Incremental Aspect Models for Mining Document Streams

Arun C. Surendran[1] and Suvrit Sra[2]

[1] Microsoft Research, 1 Microsoft Way, Redmond, WA 98052, USA
acsuren@microsoft.com
[2] Dept. of Computer Sciences, The University of Texas at Austin, TX 78712, USA
suvrit@cs.utexas.edu

Abstract. In this paper we introduce a novel approach for incrementally building aspect models, and use it to dynamically discover underlying themes from document streams. Using the new approach we present an application which we call "query-line tracking" i.e., we automatically discover and summarize different themes or stories that appear over time, and that relate to a particular query. We present evaluation on news corpora to demonstrate the strength of our method for both query-line tracking, online indexing and clustering.

1 Introduction

In this paper we present a new unsupervised framework for mining a stream of documents to extract and summarize the set of underlying themes. The system discovers and isolates themes one by one using a novel approach that combines EM and functional gradient methods. We show that our algorithm naturally leads to incorporating a HITS-like spectral technique within a probabilistic framework, thereby combining the power of both to create a unique document mining framework. The most important functionality of our approach is its ability to handle streaming data without having to retrain the entire model. As a result, the model can grow or shrink as needed leading to a faster, scalable system that permits easier model selection as compared to a batch system.

Based on our incremental framework we present a new application called "queryline tracking" which collects all documents relating to a query over time, and automatically groups and summarizes these into themes. The system automatically keeps track of themes that a user has seen and alerts the user to new themes not seen by him/her, as soon as they are discovered. We further show that our system makes meaningful summaries that correlate well with human concepts, and also has good indexing properties (the model preserves the original distances between documents as much as possible).

2 Incrementally Built Aspect Models (BAM)

BAM is motivated by ideas from density boosting [15,16], incremental EM [15], and aspect models [5]. In spite of the word "boosting" in the title, density

J. Fürnkranz, T. Scheffer, and M. Spiliopoulou (Eds.): PKDD 2006, LNAI 4213, pp. 633–640, 2006.

boosting (also referred to as unsupervised boosting) is closer to semi-parametric maximum likelihood methods [3] than to traditional boosting.

Let the input be a set of documents $D = \{d_1, d_2, \ldots, d_N\}$, where each document is represented by an M-dimensional vector of words taken from a vocabulary $W = \{w_1, w_2, \ldots, w_M\}$. The data is represented by a word-document co-occurrence matrix of size $M \times N$. The frequency of word w in document d is given by n_{wd} (or appropriately weighted versions of it like *tfidf*). Both M and N can vary as new documents are added or older documents are deleted.

The aspect model is a latent variable model [5] where each document is a mixture of underlying aspects, or themes. Each theme is represented by an the distribution of words $p(w|z)$. In PLSI [5] the word and document probabilities are conditionally independent given the aspect z. The joint word-document probability (with K hidden aspects) is

$$F_K(w,d) = \sum_{k=1}^{K} P(z_k)P(d|z_k)P(w|z_k) = (1-\alpha)F_{K-1} + \alpha h_K, \tag{2.1}$$

where $\alpha = P(z_k)$ gives the prior probability that any word-document pair belongs to the K_{th} aspect z_K. Thus, given the current model $F_{K-1}(w,d)$ we wish to compute h_K (the distribution for the individual aspect K) and α (the combining parameter), to obtain F_K using (2.1). Sometimes we will abuse the notation and also refer to h_K as the aspect or latent variable.

A natural objective function for this estimation is the empirical log-likelihood $L_K = \sum_{wd} n_{wd} \log F_K(w,d)$, which may be written as,

$$L_K = \sum_{w,d} n_{wd} \log\big((1-\alpha)F_{K-1}(w,d) + \alpha h_K(w,d)\big) \tag{2.2}$$

$$\geq \sum_{w,d} n_{wd} \left[(1-p_{wd}) \log \frac{(1-\alpha)F_{K-1}}{(1-p_{wd})} + p_{wd} \log \frac{\alpha h_K}{p_{wd}} \right] \equiv Q(\tilde{P}, h_K, \alpha), \tag{2.3}$$

where $\tilde{P} = \{p_{wd}\}\ \forall\ w,d$, and (2.3) provides a "surrogate function" that lower-bounds L_K, and can be maximized instead. Thus, the E-step maximizing (2.3) over p_{wd} is

$$p_{wd} = \frac{\alpha h_K(w,d)}{(1-\alpha)F_{K-1}(w,d) + \alpha h_K(w,d)}. \tag{2.4}$$

