# Real-Time Presentation Tracking Using Semantic Keyword Spotting

*Reza Asadi, Harriet J. Fell, Timothy Bickmore, Ha Trinh*

Northeastern University, Boston, MA, USA
{asadi, fell, bickmore, hatrinh}@ccs.neu.edu

## Abstract

Given presentation slides with detailed written speaking notes, automatic tracking of oral presentations can help speakers ensure they cover their planned content, and can reduce their anxiety during the speech. Tracking is a more complex problem than speech-to-text alignment, since presenters rarely follow their exact presentation notes, and it must be performed in real-time. In this paper, we propose a novel system that can track the current degree of coverage of each slide's contents. To do this, the presentation notes for each slide are segmented into sentences, and the words are filtered into keyword candidates. These candidates are then scored based on word specificity and semantic similarity measures to find the most useful keywords for the tracking task. Real-time automatic speech recognition results are matched against the keywords and their synonyms. Sentences are scored based on detected keywords, and the ones with scores higher than a threshold are tagged as covered. We manually and automatically annotated 150 slide presentation recordings to evaluate the system. A simple tracking method, matching speech recognition results against the notes, was used as the baseline. The results show that our approach led to higher accuracy measures compared to the baseline method.

**Index Terms**: presentation tracking, keyword spotting, semantic similarity

## 1. Introduction

Oral presentations are a critical and challenging aspect of academic and professional life. During presentation delivery, presenters often face the fear of failing to remember what to say in the moment, which may lead to speech anxiety and poor presentation quality. One way to address this problem is to continuously track the location in the presentation notes currently being spoken by the presenter, and identify parts that have not yet been covered. This tracking information could then be used to provide intelligent teleprompters, automatically highlighting key phrases to remind the presenter what to say. Alternatively, it could also be used to develop intelligent virtual co-presenter systems [1], in which a virtual agent could track the presentation progress and automatically deliver parts that have been forgotten by the human presenter.

There are many challenges in achieving a real-time presentation tracking system. For example, tracking depends on the accuracy of the automatic speech recognition (ASR) system used, and although the accuracy and performance of ASR systems have been greatly improved over the last decade, they are still far from perfect. Even having a perfect transcription of the speech doesn't solve the tracking problem, since the presenters rarely follow their speaking notes exactly. The system cannot depend on exact forms of words and sentences, and should be able to detect the semantic relatedness between the spoken terms and the source text. Another challenge is the need for real-time performance, which limits the system input to a continuous stream from the ASR system. This adds uncertainty about the overall structure of the spoken content that could help in associating utterance segments with slide content.

In this paper we present a real-time presentation tracking system that identifies which sentences have been covered in the notes of the current presentation slide, using semantic keyword spotting. In section 2 we discuss some of the previous work on this topic, in section 3 we explain the methodology used in our system and in section 4 we present our evaluation results, comparing different tracking approaches.

## 2. Related Work

There have been limited studies on presentation tracking and alignment. In these studies, automatic speech recognition (ASR) was used to transcribe the presentations and text alignment methods were used to match the script and the transcriptions. Rogina et al [2] used dynamic time warping to match slides with the ASR output hypothesis. Okada et al [3] computed the minimum distance between ASR hypothesis and speech script in order to track the current state of speech. We use a similar approach as our baseline method. In [4] Yamamoto et al. segmented the lecture transcriptions into topics by associating them with the textbook used in the lecture. To do so, term frequency - inverse document frequency (*tf.idf*) vectors were calculated for text topics and speech transcripts and then cosine similarity between vectors was used for association. Lu et al [5] used entropy-based word filtering, reliability-propagated word-based matching, and structured support vector machines to align utterances clusters with slide subsections. To our knowledge, this is the only work to date on within slide alignment. However, it assumed in-order presentation, and did not provide real-time support.

Accurate tracking depends on the accuracy of the ASR system used. The accuracy of the lecture transcriptions can be improved by retrieving text related to the presentation from the web or supplementary material and adapting the vocabulary and language model [2][6][7][8]. Another approach is to reduce the dependency on the accuracy of the ASR system. Acoustic keyword spotting instead of speech transcription can eliminate the need for a language model. The use of confusion networks is also an efficient method for spoken term detection [9].

