Portrait of an Indexer—Computing Pointers Into Instructional Videos

Andrew Lamb
Stanford University
andrew.lamb@stanford.edu

Jose Hernandez
Stanford University
josehdz@stanford.edu

Jeffrey Ullman
Stanford University
ullman@cs.stanford.edu

Andreas Paepcke
Stanford University
paepcke@cs.stanford.edu

ABSTRACT

As enrollment in online courses grows, it is becoming increasingly difficult for course staff to support students on an individual basis. We are building an autonomous system that will aid instructors and learners in online education; for example, the system might link a forum post with a relevant section of lecture if it detects the poster is confused, or suggest review materials based on a student's history. As a foundation of this system we developed an indexing module that will detect phrases relevant to the key concepts of each lecture, a task often termed keyword extraction. Most research on keyword extraction has focused on heterogenous corpuses, such as abstracts from scientific journals, while our system operates on lecture videos, which are relatively homogenous. To deal with this new setting, we designed algorithms that incorporate external knowledge, using the entire Wikipedia document collection. To evaluate the keyword extraction module, we annotated an online databases course with keywords chosen by humans.

1. INTRODUCTION

One of the biggest challenges of delivering education with massively open online courses (MOOCs) is the high student to instructor ratio, which is further compounded in courses where students don't arrive at traditional term boundaries. In the extreme, *untended* courses may not have any active teaching staff. Without support, students fend for themselves when they don't understand a concept or need to review for a test. Our team is building a system to autonomously help students enrolled in unattended courses; this will enhance, not replace, human instruction and improve the support for all learners.

Extracting important phrases and concepts both directly aids learners and serves as a foundation for the rest of the system. For example, the system produces a set of keywords that index into segments of lecture videos, allowing learners to quickly find parts of lecture relevant to a topic, similarly to the way a textbook index would be used. These keywords could also be used by the system to autonomously find the lecture segment most relevant to a forum post, or other information retrieval tasks.

This task of forming an index over the lectures in an online course, which we will also refer to as the keyword extraction task, is inherently difficult. A human performing the task must have a deep understanding of the concepts in each lecture, as well as their role in the broader context of the entire course. A 25 minute lecture video might have around 4500 individual words, from which the algorithm must form around 20 phrases - which might consist of more than 1 word - to be summarize the content of the lecture. There is a certain amount of subjectivity inherent in the task, and even there can be differing reasonable interpretations of what should be considered a keyword. When humans select important phrases from a lecture they draw on previous knowledge - for example, in a databases course one might use previous knowledge of functional dependencies to decide that "Armstrong's Axiom" should be one of the keyphrases - and we draw on this strategy to automatically decide on keyphrases.

Many keyword extraction systems are designed for use on large collections of loosely related documents, such as newspaper articles or abstracts from an academic journal. In contrast, our indexing system takes transcripts of lecture videos as input, which presents several interesting challenges. Lectures for a class are tightly linked to each other, have a well-defined sequence, and are usually delivered by only one or two instructors. Many keyword extraction algorithms leverage the fact that documents on very different topics will have mostly disjoint sets of words, which may not work as well in our setting.

We used our system to extract keywords from the content of a online introductory databases course, and then tested for agreement against a gold set of phrases produced by humans. Our first experiment took a statistical approach, selecting words that appeared frequently throughout the class, and disproportionately in certain lectures. We then incorporated lexical information, by only considering phrases that followed certain part-of-speech patterns. Finally, we were able to improve the quality of our index by incorporating external knowledge from Wikipedia pages.

2. RELATED WORK

Much of the research on keyword extraction has focused on leveraging statistical properties of a corpus or single document. Term frequency-inverse document frequency (tf-idf) is a widely used method that weights phrases proportionally to the number of times they appear and inversely proportionally to the number of documents they appear in [9]. Statistical methods that work on a single document have also been proposed, such as selecting keywords that co-occur in sentences with frequent words unevenly [10]. The TextRank system uses sentence co-occurrence data in a graph-based approach, by forming an node for each candidate phrase and an edge between two phrases if they appear together in a sentence. The system then runs PageRank over the induced graph to select keywords [11]. Other methods have combined linguistic and statistical properties of the document by first filtering the set of possible keywords (e.g. only consider noun phrases or adjectives) and then applying frequency based methods [15].

