KPCatcher – A Keyphrase Extraction System for Enterprise Videos

Yongxin Taylor Xi, Matthias Paulik, Venkata Ramana Gadde and Ananth Sankar

Media Experience Analytics Business Unit, Cisco Systems

{yoxi, mapaulik, vegadde, asankar}@cisco.com

Abstract

This paper introduces KPCatcher (keyphrase catcher). The value of our work lies in providing concrete solutions to building a real keyphrase extraction product for enterprise videos. KPCatcher has been designed to robustly extract a ranked list of keyphrases from enterprise videos, independent of the domain. It treats noun phrases in the transcript as candidate keyphrases and scores them by aggregating word-level scores. By using confidence-based and counting-based rules, KPCatcher handles transcription errors to prevent incorrect keyphrases to be surfaced to end users. Different from previous work, we focus our experiments on automatic transcriptions of real enterprise videos from various domains. We thoroughly evaluate several well-known keyword ranking features and the denoising rules, using enterprise videos from several domains at various word error rates. We find term frequency to be the best feature and show that our denoising rules are very effective in both rejecting incorrect keyphrases and increasing the overlap between top keyphrases and human provided keyphrases. We also show that KPCatcher compares favorably to existing research systems on ICSI meeting data.

Index Terms: keyphrase extraction, enterprise videos, speech recognition, denoising, term frequency, TextRank

1. Introduction

Video is everywhere. The need to record and then access enterprise video is growing. For example, in schools lectures are recorded so that students can watch them later at any time. In companies trainings and meetings are recorded for future viewing. Therefore, an effective and scalable approach for organizing, accessing and consuming enterprise video data is required. One approach that has recently been adopted in industry [11,12,13] is to index a video by automatically extracting keyphrases from its transcripts and use them as a brief summary for the video. This is achieved by first transcribing a video's audio track using automatic speech recognition (ASR) and then applying known keyphrase extraction algorithms. Despite its emerging use in industry, keyphrase extraction on automatic speech transcripts has only seen a relatively small level of effort in the literature, when compared to written documents. Moreover, related work typically does not address domain independence. For example, authors in [1,2,3] focus their experiments on ICSI meeting data; techniques on keyphrase extraction for course lectures are evaluated in [18]; in [19] the authors propose a method to identify topics for 10-minute-long telephone conversations. Different from these works, we are interested in building a real keyphrase extraction product that works well for enterprise videos, coming from any domain. The system needs to extract meaningful keyphrases irrespective of domain. To ensure good user experience, keyphrase extraction process should be fast and the extracted keyphrases should be mostly correct even when word error rate (WER) is high. As a real product, it also has to work during the initial deployment, i.e. given the very first video that is uploaded to the system.

In this paper we propose KPCatcher (keyphrase catcher) as a solution for enterprise video keyphrase extraction. It is part of Cisco's enterprise video analytics product [12]. Our focus is to thoroughly evaluate several well-known unsupervised algorithms on multi-domain enterprise video data at different WERs. Because the product needs to work well from day one, we are more interested in document-based rather than corpus-based methods as the latter require a large, pre-existing collection of domain-specific (spoken) documents. As a result, KPCatcher operates solely on the ASR transcript of a video. It treats noun phrases as potential keyphrases and scores them by aggregating word-level scores. The word scores are computed using well-known features in keyword extraction literature. Further, to cope with ASR errors, we develop and evaluate a set of denoising rules that help us avoid surfacing misrecognized keyphrases. By collecting and annotating real enterprise video from technology, healthcare and education domains, we compare the keyword extraction features and examine denoising rules on the ASR transcripts of these videos at various WERs. We also show that KPCatcher compares favorably to several competing systems [2,3] by using manual ICSI meeting transcripts.

The remainder of the paper is organized as follows. Section 2 reviews related work on keyphrase extraction. Section 3 describes KPCatcher system workflow, its ranking approach and our post-processing rules for keyphrase denoising. We discuss the preparation of data sets for evaluation in Section 4 and report experimental results in Section 5. We conclude in Section 6 with a discussion on major findings.

