Computational Models of Human Document Keyword Selection

Michael J. Stipicevic (stipim@rpi.edu)
CogWorks Laboratory, Rensselaer Polytechnic Institute
Troy NY, USA

Vladislav D. Veksler (vekslv@rpi.edu)
CogWorks Laboratory, Rensselaer Polytechnic Institute
Troy NY, USA

Wayne D. Gray (grayw@rpi.edu)
CogWorks Laboratory, Rensselaer Polytechnic Institute
Troy NY, USA

Abstract

Computational models are presented that attempt to mimic how humans select keywords to describe documents. These semantic models are based on data mining techniques applied to large corpora of human writing. A methodology to test the merit of these models is developed; performance at matching author-chosen keywords is the basis of this test. Results indicate topic models and their derivatives outperform traditional semantic models. Finally, it is shown how these models might be incorporated into a system that automatically selects keywords for an academic publication.

Keywords: machine learning; natural language processing; Latent Semantic Analysis; Latent Dirichlet Allocation; Correlated Topic Models; VGEM

Background

After reading a text, humans are able to provide a quick summary of its contents. The smallest summary possible is simply a list of topics. These topics, or 'keywords,' represent the highest levels of human abstraction: they dramatically reduce an entire document to a few words while retaining key information. A computational model of keyword generation would allow researchers to better understand how knowledge is extracted, abstracted, and generalized in the mind.

In this paper we compare the ability of four computational models to pick out author-selected keywords from a larger set of possible keywords. The models are contrasted based on theoretical and technical differences. Finally we propose future research to better understand the underlying mechanics of these models and show how they may be useful as normative tools for automatic keyword generation. The tests of the four models (fit to data) serve as a proof of concept of such a system.

Measures of Semantic Relatedness

Measures of semantic relatedness (MSRs) are techniques that quantify the semantic relationships between two words or documents. They derive a numeric ranking of relatedness from a fitted semantic model. For example, after being trained on a large corpus of English text, an MSR might determine that 'cat' and 'dog' are highly related. The calculation of this ranking depends on each MSR, but the interpretation is the same.

All of the selected MSRs (and the models they are based on) depend on the 'bag of words' assumption (Landauer, Laham, et al., 1997). This means that word order or context does not factor into the relatedness computation. As keywords merely describe the topics of the text (instead of the content), this assumption should not hinder the predictive power of the models.

Although all MSRs are capable of calculating relatedness between words, only a few of the measures are capable of determining relatedness between multi-word terms (e.g. documents, paragraphs). The four MSRs selected for this analysis are able to do this.

Our assumption is that keywords may be selected for a given document from a larger ontology based on document-keyword relatedness values. In other words, for a given MSR, m, a document, D, a set of appropriate keywords for this document, $k_{1..n}$, and a set of less appropriate keywords for this document (distractors), $d_{1..n}$, we assume that $m(D,k_x)>m(D,d_x)$.

Semantic Models

LSA

Latent Semantic Analysis was first proposed in the late 1980s as a way to extract meaningful relationships between text (Landauer & Dumais, 1997). It has become the basis of numerous applications including educational testing, search engines, and optical character recognition (Zhuang, Bao, Zhu, Wang, Naoi, 2004). LSA uses the singular value decomposition (SVD) to identify the strongest linear relationships within text corpora. The matrices resulting from this analysis can be used to calculate word-to-word, word-to-document, or document-to-document similarity.

Constructing the LSA model

A word-document matrix is constructed from a corpus of natural language. Each element in the matrix is the tf-idf ranking (term frequency–inverse document frequency; Salton and McGill, 1983) of the corresponding word in the corresponding document. Using tf-idf allows LSA to discount frequent words that have low semantic content ('the,' 'what'). The calculation for tf-idf is:

$$\frac{f(x \mid d)}{length(d)} \times \log \frac{m}{f(x)}$$

,

where $f(x|w)$ is the number of times that some word, x, appears in a given document, d, $length(d)$ is the number of words in d, m is the total number of documents in a corpus, and $f(x)$ is the total number of times that x appears in the corpus.