The approaches outlined so far all take an unsupervised approach to keyword extraction. There has also been work on supervised approaches, which classify each phrase by whether it should be a keyword or not. Turney trained a decision tree on different annotated corpora to choose keywords based on features such as the length of the phrase and whether it is a proper noun [17]. Hulth takes a related approach, and finds that adding part-of-speech tags as a feature leads to improved results [7]. The major downside of supervised learning is the expense associated with labeling data, as many types of documents will require a person with domain knowledge to choose keywords. As a result many of these supervised systems use academic journals with author-provided keywords as datasets.

Discerning important terms in the specific setting of instructional videos has also been investigated. This task differs from keyword extraction from a large heterogeneous document collection because of the sequential nature of the videos, the fact that they will be about closely related topics, and the additional audio-visual component. Methods have been designed to combine the statistical properties of the lecture transcripts with cues from the lecture videos, such as the introduction of a new speaker, and evaluated on governmental instruction videos [13]. Keyword extraction from Khan Academy lecture videos has been experimented with, using statistical properties of the lecture transcripts [8].

Finally, there has been research into using Wikipedia's external knowledge for natural language processing tasks such as clustering documents [6] and computing semantic similarity between words [12].

3. PREPARATION OF GOLD INDEX

In order to evaluate our algorithms, we prepared a gold standard index of terms extracted from an introductory databases course. Because this task requires to annotating keywords for a technical, semester-length course, the workers were compensated. Annotators were instructed to mark keywords in the lecture transcripts if they felt the lecture was important for students searching for the keyword. There was no limit to how many keywords annotators could mark in a lecture, and at the end they were asked to rank the top five keywords from that lecture, and were allowed to mark fewer than five keywords in a lecture. Annotators were only allowed to use phrases that appeared in the text as keywords, i.e. they could not generate their own original phrases. The annotators extracted keywords for 81 lecture videos (the course was broken into a large number of short videos), with an average of about 8 keywords per video. We evaluated agreement between humans using Fleiss' Kappa, which generalizes Cohen's Kappa to settings with more than two annotators [5]. The κ is a measure of inter-rater agreement, with range $[-1, 1]$, where 1 means the raters are in complete agreement, -1 complete disagreement, and 0 the agreement expected by chance. We formulate keyword extraction as a binary classification task, where the two categories are "keyword" and "not keyword". Given a set of raters and a collection of lectures $\ell_1, \ldots, \ell_n$, where each lecture contains phrases $p_1, \ldots, p_n \in \ell_i$, we define the set of examples to be classified as the set of lecture-phrase pairs (ℓ, p) such that p appears in ℓ, and p is either a 1-gram (i.e. a word) or (ℓ, p) was tagged as a keyword by at least one rater.

We computed a κ of 0.325 between all human raters in the gold set as a upper bound on the performance we could reasonably expect from any keyword extraction algorithm. The largest pairwise κ between raters was 0.336 and the lowest was 0.309 (note that Fleiss' Kappa is not generally the average of the pairwise Cohen's Kappas).

4. EXPERIMENTS

We implemented different keyword extraction algorithms and measured how closely they agreed with the gold index formed by human annotators.

For each experiment we apply the Porter stemming algorithm to each word in the document and each word in a keyphrase, so there will be a match between phrases if all of the individual stemmed tokens match. In experiments using n-grams as candidate phrases, stopwords were removed from the document before the n-grams were formed, using the stopword list used in the SMART system [16].

4.1 Naive Approach: TF-IDF

Term frequency-inverse document frequency, is defined for each phrase-lecture pair as the product of the number of times the phrase appears in the lecture divided by the logarithm of the number of proportion of lectures the phrase appears in. This is the standard definition of TF-IDF, with each lecture taken as a document. We then rank the phrases for each document by their TF-IDF score in that document.

4.2 Leveraging Linguistic Information

Considering all of the n-grams in the collection of lectures as candidate keywords has the potential to add significant noise. By selecting only certain linguistic patterns for consideration as keywords, it is possible to reduce the size of the candidate set, while still covering most important phrases. We experimented with an algorithm that runs a part-of-speech tagger over the lecture transcripts, and then selects only phrases that consist of an optional number of adjectives followed by one or more nouns. For example, "equality condition" or "XML data". We then run TF-IDF over this reduced candidate set.

