Iterative Hard Thresholding for Keyword Extraction from Large Text Corpora

Steve Yadlowsky, Preetum Nakkarin, Jingyan Wang, Rishi Sharma, Laurent El Ghaoui
Electrical Engineering and Computer Sciences
University of California, Berkeley
{steve.yadlowsky,preetum,jingyanw,rsh,elghaoui}@berkeley.edu

***Abstract*—To better understand and analyze text corpora, such as the news, it is often useful to extract keywords that are meaningfully associated with a given topic. A corpus of documents labeled by their topic can be used to approach this as a learning problem. We consider this problem through the lens of statistical text analysis, using bag-of-words frequencies as features for a sparse linear model. We demonstrate, through numerical experiments, that iterative hard thresholding (IHT) is a practical and effective algorithm for keyword-extraction from large text corpora. In fact, our implementation of IHT can quickly analyze more than 800,000 documents, returning keywords comparable to algorithms solving a Lasso problem-formulation, with significantly less computation time. Further, we generalize the analysis of the IHT algorithm to show that it is stable for rank deficient matrices, as those arising from our bag-of-words model often are.**

I. Introduction

Identifying meaningful keywords in a large corpora of documents is useful and interesting across many disciplines. For example, we may wish to use news articles to explore current or historical events, using keywords to find other concepts related to a given topic [1]. We can also quickly summarize a topic, by representing it with a small set of keywords – which can be used both to understand the topic, and to efficiently query other relevant documents for that topic. Keyword extraction can also be used for investigative purposes, for example to suggest causal factors that influence plane safety by analysing ASRS flight reports [2], or to investigate public perception and bias in media reporting [1].

There are two broad categories of keyword identification: one in which the goal is to extract keywords associated with a specific topic [1], [3], and the other in which the goal is to find keywords of implicit topics in the corpus [4]–[6]. This article is focused on the former category. To be able to find keywords associated with a specific topic, the corpus must be labeled with an indication of whether or not each document is about the topic of interest – that is, we are exploring a form of supervised learning. The goal is to identify a set of words that indicate whether or not a document is about the topic of interest. By restricting the size of this set to be small, we get a set of words that can be interpreted as keywords for the topic.

This problem can be posed in different ways. For example, one approach is to analyze the syntax and semantics of sentences to identify keywords according to the rules of the natural language [7], [8]. In the present work, we adopt a statistical approach to text analysis that analyzes only the number of occurrences of each word in the documents. This approach could be extended in the same way as many statistical text analytics approaches by considering n-*grams*, n contiguous words or characters [9]–[11], and counting the occurrences or term frequency-inverse document frequency (tf-idf score) of each of these [12]. For the purposes of this article, we will focus only on unigrams ($n = 1$). The advantage of taking a statistical approach is especially significant for large corpora: statistical approaches require much less processing per document, and thus can much more efficiently scale to larger datasets [13]. At the same time, statistical approaches benefit tremendously from large corpora, as a large number of documents improves the statistical significance of correlations that are identified [14].

In the present work, news archives are used as a corpus for documents. Besides covering a broad range of topics, the news is an interesting corpus to analyze through the lens of keyword extraction to understand how it affects or is affected by public opinion on divisive topics [1]. These corpora can be very large: the New York Times archive that is analyzed in the Experiments section has some 800,000 paragraphs, containing nearly one million unique words, which is an exciting scale to use for statistical approaches to this problem.

There has been recent interest in applying sparse machine learning techniques to the text classification and keyword extraction problems, drawing on extensive literature in compressed sensing and convex optimization. [15]–[19]. We can formulate keyword extraction as the problem of finding a sparse set of word-predictors for a topic (using bag-of-words frequencies as features). Enforcing this sparsity is highly non-convex and solving it exactly is known to be NP-hard [20], so many of these techniques involve convex relaxations, such as the ℓ_1-penalization [16], which encourage sparsity indirectly. Blumensath and Davies recently introduced the IHT algorithm for use in compressed sensing [17], which directly enforces the sparsity constraint and is guaranteed to converge under certain conditions.