2. Related Work

Two major approaches to automatic keyphrase extraction can be distinguished: unsupervised and supervised methods. Unsupervised methods [4,5] view the problem as a ranking problem; they identify keywords by ranking word-level features. Supervised approaches treat keyphrase extraction as a classification problem and learn a classification model from training examples. The KEA algorithm, for example, trains a Bayesian prediction model [6]. Unsupervised methods are much simpler than supervised ones as they do not require any annotated data for training and can be easily applied to a new domain. Given our product constraints, we adopt the unsupervised approach. Prior art closest to our work includes [1] (unsupervised) and [2,3] (supervised). We directly compare our system to them in Section 5.2.

Another common classification scheme for keyphrase extraction approaches is to distinguish between corpus-based and document-based approaches. While corpus-based approaches assume a sufficiently large corpus from a matching domain, document-based approaches use the underlying document as the only input to the system. Corpus-based methods either use corpus to calculate corpus-level features such as *IDF* and *TF-IDF* (defined later), or estimate latent topic models to facilitate keyphrase extraction [17, 18, 19].

Copyright © 2013 ISCA

Here we are more interested in document-based approaches as they run faster and can be easily applied to new customers without requirement of a large domain-matching corpus.

There are also works on spoken document retrieval for a large-scale public audio repository across different media, e.g. the SpeechFind system [20, 21] for the NGSW project [22]. Yet these works focus on retrieving relevant document given a query rather than extracting keyphrases from documents.

With our focus on unsupervised, document-based keyphrase extraction approaches, let us first briefly discuss some features that are commonly used in automatic keyword extraction. We include *TF-IDF* in our discussion and will compare its performance to *TF* when only a small-sized corpus is available as it is typical during initial product deployment.

— *TF(t,d)*, term frequency, is the number of times term *t* (here a term is a word) occurs in document *d* divided by the total number of terms in *d*.

— *TF-IDF(t,d)=TF(t,d) ∗ IDF(t)*, term frequency inverse document frequency, is the product of *TF* and *IDF*, where *IDF(t)*= $-log_2(n\,(t)/N)$. Here $n\,(t)$ is the number of documents containing term *t* among total *N* documents.

— *RP(t,d)*=$1 - pos(t,d)/L$ is relative position of the first occurrence of term *t* in *d*. Here $pos(t,d)$ is the number of words that occur before the first occurrence of *t*, and *L* is the number of words in *d*.

— *TextRank(t,d)* is a graph-based ranking feature proposed in [4] for keyword extraction. It creates a graph by linking words that co-occur in the same window and ranks words through "recommendation" similar to Google's PageRank. *TextRank* was shown to be the best system on a set of *Inspec* technical paper abstracts [4].

3. Approach

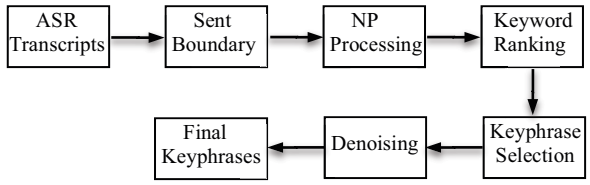

Fig. 1 KPCatcher system flow diagram

A traditional written-text keyphrase extraction system typically contains three major components: part-of-speech (POS) processing, keyword ranking and keyphrase selection [4,6,7]. Since KPCatcher is designed for automatic speech transcripts, we add a preprocessing step that inserts punctuation to ASR transcripts using a sentence boundary detector. More importantly, we add a postprocessing step to reject incorrect keyphrases.

We leverage noun phrases (NPs) for keyphrase extraction because it is widely accepted that NPs are good candidates for keyphrases [4,7]. A noun phrase chunking utility, based on openNLP [8], is used to obtain NPs from the text. We filter and normalize the NPs by stopword removal and word lemmatization. Words within the processed NPs are then scored using features discussed in Section 2, with top 50% ([4] used a third) selected as keywords. Then, during keyphrase selection, we keep the NPs that contain only keywords. Remaining phrases are scored by the sum of word scores, to promote longer phrases. If two phrases are overlapping (one as subphrase of the other), we select the one with higher unique counts. Finally, we denoise the selected keyphrases by rejecting those with low confidence (defined in Section 5.3) as well as keyphrases that occur only once. To our knowledge, methods for denoising keyphrases extracted from automatic speech transcripts has not been carefully studied before.