If a keyword spotting approach is taken, the selection of proper keywords from the text becomes a very important problem. There are several studies on keyword extraction from lecture speech [10] [11] [12]. Another useful approach is semantic retrieval of spoken keywords, since speakers often utter words that are semantically related to the text keywords, without speaking any of the exact keywords [15]. In [14], the authors argued that keywords should be semantically relevant

http://dx.doi.org/10.21437/Interspeech.2016-617

to the document theme and also provide a good coverage of the concepts. They clustered terms based on semantic relatedness and extracted key phrases from exemplar terms in these clusters. We used a similar approach to score keyword candidates.

# 3. Methodology

Our tracking system consists of 4 main units: Note Processing, ASR, Keyword Spotting, and Sentence Tagging. At first, slide notes are filtered to extract keyword candidates. Then the keywords are prepared and scored based on their usefulness for tracking. At runtime, ASR results are passed to the Keyword Spotting unit to be matched against the keyword candidates. Sentences are scored based on the detected keywords and their scores. Finally, sentences that have scores higher than a threshold are tagged as covered. Figure 1 shows the overall architecture of the system.

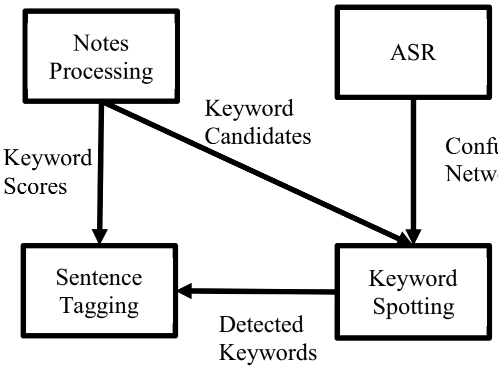


Figure 1: *Overall system architecture.*

## 3.1. Notes Processing

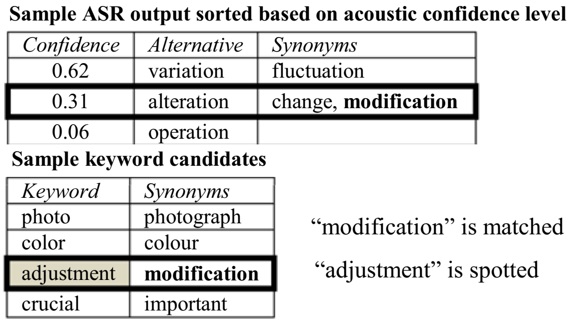
Slide notes should be segmented into smaller sections to make tracking more refined. But if the sections are too small, such as at word level, we will have over-fitting problems and even small deviations from the notes can result in false negative results. Therefore, sentences were chosen as the optimal segmentation granularity. Stanford CoreNLP tools [15] were used for sentence segmentation, part of speech tagging and word lemmatization. Stop words, punctuation marks and symbols were removed from the notes and numbers were converted into their word representations. The remaining word lemmas were selected as keyword candidates for each sentence.

WordNet [16] was used to extract synonyms. To do so, the most common synset for each word was retrieved and the words in that synset were extracted as that word's synonyms. If a word is a synonym for multiple words, the word in the synset with highest tagged frequency in WordNet was chosen. WordNet stemming was used in addition to CoreNLP lemmatization for comparative and superlative adjectives. The keyword candidates in their base form and their synonyms are stored in a table.

## 3.2. Speech Recognition and Keyword Spotting

Automatic speech recognition was performed using IBM's Watson cloud-based service [17]. This service provides both n-best transcripts and confusion networks. Confusion networks contain acoustically similar hypotheses for each time frame, ordered based on their acoustic confidence. These hypotheses are called the word alternatives. The alternatives with the level of confidence lower than 0.01 were discarded.

The ASR results are filtered to remove stop words and symbols. The remaining words are lemmatized and their synonyms are extracted. The system iterates through the ordered list of alternative words in each time frame and compares each word and its synonyms with the keyword candidates and their synonyms. If there is a match, other alternatives in that time slot are discarded, and the matched keyword candidate is tagged as spotted in all sentences containing that candidate. Each candidate can only be spotted once in a slide. Figure 2 shows a sample keyword spotting scenario:



**Sample ASR output sorted based on acoustic confidence level**

| *Confidence* | *Alternative* | *Synonyms* |
|---|---|---|
| 0.62 | variation | fluctuation |
| 0.31 | alteration | change, **modification** |
| 0.06 | operation | |

