts and email messages. Table 1 presents a listing of the corpora grouped by their sources as well as their statistics.[1] There are at least four corpus-related factors that affect the difficulty of keyphrase extraction.

Length The difficulty of the task increases with the length of the input document as longer documents yield more candidate keyphrases (i.e., phrases that are eligible to be keyphrases (see Section 3.1)). For instance, each *Inspec* abstract has on average 10 annotator-assigned keyphrases and 34 candidate keyphrases. In contrast, a scientific paper typically has at least 10 keyphrases and hundreds of candidate keyphrases, yielding a much bigger search space (Hasan and Ng, 2010). Consequently, it is harder to extract keyphrases from scientific papers, technical reports, and meeting transcripts than abstracts, emails, and news articles.

Structural consistency In a structured document, there are certain locations where a keyphrase is most likely to appear. For instance, most of a scientific paper's keyphrases should appear in the abstract and the introduction. While structural information has been exploited to extract keyphrases from scientific papers (e.g., title, section information) (Kim et al., 2013), web pages (e.g., metadata) (Yih et al., 2006), and chats (e.g., dialogue acts) (Kim and Baldwin, 2012), it is most useful when the documents from a source exhibit structural similarity. For this reason, structural information is likely to facilitate keyphrase extraction from scientific papers and technical reports because of their standard format (i.e., standard sections such as abstract, introduction, conclusion, etc.). In contrast, the lack of structural consistency in other structured sources (e.g., a web page can be a blog, a forum, a review, etc.) may

[1]Many of the publicly available corpora can be found in http://github.com/snkim/AutomaticKeyphraseExtraction/ and http://code.google.com/p/maui-indexer/downloads/list.

Source	Dataset/Contributor	Statistics		
		Documents	Tokens/doc	Keys/doc
Paper abstracts	*Inspec* (Hulth, 2003)∗	2,000	$<$200	10
Scientific papers	NUS corpus (Nguyen and Kan, 2007)∗	211	$\approx$8K	11
	citeulike.org (Medelyan et al., 2009)∗	180	-	5
	SemEval-2010 (Kim et al., 2010b)∗	284	$>$5K	15
Technical reports	NZDL (Witten et al., 1999)∗	1,800	-	-
News articles	DUC-2001 (Wan and Xiao, 2008b)∗	308	$\approx$900	8
	Reuters corpus (Hulth and Megyesi, 2006)	12,848	-	6
Web pages	Yih et al. (2006)	828	-	-
	Hammouda et al. (2005)∗	312	$\approx$500	-
	Blogs (Grineva et al., 2009)	252	$\approx$1K	8
Meeting transcripts	ICSI (Liu et al., 2009a)	161	$\approx$1.6K	4
Emails	Enron corpus (Dredze et al., 2008)∗	14,659	-	-
Live chats	Library of Congress (Kim and Baldwin, 2012)	15	-	10

Table 1: Evaluation datasets. Publicly available datasets are marked with an asterisk (∗).

render structural information less useful.

Topic change An observation commonly exploited in keyphrase extraction from scientific articles and news articles is that keyphrases typically appear not only at the beginning (Witten et al., 1999) but also at the end (Medelyan et al., 2009) of a document. This observation does not necessarily hold for conversational sources (e.g., meetings, chats), however. The reason is simple: in a conversation, the topics (i.e., its talking points) change as the interaction moves forward in time, and so do the keyphrases associated with a topic. One way to address this complication is to detect a topic change in these sources (Kim and Baldwin, 2012). However, topic change detection is not always easy: while the topics listed in the form of an agenda at the beginning of formal meeting transcripts can be exploited, such clues are absent in casual conversations (e.g., chats).

Topic correlation Another observation commonly exploited in keyphrase extraction from scientific articles and news articles is that the keyphrases in a document are typically *related* to each other (Turney, 2003; Mihalcea and Tarau, 2004). However, this observation may not hold for informal text (e.g., emails, chats, informal meetings, personal blogs), where people can talk about any number of potentially uncorrelated topics. The presence of uncorrelated topics implies that it may no longer be possible to exploit relatedness and therefore increases the difficulty of keyphrase extraction.

3 Keyphrase Extraction Approaches

A keyphrase extraction system typically operates in two steps: (1) extracting a list of words/phrases that serve as *candidate keyphrases* using some heuristics (Section 3.1); and (2) determining which of these candidate keyphrases are correct keyphrases using supervised (Section 3.2) or unsupervised (Section 3.3) approaches.

3.1 Selecting Candidate Words/Phrases

As noted before, a set of phrases and words is typically extracted as candidate keyphrases using heuristic rules. These rules are designed to avoid spurious instances and keep the number of candidates to a minimum. Typical heuristics include (1) using a stop word list to remove stop words (Liu et al., 2009b), (2) allowing words with certain part-of-speech tags (e.g., nouns, adjectives, verbs) to be candidate keywords (Mihalcea and Tarau, 2004; Wan and Xiao, 2008b; Liu et al., 2009a), (3) allowing n-grams that appear in Wikipedia article titles to be candidates (Grineva et al., 2009), and (4) extracting n-grams (Witten et al., 1999; Hulth, 2003; Medelyan et al., 2009) or noun phrases (Barker and Cornacchia, 2000; Wu et al., 2005) that satisfy pre-defined lexico-syntactic pattern(s) (Nguyen and Phan, 2009).

Many of these heuristics have proven effective with their high recall in extracting gold keyphrases from various sources. However, for a long document, the resulting list of candidates can be long. Consequently, different *pruning* heuristics have been designed to prune candidates that are unlikely to be keyphrases (Huang et al., 2006; Kumar and Srinathan, 2008; El-Beltagy and Rafea, 2009; You et al., 2009; Newman et al., 2012).

