Automatic Keyphrase Extraction and Segmentation of Video Lectures

Arun Balagopalan, Lalitha Lakshmi Balasubramanian, Vidhya Balasubramanian,
Nithin Chandrasekharan, Aswin Damodar
Department of Computer Science and Engineering, Amrita School of Engineering
Amrita Vishwa Vidyapeetham, Coimbatore, India
arunbgg@gmail.com, lalithalb@am.amrita.edu, b_vidhya@cb.amrita.edu,
nithin.cs27@gmail.com, djs.aswin@gmail.com

***Abstract*—Keyphrases are essential meta-data that summarize the contents of an instructional video. In this paper, we present a domain independent, statistical approach for automatic keyphrase extraction from audio transcripts of video lectures. We identify new features in audio transcripts, that capture key patterns characterizing keyphrases in lecture videos. A system for keyphrase extraction is designed that uses a supervised machine learning algorithm, based on a Naive-Bayes classifier to extract relevant keyphrases. Our extensive experimental studies show that our system extracts more relevant keywords than existing approaches. The paper also evaluates the performance of the proposed keyphrase extraction method for different categories of lectures. The extracted keyphrases are used further as features for automatic topic based segmentation of the video lectures. This process of automatic keyphrase extraction and segmentation results in a section-wise annotated video lecture which can be effectively viewed in a lecture browser.**

***Index Terms*—Automatic keyphrase extraction, meta-data extraction, lecture browser, segmentation, video lectures**

I. Introduction

Universities around the world are increasingly using multimedia based instruction to augment classroom learning. Several major universities have taken one step further, by making portions of their courses in digital form available to the public over the Internet. Video lectures in university channels, hosted on video sharing websites have become immensely popular, garnering millions of viewers. The proliferation of such content has led to increasing research into developing smart lecture browsers, that improve the overall learning experience of the student.

To improve the lecture browser's utility to students, video lectures are often accompanied by lecture notes, slides or syllabus books. A few advanced lecture browsers synchronize text [1] with lecture video and also allow search within the transcript. However, these systems are of little use if for instance, a student wants to quickly scan the contents of a particular lecture among a series of lectures. To facilitate such browsing, some form of lecture summary needs to be made available. A possible solution is to display lectures with section-wise annotations or titles provided by the instructor or students [2]. While the benefits of such an approach are high, the process is tedious, and unviable given the large repositories, long durations (40-60 minutes on average) and sequential nature of such lectures. We note that user-invited tagging or annotation is highly prone to errors, due to variations in levels of academic sophistication and subject knowledge amongst users. Again, the large size of e-learning repositories today makes this approach cumbersome. Thus the major proportion of video lectures produced come untagged, providing very little skimming information to the viewer.

In this paper, we introduce a system that can automatically generate and display section-wise annotations using lecture transcripts. Existing systems for information retrieval and summarization from spoken documents rely on a variety of lexical, structural, prosodic and disfluency features [3] or dictionary-based semantic processing [4]. However, the performance of such techniques when scaled to large corpora or their suitability to the domain of lectures have not been well studied. Our approach uses a simpler keyphrase-based annotation technique, which functionally strikes a middle ground between detailed annotation and basic video tagging.

Keyphrases are commonly identified in technical documents such as journal articles and technical reports, to highlight key topics and concepts. Similarly, when applied to lectures, keyphrases can give important summarizing information and can allow students to identify lectures that correspond to their learning requirements. To achieve this, we need an automatic keyphrase extractor and segmenter. Automating the process of keyphrase extraction and segmentation in lecture transcripts presents several challenges. The lack of proper content structuring, imperfections in language usage, conversational nature and variations of style across domains are common features of lectures that hamper performance of information retrieval systems.

In this paper we address these issues in the specific context of classroom lectures ordinarily seen in university video channels and webcasts. Non-classroom discourse (technical talks, guest lectures, conference presentations etc.) is not considered. Class room lectures range across varying levels of structuredness, and can span multiple domains. The longer duration of these lectures help provide more information for segmentation and keyphrase extraction. Variations in the delivery styles, classroom interactions, etc also make it a much more interesting area to study.

We show that a machine learning based approach, coupled with effective feature extraction specifically tailored for the lecture domain can address challenges in keyphrase extraction from lecture transcripts.

The following are our specific contributions in this paper

- We make use of a corpus of lectures for training the classifier. To the best of our knowledge, such supervised techniques have not been applied to lecture transcripts. To address the challenges of mining from transcripts with limited structure, we propose a set of features based on commonly observed characteristics in lecture speech.
- To perform segmentation within the transcript, we deviate from traditional text segmentation methods and instead use the automatically extracted keyphrases as features for identifying topic cohesion.
- Empirical evaluation of our techniques on a large corpus of lectures and comparing our techniques with existing keyphrase extraction and automatic segmentation solutions.

We will discuss the existing work and contrast our methodology in the related work section (Section 2). Section 3 describes the architecture of our system. Sections 4 and 5 explain the keyphrase extraction module in detail and show experimental results respectively. In section 6, we show how the extracted keyphrases can be used for topic segmentation of lectures, and then present corresponding experimental results by comparing with existing segmentation algorithms.

II. Related Work

Several methodologies have been proposed to extract keyphrases from structured documents like scientific articles, journals, etc. However very little work has been done to extract keyphrases from unstructured documents like audio transcripts. Automatic keyphrase extraction from written documents has been a subject of extensive study and research for well over a decade. Keyphrase extraction is often modeled as a classification problem that requires application of machine learning techniques. The popular KEA system [5] uses supervised machine learning to build a domain-specific classifier model from a training corpus. The trained classifier (Naive Bayes) then extracts keyphrases in new documents ranked according to feature values. Classic tf-idf and relative position of first occurrence of candidate phrases within the document are basic features employed by KEA. The GenEx system developed by Turney [6] uses a genetic algorithm operating with several features such as n-gram size and frequency measures, and gives comparable performance. Hulth [7] suggested use of linguistic features such as noun phrases and part-of-speech information to improve keyphrase extraction. Improvements to KEA when a domain vocabulary is available has been studied by Medelyan and Witten [8]. More recently, Kim and Kan [9] analyzed common linguistic patterns in keyphrases to develop regular expressions for candidate phrase extraction. The work also assesses performance of a large number of features in both supervised and unsupervised extraction algorithms. Our keyphrase extraction system draws from KEA, and incorporates improvements specific to lectures motivated by work in [7], [8].