**Sample keyword candidates**

| *Keyword* | *Synonyms* |
|---|---|
| photo | photograph |
| color | colour |
| adjustment | **modification** |
| crucial | important |

Figure 2: *A sample keyword spotting scenario.*

## 3.3. Sentence Tagging

Each time a keyword from a sentence is spotted the chance of that sentence being covered is increased. The amount of increase depends on the discrimination power of the keyword in the sentence. In order to model this effect, we assigned scores to keywords and each time a keyword is spotted the score for the sentences containing it increases by the amount equal to the score for that keyword. Therefore:

$$ss(s_i) = \sum_{w_j \in W_{s_i}} c(w_j) sw(w_j) \qquad (1)$$

Where *ss(si)* is the coverage score for the sentence $s_i$ and *sw(wj)* is the score for keyword $w_j$ in $W_{s_i}$ which is the set of keyword in $s_i$. *c(wj)* is equal to 1 when $w_j$ is spotted, otherwise it is 0. When the score for a sentence is higher than a threshold we tag that sentence as covered. The system assigns scores to keywords using two methods: *tf.idf* and semantic similarity.

### 3.3.1. *tf.idf score*

*tf.idf* is used for scoring the word based on its important in a corpus of documents. In our case we treat each sentence as a document:

$$tf.idf = tf(w_i, s_j) idf(w_i)$$
$$tf(w_i, s_j) = frequncy\ of\ w_i\ in\ s_j$$
$$idf(w_i) = \log\left(\frac{N}{n_t}\right) \qquad (2)$$
$$= \log\left(\frac{total\ number\ of\ sentences}{number\ of\ sentences\ containing\ w_i}\right)$$

The reasoning behind this score is that if a word is used in multiple sentences it has low specificity for each of those sentences. Therefore compared to a unique word, it is less useful for detecting a sentence. The issue with this method is that some unique words are not essential for the sentence concept and thus can be omitted during the presentation. *tf.idf* gives high scores to such words, and if the word is ignored the score for the sentence might stay lower than the threshold which leads to false negative results.

### *3.3.2. Semantic similarity score*

To fix the above issue, we also consider the semantic similarity of keywords to the sentences. Inspired by [14], we can argue that a good keyword should be more semantically relevant to the sentence containing it compared to other sentences. We model this concept using the similarity ratio score *sr*:

$$sr(w_i) = \frac{local\ similairty}{global\ similarity} \tag{3}$$

Local similarity is the similarity of a sentence keyword to other keywords in that sentence. Global similarity is the similarity of a sentence keyword to the keywords in other sentences.

To measure the semantic similarity between words, we use a vector space representation of words called Global Vectors for Word Representation (GloVe) [18]. Word vectors representing more semantically similar words have smaller Euclidean distance and bigger cosine similarity. GloVe vectors are trained using non-zero elements of a word-word co-occurrence matrix gathered from a large corpus. We use a pre-trained vector representation with 1.9 million uncased words and vectors with 300 elements. It was trained using 42 billion tokens of web data from Common Crawl. We will use both the Euclidean distance and cosine similarity of word vectors for calculating their semantic similarity.

To calculate the cosine similarity between a word and a word set containing it, we used the average cosine similarity between the word and all of the other words in that word set:

$$\forall w_i \in W \quad s_c(w_i, W) = \frac{\sum_{w_{j \neq i} \in W} sim(w_i, w_j)}{|W| - 1}$$

$$sim(w_i, w_j) = \frac{\sum_{k=0}^{n} v(w_i)_k . v(w_j)_k}{\sqrt{\sum_{k=0}^{n} v(w_i)_k^{\,2}}\sqrt{\sum_{k=0}^{n} v(w_j)_k^{\,2}}} \tag{4}$$

To calculate the similarity using the Euclidean distance we used a form of Closeness Centrality measure which has been used in graph based key phrase extraction [19]:

$$\forall w_i \in W \quad s_d(w_i, W) = \frac{|W| - 1}{\sum_{w_{j \neq i} \in W} dist(w_i, w_j)}$$

$$dist(w_i, w_j) = \sqrt{\sum_{k=0}^{n} (v(w_i)_k - v(w_j)_k)^2} \tag{5}$$

The value of *n* in equation 4 and 5 is equal to the number word vector dimensions which in our case was 300. Finally the similarity ratio in equation 3 is calculated using the cosine similarity or Euclidean distance:

$$sr(w_i) = \frac{s_c(w_i, W_L)}{s_c(w_i, W_G)} \ or \ \frac{s_d(w_i, W_L)}{s_d(w_i, W_G)} \tag{6}$$