3.2 Supervised Approaches

Research on supervised approaches to keyphrase extraction has focused on two issues: *task reformulation* and *feature design*.

3.2.1 Task Reformulation

Early supervised approaches to keyphrase extraction recast this task as a binary *classification* problem (Frank et al., 1999; Turney, 1999; Witten et al., 1999; Turney, 2000). The goal is to train a classifier on documents annotated with keyphrases to determine whether a candidate keyphrase is a keyphrase. Keyphrases and non-keyphrases are used to generate positive and negative examples, respectively. Different learning algorithms have been used to train this classifier, including naïve Bayes (Frank et al., 1999; Witten et al., 1999), decision trees (Turney, 1999; Turney, 2000), bagging (Hulth, 2003), boosting (Hulth et al., 2001), maximum entropy (Yih et al., 2006; Kim and Kan, 2009), multi-layer perceptron (Lopez and Romary, 2010), and support vector machines (Jiang et al., 2009; Lopez and Romary, 2010).

Recasting keyphrase extraction as a classification problem has its weaknesses, however. Recall that the goal of keyphrase extraction is to identify the most representative phrases for a document. In other words, if a candidate keyphrase c_1 is more representative than another candidate c_2, c_1 should be preferred to c_2. Note that a binary classifier classifies each candidate keyphrase independently of the others, and consequently it does not allow us to determine which candidates are better than the others (Hulth, 2004; Wang and Li, 2011).

Motivated by this observation, Jiang et al. (2009) propose a *ranking* approach to keyphrase extraction, where the goal is to learn a ranker to rank two candidate keyphrases. This pairwise ranking approach therefore introduces competition between candidate keyphrases, and has been shown to significantly outperform KEA (Witten et al., 1999; Frank et al., 1999), a popular supervised baseline that adopts the traditional supervised classification approach (Song et al., 2003; Kelleher and Luz, 2005).

3.2.2 Features

The features commonly used to represent an instance for supervised keyphrase extraction can be broadly divided into two categories.

3.2.2.1 Within-Collection Features

Within-collection features are computed based solely on the training documents. These features can be further divided into three types.

Statistical features are computed based on statistical information gathered from the training documents. Three such features have been extensively used in supervised approaches. The first one, *tf*idf* (Salton and Buckley, 1988), is computed based on candidate frequency in the given text and inverse document frequency (i.e., number of other documents where the candidate appears).[2] The second one, the *distance* of a phrase, is defined as the number of words preceding its first occurrence normalized by the number of words in the document. Its usefulness stems from the fact that keyphrases tend to appear early in a document. The third one, *supervised keyphraseness*, encodes the number of times a phrase appears as a keyphrase in the training set. This feature is designed based on the assumption that a phrase frequently tagged as a keyphrase is more likely to be a keyphrase in an unseen document. These three features form the feature set of KEA (Witten et al., 1999; Frank et al., 1999), and have been shown to perform consistently well on documents from various sources (Yih et al., 2006; Kim et al., 2013). Other statistical features include *phrase length* and *spread* (i.e., the number of words between the first and last occurrences of a phrase in the document).

Structural features encode how different instances of a candidate keyphrase are located in different parts of a document. A phrase is more likely to be a keyphrase if it appears in the abstract or introduction of a paper or in the metadata section of a web page. In fact, features that encode how frequently a candidate keyphrase occurs in various sections of a scientific paper (e.g., introduction, conclusion) (Nguyen and Kan, 2007) and those that encode the location of a candidate keyphrase in a web page (e.g., whether it appears in the title) (Chen et al., 2005; Yih et al., 2006) have been shown to be useful for the task.

Syntactic features encode the syntactic patterns of a candidate keyphrase. For example, a candidate keyphrase has been encoded as (1) a *PoS tag sequence*, which denotes the sequence of part-of-speech tag(s) assigned to its word(s); and (2) a *suffix sequence*, which is the sequence of morphological suffixes of its words (Yih et al., 2006; Nguyen and Kan, 2007; Kim and Kan, 2009). However, ablation studies conducted on web pages (Yih et al., 2006) and scientific articles

[2] A tf*idf-based baseline, where candidate keyphrases are ranked and selected according to tf*idf, has been widely used by both supervised and unsupervised approaches (Zhang et al., 2005; Paukkeri et al., 2008; Dredze et al., 2008; Grineva et al., 2009).

(Kim and Kan, 2009) reveal that syntactic features are not useful for keyphrase extraction in the presence of other feature types.

3.2.2.2 External Resource-Based Features

External resource-based features are computed based on information gathered from resources other than the training documents, such as lexical knowledge bases (e.g., Wikipedia) or the Web, with the goal of improving keyphrase extraction performance by exploiting external knowledge. Below we give an overview of the external resource-based features that have proven useful for keyphrase extraction.

Wikipedia-based keyphraseness is computed as a candidate's document frequency multiplied by the ratio of the number of Wikipedia articles where the candidate appears as a link to the number of articles where it appears (Medelyan et al., 2009). This feature is motivated by the observation that a candidate is likely to be a keyphrase if it occurs frequently as a link in Wikipedia. Unlike supervised keyphraseness, Wikipedia-based keyphraseness can be computed without using documents annotated with keyphrases and can work even if there is a mismatch between the training domain and the test domain.

Yih et al. (2006) employ a feature that encodes whether a candidate keyphrase appears in the *query log* of a search engine, exploiting the observation that a candidate is potentially important if it was used as a search query. Terminological databases have been similarly exploited to encode the salience of candidate keyphrases in scientific papers (Lopez and Romary, 2010).