4. Data

We make use of one dataset for parameter tuning and two test sets for evaluation. Kaldi [9], an open-source speech recognition toolkit, is used for the ASR system. Our first dataset (CTV) consists of 32 videos with an average length of 20 minutes. These videos were recorded from Cisco internal corporate communications and have low word error rates. To simulate various WER ranges for a thorough test of KPCatcher, we artificially increase the WER by limiting the search beam and/or applying only simple acoustic and language models during recognition. Our test sets were obtained from a partner operating in educational domain (edu test set) and a customer in medical domain (med test set). The edu set contains 12 video clips with mostly formal speech. The med set contains 16 video clips of lectures. For both sets the average video length is 15 minutes.

To create ground truth keyphrases for our datasets, we employ a process that ensures that each video is annotated by two annotators. Annotators are given both reference transcript and audio, and are asked to extract keyphrases and assign them to three categories of descending importance: *top-ten*, *high priority* and *low priority*. It is also requested that at least 20 keyphrases total should be provided. The average annotator agreement for the top-ten keyphrases ranges only between 32-37%. Similarly low human agreement for this task is found in [1,2,3]. This demonstrates the difficulty to define a ground truth. We assign scores ranging from 3-1, respectively to keyphrases in the three categories, with descending importance. The final score of a keyphrase is defined as the score received or the sum of two scores if selected by both annotators. Finally we define the ground truth as keyphrases that score 3 and above. This method yields on average about 20 keyphrases per video.

We also use manually transcribed ICSI meeting transcripts to compare KPCatcher directly to some existing research systems in [1,2,3]. Each meeting comes with manually divided topic sections and each topic is annotated with respective keyphrases. Since [1,2,3] evaluate only keywords, we run KPCatcher to the keyword ranking step to only produce keywords on this dataset.

5. Experiments

5.1. Feature evaluation

We first examine the keyphrases (without denoising) on the CTV dataset, using the four ranking features *TF*, *TF-IDF*, *RP* and *TextRank*. For our use case, the calculation of *TF-IDF* can be nontrivial. *TF-IDF* estimation tends to be unreliable in speech corpora with ASR errors [14]. Another approach is to prepare a corpus to estimate *IDF*, however, obtaining a decent corpus that matches well with the domain at hand is itself difficult. We choose to estimate *IDF* from the collection of reference transcripts. We also use *TF* and *TF-IDF* as baselines due to their reasonably good performances in prior works and investigate the following score combinations:

$$s(t) = TF(t) + a * TextRank(t) + b * RP(t) \quad (1)$$

$$s(t) = TF\text{-}IDF(t) + a * TextRank(t) + b * RP(t) \quad (2)$$

Each feature is normalized among all terms such that the minimum and maximum become 0 and 1, respectively. We refer to (1) as *TF* combo and (2) as *TF-IDF* combo.

Since there are on average 20 keyphrases as ground truth for each video, we choose the top 20 keyphrases returned by KPCatcher. For evaluation, we adopt the F-measure, the harmonic mean of precision and recall: $F=2*P*R/(P+R)$ and relax it to account for subphrase matches.

We define the relaxed match of phrase *s* and *g* as:

$$m(s,g) = \begin{cases} \frac{\min(len(s), len(g))}{\max(len(s), len(g))}, & \text{if one is subphrase of other} \\ 0, & \text{otherwise.} \end{cases}$$

where *len(s)* is the number of words in *s*. Precision of a set of keyphrases (*S*) against the ground truth (*G*) is: $P(S,G) = (\sum_{s\in S} m(s,G))/|S|$ with $m(s,G) = \max_{g\in G} m(s,g)$ and similarly recall is $R(S,G) = (\sum_{g\in G} m(g,S))/|G|$. The precision here with the relaxed match is the same as the R-precision metric, which has been found to achieve the highest correlation with human annotations [16].