This article extends the analysis of the IHT algorithm, demonstrating that convergence guarantees depend solely on bounding the ℓ_2-norm of the data matrix, $\|X\|_2 < 1$. This analysis includes all matrices arising from word-document occurrence matrices, since matrices can always be scaled by a constant factor to satisfy this requirement (without loss of generality).

Furthermore, numerical experiments demonstrate that iterative hard thresholding is a practical algorithm for working with large text corpora. Keywords extracted from the New

York Times archive on a number of topics considered indicate that the formulation and algorithm are an effective means of performing keyword extraction.

II. Problem Formulation

This article uses the following notation for representing the bag of words model. The j^{th} element of the feature vector x_i for document i encodes the term frequency of the j^{th} dictionary word in the document. Each document i is labelled with an indicator $y_i \in \{-1, 1\}$ according to whether or not the document is about the topic to be analyzed. We can represent the entire dataset as $\{(x_i, y_i)\}_{i=1}^m$, where m is the number of documents, and each $x_i \in \mathbb{R}^n$, where n is the number of words in the dictionary.

We model a document as likely to be about a topic if a weighted sum of the term frequencies of k keywords related to the topic is above some threshold. Thus, we restrict ourselves to a model which is a sparse linear combination of the elements of x_i : For some weight-vector β, document i belongs to class $y_i = 1$ if $x_i^T \beta > 0$, otherwise it belongs to $y_i = -1$. The task is to find a weight-vector β such that only k entries are nonzero – which can be interpreted as finding the k most significant keywords.

To learn the weights β, we formulate an optimization problem, where we have some loss function L of incorrectly predicting y_i:

$$\begin{aligned} \underset{\beta}{\text{minimize}} \quad & \sum_i L\left(y_i, x_i^\intercal \beta\right) \\ \text{s.t.} \quad & \operatorname{card}\left(\operatorname{supp}\left(\beta\right)\right) \leq k \end{aligned} \tag{1}$$

Common loss functions include squared error, log-loss. and hinge loss. All of these are convex loss functions, however the feasible set of this problem is not convex [20]. In this paper, the squared error loss is considered, as [1] shows that although it is not ideal for binary classification, it is found to do well in practice. This loss function also makes our mathematical problem very similar to one studied in compressive sensing, the sparse least squares problem:

$$\begin{aligned} \underset{\beta}{\text{minimize}} \quad & \|y - X\beta\|_2 \\ \text{s.t.} \quad & \operatorname{card}\left(\operatorname{supp}\left(\beta\right)\right) \leq k \end{aligned} \tag{2}$$

where the i^{th} row of X is the occurrence vector, x_i, associated with the i^{th} document.

III. Algorithms

Unfortunately, this optimization problem is not convex, and finding the global minimum is known to be NP-hard. Two approaches for efficiently finding an approximate solution have been proposed and analyzed by the compressive sensing community: a convex relaxation of the constraint based on the ℓ_1-norm, and methods for finding local a local minimum of the objective directly over the non-convex feasible set.

The ℓ_1 relaxation involves relaxing the constraint to $|\beta|_1 \leq \lambda$, and has been shown to encourage sparsity [19]. [2] uses a formulation based on the Lasso to extract keywords using a bag of words model to identify topic descriptions. [1] uses an ℓ_1-penalized logistic regression formulation and demonstrates the quality of the keywords selected by investigating the human interpretability of the model in the experimental results. However, finding a choice of λ such that the original cardinality constraint is satisfied can be computationally expensive, as a line search over λ must be performed, until a large enough λ is selected that the solution is sparse [6]. [15] shows that the Lasso path can be found much more efficiently by warm-starting based on the solution for the previous value of λ in the Lasso path, however they do not present results for problems as large as those found in statistical text analysis, and we find in our experiments that this method does not scale as well as iterative hard thresholding for such large problems. [18] shows that the least angular regression (LAR) algorithm can be adapted to solve the entire Lasso path. However, [15] points out that performance is poor for very large problem sizes, and impractical for the size of matrices we analyze in this work.