While the aforementioned external resource-based features attempt to encode how salient a candidate keyphrase is, Turney (2003) proposes features that encode the semantic relatedness between two candidate keyphrases. Noting that candidate keyphrases that are not semantically related to the predicted keyphrases are unlikely to be keyphrases in technical reports, Turney employs *coherence features* to identify such candidate keyphrases. Semantic relatedness is encoded in the coherence features as two candidate keyphrases' pointwise mutual information, which Turney computes by using the Web as a corpus.

3.3 Unsupervised Approaches

Existing unsupervised approaches to keyphrase extraction can be categorized into four groups.

3.3.1 Graph-Based Ranking

Intuitively, keyphrase extraction is about finding the important words and phrases from a document. Traditionally, the *importance* of a candidate has often been defined in terms of how related it is to other candidates in the document. Informally, a candidate is important if it is related to (1) a large number of candidates and (2) candidates that are important. Researchers have computed *relatedness* between candidates using co-occurrence counts (Mihalcea and Tarau, 2004; Matsuo and Ishizuka, 2004) and semantic relatedness (Grineva et al., 2009), and represented the relatedness information collected from a document as a graph (Mihalcea and Tarau, 2004; Wan and Xiao, 2008a; Wan and Xiao, 2008b; Bougouin et al., 2013).

The basic idea behind a graph-based approach is to build a graph from the input document and rank its nodes according to their importance using a graph-based ranking method (e.g., Brin and Page (1998)). Each node of the graph corresponds to a candidate keyphrase from the document and an edge connects two *related* candidates. The edge weight is proportional to the syntactic and/or semantic relevance between the connected candidates. For each node, each of its edges is treated as a "vote" from the other node connected by the edge. A node's score in the graph is defined recursively in terms of the edges it has and the scores of the neighboring nodes. The top-ranked candidates from the graph are then selected as keyphrases for the input document. TextRank (Mihalcea and Tarau, 2004) is one of the most well-known graph-based approaches to keyphrase extraction.

This instantiation of a graph-based approach overlooks an important aspect of keyphrase extraction, however. A set of keyphrases for a document should ideally cover the main topics discussed in it, but this instantiation does not guarantee that all the main topics will be represented by the extracted keyphrases. Despite this weakness, a graph-based representation of text was adopted by many approaches that propose different ways of computing the similarity between two candidates.

3.3.2 Topic-Based Clustering

Another unsupervised approach to keyphrase extraction involves grouping the candidate keyphrases in a document into *topics*, such that each topic is composed of all and only those candidate keyphrases that are related to that topic (Grineva et al., 2009; Liu et al., 2009b; Liu et al., 2010). There are several motivations behind this topic-based clustering approach. First, a keyphrase should ideally be relevant to one or more main topic(s) discussed in a document (Liu et al., 2010; Liu et al., 2012). Second, the extracted keyphrases should be comprehensive in the sense that they should cover all the main topics in a document (Liu et al., 2009b; Liu et al., 2010; Liu et al., 2012). Below we examine three representative systems that adopt this approach.

KeyCluster Liu et al. (2009b) adopt a clustering-based approach (henceforth KeyCluster) that clusters semantically similar candidates using Wikipedia and co-occurrence-based statistics. The underlying hypothesis is that each of these clusters corresponds to a topic covered in the document, and selecting the candidates close to the centroid of each cluster as keyphrases ensures that the resulting set of keyphrases covers all the topics of the document.

While empirical results show that KeyCluster performs better than both TextRank and Hulth's (2003) supervised system, KeyCluster has a potential drawback: by extracting keyphrases from each topic cluster, it essentially gives each topic equal importance. In practice, however, there could be topics that are not important and these topics should not have keyphrase(s) representing them.

Topical PageRank (TPR) Liu et al. (2010) propose TPR, an approach that overcomes the aforementioned weakness of KeyCluster. It runs TextRank multiple times for a document, once for each of its topics induced by a Latent Dirichlet Allocation (Blei et al., 2003). By running TextRank once for each topic, TPR ensures that the extracted keyphrases cover the main topics of the document. The final score of a candidate is computed as the sum of its scores for each of the topics, weighted by the probability of that topic in that document. Hence, unlike KeyCluster, candidates belonging to a less probable topic are given less importance.

TPR performs significantly better than both tf*idf and TextRank on the DUC-2001 and *Inspec* datasets. TPR's superior performance strengthens the hypothesis of using topic clustering for keyphrase extraction. However, though TPR is conceptually better than KeyCluster, Liu et al. did not compare TPR against KeyCluster.

CommunityCluster Grineva et al. (2009) propose CommunityCluster, a variant of the topic clustering approach to keyphrase extraction. Like TPR, CommunityCluster gives more weight to more important topics, but unlike TPR, it extracts *all* candidate keyphrases from an important topic, assuming that a candidate that receives little focus in the text should still be extracted as a keyphrase as long as it is related to an important topic. CommunityCluster yields much better recall (without losing precision) than extractors such as tf*idf, TextRank, and the Yahoo! term extractor.

3.3.3 Simultaneous Learning

Since keyphrases represent a dense summary of a document, researchers hypothesized that text summarization and keyphrase extraction can potentially benefit from each other if these tasks are performed simultaneously. Zha (2002) proposes the first graph-based approach for simultaneous summarization and keyphrase extraction, motivated by a key observation: a sentence is important if it contains important words, and important words appear in important sentences. Wan et al. (2007) extend Zha's work by adding two assumptions: (1) an important sentence is connected to other important sentences, and (2) an important word is linked to other important words, a TextRank-like assumption. Based on these assumptions, Wan et al. (2007) build three graphs to capture the association between the sentences (S) and the words (W) in an input document, namely, a S–S graph, a bipartite S–W graph, and a W–W graph. The weight of an edge connecting two sentence nodes in a S–S graph corresponds to their content similarity. An edge weight in a S–W graph denotes the word's importance in the sentence it appears. Finally, an edge weight in a W–W graph denotes the co-occurrence or knowledge-based similarity between the two connected words. Once the graphs are constructed for an input document, an iterative reinforcement algorithm is applied to assign scores to each sentence and word. The top-scored words are used to form keyphrases.