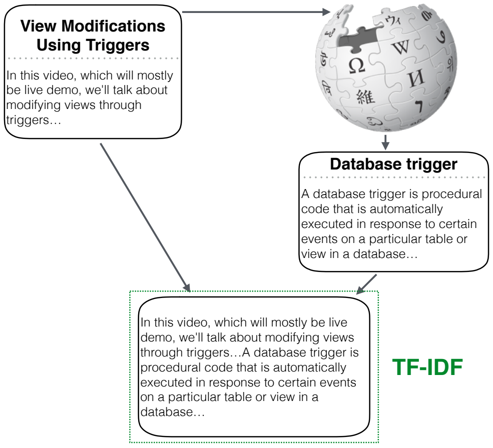
4.3 Adding External Knowledge

Motivated by the intuition that phrases gain importance because of both their role in a document and their semantic meaning in the broader world, we experimented with multiple algorithms that incorporate outside knowledge. Each algorithm integrates Wikipedia as a knowledge source in different ways.

4.3.1 Boosting Documents

The first algorithm attaches each lecture with a closely related Wikipedia page, and then uses the previously described statistical and linguistic techniques to choose keywords from the combined document. Formally, the procedure is as follows. First, for each lecture, the algorithm takes the title of the lecture, removes stopwords, and uses the result as a query to Wikipedia. Then, the best Wikipedia page match is concatenated to the lecture transcript. After attaching a Wikipedia page to each transcript, we run the previously described procedures over the collection of concatenated lectures and Wikipedia pages.

For example, for the lecture titled “View Modifications Using Triggers”, the Wikipedia query will return the Wikipedia page titled “Database trigger”, which is then concatenated to the transcript from the lecture. Then, using either n-grams or adjective-noun phrases as candidate keywords, the algorithm chooses keywords with TF-IDF over the combined document.



In this video, which will mostly be live demo, we'll talk about modifying views through triggers…

A database trigger is procedural code that is automatically executed in response to certain events on a particular table or view in a database…

In this video, which will mostly be live demo, we'll talk about modifying views through triggers…A database trigger is procedural code that is automatically executed in response to certain events on a particular table or view in a database…

Figure 1: The Document Boosting algorithm searches for a Wikipedia page using the title of the lecture, concatenates the result to the lecture, and then runs TF-IDF over the combined document.

4.3.2 Boosting Phrases

This algorithm first creates a list of candidate keywords using adjective-noun phrases. Then, the candidates are ranked by their TF-IDF score summed over all Wikipedia documents. That is, for a phrase p, we define its term frequency $TF_w(p)$ as

$$TF_w(p) := \log\left(\text{number of times } p \text{ appears on Wikipedia}\right)$$

state and its inverse document frequency $IDF_w(p, W)$ for a phrase p and Wikipedia article collection W as

$$IDF_w(p, W) := \log\left(\frac{|W|}{\text{number of documents in } W \text{ with } p} + 1\right)$$

and finally $TF\text{-}IDF_w(p, W)$ is defined as

$$TF\text{-}IDF_w(p, W) := TF_w(p) \cdot IDF_w(p, W)$$

Next, this global candidate ranking is combined with the basic TF-IDF approach described in section 4.1. First, we create a normalized Wikipedia candidate ranking

$$TF\text{-}IDF_{w-norm}(p, W) := \frac{TF\text{-}IDF_w(p, W)}{\sum_{p'} TF\text{-}IDF_w(p', W)}$$

and normalized lecture collection ranking

$$TF\text{-}IDF_{norm}(p, l, L) := \frac{TF\text{-}IDF(p, l, L)}{\sum_{p'} TF\text{-}IDF(p', l, L)}$$

Then, we combine the two normalized rankings to form a final score

$$\begin{aligned} &TF\text{-}IDF_{combined}(p, l, L, W) := \\ &\qquad TF\text{-}IDF_{w-norm}(p, W) + TF\text{-}IDF_{norm}(p, l, L) \end{aligned}$$

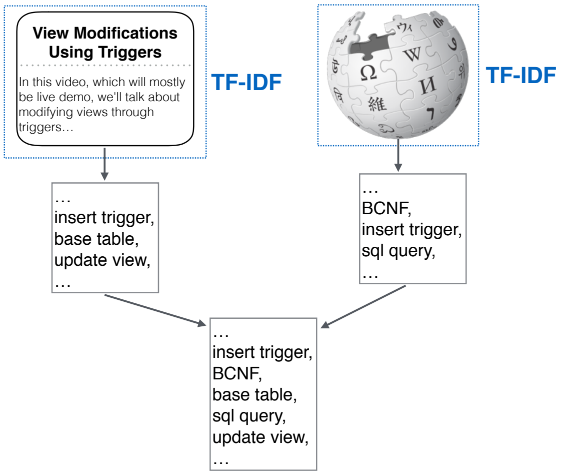
We also experimented with only boosting phrases that were at least two words, based on the intution that longer phrases are often meaningful, but appear infrequently and therefore given low scores by TF-IDF. This is called “Phrase Boosting N-Grams” in Figure 3, and can be thought of as multiplying $TF_w(p)$ by 0 if p is not at least two tokens:

$$TF_{w-ngram}(p) := TF_w(p) \cdot \mathbf{1}\left\{p \text{ is at least two tokens}\right\}$$