Blumensath and Davies [17] present the iterative hard thresholding algorithm (IHT) as an efficient algorithm for finding a local minimum of the least-squares objective in (2). IHT has the advantage of explicitly working with the cardinality-constrained feasible set, instead of the ℓ_1 convex relaxation, controlled by λ as a proxy for the cardinality. They show convergence of the algorithm under certain conditions, and provide an analysis of the reconstruction error that can be achieved. We observe that IHT, in [17] is shown to solve a very similar problem to the one that we look to solve. In fact, their formulations are identical to (2), however the presentation of the algorithm's convergence and performance guarantees in [17] are restricted to a small set of matrices: those which are full column rank and meet the 3k-restricted isometry property [21].

Satisfying these properties is very restrictive, and presents a problem for applications to matrices derived from bag of words modelling of text. These matrices are often low rank, or can be well approximated by a low rank matrix [22]. In practice, they also are extremely unlikely to satisfy the $3k$-RIP assumption.

However, we show practical and theoretical results to support applying IHT to text matrices.

IV. Analysis

To justify applying IHT to text, we show here that a less restrictive set of conditions are sufficient to guarantee convergence of the IHT algorithm. While we are unable to show that estimates produced by this algorithm will perform within a constant factor of the optimal value as can be shown in the compressive sensing context, we can still guarantee that the error will be finite, and will not increase from the initial estimate.

A. IHT Algorithm

The original iterative hard thresholding algorithm presented in [17] is as follows. Starting from an initial weight vector $\beta^{(0)}$, often selected by running matching pursuit [23], this algorithm

converges to a local minimum $\beta^\star$ by iteratively performing the step:

$$\beta^{(n+1)} = H_k\left(\beta^{(n)} + X^\intercal(y - X\beta^{(n)})\right) \tag{3}$$

where H_k is the k-sparse hard-thresholding operator, which preserves only the largest k coefficients. Notice that this iteration can be efficiently performed, and directly enforces the k-sparsity of the weight vector β, making it well-suited to the k-keyword-extraction problem.

We use the accelerated version of this algorithm (AIHT), which uses double over-relaxation to improve convergence, as described in [24]. The accelerated version essentially tries to performs a more-informed step before hard-thresholding in (3), and falls back on the normal IHT step otherwise. Thus it converges under the same criteria as IHT and performs at least as well – and our analysis generalizing the convergence criteria for IHT applies to AIHT as well.

Blumensath and Davies provide sufficient conditions for convergence and bounds on the residual error [17], however, their assumptions greatly restrict the range of applications of their results. Specifically, they require that X be full column rank, and that the cumulative coherence, $\mu_1(X, k+1)$ be bounded by a constant factor, where

$$\mu_1(X, m) = \sup_{|\Gamma|=m} \sup_{\omega \notin \Gamma} \sum_{\gamma \in \Gamma} |\langle x_\omega, x_\gamma \rangle|.$$

It should be intuitive that in the case of matrices that are well approximated by a low rank matrix, a bound on the cumulative coherence is not possible. Word frequency matrices are often either low rank, or well approximated with a low rank matrix (ie. they have a poor condition number). Thus, these conditions are too strong to justify the application to text matrices. In fact, this is a strong condition to expect of any matrix in general. However, Section IV-B of this article presents an analysis of this algorithm which shows that these conditions are not necessary for the algorithm to be stable. The concessions are that strong statements about the bounds of the difference between the local minimum and the global minimum can no longer be made.

B. Convergence Generalization

Despite the fact that [17] assumes a strong set of conditions on the design matrix X, given a weaker set of properties, we can still guarantee the convergence of iterative hard thresholding to a local minimum of (2). These conditions are very general: the only assumption is that $\|X\|_2 < 1$. Any design matrix can be trivially modified to satisfy this condition by using

$$\tilde{y} = \frac{y}{\|X\|_2 + \epsilon}$$
$$\tilde{X} = \frac{X}{\|X\|_2 + \epsilon}$$

where ϵ is a small, but significant number such that $\|\tilde{X}\|_2$ is strictly less than 1.