The main advantage of this approach is that it combines the strengths of both Zha's approach (i.e., bipartite S–W graphs) and TextRank (i.e., W–W graphs) and performs better than both of them. However, this approach has a weakness: like TextRank, it does not ensure that the extracted keyphrases will cover all the main topics. To address this problem, one can employ a topic clustering algorithm on the W–W graph to produce the topic clusters, and then ensure that keyphrases are chosen from every main topic cluster.

3.3.4 Language Modeling

Many existing approaches have a separate, heuristic module for extracting candidate keyphrases prior to keyphrase ranking/extraction. In contrast, Tomokiyo and Hurst (2003) propose an approach (henceforth LMA) that combines these two steps.

LMA scores a candidate keyphrase based on two features, namely, *phraseness* (i.e., the extent to which a word sequence can be treated as a phrase) and *informativeness* (i.e., the extent to which a word sequence captures the central idea of the document it appears in). Intuitively, a phrase that has high scores for phraseness and informativeness is likely to be a keyphrase. These feature values are estimated using language models (LMs) trained on a *foreground* corpus and a *background* corpus. The foreground corpus is composed of the set of documents from which keyphrases are to be extracted. The background corpus is a large corpus that encodes general knowledge about the world (e.g., the Web). A unigram LM and an n-gram LM are constructed for each of these two corpora. Phraseness, defined using the foreground LM, is calculated as the loss of information incurred as a result of assuming a unigram LM (i.e., conditional independence among the words of the phrase) instead of an n-gram LM (i.e., the phrase is drawn from an n-gram LM). Informativeness is computed as the loss that results because of the assumption that the candidate is sampled from the background LM rather than the foreground LM. The loss values are computed using Kullback-Leibler divergence. Candidates are ranked according to the sum of these two feature values.

In sum, LMA uses a language model rather than heuristics to identify phrases, and relies on the language model trained on the background corpus to determine how "unique" a candidate keyphrase is to the domain represented by the foreground corpus. The more unique it is to the foreground's domain, the more likely it is a keyphrase for that domain. While the use of language models to identify phrases cannot be considered a major strength of this approach (because heuristics can identify phrases fairly reliably), the use of a background corpus to identify candidates that are unique to the foreground's domain is a unique aspect of this approach. We believe that this idea deserves further investigation, as it would allow us to discover a keyphrase that is unique to the foreground's domain but may have a low tf*idf value.

4 Evaluation

In this section, we describe metrics for evaluating keyphrase extraction systems as well as state-of-the-art results on commonly-used datasets.

4.1 Evaluation Metrics

Designing evaluation metrics for keyphrase extraction is by no means an easy task. To score the output of a keyphrase extraction system, the typical approach, which is also adopted by the SemEval-2010 shared task on keyphrase extraction, is (1) to create a mapping between the keyphrases in the gold standard and those in the system output using *exact match*, and then (2) score the output using evaluation metrics such as precision (P), recall (R), and F-score (F).

Conceivably, exact match is an overly strict condition, considering a predicted keyphrase incorrect even if it is a variant of a gold keyphrase. For instance, given the gold keyphrase "neural network", exact match will consider a predicted phrase incorrect even if it is an expanded version of the gold keyphrase ("artificial neural network") or one of its morphological ("neural networks") or lexical ("neural net") variants. While morphological variations can be handled using a stemmer (El-Beltagy and Rafea, 2009), other variations may not be handled easily and reliably.

Human evaluation has been suggested as a possibility (Matsuo and Ishizuka, 2004), but it is time-consuming and expensive. For this reason, researchers have experimented with two types of automatic evaluation metrics. The first type of metrics addresses the problem with exact match. These metrics reward a partial match between a predicted keyphrase and a gold keyphrase (i.e., overlapping n-grams) and are commonly used in machine translation (MT) and summarization evaluations. They include BLEU, METEOR, NIST, and ROUGE. Nevertheless, experiments show that these MT metrics only offer a partial solution to problem with exact match: they can only detect a subset of the near-misses (Kim et al., 2010a).

The second type of metrics focuses on how a system ranks its predictions. Given that two systems A and B have the same number of correct predictions, binary preference measure (Bpref) and mean reciprocal rank (MRR) (Liu et al., 2010) will award more credit to A than to B if the ranks of the correct predictions in A's output are higher than those in B's output. R-precision (R_p) is an

IR metric that focuses on ranking: given a document with n gold keyphrases, it computes the precision of a system over its n highest-ranked candidates (Zesch and Gurevych, 2009). The motivation is simple: a system will achieve a perfect R_p value if it ranks all the keyphrases above the non-keyphrases.

4.2 The State of the Art

Table 2 lists the best scores on some popular evaluation datasets and the corresponding systems. For example, the best F-scores on the *Inspec* test set, the DUC-2001 dataset, and the SemEval-2010 test set are 45.7, 31.7, and 27.5, respectively.[3]

Two points deserve mention. First, F-scores decrease as document length increases. These results are consistent with the observation we made in Section 2 that it is more difficult to extract keyphrases correctly from longer documents.