Once the word-document matrix is populated, the singular value decomposition is run. This produces a representation of the original word-document space but realigned to capture important relationships. Restricting the new semantic space to the N most important dimensions provides a set of vectors associated to each word in the corpus an each document in the corpus. For this paper, N was set to 50 to provide a large enough number of dimensions without impeding the usefulness of the SVD.

Calculating a relatedness value

Two methods are available to compare words to documents: appending the document's word count to the original word-document matrix as the start of training, or summing each word's topic vector over the entire document. Since the former requires an expensive SVD computation for every test, we choose the latter to evaluate semantic relatedness.

For every keyword/document pair, a semantic relatedness measure can be calculated as follows:

1. For every keyword/document, look up each word in the reduced LSA semantic space (see above). This produces a vector of length N for every word.
2. Sum these vectors over the entire keyword/document.
3. Take the summed keyword vector and the summed document vector and determine the cosine between the two vectors. This cosine-similarity is the final score provided by LSA (Landauer & Dumais, 1997).

Implementation

LSA was implemented for this paper in custom software. A word-by-document matrix was constructed and populated with corresponding tf-idf values. This matrix was passed to a Matlab SVD routine which computed the reduced semantic model. This model correlates words to the reduced semantic space. A vector for each document or keyword phrase was calculated by summing the individual word vectors, and the final relatedness value is the cosine between the document and keyword vectors.

LDA

In attempting to rework LSA with a strong probability model, Latent Dirichlet Allocation was developed (Blei, Ng, Jordan, 2003). This technique models each document as a probability distribution of topics; each topic is modeled as a distribution of words. By inferring what topic and word distributions exist in a corpus, LDA is able to provide an intuitive notion of topic – and keyword – extraction. LDA and similarly derived methods are called 'topic models' because, unlike methods such as LSA, topics are an explicit component in the model.

Constructing the LDA model

Latent Dirichlet Allocation builds a generative model of text by fitting a proposed model against known data. Specifically, LDA constructs a hierarchical Bayesian model based on Dirichlet priors. Thus, documents comprise a Dirichlet distribution of N "topics," while topics comprise a multinomial distribution of words. Two corpus-wide parameters govern the model: the Dirichlet prior for topics are controlled by a scalar parameter α and the multinomial distribution for words in topics is controlled by the N-vector β. By estimating α and β for a corpus, a document's topical content may be computed and compared. As with LSA, N was chosen to be 50 for this paper.

1. Initialize a placeholder set of topic probabilities (γ) and a placeholder set of probabilities that each word was derived from each topic (Φ).
2. Using the expectation-maximization algorithm (Dempster, Laird, & Rubin, 1977) determine the best Dirichlet parameters to predict the word-document matrix (as calculated for LSA).
    1. For each word and using the current estimate of γ, α, and β, estimate the probabilities of which topic each word was derived from (Φ).
    2. Normalize Φ so it sums to 1.
    3. For each document and using the updated estimate of Φ, α, and β, calculate the new per-document topic probabilities γ.
3. Once Φ and γ are calculated, estimate α, and β. Repeat steps 2 and 3 until convergence.

Calculating the relatedness value

Using Bayesian inference (with the Dirichlet priors calculated above), a probability for each topic in a document can be calculated. This vector is a topical "fingerprint" of the document, and is similar to the vector created by LSA. Another topic vector is created for the keyword, and the cosine similarity between document and keyword vectors provides a similarity score.

Implementation

For this paper, the LDA-C software package was used. This takes a list of word counts for each document and outputs a fitted LDA model in terms of topic, document, and word probabilities. It also infers the topic probabilities of a new document (when given a fitted model). The similarity

between a document and keyword phrase can be determined by first inferring the topic probabilities of each. These can be treated as a vector, and the cosine represents their relatedness.