Besides machine learning, keyphrase extraction systems that rely on information retrieval, NLP and graph-based techniques have been proposed. Suzuki et al., [10], used an encyclopedia with 141 different domains and a large corpus of news articles to construct feature vectors for each domain. Document sections were assigned keyphrases from the feature vector that was found most similar to that section. Matsuo and Ishizuka [11] showed how co-occurrence probability distributions of terms can be used to obtain keyphrases. The method does not use a large external corpus but still manages to give results comparable to algorithms that use tf-idf. The suitability of these approaches when applied to spoken audio documents such as lecture transcripts remains to be studied.

Keyphrase extraction from speech transcripts has not received much attention when compared to written documents. Several lexical features pertaining to written documents are not effective in the context of unstructured speech transcripts. Plas et al., [12] explored usage of linguistic tools such as Wordnet to extract keyphrases from transcripts in the multiparty meeting corpus. Liu et al., [13] also studied meeting transcripts, comparing keyphrase extraction performance of an unsupervised tf-idf based method and a graph based approach. Applying techniques used with other spoken data to lectures is not possible, as almost all approaches are domain centric and exploit very specialized features.

Very limited work can be found on information retrieval from classroom lecture transcripts. Haubold [14] studied index terms that appear in an ASR generated lecture transcript, and presented several ways to visualize them. The work also explored techniques to match lectures to textbook chapters and to cluster lectures using index terms. The goal is to provide quick browsing in a series of lectures, rather than in a single lecture. It also assumes the availability of a high-level domain and sub-domain information (e.g. Computer Science - Data structures). If such classifications are unavailable, finding out the text book for comparison will be difficult. For each domain and sub-domain category, at least one text book has to be maintained and there is a dependency on external sources of data. Yamamoto et al. [15] used the entire course textbook to build a lecture segmentation and information retrieval system. The system makes use of a vector space model constructed from the textbook sections and tries to match vectors obtained from transcripts with the textbook vectors. While presence of text book material is useful for keyphrase extraction, this is often not the case and therefore the approach cannot be generalized to all classroom lectures.

Although segmentation of spoken documents is a relatively new area of research, several different approaches have already been applied. The TDT initiative [16], attempts to segment broadcast news streams using corpus based learning of features. Halliday and Hassan's lexical cohesion theory [17] is the basis for almost all domain-independent techniques for

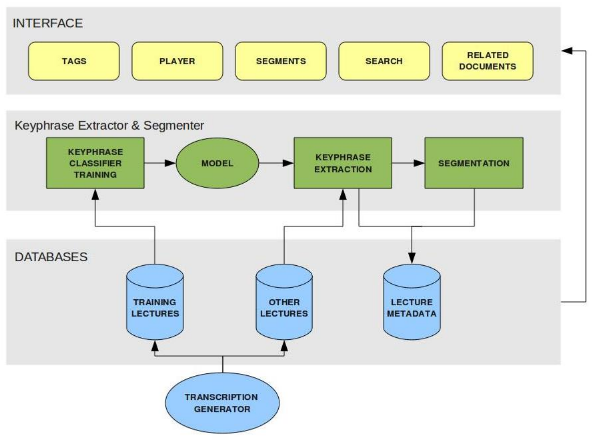


Figure 1. System Architecture

linear text segmentation. The theory states that topic coherence across text segments can be measured based on the extent of similarity in the vocabulary used in each segment. Alternately, this means that large shifts in vocabulary usage indicates a topic boundary. Researchers have used several techniques to select segments of text, making comparisons using varied lexical features. One of the earliest approaches, by Hearst [18], used sliding windows of fixed length where text in adjacent windows were compared for topic cohesion. A vector space model with word stems as features was used to represent the windows. In our system we use a modified version of this algorithm.

III. System Architecture

Previous sections introduced the problem of automatic keyphrase extraction and showed why existing work is insufficient for classroom lectures. To be able to automatically extract keyphrase and segment lectures, we have developed an annotation system which is described in this section.

Figure 1 shows the top level architecture of the system. The transcript generator (human or ASR system) produces transcripts of lectures. The transcripts are stored in the digital repository and is assumed to be part of an existing support framework.

The system consists of three major components as follows:

A. Databases

The output from the transcript generator i.e. the lecture transcripts are stored in the database along with the lecture videos. Following keyphrase extraction and segmentation steps, the generated lecture metadata is saved as XML files in the database for use by the lecture browser.

B. Keyphrase Extraction

The keyphrase extraction component uses the database of training lectures to first build a keyphrase classifier model. Once trained, the classifier is run on untagged lectures for keyphrase extraction. The generated keyphrases are passed on to the segmentation component.

C. Segmentation

The segmentation system uses keyphrases as features to detect topic boundaries in a lecture in order to generate topically cohesive segments. A variant of Hearst's classic Text Tiling algorithm is used to perform segmentation in lecture transcripts. Each segment in the lecture video contains the start boundary, terminating boundary and the associated keyphrases. This information is used to display the annotated and segmented video in the lecture browser. A detailed description of this subsystem is given in Section VI.

D. Interface

Figure 2 shows a screen shot of our lecture browser. The lecture browser uses the automatically generated metadata, and presents it in a simple user-friendly interface. The combination of segmentation and keyphrase annotation is a powerful summarization and skimming tool. Salient features and functionalities of the browser are as follows:

- Topically cohesive sections in the lecture annotated with corresponding keyphrases are shown alongside the video. This enables the student to easily judge how well a lecture corresponds to his/her learning requirements.
- The interface allows the student to quickly jump to any required section of the lecture. Attention can be concentrated on relevant sections of the lecture, while other sections may be skimmed or skipped.
- The search feature allows users to find not just relevant lectures, but also relevant sections in the lecture. Comparison of multiple lectures covering similar topics is also made possible.
- Section-wise lecture transcript to video association is also available. Users can view the transcript corresponding to the specific section of the video if required.
- Guided by keyphrases, students can quickly correlate the video lecture with course textbook material . The browser also allows quick review for students wishing to revise.

IV. Keyphrase Extraction

The goal of keyphrase extraction is to generate an optimal set of phrases appearing in the lecture, that best summarizes its content. Generally, keyphrases are either terms spread throughout the lecture (theme phrases) or terms important to particular sections (topic phrases). The extraction algorithm needs to capture both classes of keyphrases to build a good lecture summary.