Theorem 4.1: If $\|X\|_2^2 < 1$ and the set of feasible local minima of (2) is nonempty, then iterative hard thresholding will converge to a feasible local minimum of (2).

Proof: The proof of [17, Lemma D.1] proves that if the spectrum of $I - X^\intercal X$ is strictly positive, then the iteration of (3) converges to a fixed point. That is, $\forall\epsilon\ \exists N$ such that$\forall n > N,\ \|\beta^{n+1} - \beta^n\|_2 < \epsilon$. Our condition that $\|X\|_2 < 1$ implies this result, so it is still true for all matrices we consider. From this point, they prove that this fixed point is a local minimum of (2) in two cases. We review each of these cases, and show that they can be modified for the case when X is not full rank and does not satisfy their requirement on the cumulative coherence.

Case 1: The support of $\beta^{(n)}$ is the same for all $n > N$, for some N. Just as [17] shows, this reduces to Landweber iteration. [25] shows that even when X is not full rank, Landweber iteration converges to the closest minimum of $\|X\beta - y\|_2$ from the starting point, β^N in this case, assuming $\sigma_{\max}(X) < \sqrt{2}$.

Case 2: There exist infinitely many n such that $\beta^{(n+1)}$ and $\beta^{(n)}$ have different support. [17, Theorem 4] shows that $\|\beta^{n+1} - \beta^n\|_2 < \epsilon$ implies that $\forall i$,

$$\sup_i x_i^\intercal(y - X\beta^n) \le 2\epsilon \tag{4}$$

At this point, they assume X is full rank, and show that the error must converge to 0. We generalize this, showing that the error converges to the unconstrained, unregularized minimal least squares error e_{LS}. Let α be the largest singular value of X, and π be the subspace-projection operator:

$$\sup_i x_i^\intercal(y - X\beta^n) \le 2\epsilon \tag{5}$$

$$\alpha\Big(\|y - X\beta^n\|_2$$

$$-\|\pi_{\text{null}(X^T)}(y) - \pi_{\text{null}(X^T)}(X\beta^n)\|_2\Big) \le \tag{6}$$

$$\alpha\left(\|y - X\beta^n\|_2 - \|e_{\text{ls}}\|_2\right) = \tag{7}$$

$$\|y - X\beta^n\|_2 - \|e_{\text{ls}}\|_2 \le \frac{2\epsilon}{\alpha} \tag{8}$$

$$\|y - X\beta^n\|_2 \le \frac{2\epsilon}{\alpha} + \|e_{\text{ls}}\|_2 \tag{9}$$

Equation (6) can be reduced to (7) because the fundamental theorem of linear algebra gives us that $\text{range}(X) \perp \text{null}(X^T)$. We know that the solution cannot be smaller than the unconstrained LS case, so we also have a both a lower and an upper bound on the error,

$$\|e_{\text{ls}}\|_2 \le \|y - X\beta^n\|_2 \le \frac{2\epsilon}{\alpha} + \|e_{\text{ls}}\|_2 \tag{10}$$

$$\|y - X\beta^n\|_2 \to \|e_{\text{LS}}\|_2 \tag{11}$$

Hence, in this case, the algorithm converges to e_{LS}, which we know to be the global minimum. ■

Most importantly, there is no requirement on the cumulative coherence of X. This is key to expanding the scope of relevant problems to which iterative hard thresholding can be applied. [17] proves that application of iterative hard thresholding will not increase the error from the starting point. This result is still implied by Theorem 4.1.

V. EXPERIMENTS

We used IHT to analyze text from New York Times Headline articles between January 1981 and December 2006, and found the returned keywords to be meaningfully associated with the query topic.

Further, this analysis of more than 800,000 documents was performed in under one minute per query, demonstrating that this method is well-suited to handling large text corpora. Results are also shown for applying the LASSO algorithm to the same dataset, for comparison.