CTM

Correlated Topic Models extend LDA by allowing topics to be correlated with each other (Blei & Lafferty, 2006). LDA requires topics to be statistically independent, but this may not be true in practice. For example, a document with a topic related to biology is more likely to contain chemistry related topics than topics concerning the French Revolution. CTM allows for this correlation by using a logistic distribution instead of the Dirichlet.

Constructing the CTM model

Correlated Topic Models are constructed in a similar manner to Latent Dirichlet Allocation models, the main difference is the choice of the logistic norm instead of the Dirichlet prior. The logistic norm allows topics to be correlated with each other – specified by a correlation matrix. Inference of these new parameters are performed similarly as in LDA.

Calculating the relatedness value

The results from CTM are computed exactly the same as with LDA – the distribution over topics is treated as a vector and the cosine similarity is computed between the keyword and document. The cross-topic correlation values are ignored.

Implementation

As with LDA, CTM was computed using a software package. CTM-C takes similar inputs and provides similar outputs as LDA-C. Since the cross-topic correlation values are ignored for this paper, the calculation of relatedness values in CTM is the same as in LDA.

VGEM

VGEM (Vector Generation from Explicitly-defined Multidimensional semantic space; Veksler, Govostes, & Gray, 2008) was recently proposed as an alternative to the more computationally-intensive MSRs introduced above. Like LSA, VGEM represents terms as vectors in a multidimensional semantic space, and calculates term relatedness as the cosine between their vectors. However, VGEM does not require construction or computational reduction of a document-by-word matrix (which becomes extremely expensive for sufficiently large corpora). Instead, VGEM requires a set of words to be explicitly chosen as the dimensions of the semantic space, and calculates term vectors dynamically based on term frequencies and term co-occurrences with each of the dimension-words. Various frequency/co-occurrence formulas may be used, e.g. Pointwise Mutual Information (Turney, 2001), or Normalized Google Distance, (Cilibrasi & Vitanyi, 2007).

For the purposes of this paper, VGEM dimensions were taken to be the topics derived from LDA, and frequency/co-occurrence formula used for calculating term vectors was Normalized Similarity Score (NSS), which is a variant of Normalized Google Distance. To be more precise, the value of each word, x, on dimension, y, is derived as follows:

$$\mathrm{NSS}(x, y) = 1 - \mathrm{NGD}(x, y) ,$$

where NGD is a formula derived by Cilibrasi & Vitanyi (2007):

$$\mathrm{NGD}(x,y) = \frac{\max\{\log f(x), \log f(y)\} - \log f(x,y)}{\log M - \min\{\log f(x), \log f(y)\}}$$

$f(x)$ is the frequency with which x may be found in the corpus, $f(x,y)$ is the frequency with which both x and y may be found in the corpus, and M is the total number of texts in the corpus.

Constructing the VGEM model

1. After training an LDA model(N = 50), select the two highest probability words from each topic in the corpus. This provides the dimension words for VGEM and only needs to be performed once per corpus. Note that VGEM may have up to 100 dimensions, but may have fewer due to redundant words in the LDA-derived dimensions.

Calculating relatedness value

1. For each word in the document, calculate the NSS between the word and each dimension word.
2. Over each document, sum up all word similarity vectors
3. To compare both documents, calculate the cosine similarity between the two summed vectors.

Implementation

VGEM was implemented in custom software for this paper. After the set of word-document counts were computed, LDA was trained on the corpus to provide VGEM dimension words. The two most probable word for each topic were chosen for VGEM dimensions (duplicate words were only represented once). NSS value between a given word and each dimension word generate a VGEM vector. Negative NSS values were clamped to zero. For each document or keyword phrase, VGEM vectors for each word were summed, providing a vector for the entire document or keyword phrase. The similarity between the two is simply the cosine, as for all the measures presented in this paper.

Methodology

Corpus Selection

Our testing and training corpus is the proceedings of the Annual Meeting of the Cognitive Science Society from 2004 to 2008. 100 papers from 2008 were removed to provide test cases, while the rest of the papers were used to train the models as described above. All stopwords as

defined in the Python Natural Language Toolkit (Loper & Bird, 2002) were discarded, in addition to all words less than 3 characters long and all words occurring in less than 3 documents.