Keyphrase extraction from lecture transcripts pose several unique challenges, as noted earlier. The defining characteristic of lecture speech is its spontaneity. A majority of the sentences may be incomplete or grammatically incorrect. Also, different speakers have their own unique styles of lecture delivery. For instance, instructors who use presentation slides usually follow a well planned sequence or scheme, and therefore such lectures tend to possess good topic structuring. Lecture characteristics can vary significantly based on the subject domain. For example, extraction results for lectures in engineering management will usually be better than, say for a set of calculus lectures which carry very little keyphrase information due to increased board work and the usage of specialized language. Lectures may not possess proper introductions or conclusions, and due to its conversational style often contain "non-academic speech" such as anecdotes, digressions and humor.

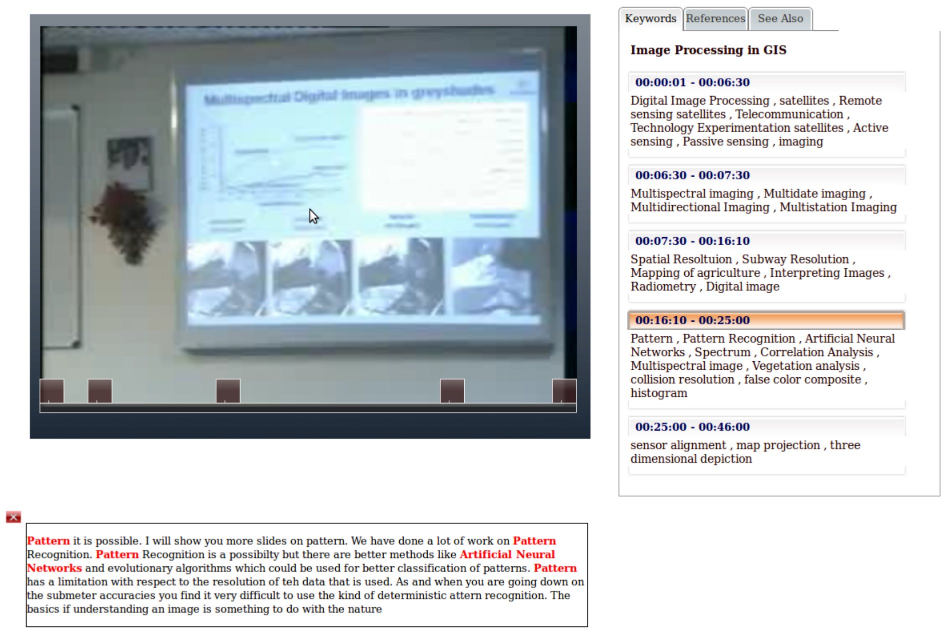


Image Processing in GIS

00:00:01 - 00:06:30
Digital Image Processing , satellites , Remote sensing satellites , Telecommunication , Technology Experimentation satellites , Active sensing , Passive sensing , imaging

00:06:30 - 00:07:30
Multispectral imaging , Multidate imaging , Multidirectional Imaging , Multistation Imaging

00:07:30 - 00:16:10
Spatial Resoltuion , Subway Resolution , Mapping of agriculture , Interpreting Images , Radiometry , Digital image

00:16:10 - 00:25:00
Pattern , Pattern Recognition , Artificial Neural Networks , Spectrum , Correlation Analysis , Multispectral image , Vegetation analysis , collision resolution , false color composite , histogram

00:25:00 - 00:46:00
sensor alignment , map projection , three dimensional depiction

Pattern it is possible. I will show you more slides on pattern. We have done a lot of work on **Pattern** Recognition. **Pattern** Recognition is a possibilty but there are better methods like **Artificial Neural Networks** and evolutionary algorithms which could be used for better classification of patterns. **Pattern** has a limitation with respect to the resolution of teh data that is used. As and when you are going down on the submeter accuracies you find it very difficult to use the kind of deterministic attern recognition. The basics if understanding an image is something to do with the nature

Figure 2. Screenshot of the Lecture Browser

More difficulties arise if the transcripts are obtained from an ASR system, as extraction performance is directly influenced by the Word Error Rate (WER). The impact of transcription errors can be reduced by using an external corpus of expected terms as shown in [14]. Currently this paper focuses on keyphrase extraction based on manually generated transcripts, and a study of the impact of ASR errors will be part of future work.

Our system follows a supervised machine learning approach to keyphrase extraction, similar to KEA. We define features which takes into account the characteristics of spoken lectures, which will be used by the machine learning system for classifying keyphrases. The details of the keyphrase extraction system is described in the following sections.

A. Proposed Features

Features in the context of machine learning algorithms, are parameters used to characterize and classify input data. Our studies on keyphrase extraction from lecture transcripts show that features used for keyphrase extraction from written documents, are insufficient when applied to lecture transcripts, even with domain knowledge, due to the challenges discussed earlier. We therefore define a new set of features (Dispersion, Local-Span and Cuewords) for classroom lectures, in order to improve accuracy of keyphrase extraction. In addition, suitable features used with existing keyphrase extraction systems have been identified. Both these set of features are described in detail in the following paragraphs.

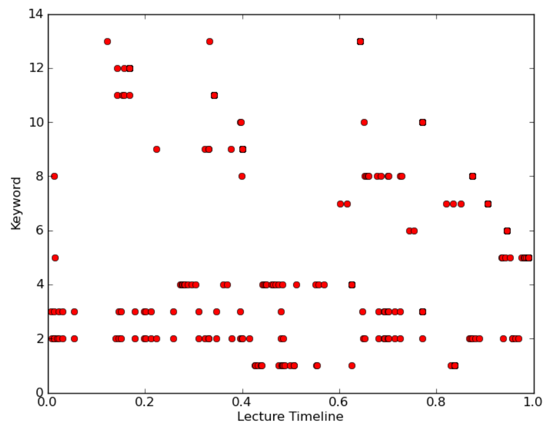
1) Dispersion and Local-Span

One of the important tasks of keyphrase extraction methods in classroom lectures is to identify both the topic keyphrases and subtopics which are specific to segments. The timeline spread of the keyphrase in the lecture is a key factor in determining both these types of keyphrases. This is illustrated in Figure 3. The dispersion plot in Figure 3a shows that keyphrases are more spread out than non keyphrases, shown in the second plot in Figure 3. Topic keyphrases have a more uniform spread, while segment specific keyphrases are more clustered in some regions. The non-keyphrase terms have no proper configuration or recognizable pattern in their dispersion plot. The terms are either too local, or form unorganized clusters. Some exceptions do exist, but such terms maybe eliminated by tf-idf and the other features.