$W_L$ is the set of words in the sentence containing $w_i$ and $W_G$ is the set of words in other sentences. Similarity ratios are used as weighting coefficients of *tf.idf* scores and these weighted scores are normalized by dividing the score of each keyword by the sum of scores of all of the keywords in the sentence:

$$\forall w_i \in W_{s_j} \ sw(w_i) = \frac{sr(w_i) tfidf(w_i)}{\sum_{w_k \in W_{s_j}} sr(w_k) tfidf(w_k)} \tag{7}$$

### *3.3.3. Sample scoring scenario*

In this subsection we present a sample keyword scoring scenario. Figure 3 shows the notes for a sample slide with 7 sentences. The keyword candidates are in bold.

1. The **tiger** is the **largest cat species**.
2. An **adult male wild tiger** can **reach** a **total body length** of up to **11.5 feet**, and **weigh** up to **850 pounds**.
3. The **most common color** of **tigers** is **orange**, with **black stripes**.
4. But every **tiger** has a **unique stripe pattern**, much like our **fingerprints**.
5. We have also **seen** some **color variations**, with **white**, **black**, **golden tabby**, and **blue tigers**.
6. The **current population** of **wild tigers** is **estimated** to be about **3200 individuals**.
7. There are **10 recognized tiger subspecies**, but **four** of them are **considered extinct**.

Figure 3: *Sample slide notes with keyword candidates in bold.*

| | *tf.idf* | *sr<sub>c</sub>* | *sr<sub>d</sub>* | *sw<sub>c</sub>* | *sw<sub>d</sub>* |
|---|---|---|---|---|---|
| **orange** | 0.845 | 1.113 | 1.3 | 0.293 | 0.284 |
| **common** | 0.845 | 0.668 | 1.106 | 0.176 | 0.242 |
| **color** | 0.477 | 1.147 | 1.295 | 0.170 | 0.160 |
| **black** | 0.477 | 1.086 | 1.281 | 0.161 | 0.158 |
| **stripes** | 0.477 | 1.347 | 1.267 | 0.200 | 0.156 |
| **tiger** | 0 | 0.887 | 1.15 | 0 | 0 |

Table 1: *keywords from sentence 3 of Figure 3. $sr_c$ and $sr_d$ are similarity ratios using distance and cosine, $sw_d$ and $sw_c$ are normalized word scores using $sr_d$ and $sr_c$*

Table 1 shows the scores for keywords of sentence 3. The table is ordered by normalized similarity scores. We can see that *tf.idf* discards the word “tiger” since it is used in all sentences. Similarity scores give higher scores to “orange”, “black” and “color” since they are semantically close to each other and represent the main sentence concept. The word “common” is scored highly by *tf.idf* since it is only used in this sentence but it has the lowest score in similarity ratios. Normalizing the score results in lowering the final score for “common”. This effect is more evident in $sw_c$ which uses the cosine similarity score.

# 4. Evaluation

## 4.1. Experiments

We evaluated our system using a corpus of 30 videotaped presentations delivered by 15 speakers (6 female, 9 male, 13 non-native English speakers), on the topics of lions and tigers. Each presentation contained 5 slides, with an average length of 5 minutes. Each slide had detailed speaking notes containing 6-10 sentences with an average of 8 sentences. Recordings were split for each slide, resulting in 150 slide presentation recordings.

In order to simulate real-time tracking conditions, each slide presentation recording was segmented into 3 sections in a semirandom manner. This was done by detecting the 2 longest pauses in speech and splitting the recording around them. We also made sure that each segment is at least 6 seconds long. Using this segmentation method, each speech segment might cover a random number of sentences from zero to the total number of sentences in the slide. A few recordings were too short to be split and were discarded. After segmentation we had a total of 426 audio files.