Segmentation

Each document was processed in two separate modes: by -document and by-paragraph. In paragraph mode, each document was split by software into 50-word non-overlapping segments (the final segment, even if less than 50 words, is kept). Sentence and true paragraph boundaries are ignored (to preserve the 'bag of words' assumption). Document mode leaves the document intact. Thus, in by-paragraph mode, a word-by-paragraph matrix is constructed for LSA, LDA, and CTM instead of the word-by-document matrix mentioned above (splitting up corpora into smaller segments is standard practice, e.g. Landauer & Dumais, 1997).

Segmenting documents by paragraph allows insight into what information these models incorporate. In paragraph mode, each model is only given a small window to process text. This emphasizes locality in word-word association. However, because the windows do not overlap, paragraph mode may break word-word associations that lie across 'paragraph' boundaries. This would lead to an artificially inflated number of topics as there would be less second-order word-word correlations. Future research will examine the use of a sliding window to solve this problem.

Testing Method

Test Cases

In order to evaluate the predictive power of these models, we determine how well each model can select author-chosen keywords. Each test case consisted of a *cue*, *targets*, and *distractors*. The *cue* was one of the 100 selected documents from Cognitive Science 2008 conference proceedings, as mentioned in the Corpus Selection section. The *targets* were the author-picked keywords from the *cue* document, and the *distractors* were random keywords from other documents. The number of *distractors* was twice the number of *targets,* a random guess would be correct 33% of the time. It should be noted that each 'keyword' may actually be a keyword phrase (e.g., 'natural language processing'). This does not pose a problem as each MSR is able to process multi-word terms.

For each MSR, the *targets*+*distractors* list was ranked and sorted in accordance with MSR's relatedness values between the *cue* and each of the keywords. Finally, the score for each MSR on each *cue-targets-distractors* test case was calculated as follows:

$$Score_{case} = \frac{Number\ of\ targets\ in\ top\ n\ words}{n}$$

where *n* is the number of *targets*, and "*top n words*" refers to the top third of the sorted *targets-distractors* list. Thus, if all target keywords are more related to *cue* than any of the distractor keywords, the score for that test case is 100%. If none of the target words are picked by the MSR to be more related to the cue than any of the distractor words, the score is 0. The overall score for a given MSR is the average of all 100 test case scores.

Results

The mean performance, measured as the ability to select author generated keywords from among distractors, is shown in (Figure 1). We compared the best results from each measure (by-document mode for LSA and LDA; by-paragraph mode for CTM and VGEM) by means of a repeated measures ANOVA. The analysis revealed a significant main effect of Measure, $F(3, 395)=13.011$, $p<.01$. Posthoc Tukey HSD comparisons revealed significant differences between LSA (M=.53, SE=.02) and LDA (M=.69, SE=.02), LSA and CTM (M=.63, SE=.02), and LSA and VGEM (M=.68, .02). No significant differences were found between LDA, CTM, and VGEM.

Table 1. Mean and standard error values for the performances of four MSRs on selecting author-generated keywords among distractor keywords, using by-document and by-paragraph modes of training. Chance-level performance is .33.

Mode	Document		Paragraph	
	Mean	Std. Err	Mean	Std. Err
LSA	0.53	0.020	0.38	0.021
LDA	0.68	0.021	0.65	0.019
CTM	0.59	0.022	0.63	0.018
VGEM	0.42	0.021	0.68	0.019

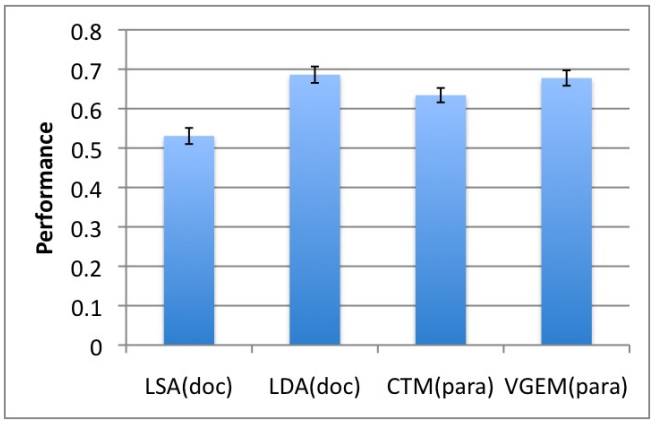

Figure 1. Best mean performances of four MSRs on selecting author-generated keywords among distractor keywords. Chance-level performance is .33. Error bars represent standard error.