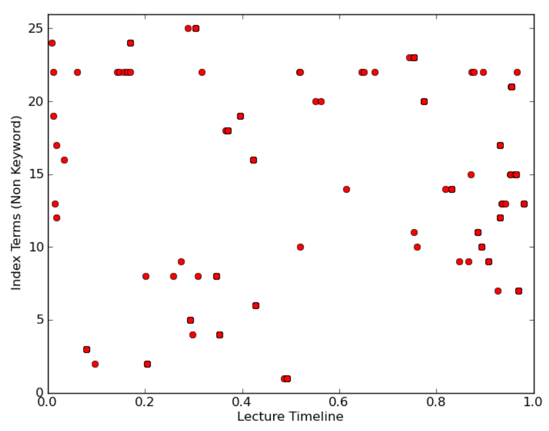
To identify topic phrases we define a feature “Dispersion“ which is a measure of the spread of the keyphrase in the lecture. Dispersion captures phrases that have both high spread across the lecture and have a high frequency of occurrence. Such phrases generally capture the main topics of the lecture. Dispersion is determined by calculating the intervals during which the phrase appears in the document. For a high dispersion the length of the interval is small and the number of intervals is large.

Let $[h_1, h_2, ..h_m] \in I$ where, I is the set of m intervals of term t. Each interval h_i is calculated by finding the difference between adjacent occurrence of the phrase, ie. $h_i = p_{i+1} - p_i$, where $(p_1, p_2, ..p_n)$ are the word positions of phraseterm in the lecture. If the terms occur very close to each other, ie occuring small range clusters, they are recognized as a single position. This step is done to reduce variance when the term occurs in such a cluster. Dispersion is defined as the ratio of the number of intervals where the term occurs to the variance of the interval set I i.e.,

$$Dispersion = \frac{m}{var(I)} \text{ where,}$$
$$var(I) = \sum_{k=1}^{m} (\mu - h_i)^2 \text{ and } \mu = \frac{\sum_{k=1}^{m} (h_i)}{m}$$



(a)



(b)

Figure 3. (3.a) Lexical Dispersion Plot for Keyphrases in a lecture (3.b) Lexical Dispersion Plot for Non-Keyphrases in a lecture

2) Local-Span

Phrases that are sub-topics, are often clustered in few specific segments of a lecture. For instance the word ”sorting” will occur throughout the lecture which is on Sorting Techniques. However the subtopics like “Quicksort“, or allied terms like “divide and conquer“ usually occur in some parts of the lecture where that particular subtopic or method is being discussed. These phrase mostly have low dispersion values but are important to the lecture or to the segment of the lecture. We propose a feature called “local-span“ to identify such locally clustered phrases.

To calculate local-span, each lecture transcript is divided in to ’m’ overlapping segments. The word count for each segment is the number of times the term occurs in a time span of τ minutes. Let c_{avg} be the average occurrence of term ’t’ across m segments i.e, $c_{avg} = n/m$. Let c_i be the count of term ’t’ in segment s_i. Local-span, $\lambda(t)$ is therefore defined as follows:

$$\lambda(t) = max(\sum_{i=1}^{m}(c_i - c_{avg}))$$

Since dispersion and local-span capture essential properties of keyphrases, they can also be used to improve the video lecture search. Lectures in which the search phrase has a better dispersion value are more likely to be relevant, as the phrase can be the topic of the lecture. Similarly local-span can be used to output the relevant segments matching the search query.

3) C-Value

C-Value [19] by Frantzi et al., is useful to identify nested collocations thereby improving the accuracy of keyphrase extraction. For e.g. in the phrase ’data mining’, the terns ’data’ and ’mining’ are meaningful. However ’data’ and ’mining’ can be considered as keyphrases for a lecture only if the lecture explains or defines each of them individually. C-Value combines linguistic and statistical information to extract the keyphrase, emphasis being placed on the statistical part. This measure is built based on the following parameters:

- The total frequency of occurrence of the candidate phrase in the corpus
- The frequency of candidate phrase as a part of other longer candidate phrases
- Number of these longer candidate phrases
- The length of the candidate phrase (in number of words).

Using these parameters, C-Value is calculated as shown below.

$C - Value = log_2|len(a)|.f(a)$, if a is not nested, else
$C - Value = log_2|len(a)|.(f(a) - (\frac{1}{P(T_a)}. \sum_{b \in T_a} f(b)))$

where a is the candidate string,
len(a) is the length of candidate phrase (in number of words)
f(a) is the frequency of occurrence of phrase ’a’ in the corpus
T_a is the set of extracted candidate terms that contain a,
$P(T_a)$ is the number of these candidate terms
The negative effect of higher order frequency term reduces the C-Value of phrases that appear only as substring of higher order phrases.

4) Cuewords

While frequency of occurrence of the term is a useful parameter to identify keyphrases, it can be observed in many video lectures that there are phrases that are spoken very rarely, though being highly relevant to the lecture. Such phrases cannot be easily detected using the previously defined features due to limited occurrence. However they are observed to follow cuewords like ’called as’, ’defined as’ etc. Based on the normalized frequency of occurrence of each such term, their relevance is computed.

5) TF-IDF

TF-IDF is a common measure used to determine the importance of a term to a document. TF-IDF is the product of the term frequency of term t in a document d, and its inverse document frequency [20]. $tf(t, d)$ is the frequency of the term t in the lecture d.

$idf(t) = log_2\left(\frac{D}{df(t)}\right)$, where $df(t)$ is the document frequency i.e, the number of documents in the global domain corpus containing the term t, and D is total number of documents in the domain corpus.

$TF - IDF$ for term t in document d is hence defined as $tf - idf(t,d) = tf(t,d) \times idf(t)$

Since $idf(t)$ is high when the term t occurs in few documents, term can be assumed to be important to those documents, and is not a stop word. Usually terms with high tf-idf values occur many times in fewer documents, while terms that occur more frequenctly in most documents have lower tf-idf. Thus, the higher the value of tf-idf, the more important the term is, and the higher the probability of the term being a keyphrase.

The features we have defined captures most of the identifying properties of keyphrases in video lectures.