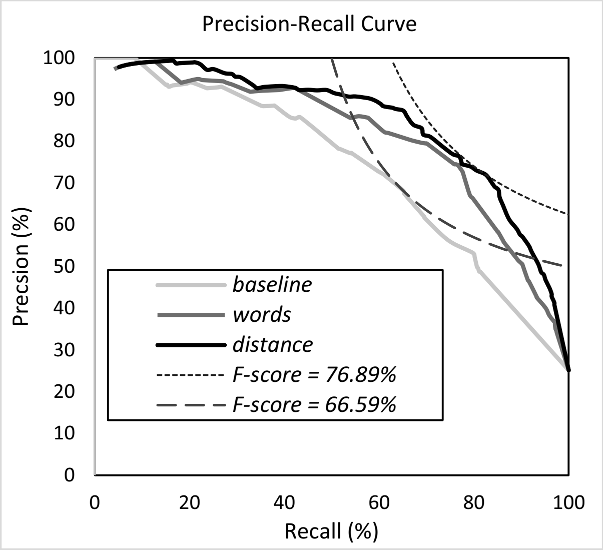


Figure 4: *Precision-Recall curves*

Each audio file was manually annotated for content coverage by a human annotator. The annotator was instructed to subjectively tag a sentence as covered if she found that the main points of the sentence were covered in sufficient detail. 100 recordings were randomly chosen and annotated by another annotator to check the inter-rater agreement. The Cohen's Kappa coefficient was 84%, which indicates a high degree of agreement. The files were also automatically annotated using different methods and thresholds. In order to evaluate the results, manual and automatic annotations were compared and precision, recall, and f-score measures were calculated for each method.

We evaluated a method similar to [3] as our baseline in which the 1-best ASR results were used instead of the confusion network. The stop words were removed and remaining words were used for matching the slide notes and the speech transcription without using synonyms. Sentences were scored based on the number of their spotted keywords over total number of their keywords. Table 2 lists the evaluated tracking methods and their reference names in the Results section:

| Name | Description |
|---|---|
| *baseline* | Using 1-best ASR output |
| *synons* | *baseline* + synonyms |
| *words* | confusion network ASR output + synonyms |
| *tfidf* | *words* method + tf.idf score |
| *cosine* | *tfidf* method + cosine similarity weighting |
| *distance* | *tfidf* method + Euclidean distance weighting |

Table 2: *Evaluated tracking methods' reference names and their descriptions*

### 4.2. Results

Figure 4 shows the precision-recall curves for 3 tracking methods with thresholds changing from 0 to 1. It also includes the curves for the highest F-score values for *distance* and *baseline* methods. We can see the improvements in precision, recall and F-score compared to the baseline method. Increasing the threshold generally results in lower recall but higher precision values. In our case, threshold values between 0.2 and 0.3 led to the best F-scores for all methods. The unsupervised nature of the system and different application requirements discourage us from determining an optimal threshold for all applications.

Table 3 shows the values for precision, recall and F-score for each experiment using threshold values optimized for F-score. A random approach that randomly tags sentences as covered is also included in the table for comparison. We can see the positive effect of using synonyms and the confusion network in improvements from *baseline* to *synons* and from *synons* to *words* method.

| Method Name | Precision (%) | Recall (%) | F-score (%) |
|---|---|---|---|
| *random* | 25.53 | 53.44 | 34.55 |
| *baseline* | 68.13 | 65.11 | 66.59 |
| *synons* | 70.32 | 71.06 | 70.69 |
| *words* | 74.30 | 76.55 | 75.40 |
| *tfidf* | 76.24 | 76.78 | 76.51 |
| *cosine* | **79.70** | 73.75 | 76.61 |
| *distance* | 72.00 | **82.50** | **76.89** |

Table 3: *Evaluation measures using thresholds optimized for best F-score*

Using the scoring methods improves the F-score compared to the *words* method and the similarity weighting methods have the best F-scores. Using the Euclidean distance similarity weighting results in the best recall value and the *cosine* method has the best precision. Depending on the application requirement, we can choose between these two similarity weighting methods.

## 5. Conclusion

In this paper we presented a real-time presentation tracking system that can track the current state of the presentation based on slide notes. To do so, the system extracts keyword candidates from the slide notes, and scores them based on specificity and semantic similarity. We showed that our system resulted in 10.3% increase in F-score compared to our baseline method.

We are planning to integrate our tracking system into a virtual co-presenter agent application [1] and evaluate it in a user study. There are also several aspects of our tracking system that can be improved, including the accuracy of the ASR system, alternative scoring methods, the use of slide content in addition to notes, and automatic topic segmentation of slide notes.