CTV		Relaxed F-measure %					
Algo		*TF*	*TF-IDF*	*Text Rank*	*RP*	*TF* combo	*TF-IDF* combo
Ref		35.4	28.7	27.1	29.9	**36.9**(0.1,0.1)	33.6(0.4,0.1)
WER %	20.3	32.4	26.9	23.7	26.4	**32.9**(0, 0.1)	29.4(0, 0.3)
	25.5	30.2	23.7	22.0	23.4	**30.6**(0, 0.1)	27.3(0.1,0.2)
	29.8	**30.8**	23.8	20.7	23.5	**30.8**(0, 0)	26.6(0.1,0.1)
	35.2	**29.9**	23.9	20.7	22.6	**29.9**(0, 0)	25.9(0.3,0.2)
	40.4	**29.7**	23.0	20.5	22.1	**29.7**(0, 0)	26.0(0.3,0.1)

Table 1. F-measure of top 20 keyphrases on CTV set. Optimal weights for (*TextRank*, *RP*) are reported in brackets.

Table 1 lists relaxed F-measures of keyphrases extracted from reference and ASR transcripts at 5 WER levels for the CTV set. The results are averaged over the 32 videos. We use grid search to select optimal parameter pairs (a, b) for both *TF* combo and *TF-IDF* combo from 64 possible pairs by letting *a, b* = 0, 0.1, 0.2, 0.3, 0.4, 0.5, 1.0, 2.0. The findings are:

1. Not surprisingly, F-measure tends to drop w.r.t WER.
2. Both *TF* and *TF* combo achieve a F-measure of more than 35% on the reference transcripts, which is quite remarkable as human annotators, on average, only achieve 32% on the top-ten keyphrases (evaluating one annotator against the other and vice versa).
3. We find that *TF-IDF* performs uniformly worse, almost absolute 7% lower than *TF* at all WER levels. This suggests that *IDF* estimates made from a small video transcript corpus are inaccurate. Adding other features to *TF-IDF* helps, although *TF* combo still outperforms *TF-IDF* combo by a large margin.
4. Also notice that *RP* and *TextRank* contribute very little to *TF* combo and not at all at higher WER levels. This suggests that *TF* is very robust and effective feature for keyword ranking in erroneous transcripts.
5. *TextRank* alone does quite poorly, indicating that graph-based methods may be less effective on spoken document, which is similar to the findings in [2].

Next we evaluate these features on the test sets (see Table 2). The med test set has a higher WER (41.8%) than the edu test set (30.5%). When reporting the result of *TF* combo and *TF-IDF* combo, we use the optimal weights estimated from the CTV set. Since we investigated multiple WER levels on the CTV set, we take the optimal weights of the respective CTV WER level closest to the test set WER. Again *TF* performs much better than the others. *TF* combo and *TF-IDF* combo perform the same or worse than the baselines *TF* and *TF-IDF*, likely caused by domain mismatch.

5.2. Comparison to existing research systems

In Table 3 and 4, we compare the performance of KPCatcher with the *TF* ranking against the results reported in [2, 3] on the ICSI data set. Although [1] is a more direct match with our approach (unsupervised), its results are not comparable because it treats nouns in singular and plural forms as different words whereas we use lemmatization to map them to the same word. Each ICSI meeting transcript contains multiple segmented topics, with each topic annotated by several annotators (3 annotators in [3] and 2 annotators in [2]). We prepare the data in the same way as [2,3], by either only keeping "on-topic" topics [2], or by rejecting topics classified as "Chitchat" or "Digits" [3]. Following [2,3], we obtain the F-measure by comparing the top 5 keywords of each topic against the ground truth. The first two rows in Table 3 are reported results reported in [3]. *TF-IDF*+POS is an unsupervised method that uses POS tag filtering in addition to *TF-IDF*. Supervised+Bigram is the proposed supervised method in [3] which uses web resources to verify the quality of selected bigrams. To train a supervised model, 6 other meetings were used as training data. Table 3 shows that KPCatcher outperforms the rest.