A. Methods

Preprocessing is performed to create the word occurrence matrix used in these experiments. Articles are cleaned by removing stop words, short words (3 or less characters), lowercasing all capitalized words, and scrubbing all punctuation from the corpus. Tokenization is performed at the paragraph level, creating a total of 805,772 documents. After removing all documents with less than 30 words, 761,727 documents remain. Irrelevant feature removal is performed by removing all words that appear less than twice in the entire corpus, and the number of features is further pruned by only selecting the 30,000 words with the high frequency variance among documents.

To label the corpus according to topic, all paragraphs that refer to a word stemming from the root word for the topic are found. For example, if the topic is 'economy', it is appropriate to include words such as 'economic', 'economist', and 'economies'. These words are removed from the features of each document to prevent the algorithm from selecting only these features as keywords, and ignoring other words associated with the topic. This method for selecting the labels is not perfect; future work includes using hand-labelled examples or more develop a more sophisticated labelling system that could be used to generate labels on a large number of documents, which could then be analyzed using our algorithm. However, this method is simple and provides a high enough quality baseline to perform keyword extraction.

The AIHT algorithm as described in Section IV-A was implemented in Python using the `numpy` [26] and `scipy` [27] libraries for sparse matrix operations. This is an identical environment as the `sklearn` LASSO implementation [28], allowing for fair comparison of runtime. An implementation of this algorithm can be found at http://www.eecs.berkeley.edu/~elghaoui/pubs_icmla14.html.

Features are normalized to reduce the sensitivity of regularization and thresholding to scaling factors. All features were mean-centered and given unit variance. This was a practical challenge in implementation, as such large matrices can only be stored in memory in a sparse representation. Mean-centering would remove this sparsity, and make storage of the matrices unwieldy. To overcome this, the features are implicitly centered by representing the centered and normalized matrix, $\hat{X}$, by the normalized matrix $\tilde{X}$ and a dyad with column means, $\bar{x}$, $\hat{X} = \tilde{X} - \mathbf{1}\bar{x}$.

All experiments were run with the sparsity $k = 24$, and the predictive words corresponding to nonzero coefficients are shown (in descending order of coefficient weight).

TABLE I. KEYWORDS FOR TOPIC 'ECONOMY'.

1	percent	9	trade	17	sanctions
2	development	10	industrial	18	nation
3	political	11	growth	19	countries
4	prices	12	russia	20	foreign
5	jobs	13	recession	21	indicators
6	business	14	rates	22	markets
7	government	15	budget	23	unemployment
8	social	16	country	24	policy

TABLE II. KEYWORDS FOR TOPIC 'TERROR'.

1	attacks	9	homeland	17	obituaries
2	security	10	osama	18	peace
3	palestinian	11	bombing	19	bomb
4	war	12	guantnamo	20	plot
5	antiterrorism	13	threats	21	sept
6	iraq	14	suspects	22	torture
7	islamic	15	qaeda	23	threat
8	bombings	16	attack	24	liberties

TABLE III. KEYWORDS FOR TOPIC 'DATA'.

1	percent	9	report	17	technology
2	computer	10	software	18	recorder
3	study	11	fed	19	wireless
4	statistics	12	web	20	communications
5	analysis	13	census	21	analyzed
6	scientists	14	researchers	22	processing
7	systems	15	numbers	23	traders
8	privacy	16	companies	24	consumer

TABLE IV. KEYWORDS FOR TOPIC 'PRIVACY'.

1	liberties	9	protect	17	intrusion
2	records	10	internet	18	eavesdropping
3	encryption	11	consent	19	database
4	confidentiality	12	private	20	rooms
5	personal	13	testing	21	bathroom
6	surveillance	14	roe	22	searches
7	data	15	living	23	intrusive
8	mail	16	doubleclick	24	law

TABLE V. KEYWORDS FOR TOPIC 'MICROSOFT'.