A few subtleties deserve further description. First, in our naive TF-IDF approach, a score is calculated for each lecure l and phrase p, while in our Wikipedia calculation a score is only calculated for each p. This is because in this algorithm we are not interested in keywords for every Wikipedia document, only in getting a global sense of the importance of a phrase. This is equivalent to summing over all Wikipedia documents $d \in W$

$$TF_w(p) = \log\left(\sum_{d \in W} \text{number of times } p \text{ appears in } d\right)$$

Second, we take logarithm of the phrase count in the Wikipedia ranking. This seems to produce better empirical results.



In this video, which will mostly be live demo, we'll talk about modifying views through triggers…

Figure 2: The Phrase Boosting algorithm runs TF-IDF over the entirety of Wikipedia and then combines the global ranking with a local ranking for a document.

Algorithm	κ	$\mathbf{P}_{\text{pos}}$	$\mathbf{P}_{\text{neg}}$	**PABAK**
TF-IDF	0.178	0.200	0.973	0.896
TF-IDF with Adjective-Noun Chunks	0.084	0.114	0.968	0.875
Document Boosting	0.180	0.199	**0.974**	**0.901**
Document Boosting with Adjective-Noun Chunks	0.131	0.154	0.971	0.890
Phrase Boosting	0.180	0.203	0.973	0.895
Phrase Boosting N-Grams	**0.191**	**0.213**	**0.974**	0.899

Figure 3

4.4 Summary of Results

As stated in section 3, we computed agreement between humans using Fleiss' Kappa. We evaluate each algorithm by computing Cohen's Kappa agreement between the algorithm and the union of the lecture-phrase pairs in the gold index, i.e. a lecture-phrase pair is classified as "keyword" if any human rater classified it as a keyword, and classified as "not keyword" otherwise. When using a single metric for agreement, such as Cohen's Kappa, paradoxes can arise where one category is much more prevalent than another and the chance-correction term in the agreement metric overcompensates, leading to low values of agreement [4]. In our case, because there are many more possible phrases than keywords in a lecture, most phrases are calculated as "not keyword". To resolve this paradox, we report three additional metrics: agreement on keywords (P_{pos}), agreement on other phrases (P_{neg}) [2], and prevalence-adjusted bias-adjusted kappa (PABAK) [1].

These three metrics are defined in terms of a concordance table. Let a be the number of terms both humans and the algorithms marked as "keyword", "d" be the number of terms both marked as "not keyword", and $n = (a + d)/2$ be their average. Let f_1 be the total number of terms humans marked as "keyword", f_2 be the total number of terms humans marked as "not keyword", g_1 be the total number of terms the algorithm marked as "keyword", g_2 be the total number of terms the algorithm marked as "not keyword", and N be the total number of possible phrases. Then, positive agreement is the number of terms both marked as "keyword" over the average number of terms marked as "keyword"

$$P_{\text{pos}} := \frac{a}{\frac{f_1+g_1}{2}}$$

negative agreement is the number of terms both marked as "not keyword" over the average number of terms marked as "not keyword"

$$P_{\text{neg}} := \frac{d}{\frac{f_2+g_2}{2}}$$

and PABAK is defined as

$$\text{PABAK} = \frac{(2n/N) - 0.5}{1 - 0.5}$$

Kappa values do not have a universally agreed upon interpretation, but values in the range we observe (about 0.15 to 0.3) have been interpreted as indicating "slight" to "fair" agreement. The fact that agreement between humans is only 0.336, and the lowest pairwise Kappa between humans was 0.309 suggests that the keyword extraction task is inherently subjective, and there are mutiple valid interpretations of what phrases are important enough to be attached to a document.