Test sets		Relaxed F-measure %					
Algo		*TF*	*TF-IDF*	*Text Rank*	*RP*	*TF* combo	*TF-IDF* combo
med	ref	**40.8**	36.5	28.9	28.7	39.4(0.1,0.1)	33.5(0.4,0.1)
	WER 41.8	**33.7**	28.9	22.6	21.5	**33.7(0, 0)**	28.9(0.3,0.1)
edu	ref	**42.6**	40.1	31.8	32.5	42.5(0.1,0.1)	37.3(0.4,0.1)
	WER 30.5	**38.4**	35.5	25.2	26.6	**38.4(0, 0)**	34.3(0.1,0.1)

Table 2. F-measure on two test sets. We adopted optimal weights from the CTV set closest to the CTV WER level.

Methods	P (%)	R (%)	F (%)
TF-IDF + POS	24.6	27.2	25.4
Supervised + Bigram	25.5	35.3	29.1
KPCatcher (*TF*)	**33.0**	**36.7**	**32.3**

Table 3. Comparison with [3] on ICSI data using 20 test meetings. Results are averaged among three annotators and topics. The first two rows are extracted from [3].

Methods	P (%)	R (%)	F (%)
TF-IDF	35.3	32.5	33.8
KEA	36.1	33.2	34.5
Supervised-CONF	**42.2**	**38.3**	**40.0**
KPCatcher (*TF*)	38.2	36.5	35.3

Table 4. Comparison with [2] on ICSI data using 26 videos. Results are averaged among two annotators and topics. The first three rows are extracted from [2].

In Table 4, the first three rows are results of *TF-IDF*, supervised method KEA and the supervised-CONF method proposed in [3]. The leave-one-out method was adopted for the supervised methods in [3] such that in each round 25 videos were used for training and the left-out one used for testing. Supervised-CONF appears to perform much better than Supervised+Bigram, likely due to a richer feature set and more training data. Here KPCatcher performs slightly better than the supervised KEA but worse than supervised-CONF. This is reasonable since KPCatcher is an unsupervised

approach that has no domain knowledge at all about the ICSI meetings. Its model is simple and can be quickly and easily applied to new domains.

5.3. De-noising rules evaluation

As pointed out before, we are particularly interested in methods that ensure a low keyphrase error rate (KPER). We declare a keyphrase occurring at time t in the ASR transcript to be an error if the same keyphrase cannot be found within $t \pm 2$ seconds in the reference transcript. KPER is the percentage of such errors among all keyphrase occurrences.

In this section we report results of KPCatcher with *TF* as its ranking feature. The first KPER column in Table 5 lists the baseline KPER of KPCatcher keyphrases at several WER levels. Interestingly, KPER is already typically around a third of the WER, likely due to the robustness of term frequency feature. To further reduce KPER, we discard the keyphrase occurrences that have a confidence score below threshold T. The confidence of a keyphrase occurrence is defined as the average of the word-level confidences of its words estimated by the ASR system. A higher threshold typically results in a lower KPER, but also more falsely rejected keyphrases. To tune the threshold value T, we control the percentage of false rejects at 2%, 5%, 10% and 15% and report respective threshold T and KPER in Table 5. In our final KPCatcher implementation, we decided to accept a 5% false rejects, corresponding to an average threshold value of $T = 0.70$. For this value, the reduction in PER ranges between 2-3% absolute, with larger reductions at higher WER levels.

WER	Percentage of false rejects %									
	0		2		5		10		15	
	KPER	*T*	KPER	*T*	KPER	*T*	KPER	*T*	KPER	*T*
20.3	5.2	0.00	4.6	0.64	**4.1**	**0.86**	3.7	0.96	3.7	0.99
25.5	8.3	0.00	7.2	0.56	**6.5**	**0.71**	5.6	0.86	5.0	0.93
29.8	8.2	0.00	7.1	0.57	**6.6**	**0.72**	5.7	0.85	5.0	0.93
35.2	9.7	0.00	8.2	0.54	**7.3**	**0.70**	6.4	0.84	5.7	0.91
40.4	13.4	0.00	11.4	0.51	**10.7**	**0.63**	9.4	0.74	8.2	0.84

Table 5. Without the counting rule, averaged KPER (%) and confidence thresholds at different levels of recall.