1	software	9	google	17	browser
2	windows	10	technology	18	yahoo
3	antitrust	11	stocks	19	msnbc
4	gates	12	msn	20	operating
5	xbox	13	nasdaq	21	redmond
6	computer	14	playstation	22	spreadsheet
7	internet	15	apple	23	intel
8	microsystems	16	composite	24	corporation

To provide a useful comparison for the computational advantage IHT presents over an algorithm such as Lasso, we also ran the Lasso algorithm using both `sklearn` [29] and `glmnet` [15] for comparison. To get a k-sparse solution, we computed the Lasso path [15] for a sequence of λ, the regularization parameter, and chose the one with the closest solution support size to our choice of $k = 24$. If there were more than 24 nonzero elements, we chose the largest 24 as the keywords.

TABLE VI. LASSO KEYWORDS FOR TOPIC 'ECONOMY'.

1	percent	9	industrial	17	nation
2	growth	10	prices	18	billion
3	government	11	rates	19	trade
4	years	12	markets	20	political
5	country	13	unemployment	21	interest
6	states	14	recession	22	countries
7	inflation	15	companies	23	jobs
8	higher	16	business	24	rise

B. Results

1) Economy: We first explore keywords related to the topic 'economy'. The economy is frequently discussed in the news, making it a prototypical topic one may be interested in summarizing. The selected keywords are listed in Table I. These keywords are all words commonly associated with the economy. As expected, this topic is one for which keyword extraction from news articles worked very well. Computation using our method takes 37.9 seconds in total per query. The computation times for all other queries are comparable in order of magnitude and are not be listed for brevity's sake.

Although the intent of the present work is not to present a high-performance classifier, the classification rates are presented below to discuss the predictive power of these keywords. The sensitivity of the predictor was 0.953 and the specificity was 0.182. The keywords are strong indicators of positive examples, however they have low specificity. This is not particularly surprising, as many of these words are often used in other contexts as part of different topics, and the simplicity of keywords cannot capture all of these nuances.

This same query is run using the Lasso algorithm as described above as well, and the keywords are shown in Table VI. Using the `sklearn` implementation, implemented in `python` that interfaces using `cython` to `C` to take advantage of `libblas3` for BLAS computations, the algorithm takes 26.3 hours to run, computing the Lasso path for only 10 choices of λ. This is the closest environment for comparison to the computation time of the implementation of iterative hard thresholding in this algorithm, which is also implemented in `python` and uses the optimized implementations provided by `scipy` for any linear algebra computations, similar to `sklearn`. The solution is also found using `glmnet` in `R`, a highly optimized package for solving ℓ_1-regularized problems, among others [15]. This algorithm took 354 seconds to find the lasso path for 100 choices of λ.

2) Terrorism: The extent and impact of terrorism has spread in recent years, and news media reflects this. The selected keywords in Table II are clearly associated with terrorism, and in fact reveal the range of reactions to terrorism – from 'war' and 'torture' to issues of 'liberties' and 'peace'. These words are also highly related to each other, suggesting that they are a meaningful summary of the abstract concept of terrorism.

3) Data: The topic of data is analyzed to better understand public perception of data in the news. Not surprisingly, the results of the IHT algorithm (Table III), show that data analysis is represented by a set of cohesive keywords such as 'analyzed', 'processing', 'statistics', and 'analysis', along with the cohesive group of keywords about data acquisition, such as 'measurements', 'recorder', 'census', and 'study'. One of the most interesting keywords related to data is 'privacy', an excellent example of how keyword analysis can be a useful tool for understanding topics in the news.

4) Privacy: Motivated by the appearance of 'privacy' as a keyword for 'data', we explore the topic of privacy itself in the news. The results in Table IV paint an extensive picture of privacy, covering modern issues from eavesdropping to abortion to the internet.

5) Microsoft: [1] investigates keywords associated with Microsoft in the news, using the same corpus of New York Times articles analyzed in the present work. The 25 keywords identified by IHT, shown in Table V share a number of common words with those found previously. We see that competitors such as Yahoo! and Google are listed as keywords for Microsoft, suggesting that their co-occurrence is high in the news. One likely explanation for this is the common practice of mentioning competitors to provide a baseline for comparison of stock prices or earning reports.