Analysis

It is not surprising that topic models (LDA and CTM) consistently perform well in this task. Keywords are nothing more than descriptions of topics; while LSA and VGEM can be interpreted to use topics, they do not model them as explicitly as LDA and CTM.

In paragraph mode, LSA barely outperforms random guessing (0.33). This implies that LSA heavily depends on second and even higher order word correlations, which has been confirmed in several contexts (Kontostathis & Pottenger, 2002). However, the exact opposite has occurred with VGEM – non-overlapping segmentation of text has significantly increased its score. VGEM's underlying MSR – NSS – explicitly uses first order co-occurrence in its model, but nothing else.

LDA and CTM were not statistically different from each other. This is striking because CTM was developed as an improvement upon LDA. Perhaps the utility of CTM is not realized without very disparate topics, the narrow scope of cognitive science papers might render the topic correlation of CTM ineffective.

VGEM is statistically equivalent to LDA and CTM. This is interesting as it is a much simpler measure. Of course, in this paper VGEM used LDA-derived dimensions; this means a full LDA training step must be performed to obtain these results. However, that only needs to be performed once per corpus. With dimensions, VGEM only depends on two tabulations: how many documents contain a given word, and how many documents contain two given words. These can be performed quickly on large or even streaming databases. Additionally, the VGEM approach is adaptable to new vocabulary: as long as the new word appears in a document with a dimension word, a VGEM score can be computed for it. The other three models would require computationally expensive retraining.

Future Development

Sliding Window

As mentioned in analysis, the paragraph model emphasizes locality, but interferes with higher-order word correlation. Preprocessing the text as a sliding window will eliminate artificial barriers by overlapping the selected text. This has the unfortunate side-effect of greatly increasing computation time, which is why it was left out of this analysis. The size of the window (either sliding or non-overlapping) could be modulated to find parameters that best suit the data. This analysis might not benefit LDA (which performs worse on paragraphs) but might boost VGEM performance even higher.

General Corpus

In this analysis, the MSRs were trained on a corpus with a narrow technical focus – cognitive science articles. Although a substantial number of documents were used, the limited breadth of this corpus might be an issue, especially when specialized vocabulary is considered. Lindsey, Veksler, Grintsvayg, & Gray (2007) explore the performance of MSRs when used on different types of corpora, a similar analysis could be performed on the keyword-matching test.

Larger Parameter Search

In addition to the corpus used and the text windowing chosen, there are several variables that affect the outcome of the results. The number of topics/dimensions used, the selection of topics for VGEM, and the choice of MSR all might interact in complicated ways that can only be determined by a more rigorous examination of the parameter space.

Automated Keyword Generation

This work lays the foundation for an automated keyword selection system. Editors of scholarly publications solicit keywords for each submission, mainly to assist in assigning reviewers with relevant interest. However, these keywords can describe topics too narrow or too broad and are rarely consistent across authors (Furnas, Landauer, Gomez, & Dumais, 1987). This is not fully alleviated by a fixed set of keywords: authors may pick too few or too many keywords, and the keyword set may be redundant or omit crucial topics.

A system built upon the keyword-matching test could provide a solution to these two problems. For the first, any of the MSRs described can rank the relationship between each paper and an ontology of keywords. A relatedness threshold can determine which keywords to retain for the document and which to discard. As for the second problem, the ontology of keywords itself can be generated by topic models such as LDA and CTM, much like the VGEM dimensions extracted from LDA as described above. Theoretically, these would represent the 'true' topic distribution.