Second, recent unsupervised approaches have rivaled their supervised counterparts in performance (Mihalcea and Tarau, 2004; El-Beltagy and Rafea, 2009; Liu et al., 2009b). For example, KP-Miner (El-Beltagy and Rafea, 2010), an unsupervised system, ranked third at SemEval-2010 shared task with an F-score of 25.2, which is comparable to the best supervised system scoring 27.5.

Dataset	Approach and System [Supervised?]	Score		
		P	R	F
Abstracts (*Inspec*)	Topic clustering (Liu et al., 2009b) [×]	35.0	66.0	45.7
Blogs	Topic community detection (Grineva et al., 2009) [×]	35.1	61.5	44.7
News (DUC -2001)	Graph-based ranking for extended neighborhood (Wan and Xiao, 2008b) [×]	28.8	35.4	31.7
Papers (SemEval -2010)	Statistical, semantic, and distributional features (Lopez and Romary, 2010) [✓]	27.2	27.8	27.5

Table 2: Best scores achieved on various datasets.

5 Analysis

With the goal of providing directions for future work, we identify the errors commonly made by state-of-the-art keyphrase extractors below.

5.1 Error Analysis

Although a few researchers have presented a sample of their systems' output and the corresponding gold keyphrases to show the differences between them (Witten et al., 1999; Nguyen and Kan, 2007; Medelyan et al., 2009), a systematic analysis of the major types of errors made by state-of-the-art keyphrase extraction systems is missing.

To fill this gap, we ran four keyphrase extraction systems on four commonly-used datasets of varying sources, including *Inspec* abstracts (Hulth, 2003), DUC-2001 news articles (Over, 2001), scientific papers (Kim et al., 2010b), and meeting transcripts (Liu et al., 2009a). Specifically, we randomly selected 25 documents from each of these four datasets and manually analyzed the output of the four systems, including tf*idf, the most frequently used baseline, as well as three state-of-the-art keyphrase extractors, of which two are unsupervised (Wan and Xiao, 2008b; Liu et al., 2009b) and one is supervised (Medelyan et al., 2009).

Our analysis reveals that the errors fall into four major types, each of which contributes significantly to the overall errors made by the four systems, despite the fact that the contribution of each of these error types varies from system to system. Moreover, we do not observe any significant difference between the types of errors made by the four systems other than the fact that the supervised system has the expected tendency to predict keyphrases seen in the training data. Below we describe these four major types of errors.

Overgeneration errors are a major type of precision error, contributing to 28–37% of the overall error. Overgeneration errors occur when a system correctly predicts a candidate as a keyphrase because it contains a word that appears frequently in the associated document, but at the same time erroneously outputs other candidates as keyphrases because they contain the same word. Recall that for many systems, it is not easy to reject a non-keyphrase containing a word with a high term frequency: many unsupervised systems score a candidate by summing the score of each of its component words, and many supervised systems use unigrams as features to represent a candidate. To be more concrete, consider the news article on athlete *Ben Johnson* in Figure 1, where the keyphrases are boldfaced. As we can see, the word *Olympic(s)* has a significant presence in the document. Consequently, many systems not only correctly predict *Olympics* as a keyphrase, but also erroneously predict *Olympic movement* as a keyphrase, yielding overgeneration errors.

Infrequency errors are a major type of re-

[3] A more detailed analysis of the results of the SemEval-2010 shared task and the approaches adopted by the participating systems can be found in Kim et al. (2013).

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.

Figure 1: A news article on *Ben Johnson* from the DUC-2001 dataset. The keyphrases are boldfaced.

call error contributing to 24–27% of the overall error. Infrequency errors occur when a system fails to identify a keyphrase owing to its infrequent presence in the associated document (Liu et al., 2011). Handling infrequency errors is a challenge because state-of-the-art keyphrase extractors rarely predict candidates that appear only once or twice in a document. In the *Ben Johnson* example, many keyphrase extractors fail to identify *100-meter dash* and *gold medal* as keyphrases, resulting in infrequency errors.

Redundancy errors are a type of precision error contributing to 8–12% of the overall error. Redundancy errors occur when a system correctly identifies a candidate as a keyphrase, but at the same time outputs a semantically equivalent candidate (e.g., its alias) as a keyphrase. This type of error can be attributed to a system's failure to determine that two candidates are semantically equivalent. Nevertheless, some researchers may argue that a system should not be penalized for redundancy errors because the extracted candidates are in fact keyphrases. In our example, *Olympics* and *Olympic games* refer to the same concept, so a system that predicts both of them as keyphrases commits a redundancy error.

Evaluation errors are a type of recall error contributing to 7–10% of the overall error. An evaluation error occurs when a system outputs a candidate that is semantically equivalent to a gold keyphrase, but is considered erroneous by a scoring program because of its failure to recognize that the predicted phrase and the corresponding gold keyphrase are semantically equivalent. In other words, an evaluation error is not an error made by a keyphrase extractor, but an error due to the naivety of a scoring program. In our example, while *Olympics* and *Olympic games* refer to the same concept, only the former is annotated as keyphrase. Hence, an evaluation error occurs if a system predicts *Olympic games* but not *Olympics* as a keyphrase and the scoring program fails to identify them as semantically equivalent.

5.2 Recommendations

We recommend that *background knowledge* be extracted from external lexical databases (e.g., YAGO2 (Suchanek et al., 2007), Freebase (Bollacker et al., 2008), BabelNet (Navigli and Ponzetto, 2012)) to address the four types of errors discussed above.