C. Discussion

The keywords extracted from topics in the New York Times corpus provide compelling results that iterative hard thresholding is an effective algorithm for this purpose. Efficiently analyzing a corpus of 800,000 documents in less than a minute indicates that the algorithm is practical for a learning problem of this scale. Optimization of this approach by improving the computation of the initial estimate made by matching pursuit would reduce the runtime further.

The iterative hard thresholding algorithm was an order of magnitude faster than the `glmnet` Lasso solver, and many orders of magnitude faster than the `sklearn` implementation. The results are comparable in terms of quality of keywords selected by the algorithm. This suggests that IHT is a not just viable alternative to these algorithms for for keyword extraction, but may be preferable in many cases when quick results are necessary.

VI. CONCLUSION

This article presents a statistical approach to topic keyword extraction using the iterative hard thresholding algorithm. We formulate the problem as a sparse linear model using bag-of-words frequencies as features for a document. Iterative hard thresholding is an efficient algorithm to quickly find an prediction model that is guaranteed to meet the desired level of sparsity. This article extends the analysis of the IHT algorithm, demonstrating that convergence guarantees depend solely on bounding the ℓ_2-norm of the data matrix, $\|X\|_2 < 1$. This analysis includes all matrices arising from word-document occurrence matrices, as the matrix can always be scaled by a constant factor, without loss of generality, to satisfy this requirement.

The present work demonstrates compelling practical results that IHT is both efficient and effective for analyzing large text corpora. The predictors that it finds represent meaningful, interesting keywords for the topic of interest, and the run time is fast enough to be suitable even for interactive analysis.

In a broader sense, the results demonstrated in this article provide an encouraging example that approaching keyword extraction from a statistical text analysis perspective is practical and provides meaningful results. The simplicity of the bag-of-words model provides efficiently computable and easily interpretable results. The iterative thresholding algorithm demonstrates this power, allowing quick analysis of a news archive corpus with over 800,000 documents in less than a minute.

ACKNOWLEDGMENT

The authors would like to thank Jeff Huang and Andrew Godbehere for providing the New York Times dataset and contributing to the pre-processing of data for the analysis in this article.

REFERENCES

[1] B. Gawalt, J. Jia, L. Miratrix, L. El Ghaoui, B. Yu, and S. Clavier, "Discovering word associations in news media via feature selection and sparse classification," in *Proceedings of the international conference on Multimedia information retrieval*. ACM, 2010, pp. 211–220.

[2] L. El Ghaoui, V. Pham, G.-C. Li, V.-A. Duong, A. Srivastava, and K. Bhaduri, "Understanding large text corpora via sparse machine learning," *Statistical Analysis and Data Mining*, vol. 6, no. 3, pp. 221–242, 2013.

[3] A. Hulth, J. Karlgren, A. Jonsson, H. Boström, and L. Asker, "Automatic keyword extraction using domain knowledge," in *Computational Linguistics and Intelligent Text Processing*. Springer, 2001, pp. 472–482.

[4] F. Liu, D. Pennell, F. Liu, and Y. Liu, "Unsupervised approaches for automatic keyword extraction using meeting transcripts," in *Proceedings of human language technologies: The 2009 annual conference of the North American chapter of the association for computational linguistics*. Association for Computational Linguistics, 2009, pp. 620–628.

[5] Y.-N. Chen, Y. Huang, S.-Y. Kong, and L.-S. Lee, "Automatic key term extraction from spoken course lectures using branching entropy and prosodic/semantic features," in *Spoken Language Technology Workshop (SLT), 2010 IEEE*. IEEE, 2010, pp. 265–270.

[6] L. El Ghaoui, G.-C. Li, V.-A. Duong, V. Pham, A. N. Srivastava, and K. Bhaduri, "Sparse machine learning methods for understanding large text corpora." in *CIDU*, 2011, pp. 159–173.

[7] Y. Park, R. J. Byrd, and B. K. Boguraev, "Automatic glossary extraction: beyond terminology identification," in *Proceedings of the 19th international conference on Computational linguistics-Volume 1*. Association for Computational Linguistics, 2002, pp. 1–7.