B. Generating the Feature Table

The next step in the pipeline is to compute the feature values for the candidate phrases in the lectures. A feature table is built, for use in the training and testing module of the keyphrase extraction system. Before keyphrase extraction, the transcript needs to be preprocessed, after which feature extraction and discretization is performed on the preprocessed transcript.

1) Text Preprocessing

Removing unnecessary information from the transcript is an essential step in processing noisy data and preventing wrongly identified keywords. Basic text filtering steps include punctuation processing, stemming, ngram formation, and stop-gram removal. As the size of the transcripts tends to be large, as an additional filtering step, we exclude all terms which appear once in the transcript [5].

The first step is punctuation processing where punctuations that appear in manual or perfect transcripts are removed. Hyphens are replaced by spaces, for better recognition of bigrams. Dots are kept as such, as it helps in ngram formation, or removed if an abbreviation can be detected (close proximity dots). Next stemming is peformed to reduce terms to root forms since it allows the classifier to treat similar words in different forms of speech as the same (e.g. compiling and compiler). The Porter Stemmer [20] has been found most suitable for our purposes. After stemming the next step is Part-of-Speech (POS) Tagging which is required to filter the phrases during N-gram extraction based on its part of speech classification. Then N-grams are extracted using a selected set of linguistic filters. Based on the analysis done for 40 lectures, only phrases that satisfy these linguistic filters have high probability of being a keyphrase.

2) Feature extraction and discretization

The feature values explained earlier are calculated for the given transcript. With the exception of IDF, all other feature values can be calculated from the single transcript alone. The Naive-Bayes algorithm works best for feature ranges rather than feature values are given. It then assigns probabilities for each range. Existing methods for data discretization include the Entropy-MDL method, and divisions based on equal frequency or width. Discretization using Entropy-MDL method was found to provide best results and hence it is the chosen method in our system. Determining discretization points is a step before training.

After determining the discretization points of each feature, each term in the feature table is modified to represent the feature ranges, and the final feature table is generated. This is sent to a classifier to classify the terms as keyphrases or nonkeyphrases.

C. The Learning Algorithm

We use a Naive-Bayes classifier for classifying keyphrases from other terms. The Naive-Bayes classifier is a probabilistic model that uses Bayes' theorem and assumes feature independence. Naive Bayes is a simple yet effective statistical machine learning tool, well suited for text classification purposes [20]. As it assumes feature independence, simple multiplication of the probability distribution of each feature can be done, given the value of the class variable.

$$p(C|F_1, ..., F_n) = \frac{p(C)}{Z} \prod_{i=1}^{n} p(F_i|C)$$

where C is the class variable,$F_1, ..., F_n$ are the n features, $p(C|F_1, ..., F_n)$ is the conditional distribution over C , and Z is a scaling factor dependent on $F_1, ..., F_n$

D. Training

A key component of the keyphrase extraction system is the training module in which the supervised machine learning algorithm (Naive-Bayes learner) is given sample transcripts with manual keyword information. The text transcript is first converted into feature vectors, which are given as input to the training module. The classifier then tries to create the most probable feature profile for keyphrases using feature statistics of manual keywords. A model is built that stores the training information, and used later for testing.

A set of 25 lectures from different domains is fed into the trainer, and the probability information of each feature is stored. We limit the training to 25 lectures empirically, as improvements in the system performance were insignificant with further addition of lectures. We use different domains, to account for any feature probability variance in different domains. We also train the system with different kinds of lectures (introductory, descriptive or analytical).

E. Testing of keyword extraction

The testing module is used to identify and extract keywords from new, untrained lectures. Each transcript is converted into a feature vector, and the feature vector is fed into the classifier. The algorithm then, uses the probabilistic information from the learned keyphrase profile, and computes the likelihood of the term being a keyword. The terms so identified are ranked by the highest probability. Feature ranking is also used in cases of equal probabilities. We also include segment size as a feature ranking criteria, as terms uttered in the entire lecture are more probable keywords, but this criteria is only applied after segmentation is performed.

V. Evaluation

We have described the details of our keyphrase extraction system in the previous sections. We now evaluate the perfor-

Table I
COMPARISON : PROPOSED SYSTEM VS KEA VS C-VALUE

Lecture Name	**Proposed System**			**KEA**			**C-Value**		
	Precision	**Recall**	**F-Score**	**Precision**	**Recall**	**F-Score**	**Precision**	**Recall**	**F-Score**
Lecture 1	0.778	0.500	0.609	0.400	0.118	0.182	0.444	0.286	0.348
Lecture 2	0.867	0.52	0.65	0.433	0.232	0.302	0.278	0.385	0.323
Lecture 3	0.556	0.417	0.476	0.322	0.112	0.166	0.556	0.417	0.476
Lecture 4	0.667	0.462	0.545	0.143	0.200	0.167	0.333	0.231	0.273
Lecture 5	0.778	0.500	0.609	0.340	0.230	0.274	0.556	0.357	0.435
Lecture 6	0.750	0.545	0.632	0.480	0.234	0.315	0.375	0.273	0.316
Lecture 7	0.750	0.800	0.774	0.456	0.340	0.390	0.375	0.400	0.387
Lecture 8	0.500	0.667	0.571	0.432	0.367	0.397	0.375	0.500	0.429
Lecture 9	0.571	0.615	0.593	0.400	0.167	0.236	0.357	0.385	0.370
Lecture 10	0.455	0.714	0.556	0.143	0.200	0.167	0.273	0.429	0.333
Lecture 11	0.800	0.400	0.533	0.340	0.250	0.288	0.800	0.400	0.533
Lecture 12	0.714	0.625	0.667	0.484	0.234	0.315	0.417	0.455	0.435
Lecture 13	0.667	0.500	0.571	0.443	0.340	0.385	0.444	0.333	0.381
Lecture 14	0.438	0.636	0.519	0.432	0.367	0.397	0.250	0.364	0.296
Lecture 15	0.50	0.63	0.56	0.320	0.223	0.263	0.250	0.222	0.235
Lecture 16	0.500	0.417	0.454	0.330	0.230	0.271	0.400	0.235	0.296

mance of our keyphrase extraction system by investigating the following aspects:

- Improvement in keyphrase extraction over existing systems.
- Impact of different features in automatic keyphrase extraction
- Impact of the number of potential keyphrases on the accuracy
- Performance across various classes of lectures

A. Experimental Setup

Due to the requirement of large corpora of lecture transcripts for training and testing, we had to compile and build our datasets from various sources. Our major source is a large corpus of around 300 manual transcripts of lectures in computer science, obtained with permission from NPTEL [21]. These lectures are aimed at undergraduate students, and cover a wide range of subjects in computer science. Most lectures are part of a semester-long series comprising 30-40 lectures. The rest of the dataset is comprised of publicly available lectures for graduate and undergraduate students from university websites [2], [22], that included manual lecture transcripts. These lectures primarily belong to the computer science domain and includes subjects like Algorithms and Data Structures, Computer Graphics, Machine Learning etc. All lectures in the dataset are held in a classroom environment. The duration of a lecture is between 40 to 60 minutes on an average, with transcripts containing 8000 to 14000 words.