In our studies, we observed that keyphrases that occur only once in the ASR transcript tend to either be incorrect or unimportant. Hence, we discard the keyphrases that occur only once. From our experiments this simple heuristic further lowers KPER by 1-3% absolute. We then evaluate the two denoising rules on the test sets and report results in Table 6. Both rules significantly improve the KPER for both test sets, and their combination reduces the KPER by a relative of 40%. The percentage of false rejects after imposing the rules is 4.8% and 5.2% respectively for edu and med sets.

Finally we examine the effect of the confidence thresholding on the relaxed F-measure. At a reasonable threshold, we expect the F-measure to increase as a result of rejecting incorrect or unimportant keyphrases. This is especially true for algorithms that do not use frequency-based features. For example, in Fig. 2, the F-measure of *TextRank* gains remarkably when the threshold increases, despite the drop at 0.9 for the case of 20.3% WER. On the other hand, frequency-based features such as *TF* are intrinsically robust to ASR errors, so as shown in Fig. 3 such improvement is not as obvious. From these plots a selection of $T = 0.70$ seems reasonable. We then report F-measure on the test sets after applying the rules in Table 7. The relaxed F-measure always improves after introducing both rules, especially for *TextRank* and *RP* features.

Test set (WER)	KPER %			
	No rules	Post-processing rules		
		$T = 0.70$	*Count* > 1	Both
edu (30.5%)	11.4	8.7	8.6	**6.9**
med (41.8%)	22.2	16.9	15.8	**12.9**

Table 6. KPER after applying the post-processing rules

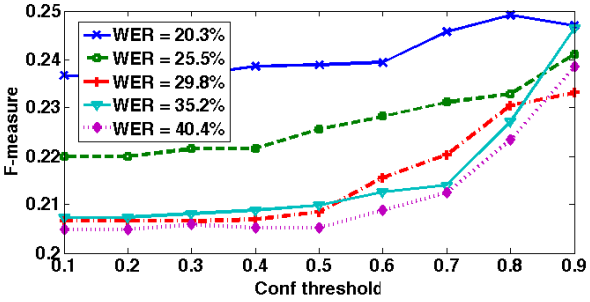

Fig.2 F-measure of TextRank (CTV) with conf thresholding

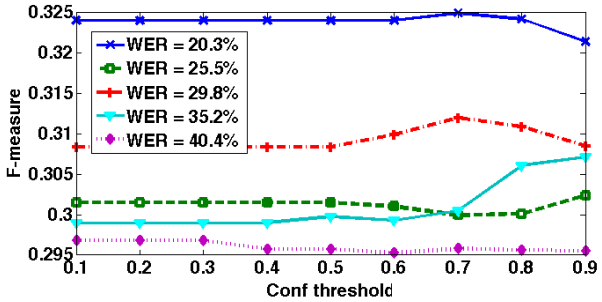

Fig.3 F-measure of TF (CTV) with conf thresholding

Test set (WER)	Algorithm	Relaxed F-measure %			
		No rules	Post-processing rules		
			$T = 0.70$	*Count* > 1	Both
edu (30.5%)	*TextRank*	25.2	26.4	**33.5**	**33.5**
	RP	26.6	26.9	**33.6**	**33.6**
	TF combo	38.4	38.4	38.3	**38.7**
med (41.8%)	*TextRank*	22.6	23.5	32.9	**33.0**
	RP	21.5	23.3	**32.4**	**32.4**
	TF combo	33.7	34.4	**35.4**	**35.4**

Table 7. F-measure of top 20 keyphrases on the two test sets after introducing the post-processing denoising rules.

6. Conclusion

In this paper, we introduced KPCatcher, a solution to keyphrase extraction product for enterprise videos which is domain independent, easy to deploy and robust to noise. We thoroughly examined well-known keyword extraction features on speech transcripts of real enterprise videos from multiple domains at different WER levels. We found that term frequency is the most important and robust feature for this problem. Furthermore, the proposed denoising rules not only effectively reduced the keyphrase error rate but also noticeably improved the quality of the keyphrases. We think this work provides concrete and valuable guidelines to building real keyphrase extraction system for enterprise videos.

7. Acknowledgement

We thank all the colleagues who contributed to the tasks of annotating keyphrases for the internal video datasets. We also sincerely thank Y. Liu and F. Liu for providing us the ICSI data set with human annotations.