## 6. Acknowledgements

This work is supported in part by the National Science Foundation under award IIS-1514490. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

# 7. References

[1] H. Trinh, L. Ring, and T. Bickmore, “Dynamicduo: Co-presenting with virtual agents,” in Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems. ACM, 2015, pp. 1739–1748.

[2] I. Rogina and T. Schaaf, “Lecture and presentation tracking in an intelligent meeting room,” in Multimodal Interfaces, 2002. Proceedings. Fourth IEEE International Conference on. IEEE, 2002, pp. 47–52.

[3] T. Okada, T. Yamamoto, T. Terada, and M. Tsukamoto, “Wearablemc system a system for supporting mc performances using wearable computing technologies,” in Proceedings of the 2nd Augmented Human International Conference. ACM, 2011, p. 25.

[4] N. Yamamoto, J. Ogata, and Y. Ariki, “Topic segmentation and retrieval system for lecture videos based on spontaneous speech recognition.” in INTERSPEECH, 2003.

[5] H. Lu, S.-s. Shen, S.-R. Shiang, H.-y. Lee, and L.-s. Lee, “Alignment of spoken utterances with slide content for easier learning with recorded lectures using structured support vector machine (svm).” in INTERSPEECH, 2014, pp. 1473–1477.

[6] C. Munteanu, G. Penn, and R. Baecker, “Web-based language modelling for automatic lecture transcription.” in INTERSPEECH, 2007, pp. 2353–2356.

[7] P. Maergner, A. Waibel, and I. Lane, “Unsupervised vocabulary selection for real-time speech recognition of lectures,” in Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on. IEEE, 2012, pp. 4417–4420.

[8] A. Park, T. J. Hazen, and J. R. Glass, “Automatic processing of audio lectures for information retrieval: Vocabulary selection and language modeling.” in ICASSP (1), 2005, pp. 497–500.

[9] L. Mangu, B. Kingsbury, H. Soltau, H.-K. Kuo, and M. Picheny, “Efficient spoken term detection using confusion networks,” in Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on. IEEE, 2014, pp. 7844–7848.

[10] Y. Fujii, N. Kitaoka, and S. Nakagawa, “Automatic extraction of cue phrases for important sentences in lecture speech and automatic lecture speech summarization.” in INTERSPEECH, 2007, pp. 2801–2804.

[11] T. Kawahara, K. Shitaoka, T. Kitade, and H. Nanjo, “Automatic indexing of key sentences for lecture archives,” in Automatic Speech Recognition and Understanding, 2003. ASRU’03. 2003 IEEE Workshop on. IEEE, 2003, pp. 141–144.

[12] H. Yang and C. Meinel, “Content based lecture video retrieval using speech and video text information,” Learning Technologies, IEEE Transactions on, vol. 7, no. 2, pp. 142–154, 2014.

[13] L.-s. Lee, J. Glass, H.-y. Lee, and C.-a. Chan, “Spoken content retrieval beyond cascading speech recognition with text retrieval,” Audio, Speech, and Language Processing, IEEE/ACM Transactions on, vol. 23, no. 9, pp. 1389–1420, 2015.

[14] Z. Liu, P. Li, Y. Zheng, and M. Sun, “Clustering to find exemplar terms for keyphrase extraction,” in Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1. Association for Computational Linguistics, 2009, pp. 257–266.

[15] C. D. Manning, M. Surdeanu, J. Bauer, J. R. Finkel, S. Bethard, and D. McClosky, “The Stanford CoreNLP natural language processing toolkit.” in ACL (System Demonstrations), 2014, pp. 55–60.

[16] G. A. Miller, “Wordnet: a lexical database for english,” Communications of the ACM, vol. 38, no. 11, pp. 39–41, 1995.

[17] “Speech to Text | IBM Watson Developer Cloud”, ibm.com, 2016. [Online].Available: http://www.ibm.com/smarterplanet/us/en/ibmwatson/developercloud/speech-to-text.html. [Accessed: 30- Mar- 2016].

[18] J. Pennington, R. Socher, and C. D. Manning, “Glove: Global vectors for word representation.” in EMNLP, vol. 14, 2014, pp. 1532–1543.

[19] F. Boudin, “A comparison of centrality measures for graph-based keyphrase extraction,” in International Joint Conference on Natural Language Processing (IJCNLP), 2013, pp. 834–838.