First, we discuss how **redundancy errors** could be addressed by using the background knowledge extracted from external databases. Note that if we can identify semantically equivalent candidates, then we can reduce redundancy errors. The question, then, is: can background knowledge be used to help us identify semantically equivalent candidates? To answer this question, recall that Freebase, for instance, has over 40 million *topics* (i.e., real-world entities such as people, places, and things) from over 70 domains (e.g., music, business, education). Hence, before a system outputs a set of candidates as keyphrases, it can use Freebase to determine whether any of them is mapped to the same Freebase topic. Referring back to our running example, both *Olympics* and *Olympic games* are mapped to a Freebase topic called *Olympic games*. Based on this information, a keyphrase extractor can determine that the two candidates are aliases and should output only one of them, thus preventing a redundancy error.

Next, we discuss how **infrequency errors** could be addressed using background knowledge. A natural way to handle this problem would be to make an infrequent keyphrase frequent. To accomplish this, we suggest exploiting an influential idea in the keyphrase extraction literature: the importance of a candidate is defined in terms of how related it is to other candidates in the text (see Section 3.3.1). In other words, if we could relate an infrequent keyphrase to other candidates in the text, we could boost its importance.

We believe that this could be accomplished using background knowledge. The idea is to boost the importance of infrequent keyphrases using their frequent counterparts. Consider again our running example. All four systems have managed to identify *Ben Johnson* as a keyphrase due to its significant presence. Hence, we can boost the importance of *100-meter dash* and *gold medal* if we can relate them to *Ben Johnson*.

To do so, recall that Freebase maps a candidate to one or more pre-defined topics, each of which is associated with one or more types. Types are similar to entity classes. For instance, the candidate *Ben Johnson* is mapped to a Freebase topic with the same name, which is associated with Freebase types such as *Person*, *Athlete*, and *Olympic athlete*. Types are defined for a specific domain in Freebase. For instance, *Person*, *Athlete*, and *Olympic athlete* are defined in the *People*, *Sports*, and *Olympics* domains, respectively. Next, consider the two infrequent candidates, *100-meter dash* and *gold medal*. *100-meter dash* is mapped to the topic *Sprint* of type *Sports* in the *Sports* domain, whereas *gold medal* is mapped to a topic with the same name of type *Olympic medal* in the *Olympics* domain. Consequently, we can relate *100-meter dash* to *Ben Johnson* via the *Sports* domain (i.e., they belong to different types under the same domain). Additionally, *gold medal* can be related to *Ben Johnson* via the *Olympics* domain.

As discussed before, the relationship between two candidates is traditionally established using co-occurrence information. However, using co-occurrence windows has its shortcomings. First, an *ad-hoc* window size cannot capture related candidates that are not inside the window. So it is difficult to predict *100-meter dash* and *gold medal* as keyphrases: they are more than 10 tokens away from frequent words such as *Johnson* and *Olympics*. Second, the candidates inside a window are all assumed to be related to each other, but it is apparently an overly simplistic assumption. There have been a few attempts to design Wikipedia-based relatedness measures, with promising initial results (Grineva et al., 2009; Liu et al., 2009b; Medelyan et al., 2009).[4]

Overgeneration errors could similarly be addressed using background knowledge. Recall that *Olympic movement* is not a keyphrase in our example although it includes an important word (i.e., *Olympic*). Freebase maps *Olympic movement* to a topic with the same name, which is associated with a type called *Musical Recording* in the *Music* domain. However, it does not map *Olympic movement* to any topic in the *Olympics* domain. The absence of such a mapping in the *Olympics* domain could be used by a keyphrase extractor as a supporting evidence against predicting *Olympic movement* as a keyphrase.

Finally, as mentioned before, **evaluation errors** should not be considered errors made by a system. Nevertheless, they reveal a problem with the way keyphrase extractors are currently evaluated. To address this problem, one possibility is to conduct human evaluations. Cheaper alternatives include having human annotators identify semantically equivalent keyphrases during manual labeling, and designing scoring programs that can automatically identify such semantic equivalences.

6 Conclusion and Future Directions

We have presented a survey of the state of the art in automatic keyphrase extraction. While unsupervised approaches have started to rival their supervised counterparts in performance, the task is far from being solved, as reflected by the fairly poor state-of-the-art results on various commonly-used evaluation datasets. Our analysis revealed that there are at least three major challenges ahead.

1. Incorporating background knowledge. While much recent work has focused on algorithmic development, keyphrase extractors need to have a deeper "understanding" of a document in order to reach the next level of performance. Such an understanding can be facilitated by the incorporation of background knowledge.

2. Handling long documents. While it may be possible to design better algorithms to handle the large number of candidates in long documents, we believe that employing sophisticated features, especially those that encode background knowledge, will enable keyphrases and non-keyphrases to be distinguished more easily even in the presence of a large number of candidates.

3. Improving evaluation schemes. To more accurately measure the performance of keyphrase extractors, they should not be penalized for evaluation errors. We have suggested several possibilities as to how this problem can be addressed.

Acknowledgments

We thank the anonymous reviewers for their detailed and insightful comments on earlier drafts of this paper. This work was supported in part by NSF Grants IIS-1147644 and IIS-1219142.

[4]Note that it may be difficult to employ our recommendations to address infrequency errors on informal text with uncorrelated topics because the keyphrases it contains may not be related to each other (see Section 2).

References

Ken Barker and Nadia Cornacchia. 2000. Using noun phrase heads to extract document keyphrases. In *Proceedings of the 13th Biennial Conference of the Canadian Society on Computational Studies of Intelligence*, pages 40–52.

Gábor Berend. 2011. Opinion expression mining by exploiting keyphrase extraction. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 1162–1170.

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data*, pages 1247–1250.

Adrien Bougouin, Florian Boudin, and Béatrice Daille. 2013. Topicrank: Graph-based topic ranking for keyphrase extraction. In *Proceedings of the Sixth International Joint Conference on Natural Language Processing*, pages 543–551, Nagoya, Japan, October. Asian Federation of Natural Language Processing.