For each lecture in the development set, a human annotator manually extracts keyphrases, which is taken to be the gold standard [5]. We collected manual keyphrases for each of the lectures from 3 users, one of them being a domain expert. Keyphrases identified by the domain expert is given higher priority, followed by phrases identified by atleast 2 users. The number of keyphrases extracted from a lecture varies between 10 to 25. It must noted that while the majority of literature in the field consider human picked keyphrases as the gold standard [5], there is no real consensus on managing variations arising from subjectivity of judges. We also categorize the lectures approximately based on the delivery style and content covered as introductory, descriptive, illustrative and analytical. We do so inorder to identify the impact of the lecture style on the keyphrase extraction system. For evaluation of keyphrase extraction performance we use the traditional Precision, Recall and F-score metrics.

Given a brief description of the experimental setup the following paragraphs will describe and discuss about experiments and their results.

1) *Comparison of our system with existing systems*

We first compared the performance of our keyphrase extraction system with existing systems like KEA and a system that primarily used C-Value as its feature. The different systems extracted keyphrases from a selected number of lectures and the extracted keyphrases evaluated for precision and recall. Table I shows the results of this experiment. From the results, it was observed that the F-Score of our system is 46 percent higher than KEA and 33 percent better than the system using C-Value. This is because, the fundamental features in both systems use frequency of occurrence of a phrase [5]. However that is insufficient to capture many phrases in an unstructured domain like lectures where phrases can occur less frequently yet be highly relevant. Integration of the features like cuewords, dispersion and local-span in our system ensures that even less frequent yet highly relevant phrases are also extracted. This firmly establishes the improved performance of our system.

2) *Impact of different features in automatic keyphrase extraction*

The next set of experiments evaluate the effectiveness of our technique. We first evaluate the role, the different features play in keyphrase extraction.

Table II shows keyphrases extracted and the impacting feature (in bold). It is observed that dispersion captures the major topics of the lecture. Here the lecture is on ’binary trees’ and hence ‘binary tree’ has been extracted using dispersion. Cuewords also help in identifying theme phrases. Using local-span we can get the subtopics or aspects of the main topic. Here for instance a part of the lecture discusses “height of

Table III
NUMBER OF KEYPHRASES PER LECTURE VS F-SCORE

Range of keyphrase count	Average Precision	Average Recall	Average F-Score
<=9	0.713	0.475	0.568
>=10 and <=19	0.507	0.559	0.524
>=20 and <=30	0.422	0.607	0.497

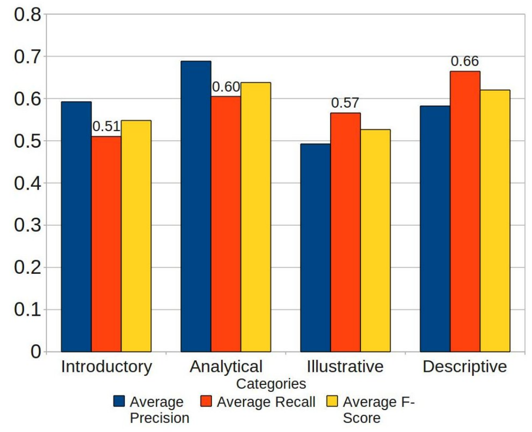


Figure 4. Evaluating the Keyphrase Extraction for different lecture categories

a tree", and hence that is identified as a keyphrase. C-Value and tf-idf features complement the other features by capturing the frequency of the keywords. The diverse roles played by the different features amounts to significant contribution in improving the performance of our system.

3) Impact of various factors on keyphrase extraction accuracy

Now we study the effect the following have on the accuracy of automatic keyphrase extraction from lecture videos

1) Number of potential keyphrases in the lecture
2) Lecture category

Table III shows the precision, recall and F-Score for different ranges of potential number of keywords in each lecture. It is interesting to note that precision is high when the number of keyphrases are lower, while recall is better when the number of keyphrases are higher. We believe this could be primarily due to the type of lecture, rather than actual number of keyphrases which is impacting the results thus. We see this more clearly in Figure 4 where the average precision,recall and F-score for each category of lectures is presented.

We see that the F-Score is high for analytical and descriptive lectures when compared to other categories.This is because, a particular topic is described in more detail in analytical and descriptive lectures, and we observe that this is directly proportional to the dispersion of the keyphrase.

However in an introductory lecture, the features such as dispersion, local-span, tf-idf and c-value yield lower scores, primarily since the lecturer does not dwell on a specific topic for much time. Precision for illustrative lectures is less when compared to other categories of lectures. This is because, illustrative lectures normally use analogy of real world entities to explain the topic, which we term illustrative phrases. Though illustrative phrases usually are not keyphrases for a lecture, they tend to occur frequently. This apparently leads to higher dispersion and frequency of occurrence of illustrative phrases. In such scenarios, our system will capture such illustrative phrases as key phrases leading to lower precision.

VI. AUTOMATIC LECTURE SEGMENTATION

A. Introduction

To build a successful lecture browser, we need to improve the speed and ease with which a user can find what he is looking for. Besides using keyphrases which enable summarization and search, the learning experience can be further improved by guiding user navigation towards relevant sections in the lecture. This functionality can be achieved by dividing the lecture into topically cohesive segments. Segmentation of a lecture maybe performed using its video track, audio track or transcript. Since ordinary lectures contain only weak cues in video and audio for topic shifts, segmentation in both these mediums is not very effective. Therefore we use lecture transcripts, and the problem becomes purely one of text segmentation.