To test such a system, more human data would be required. Reviewers of each submission are most qualified to judge which keywords are most representative of the document. When reviewing a paper, they could be sent two sets of keywords – author developed and MSR generated – and select which set they feel better suits the paper. This could be expanded to provide keywords from multiple MSRs

Of course, with such sophisticated ranking systems, one questions the usefulness of keywords at all – could we not move straight to an automated matching of submissions to reviewers?

Summary

We describe several MSRs of various theoretical underpinnings and introduce a test that measures the ability of MSRs to match human document keyword selection. We then evaluate each MSR's performance on this test. The results demonstrate that topic models perform well and are

robust under text segmentation. Additionally, VGEM (with LDA-derived dimensions) performs as well as topic models with the added benefit of adaptability and speed. Finally, we show how this test is a proof of concept of an automatic keyword generation system.

Acknowledgments

The work was supported, in part, by grant N000140710033 to Wayne Gray from the Office of Naval Research, Dr. Ray Perez, Project Officer.

References

Blei, D., & Lafferty, J. (2006). Correlated topic models. In Advances in Neural Information Processing Systems 18.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet allocation. *Journal of Machine Learning Research* 3 (Mar. 2003), 993-1022.

Cilibrasi, R., & Vitanyi, P. (2007). The Google Similarity Distance. IEEE Transactions on Knowledge and Data Engineering, Vol. 19, No. 3, p. 370-383.

Dempster, A.P.; Laird, N.M.; Rubin, D.B. (1977). "Maximum Likelihood from Incomplete Data via the EM Algorithm". *Journal of the Royal Statistical Society.* Series B (Methodological) 39 (1): 1–38.

Furnas, G. W., Landauer, T. K., Gomez, L. M., & Dumais, S. T. (1987). The Vocabulary Problem in Human System Communication. Communications of the ACM, 30(11), 964-971.

Landauer, T. K., Laham, D., Rehder, B., & Schreiner, M. E., (1997). How well can passage meaning be derived without using word order? A comparison of Latent Semantic Analysis and humans. In M. G. Shafto & P. Langley (Eds.), Proceedings of the 19th annual meeting of the Cognitive Science Society (pp. 412-417). Mawhwah, NJ: Erlbaum.

Landauer, T. K., & Dumais, S. T. (1997). A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. Psychological Review, 104(2), 211-240.

Lindsey, R., Veksler, V. D., Grintsvayg, A., & Gray, W.D. (2007). Effects of Corpus Selection on Semantic
Relatedness. 8th International Conference of Cognitive Modeling, ICCM2007, Ann Arbor, MI.

Kontostathis, A., & Pottenger, W. M. (2002). A mathematical view of latent semantic indexing: Tracing term co-occurrences. Technical Report, LU-CSE-02-006, Department of Computer Science and Engineering, Lehigh University.

Loper, E., & Bird, S. (2002). NLTK: The Natural Language Toolkit. In Proceedings of the Workshop on Effective Tools and Methodologies for Teaching NLP and Computational Linguistics, pages 63–70, Philadelphia, PA, 7 July.

Salton, G. & M. McGill. (1983). Introduction to Modern Information Retrieval. McGraw-Hill.

Turney, P. (2001). Mining the Web for Synonyms: PMI-IR versus LSA on TOEFL. In L. De Raedt & P. Flach (Eds.). Proceedings of the Twelfth European Conference on Machine Learning (ECML-2001) (pp.491-502). Freiburg, Germany.

Veksler, V. D., Govostes, R., & Gray, W. D. (2008). Defining the Dimensions of the Human Semantic Space. Accepted to the 30th Annual Meeting of the Cognitive Science Society.

Zhuang, L., Bao T., Zhu, X., Wang, C., Naoi, S. (2004). A Chinese OCR Spelling Check Approach Based on Statistical Language Models. International Conference on Systems, Man and Cybernetics, pp. 4727-4732. Hague, Netherlands: IEEE.