The vanilla TF-IDF algorithm was already able to achieve reasonable performance, with respect to the human annotators. Limiting the candidate set to adjective-noun chunks drastically hurt the performance of the algorithm, suggesting that many important phrases do not fit this linguistic pattern, and the restriction is too severe. Document Boosting and Phrase Boosting, the two algorithms that incorporated external knowledge, were able to make improvements on the basic algorithm. Phrase Boosting with the TF-IDF scores of all candidate keywords was slightly better than TF-IDF, and only boosting longer phrases (Phrase Boosting N-Grams) was able to improve further. We can see that negative agreement was high for all algorithms, suggesting that it is relatively easy to identify phrases that shouldn't be keywords. Positive agreement was lower, implying it is indeed difficult to identify a small number of phrases to summarize the document, given that there are a large number of possible phrases, and there can be legitimate disagreement on what phrases are most important. Note that PABAK is not linearly related to Cohen's Kappa, P_{pos}, and P_{neg}, which explains how the Document Boosting algorithm can have a higher PABAK despite having a lower P_{pos} and Cohen's Kappa.

5. DISCUSSION

In order to convey intuition for how the algorithms differ, we will examine the keywords extracted from a few documents in depth.

In general, Phrase Boosting can be thought of as imposing a prior distribution over what phrases should be considered important. This prior is then combined with the local knowledge in a specific lecture. In practice, TF-IDF tends to give longer phrases unfairly low scores, because they do not appear frequently, even though they may be quite important to the content. Phrase Boosing corrects this by raising the value of longer phrases. For example, in the lecture "DTDs IDs and IDREFs", the phrase "Document Type Descriptors" is clearly important, but only appears 5 times in the document, and is ranked 36th if Phrase Boosting is not used. After incorporating the global Wikipedia ranking and the preference for longer phrases, it is boosted to the 9th rank. There are similar occurences throughout the lecture collection of longer phrases that only appear a small number of times in the document but have a high $TF\text{-}IDF_w$ score on the Wikipedia ranking. In a lecture on "Multivalued Dependencies and Fourth Normal Form", the phrase "multivalued dependencies" appears only 3 times in the lecture transcript and is therefore not considered a keyword by vanilla TF-IDF.

The phrase appears 26 times on Wikipedia in 4 documents, and is tagged as a keyword when the Wikipedia ranking is incorporated.

There are also a few nice properties of Phrase Boosting from algorithmic and computational perspectives. First, we formed the combined ranking $TF\text{-}IDF_{combined}$ by giving equal weight to $TF\text{-}IDF_{w-norm}$, the phrase's score in the Wikipedia ranking, and $TF\text{-}IDF_{norm}$, the phrase's score in the lecture ranking. One could modify the preference of the algorithm towards favoring local or external information by weighting these two components differently - if all weight was given to $TF\text{-}IDF_{norm}$ the algorithm would output the same order as TF-IDF, and if all weight was given to $TF\text{-}IDF_{w-norm}$ the algorithm would output the order extracted from Wikipedia. This principle applies more generally, and Phrase Boosting can incorporate any prior belief about keyphrases. Second, although there are a large number of documents in Wikipedia (the algorithm was run on 5027125 documents which had a total size of about 50GB), the runtime complexity is linear in the number of documents and therefore quite tractable on modern hardware.

Document Boosting	**TF-IDF**
transaction	transaction
isolation level	read
read	isolation level
lock	t1
commit	t2
concurrent	commit
serialization	client
repeat read	dirty read
concurrency control	uncommit
dirty read	transaction isolation level
transaction commit	GPA

Figure 4

The Document Boosting algorithm can be thought of as amplifying the keyphrases for a lecture, and therefore decreasing the scores of phrases that happen to occur frequently in a lecture but are not meaningful. This often occurs if a lecture has worked examples that use words from the examples frequently, as can be seen in Figure 4, which shows the top ranked phrases for the Document Boosting algorithm and plain TF-IDF on a lecture on “Isolation Levels”. In TF-IDF's keywords, we can see that there was a example involving students, and the token “GPA” appears frequently, even though it is not important to the core concept of the lecture. Similarly, the instructor used examples with transactions named “T1” and “T2”, which appeared frequently in this lecture and not others, in effect tricking the TF-IDF metric.

When Document Boosting was run, the Wikipedia article “Isolation (database systems)” was selected, which was able to decrease the frequency of phrases that were part of specific examples, and augment the frequency of words that were related to the concept of isolation levels, such as “concurrency control”, “repeat read”, and “transaction commit”. We can see that conceptually the algorithm is boosting phrases in the intersection of the Wikipedia article and lecture. This allows noise that only occurs frequently in the lecture to be identified and cut out.