Using (2.4) in (2.3) we get $Q(\tilde{P}, h_K, \alpha) = \sum_{w,d} n_{wd} E_{\tilde{P}}\big[\log F_K(w,d)\big]$. In the traditional M-step h_K is estimated such that this Q-function is maximized. Alternatively, we can perform a first-order functional gradient ascent on Q. This idea is similar to the one used in AnyBoost [11], density boosting [16] and semi-parametric methods [3]. To that end, we approximate the difference $L_K - L_{K-1} \approx \sum_{w,d} n_{wd}\alpha\langle \nabla L, h_K - F_{K-1}\rangle$, where $\nabla L = \frac{\partial L(F_{K-1} + \delta \mathbf{1}_{wd})}{\partial \delta}\,|_{\delta=0}$ is the functional derivative of L, and $\langle \nabla L, h_K - F_{K-1}\rangle$ is the directional derivative in the new direction $h_K - F_{K-1}$. To ensure an increase in Q, it is enough to maximize the expected value of the difference $L_K - L_{K-1}$. This leads to a generalized EM approach, wherein the objective function is increased using a unique

combination of EM and functional gradient based approaches. Consequently, we call this new method *Expectation Functional Gradient (EFG)*.

If a data point (w, d) is well explained by the existing model, then, $F_K = F_{K-1}$, and the corresponding directional derivative is zero. Thus $\langle \nabla L, h_K - F_{K-1} \rangle$ is non-zero only for points well represented by h_K. For the log-likelihood in (2.2) the expected change in Q, $E_{\tilde{P}}[L_K - L_{K-1}]$ equals

$$\alpha \sum_{w,d} n_{wd} p_{wd} \langle \nabla L, h_K - F_{K-1} \rangle = \alpha \sum_{w,d} n_{wd}\ p_{wd} \left(\frac{h_K(w,d) - F_{K-1}(w,d)}{F_{K-1}(w,d)} \right). \quad (2.5)$$

We can now maximize $E_{\tilde{P}}[L_K - L_{K-1}]$ by solving

$$h_K = \operatorname*{argmax}_h \sum_{w,d} n_{wd}\ p_{wd} h(w,d)/F_{K-1}(w,d). \quad (2.6)$$

Once h_K has been estimated, α can be estimated using line search on (2.2). We note that as a natural consequence of the above maximization steps, the quantities $1/F_{K-1}(w,d)$ act like weights—data points that are **well represented** by the current model tend to be **down weighted**, and data points that are **poorly represented** tend to be **given more attention** in the next BAM step. This procedure is similar to the one used in boosting.

Estimating h_K: Recall that $h_K(w,d) = p(w|z)p(d|z)$. Let $\boldsymbol{w} = p(w|z_K)$ over all words w, and $\boldsymbol{d} = p(d|z_K)$ over all documents d. Introducing the matrix $\boldsymbol{V} = [n_{wd}\ p_{wd}/F_{wd}]$ we write (2.6) as

$$\min_{\boldsymbol{w},\boldsymbol{d}} \quad Q(\boldsymbol{w},\boldsymbol{d}) = -\boldsymbol{w}^T \boldsymbol{V} \boldsymbol{d}, \qquad \text{where} \quad \boldsymbol{w},\boldsymbol{d} \geq 0. \quad (2.7)$$

However, without further constraints on $\boldsymbol{w}$ and $\boldsymbol{d}$ (2.7) is unbounded. Using a constraint on the L_1 norm of the vector (which gives it a probabilistic interpretation) leads to a formulation similar to PLSI (discussed later). Alternatively we could use a regularization restricting the magnitude of the $\boldsymbol{w}$ and $\boldsymbol{d}$ vectors, and then later re-interpret them as probabilities. This can be done in a principled way starting with (2.2) to incorporate a regularizer. The rest of the analysis will remain unchanged, but the regularized form of the function to be minimized in (2.7) can be modified to be [1]

$$Q(\boldsymbol{w},\boldsymbol{d}) = -\boldsymbol{w}^T V \boldsymbol{d} + \nu(\boldsymbol{w}^T \boldsymbol{w}) + \mu(\boldsymbol{d}^T \boldsymbol{d}).$$

where μ and ν are regularization parameters. Differentiating $Q(\boldsymbol{w},\boldsymbol{d})$ with respect to $\boldsymbol{w}$ and $\boldsymbol{d}$ and setting the derivatives to equal zero we obtain the system of equations

$$\boldsymbol{w} = \frac{\boldsymbol{V}\boldsymbol{d}}{\nu}, \qquad \boldsymbol{d} = \frac{\boldsymbol{V}^T \boldsymbol{w}}{\mu}, \quad (2.8)$$

which can be solved iteratively. Setting $\boldsymbol{w}$ and $\boldsymbol{d}$ to the left and right singular vectors (with $\nu,\ \mu = 1$) of $\boldsymbol{V}$ provides the solution. Readers will notice the similarity of these steps to the popular HITS algorithm [7] which is a spectral