Sergey Brin and Lawrence Page. 1998. The anatomy of a large-scale hypertextual Web search engine. *Computer Networks*, 30(1–7):107–117.

Mo Chen, Jian-Tao Sun, Hua-Jun Zeng, and Kwok-Yan Lam. 2005. A practical system of keyphrase extraction for web pages. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, pages 277–278.

Zhuoye Ding, Qi Zhang, and Xuanjing Huang. 2011. Keyphrase extraction from online news using binary integer programming. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 165–173.

Mark Dredze, Hanna M. Wallach, Danny Puller, and Fernando Pereira. 2008. Generating summary keywords for emails using topics. In *Proceedings of the 13th International Conference on Intelligent User Interfaces*, pages 199–206.

Samhaa R. El-Beltagy and Ahmed A. Rafea. 2009. KP-Miner: A keyphrase extraction system for English and Arabic documents. *Information Systems*, 34(1):132–144.

Samhaa R. El-Beltagy and Ahmed Rafea. 2010. KP-Miner: Participation in SemEval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 190–193.

Eibe Frank, Gordon W. Paynter, Ian H. Witten, Carl Gutwin, and Craig G. Nevill-Manning. 1999. Domain-specific keyphrase extraction. In *Proceedings of 16th International Joint Conference on Artificial Intelligence*, pages 668–673.

Maria Grineva, Maxim Grinev, and Dmitry Lizorkin. 2009. Extracting key terms from noisy and multi-theme documents. In *Proceedings of the 18th International Conference on World Wide Web*, pages 661–670.

Carl Gutwin, Gordon Paynter, Ian Witten, Craig Nevill-Manning, and Eibe Frank. 1999. Improving browsing in digital libraries with keyphrase indexes. *Decision Support Systems*, 27:81–104.

Khaled M. Hammouda, Diego N. Matute, and Mohamed S. Kamel. 2005. CorePhrase: Keyphrase extraction for document clustering. In *Proceedings of the 4th International Conference on Machine Learning and Data Mining in Pattern Recognition*, pages 265–274.

Kazi Saidul Hasan and Vincent Ng. 2010. Conundrums in unsupervised keyphrase extraction: Making sense of the state-of-the-art. In *Proceedings of the 23rd International Joint Conference on Computational Linguistics: Posters*, pages 365–373.

Chong Huang, Yonghong Tian, Zhi Zhou, Charles X. Ling, and Tiejun Huang. 2006. Keyphrase extraction using semantic networks structure analysis. In *Proceedings of the Sixth International Conference on Data Mining*, pages 275–284.

Anette Hulth and Beáta B. Megyesi. 2006. A study on automatically extracted keywords in text categorization. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pages 537–544.

Anette Hulth, Jussi Karlgren, Anna Jonsson, Henrik Boström, and Lars Asker. 2001. Automatic keyword extraction using domain knowledge. In *Proceedings of the Second International Conference on Computational Linguistics and Intelligent Text Processing*, pages 472–482.

Anette Hulth. 2003. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing*, pages 216–223.

Anette Hulth. 2004. Enhancing linguistically oriented automatic keyword extraction. In *Proceedings of the HLT-NAACL 2004 Conference: Short Papers*, pages 17–20.

Xin Jiang, Yunhua Hu, and Hang Li. 2009. A ranking approach to keyphrase extraction. In *Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 756–757.

Daniel Kelleher and Saturnino Luz. 2005. Automatic hypertext keyphrase detection. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence*, pages 1608–1609.

Su Nam Kim and Timothy Baldwin. 2012. Extracting keywords from multi-party live chats. In *Proceedings of the 26th Pacific Asia Conference on Language, Information, and Computation*, pages 199–208.

Su Nam Kim and Min-Yen Kan. 2009. Re-examining automatic keyphrase extraction approaches in scientific articles. In *Proceedings of the ACL-IJCNLP Workshop on Multiword Expressions*, pages 9–16.

Su Nam Kim, Timothy Baldwin, and Min-Yen Kan. 2010a. Evaluating n-gram based evaluation metrics for automatic keyphrase extraction. In *Proceedings of the 23rd International Conference on Computational Linguistics*, pages 572–580.

Su Nam Kim, Olena Medelyan, Min-Yen Kan, and Timothy Baldwin. 2010b. SemEval-2010 Task 5: Automatic keyphrase extraction from scientific articles. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 21–26.

Su Nam Kim, Olena Medelyan, Min-Yen Kan, and Timothy Baldwin. 2013. Automatic keyphrase extraction from scientific articles. *Language Resources and Evaluation*, 47(3):723–742.

Niraj Kumar and Kannan Srinathan. 2008. Automatic keyphrase extraction from scientific documents using n-gram filtration technique. In *Proceedings of the Eighth ACM Symposium on Document Engineering*, pages 199–208.

Feifan Liu, Deana Pennell, Fei Liu, and Yang Liu. 2009a. Unsupervised approaches for automatic keyword extraction using meeting transcripts. In *Proceedings of Human Language Technologies: The Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 620–628.

Zhiyuan Liu, Peng Li, Yabin Zheng, and Maosong Sun. 2009b. Clustering to find exemplar terms for keyphrase extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 257–266.

Zhiyuan Liu, Wenyi Huang, Yabin Zheng, and Maosong Sun. 2010. Automatic keyphrase extraction via topic decomposition. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 366–376.

Zhiyuan Liu, Xinxiong Chen, Yabin Zheng, and Maosong Sun. 2011. Automatic keyphrase extraction by bridging vocabulary gap. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning*, pages 135–144.