Segmentation in lecture transcripts is also a challenging problem. Due to the smooth nature of topic transitions seen in lectures, topic boundaries tend to be fuzzy and difficult to detect. The issues with spontaneous speech and ASR errors discussed for keyphrase extraction, also hold for segmentation.

We use a variant of Hearst's classic Text Tiling algorithm [18] to perform segmentation in lecture transcripts. The motivations for our choice of Text Tiling over more recent, advanced algorithms are as follows:

- TextTiling is simple, both conceptually and implementation wise.
- Our studies showed that TextTiling is well suited for the coarse nature of segmentation required by the user (details in the Related Work section).
- TextTiling runs in linear time. Compared to other algorithms, this reduced cost becomes important when the system is used on large-scale digital repositories.

The original TextTiling algorithm [18] computes lexical scores between text segments, using word stems (other than for stopwords) as features. Instead we reformulate the algorithm to use automatically generated keyphrases for that lecture as features. We base our modification on the idea that changes in usage of keyphrases in a lecture strongly correlate with topic shifts. This technique is in many ways similar to how human beings approach the task of manual segmentation. For example, when volunteers were asked to view a lecture and later find topic segments in the transcript, it was observed that the majority relied on keyphrase occurrence patterns in the text. We show experimentally that keyphrases provide the algorithm with enough resolving power to achieve required accuracy levels for coarse segmentation. An added advantages of using only keyphrases for features, is that the algorithm runs even quicker with the reduced feature space.

B. The TextTiling Segmentation Algorithm

TextTiling uses sliding windows to locate possible boundaries in text. A text window moves through the transcript at fixed intervals assumed to approximate a typical boundary.

Table II
KEYPHRASES IDENTIFIED USING EACH FEATURE FOR A LECTURE ON "COMPLETE BINARY TREE" IN DATA STRUCTURES SUB-DOMAIN

Manual Keyphrase	Keyphrase from each feature	Keyphrase result of the system
trees, ordered tree, binary tree, complete binary tree, abstract data type, ADT for binary tree, unbounded branching, root node, degree of node, leaves of the tree, nodes of a tree, internal node, height of the tree, left sub tree, right sub tree, decision tree	**CueWords:**tree, internal node, ordered tree **Dispersion:** binary tree, complete binary tree, root node, height of the tree **LocalSpan:** height of the tree, left subtree, decision trees, tree on n node, root node **C-Score:** binary tree, complete binary tree, number of leaves, internal node	internal node, tree, left sub tree, tree on n node, complete binary tree, number of leaves, ordered tree, root node, decision tree, height of the tree, binary tree

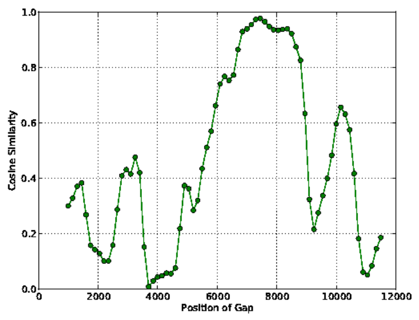


Figure 5. TextTiling window similarity

Boundaries are calculated based on the measure of dissimilarity between adjacent windows. For each window w, any keyphrase t appearing in the lecture is given a window score calculated as

$$S_w(t) = \frac{f_w(t)}{isf(t)}$$

where $f_w(t)$ is the frequency of t in w and $isf(t)$ is the inverse segment frequency of t, which denotes the importance of t in w. $isf(t)$ is analogous to tf-idf and is calculated as $log(N/n(t))$ where N denotes the total number of windows and $n(t)$ stands for the number of windows containing keyphrase t.

A vector space model is constructed for each window whose members are scores calculated for keyphrases as described. Cosine similarity scores are computed for each pair of adjacent windows with corresponding feature vectors V_i and $V_{adj(i)}$ as

$$C_{i,i+1} = \frac{V_i^T V_{adj(i)}}{|V_i||V_{adj(i)}|}$$

When the cosine scores are plotted against the transcript timeline, graphs similar to figure 5 are obtained. As Hearst explains, to obtain better scoring accuracy a moving average based smoothing needs to be done on the scores. The deepest valleys or dips are selected as probable boundaries. The actual number of boundaries is set based on the variation of scores from the mean score ($\mu - n\sigma$).

VII. EVALUATION

To evaluate our segmentation algorithm we use the same corpora as used for keyphrase extraction. Our evaluation strategy parallels the approach in [23]. We compare our results with the classic TextTiling algorithm to show that keyphrases are indeed strong features for segmentation. We also compare our results with Choi's system and a few baseline algorithms.

A. Parameters and Metrics

Parameters required for the segmentation algorithm are determined on an empirical basis. The window size is set to 500 words, which is approximately half the average segment size in a typical lecture containing 8000 words and 4-6 segments. The sliding interval is taken to be 50 words, roughly equal to 20 seconds of lecture speech. Since the granularity of topics varies based on various factors such as domain, instructor's style and lecture length, these parameter need to be varied appropriately according to the dataset chosen.

As in keyphrase extraction manual segmentation done by humans is taken as the gold standard for evaluation. Since metrics such as precision and recall are unsuited to the segmentation task, we use two recently proposed metrics, P_k and $Windowdiff$, now widely used for such evaluations [18]. P_k estimates the probability that two words chosen from the transcript at random, are correctly identified (or not) as falling in the same segment by the algorithm. The window size parameter k is set to 250, which is half the average segment size, as suggested by Beeferman [24] to give appropriate metric values for baseline systems. The Windowdiff metric counts the number of boundaries in a sliding window and penalizes the algorithm if any variation is discovered with respect to the human segmented reference. The metric reduces problems associated with P_k to do with overpenalizing near misses and false negatives, and susceptibility to variations in segment size.

B. Experiments and Results

To evaluate the segmentation technique, we find the level of annotator agreement by having 2 annotators segment a small set of lectures. We gave the annotators a range of values for the number of segments, in order to ensure that both segment at similar levels of granularity. The results are tabulated in Table IV. It is observed that our algorithm consistently performs much better than the existing methods. In addition, since we use only the extracted keyphrases for segmentation, this results in faster segmentation. However it is observed that the algorithm does not perform very well for introductory lectures due to the lack of specific topics in the lecture.