[8] A. Hulth, "Enhancing linguistically oriented automatic keyword extraction," in *Proceedings of HLT-NAACL 2004: Short Papers*, ser. HLT-NAACL-Short '04. Stroudsburg, PA, USA: Association for Computational Linguistics, 2004, pp. 17–20. [Online]. Available: http://dl.acm.org/citation.cfm?id=1613984.1613989

[9] H. Berger and D. Merkl, "A comparison of text-categorization methods applied to n-gram frequency statistics," in *AI 2004: Advances in Artificial Intelligence*. Springer, 2005, pp. 998–1003.

[10] X. Wang, A. McCallum, and X. Wei, "Topical n-grams: Phrase and topic discovery, with an application to information retrieval," in *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*, Oct 2007, pp. 697–702.

[11] F. Peng and D. Schuurmans, *Combining naive Bayes and n-gram language models for text classification*. Springer, 2003.

[12] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to information retrieval*. Cambridge university press Cambridge, 2008, vol. 1.

[13] M. Marcus, "New trends in natural language processing: statistical natural language processing," *Proceedings of the National Academy of Sciences*, vol. 92, no. 22, pp. 10 052–10 059, 1995.

[14] R. Feldman, M. Fresko, Y. Kinar, Y. Lindell, O. Liphstat, M. Rajman, Y. Schler, and O. Zamir, "Text mining at the term level," in *Principles of Data Mining and Knowledge Discovery*. Springer, 1998, pp. 65–73.

[15] J. Friedman, T. Hastie, and R. Tibshirani, "Regularization paths for generalized linear models via coordinate descent," *Journal of Statistical Software*, vol. 33, no. 1, pp. 1–22, 2010. [Online]. Available: http://www.jstatsoft.org/v33/i01/

[16] D. L. Donoho and M. Elad, "Optimally sparse representation in general (nonorthogonal) dictionaries via 1 minimization," *Proceedings of the National Academy of Sciences*, vol. 100, no. 5, pp. 2197–2202, 2003.

[17] T. Blumensath and M. E. Davies, "Iterative thresholding for sparse approximations," *Journal of Fourier Analysis and Applications*, vol. 14, no. 5-6, pp. 629–654, 2008.

[18] B. Efron, T. Hastie, I. Johnstone, R. Tibshirani *et al.*, "Least angle regression," *The Annals of statistics*, vol. 32, no. 2, pp. 407–499, 2004.

[19] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 267–288, 1996.

[20] B. K. Natarajan, "Sparse approximate solutions to linear systems," *SIAM journal on computing*, vol. 24, no. 2, pp. 227–234, 1995.

[21] T. Blumensath and M. E. Davies, "Iterative hard thresholding for compressed sensing," *Applied and Computational Harmonic Analysis*, vol. 27, no. 3, pp. 265–274, 2009.

[22] M. D. G. Underhill and J. E. Shade, "Usnatrident scholar project report; no. 362 (2007) exploring dimensionality reduction for text mining."

[23] S. G. Mallat and Z. Zhang, "Matching pursuits with time-frequency dictionaries," *Signal Processing, IEEE Transactions on*, vol. 41, no. 12, pp. 3397–3415, 1993.

[24] T. Blumensath, "Accelerated iterative hard thresholding," *Signal Processing*, vol. 92, no. 3, pp. 752–756, 2012.

[25] C. Byrne, "Iterative algorithms in inverse problems," 2006.

[26] S. v. d. Walt, S. C. Colbert, and G. Varoquaux, "The numpy array: A structure for efficient numerical computation," *Computing in Science & Engineering*, vol. 13, no. 2, 2011.

[27] E. Jones, T. Oliphant, P. Peterson *et al.*, "SciPy: Open source scientific tools for Python," 2001–, [Online; accessed 2014-08-03]. [Online]. Available: http://www.scipy.org/

[28] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg *et al.*, "Scikit-learn: Machine learning in python," *The Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[29] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.