Zhiyuan Liu, Chen Liang, and Maosong Sun. 2012. Topical word trigger model for keyphrase extraction. In *Proceedings of the 24th International Conference on Computational Linguistics*, pages 1715–1730.

Patrice Lopez and Laurent Romary. 2010. HUMB: Automatic key term extraction from scientific articles in GROBID. In *Proceedings of the 5th International Workshop on Semantic Evaluation*, pages 248–251.

Yutaka Matsuo and Mitsuru Ishizuka. 2004. Keyword extraction from a single document using word co-occurrence statistical information. *International Journal on Artificial Intelligence Tools*, 13.

Olena Medelyan, Eibe Frank, and Ian H. Witten. 2009. Human-competitive tagging using automatic keyphrase extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, pages 1318–1327.

Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing order into texts. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, pages 404–411.

Roberto Navigli and Simone Paolo Ponzetto. 2012. BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network. *Artificial Intelligence*, 193:217–250.

David Newman, Nagendra Koilada, Jey Han Lau, and Timothy Baldwin. 2012. Bayesian text segmentation for index term identification and keyphrase extraction. In *Proceedings of the 24th International Conference on Computational Linguistics*, pages 2077–2092.

Thuy Dung Nguyen and Min-Yen Kan. 2007. Keyphrase extraction in scientific publications. In *Proceedings of the International Conference on Asian Digital Libraries*, pages 317–326.

Chau Q. Nguyen and Tuoi T. Phan. 2009. An ontology-based approach for key phrase extraction. In *Proceedings of the ACL-IJCNLP 2009 Conference: Short Papers*, pages 181–184.

Paul Over. 2001. Introduction to DUC-2001: An intrinsic evaluation of generic news text summarization systems. In *Proceedings of the 2011 Document Understanding Conference*.

Mari-Sanna Paukkeri, Ilari T. Nieminen, Matti Pöllä, and Timo Honkela. 2008. A language-independent approach to keyphrase extraction and evaluation. In *Proceedings of the 22nd International Conference on Computational Linguistics: Companion Volume: Posters*, pages 83–86.

Gerard Salton and Christopher Buckley. 1988. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523.

Min Song, Il-Yeol Song, and Xiaohua Hu. 2003. KPSpotter: A flexible information gain-based keyphrase extraction system. In *Proceedings of the 5th ACM International Workshop on Web Information and Data Management*, pages 50–53.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. YAGO: A core of semantic knowledge. In *Proceedings of the 16th International World Wide Web Conference*, pages 697–706.

Takashi Tomokiyo and Matthew Hurst. 2003. A language model approach to keyphrase extraction. In *Proceedings of the ACL Workshop on Multiword Expressions*, pages 33–40.

Peter Turney. 1999. Learning to extract keyphrases from text. *National Research Council Canada, Institute for Information Technology, Technical Report ERB-1057*.

Peter Turney. 2000. Learning algorithms for keyphrase extraction. *Information Retrieval*, 2:303–336.

Peter Turney. 2003. Coherent keyphrase extraction via web mining. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, pages 434–439.

Xiaojun Wan and Jianguo Xiao. 2008a. CollabRank: Towards a collaborative approach to single-document keyphrase extraction. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 969–976.

Xiaojun Wan and Jianguo Xiao. 2008b. Single document keyphrase extraction using neighborhood knowledge. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence*, pages 855–860.

Xiaojun Wan, Jianwu Yang, and Jianguo Xiao. 2007. Towards an iterative reinforcement approach for simultaneous document summarization and keyword extraction. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 552–559.

Chen Wang and Sujian Li. 2011. CoRankBayes: Bayesian learning to rank under the co-training framework and its application in keyphrase extraction. In *Proceedings of the 20th ACM International Conference on Information and Knowledge Management*, pages 2241–2244.

Ian H. Witten, Gordon W. Paynter, Eibe Frank, Carl Gutwin, and Craig G. Nevill-Manning. 1999. KEA: Practical automatic keyphrase extraction. In *Proceedings of the 4th ACM Conference on Digital Libraries*, pages 254–255.

Yi-Fang Brook Wu, Quanzhi Li, Razvan Stefan Bot, and Xin Chen. 2005. Domain-specific keyphrase extraction. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, pages 283–284.

Wen-Tau Yih, Joshua Goodman, and Vitor R. Carvalho. 2006. Finding advertising keywords on web pages. In *Proceedings of the 15th International Conference on World Wide Web*, pages 213–222.

Wei You, Dominique Fontaine, and Jean-Paul Barthès. 2009. Automatic keyphrase extraction with a refined candidate set. In *Proceedings of the IEEE/WIC/ACM International Joint Conference on Web Intelligence and Intelligent Agent Technology*, pages 576–579.

Torsten Zesch and Iryna Gurevych. 2009. Approximate matching for evaluating keyphrase extraction. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing 2009*, pages 484–489.

Hongyuan Zha. 2002. Generic summarization and keyphrase extraction using mutual reinforcement principle and sentence clustering. In *Proceedings of 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 113–120.

Yongzheng Zhang, Nur Zincir-Heywood, and Evangelos Milios. 2004. World Wide Web site summarization. *Web Intelligence and Agent Systems*, 2:39–53.

Yongzheng Zhang, Nur Zincir-Heywood, and Evangelos Milios. 2005. Narrative text classification for automatic key phrase extraction in web document corpora. In *Proceedings of the 7th ACM International Workshop on Web Information and Data Management*, pages 51–58.

Xin Zhao, Jing Jiang, Jing He, Yang Song, Palakorn Achanauparp, Ee-Peng Lim, and Xiaoming Li. 2011. Topical keyphrase extraction from twitter. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 379–388.