VIII. CONCLUSIONS

In this paper, a lecture browser system that uses automatic keyphrase extraction and segmentation was presented. The paper has shown that a combination of automatic keyphrase extraction and segmentation enhances the functionality of

Table IV
COMPARING SEGMENTATION ALGORITHMS

Lecture Name	Type	Segments						Lecture Size	P	WD
ML1	Reference	1787	4919	6843	8633	9736		10463		
	Our algorithm	1600	2800	4150	5650	8050	9700		0.538461538	0.538461538
	Text tiling Orig	3054	5492	8392					0.711538462	0.711538462
	C99 Choi*	601	3348	4791	5106	8643			0.509090909	0.509090909
ML2	Reference	1327	3214	6210	8299			9087		
	Our algorithm	1600	2950	6500					0.314814815	0.314814815
	Text tiling Orig	2651	5152	6569					0.574074074	0.574074074
	C99 Choi*	1213	2780	6718	7876				0.345454545	0.345454545
ML3	Reference	1020	3208		6070	8734		9403		
	Our algorithm	2050	3100	4150	6400	7450			0.5172	0.5172
	Text tiling Orig	1627	1981	3397	4770	6219	7649		0.456140351	0.456140351
	C99 Choi*	2458	3660	4710	7727				0.672727273	0.672727273
ML6	Reference	1016	3395	4631	5747	7562	8950	9088		
	Our algorithm	1000		4300	6100	7500			0.290909091	0.290909091
	Text tiling Orig	2596	6098						0.725490196	0.725490196
	C99 Choi*	2239	2387	3398	4221	5748	8582		0.375	0.375
ML7	Reference	1219	2491	3340	5100	6860	8186	8925		
	Our algorithm	1300		3400	5600	7000			0.351851852	0.351851852
	Text tiling Orig	1242	1600	1958	3340	6197			0.456140351	0.421052632
	C99 Choi*	492	2198	3450	4625	5632	7974		0.543859649	0.543859649

a lecture browser system. Our experimental evaluation of the keyphrase extraction algorithm shows that our technique performs better than existing techniques. We also evaluated the system for different categories of lectures and we believe that our technique can be enhanced to perform better for introductory and illustrative lectures. The evaluation of the segmentation technique, which uses keyphrases, also performs better than existing methods. We hope to add to this work by identifying features from modalities other than lecture transcripts to capture the content of the lecture. We also plan to use the proposed features to calculate the relevance of a term to a particular document, so as to improve search results.

REFERENCES

[1] J. R. Glass, T. J. Hazen, D. S. Cyphers, K. Schutte, and A. Park, "The mit spoken lecture processing project," in *Proceedings of HLT/EMNLP on Interactive Demonstrations*, pp. 28–29, 2005.

[2] "SEE - Stanford Engineering Everywhere." http://see.stanford.edu/default.aspx.

[3] K. Zechner, "Automatic generation of concise summaries of spoken dialogues in unrestricted domains," in *SIGIR '01: Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pp. 199–207, 2001.

[4] I. Gurevych and M. Strube, "Semantic similarity applied to spoken dialogue summarization," in *COLING '04*, 2004.

[5] I. H. Witten, G. W. Paynter, E. Frank, C. Gutwin, and C. G. Nevill-Manning, "Kea: practical automatic keyphrase extraction," in *DL '99: Proceedings of the fourth ACM conference on Digital libraries*, pp. 254–255, 1999.

[6] P. D. Turney, "Learning algorithms for keyphrase extraction," *Information Retrieval*, pp. 303–336, 2000.

[7] A. Hulth, "Improved automatic keyword extraction given more linguistic knowledge," in *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pp. 216–223, 2003.

[8] O. Medelyan and I. H. Witten, "Thesaurus based automatic keyphrase indexing," in *JCDL '06: Proceedings of the 6th ACM/IEEE-CS joint conference on Digital libraries*, pp. 296–297, 2006.

[9] S. N. Kim and M.-Y. Kan, "Re-examining automatic keyphrase extraction approaches in scientific articles," in *MWE '09: Proceedings of the Workshop on Multiword Expressions*, pp. 9–16, 2009.

[10] F. Fukumoto and Y. Sekiguchi, "Keyword extraction of radio news using term weighting with an encyclopedia and newspaper articles," in *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, 1998.

[11] Y. Matsuo and M. Ishizuka, "Keyword extraction from a single document using word co-ocurrence statistical information," 2003.

[12] L. van der Plas, V. Pallotta, M. Rajman, and H. Ghorbel, "Automatic Keyword Extraction from Spoken Text. A Comparison of two Lexical Resources: the EDR and WordNet," *ArXiv Computer Science e-prints*, 2004.

[13] F. Liu, D. Pennell, F. Liu, and Y. Liu, "Unsupervised approaches for automatic keyword extraction using meeting transcripts," in *NAACL '09: Proceedings of Human Language Technologies*, pp. 620–628, 2009.

[14] A. Haubold and J. R. Kender, "Analysis and visualization of index words from audio transcripts of instructional videos," *IEEE International Symposium on Multimedia Software Engineering,*, pp. 570–573, 2004.

[15] N. Yamamoto, J. Ogata, and Y. Ariki, "Topic segmentation and retrieval system for lecture videos based on spontaneous speech recognition," in *EUROSPEECH-2003*, pp. 961–964, 2003.

[16] O. Perspective and C. L. Wayne, "Topic detection tracking (tdt)," in *In Proceedings DARPA Broadcast News Transcription and Understanding Workshop*, 1998.

[17] M.A.K. Halliday and R. Hasan, *Language, Context and Text: Aspects of language in a social-semiotic perspective*. Deakin University Press, 1989.

[18] M. A. Hearst, "Texttiling: Segmenting text into multi-paragraph subtopic passages," *Computational Linguistics*, pp. 33–64, 1997.

[19] K. T. Frantzi and S. Ananiadou, "Extracting nested collocations," in *COLING*, 1996.

[20] P. R. Christopher Manning and H. Schutze, *An Introduction to Information Retrieval*. Cambridge University Press, 2009.

[21] "NPTEL - National Programme on Technology Enhanced Education." http://nptel.iitm.ac.in.

[22] "MIT OCW - MIT OpenCourseWare." http://ocw.mit.edu.

[23] I. Malioutov and R. Barzilay, "Minimum cut model for spoken lecture segmentation," in *ACL-44: Proceedings of the 21st International Conference on Computational Linguistics*, pp. 25–32, 2006.

[24] D. Beeferman, A. Berger, and J. Lafferty, "Statistical models for text segmentation," in *Machine Learning*, pp. 177–210, 1999.