8. References

[1] F. Liu, D. Pennell, F. Liu and Y. Liu, "Unsupervised Approaches for Automatic Keyword Extraction using Meeting Transcripts," Human Language Technologies: Conference of the North American Chapter of the Association for Computational Liguistics, 2009, pp. 620-628.

[2] F. Liu, F. Liu and Y. Liu, "A Supervised Framework for Keyword Extraction from Meeting Transcripts," IEEE Transactions on Audio, Speech and Language Processing, Vol. 19, No.3, 2011, pp. 538-548.

[3] F. Liu, F. Liu and Y. Liu, "Automatic Keyword Extraction for the Meeting Corpus using Supervised Approach and Bigram Expansion," Proceedings of IEEE Spoken Language Technology, 2008, pp. 181-184.

[4] R. Mihalcea and P. Tarau, "TextRank: Bringing Order into Texts," Conference on Empirical Methods in Natural Language Processing, 2004, pp. 404-411.

[5] G. Salton, C. S. Yang and C. T. Yu, "A Theory of Term Importance in Automatic Text Analysis," Journal of the American Society for Information Science, volume 26, issue 1, 1975, pp. 33-44.

[6] I. H. Witten, G. W. Paynter, E. Frank, C. Gutwin and C. G. Nevil-Manning, "KEA: Practical Automatic KP Extraction," 4th ACM Conference on Digital Library, 1999, pp. 254-255.

[7] A. Hulth, "Improved Automatic Keyword Extraction Given More Linguistic Knowledge," Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing, 2003, pp. 216-223.

[8] openNLP, Apache Software Foundation, http://opennlp.apache.org.

[9] D. Povey et al., "The Kaldi Speech Recognition Toolkit," IEEE 2011 Workshop on Automatic Speech Recognition and Understanding, 2011.

[10] Pdftotext, open source utility, http://en.wikipedia.org/wiki/pdftotext.

[11] VoiceBase, http://www.voicebase.com.

[12] Cisco Pulse Analytics, http://www.cisco.com/en/US/products/ps12130/index.html.

[13] Autonomy Virage, http://www.virage.com.

[14] D. Karakos et al., "Estimating Documents Frequencies," Automatic Speech Recognition and Understanding (ASRU), 2011, pp. 407-412.

[15] J. Diao, V. Gadde and A. Sankar, "Cisco AutoVocab, Create Domain Vocabulary for Enterprise Video Analyics," internal publication, Cisco Technology Forum (CTech), 2011.

[16] S. N. Kim, T. Baldwin and M. Kan, "Evaluating N-gram based Evaluation Metrics for Automatic Keyphrase Extraction," COLING '10 Proceedings of the 23rd International Conference on Computational Linguistics, 2010, pp. 572-580.

[17] Z. Liu, W. Huang, Y. Zheng and M. Sun, "Automatic Keyphrase Extraction via Topic Decomposition," Conference on Empirical Methods in Natural Language Processing, 2010, pp. 366-376.

[18] Y. Chen, Y. Huang, S. Kong and L. Lee, "Automatic Key Term Extraction from Spoken Course Lectures Using Branching Entropy and Prosodic/Semantic Features," IEEE Workshop on Spoken Language Technology, 2010, pp. 265-270.

[19] D. Harwath and T. J. Hazen, "Topic Identification Based Extrinsic Evaluation of Summarization Techniques Applied to Conversational Speech," International Conference on Acoustics, Speech and Signal Processing, 2012, pp. 5073-5076.

[20] J. Hansen, R. Huang, P. Mangalath, B. Zhou, M. Seadle and J. Deller, "SpeechFind: Spoken Document Retrieval for a National Gallery of the Spoken Word," Nordic Signal Processing Symposium, 2004.

[21] R. Huang, B. Zhou, M. Seadle, J. Deller, A. Gurijala, M. Kurimo, P. Angkititrakul, "SpeechFind: Advances in Spoken Document Retrieval for a National Gallery of the Spoken Word," IEEE Transactions on Speech and Audio Processing, vol. 13, issue. 5, 2005, pp. 712-730.

[22] The National Gallery of the Spoken Word Project, http://www.ngsw.org