6. CONCLUSION

In this paper we have started to tackle the task of choosing the most important phrases from a collection of lectures, which will be used downstream to support information retreival tasks, such as answering learner questions with relevant lecture clips, and recommendation tasks, such as finding the best study materials given their progress through the course. There has been little previous work on keyword extraction in the online education setting, so we began by evaluating the performance of term frequency-inverse document frequency, a well-known metric for gauging the importance of a phrase to a document. After evaluating the weaknesses of TF-IDF, we designed algorithms that incorporated linguistic information, in the form of part-of-speech tags and chunking, and external information, with the entire Wikipedia document collection used as a knowledge source. The algorithms that incorporate Wikipedia information boost performance of TF-IDF, especially on longer phrases that do not have high raw frequencies in a lecture. We also created a human generated set of keywords for the online introductory databases course we evaluated our algorithms on.

In the future we would like to explore using the richer structure provided by the Wikipedia dataset to improve our keyword extraction algorithms. For example, Wikipedia gives access to the link structure between documents and groups documents into collections, which are explored in [6] and [12] for different tasks.

We are also interested in keyword extraction as a supervised learning task. As previously described, one of the main challenges is the cost of obtaining a large amount of labelled data, especially in the online education setting where annotators often need to be highly educated and devote a substantial amount of time to generate high quality keywords. One possible strategy is transfer learning, where a learning algorithm is trained on a different problem than the one it will make predictions on. It is possible that there are features that differentiate important phrases in journal abstracts or newspapers that could also differentiate important phrases in lectures. Indeed, transfer learning for text classification has been explored previously [14], [3].

7. REFERENCES

[1] T. Byrt, J. Bishop, and J. B. Carlin. Bias, prevalence and kappa. *Journal of clinical epidemiology*, 46(5):423–429, 1993.

[2] D. V. Cicchetti and A. R. Feinstein. High agreement but low kappa: Ii. resolving the paradoxes. *Journal of clinical epidemiology*, 43(6):551–558, 1990.

[3] C. Do and A. Y. Ng. Transfer learning for text classification. In *NIPS*, pages 299–306, 2005.

[4] A. R. Feinstein and D. V. Cicchetti. High agreement but low kappa: I. the problems of two paradoxes. *Journal of clinical epidemiology*, 43(6):543–549, 1990.

[5] J. L. Fleiss. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378, 1971.

[6] X. Hu, X. Zhang, C. Lu, E. K. Park, and X. Zhou. Exploiting wikipedia as external knowledge for document clustering. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge*

discovery and data mining, pages 389–396. ACM, 2009.

[7] A. Hulth. Improved automatic keyword extraction given more linguistic knowledge. In *Proceedings of the 2003 conference on Empirical methods in natural language processing*, pages 216–223. Association for Computational Linguistics, 2003.

[8] A. Imran, L. Rahadianti, F. Cheikh, and S. Yayilgan. Semantic tags for lecture videos. In *Semantic Computing (ICSC), 2012 IEEE Sixth International Conference on*, pages 117–120, Sept 2012.

[9] C. D. Manning, P. Raghavan, and H. Schutze. *Introduction to Information Retrieval*. Cambridge University Press, 32 Avenue of the Americas, New York, NY 10013-2473, USA, 2008.

[10] Y. Matsuo and M. Ishizuka. Keyword extraction from a single document using word co-occurrence statistical information. *International Journal on Artificial Intelligence Tools*, 13(01):157–169, 2004.

[11] R. Mihalcea and P. Tarau. Textrank: Bringing order into texts. Association for Computational Linguistics, 2004.

[12] D. Milne. Computing semantic relatedness using wikipedia link structure. In *Proceedings of the new zealand computer science research student conference*, pages 1–8, 2007.

[13] Y. Park and Y. Li. Extracting salient keywords from instructional videos using joint text, audio and visual cues. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 109–112. Association for Computational Linguistics, 2006.

[14] R. Raina, A. Y. Ng, and D. Koller. Constructing informative priors using transfer learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 713–720. ACM, 2006.

[15] S. Rose, D. Engel, N. Cramer, and W. Cowley. Automatic keyword extraction from individual documents. *Text Mining*, pages 1–20, 2010.

[16] G. Salton. The smart retrieval systemâĂŤexperiments in automatic document processing. 1971.

[17] P. Turney. Learning to extract keyphrases from text. 1999.