[1] If $\boldsymbol{w},\boldsymbol{d}$ are initialized to be positive, they will stay positive, and the solution will satisfy the non-negativity constraints.

method. Due to its similarity to a spectral algorithm we call (2.8) the *spectral M-step*. Since both $\boldsymbol{w}$ and $\boldsymbol{d}$ are non-negative, we can normalize them and interpret them as probabilities. We obtain $h_K = \boldsymbol{w}\boldsymbol{d}^T$ as the new aspect. Thereafter, we determine α using line search and then obtain the updated model $F_K = (1-\alpha)F_{K-1} + \alpha h_K$.

2.1 Understanding How BAM Works

Relation to PLSI: In PLSI, the traditional M-step leads to the solution $p(w|z) \propto \sum_d n_{wd}p_{wd}$ and $p(d|z) \propto \sum_w n_{wd}p_{wd}$, which can be re-written as

$$\boldsymbol{w} \propto \boldsymbol{V}\mathbf{1}, \qquad \boldsymbol{d} \propto \boldsymbol{V}^T\mathbf{1}, \tag{2.9}$$

where $\boldsymbol{V} = [n_{wd}p_{wd}]$. Comparing this with (2.8) we can see that this M-step (2.9) is essentially one spectral step with $\boldsymbol{w}$ and $\boldsymbol{d}$ initialized to $\mathbf{1}$ (and an unweighted $\boldsymbol{V}$). The regular PLSI M-step simply computes an average of the pre-weighted data. Due to this behavior, in an incremental setting, PLSI leads to a sequence of averaged models, whereas BAM produces a sequence of coherent topics. It is difficult to grow PLSI models to accommodate new topics, while BAM is built to do so. One has to rerun PLSI with an increased number of aspects - increasing the computational needs with no guarantee that the new themes will correspond to the old ones, or that the old ones will be rediscovered. Model selection can also be a problem with PLSI—multiple runs with different model sizes are needed. Computationally BAM is faster since, unlike PLSI, it does not need annealing to work well, plus spectral algorithms tend to exhibit fast convergence properties.

Following (2.8), we suggest a way to modify PLSI to grow the model - before estimating a new aspect, the data is weighted by $1/F_{K-1}$. Then the regular M-step is replaced by the spectral M-step, i.e., iterate as per (2.8) until convergence. This is equivalent to replacing $\boldsymbol{V}$ in (2.8) by $\boldsymbol{V} = [n_{wd}p_{wd}]$. In practice, both these approaches seem to yield similar results, so in this paper we use the latter.

Relation to HITS: The updates in (2.8) are strikingly similar to another very popular algorithm used in the IR community to rank a set of web documents into authorities and hubs—HITS [7]. BAM uses a similar idea to rank words and documents. HITS can be shown to discover tightly knit clusters (TKC) based on link structure [9]; similarly, BAM discovers TKCs based on how strongly words and documents are connected to each other, which is what we want to achieve. Thus BAM looks at the data weighted by $1/F$ (which defines the search space), and then finds an appropriate cluster within this space. We can think of these two steps as *restriction* and *discovery* steps respectively. This process is repeated till all relevant topics are found. BAM also reduces the chances of mixing up weakly connected components, i.e., it reduces the chances of discovering mixed topics. The TKC issue can be a problem for HITS because sometimes the best cluster is not the most relevant one. For BAM, this is not a problem; in fact it is an advantage. BAM can be thought of as performing a series of soft cuts on the bipartite graph to extract many tightly-knit overlapping components from it.

Relation to other spectral graph partitioning approaches: There are other ways to partition these graphs, e.g. normalized cut using the Fiedler vec-

tor (eigenvector corresponding to the second smallest generalized eigenvalue [4]. BAM uses the top left and right singular vectors. Norm-cut looks at both inter- and intra-partition properties while our algorithm is thought to target the tightness of the partition in question.

The convergence properties of BAM depend on the nature of the weighted word-document graph at each BAM step. Due to lack of space, we defer a detailed analysis of stability and convergence to future publications.

Other Relevant Work: We have briefly discussed BAM's relation to PLSI. Latent Dirichlet allocation (LDA) [2] addresses many of the issues faced by PLSI, including the ability to grow with data [2,6], but at commensurate additional computational costs. BAM could be extended to LDA, something that we defer to the future. LSI is a spectral method to finding topics but lacks generative semantics. There exist some incremental approaches to LSI (e.g. [1])

There has been a significant amount of work in topic tracking and detection [8] but we only mention a couple that are very closely related to our method. The work by Kumar, 2004 [14] uses non-probabilistic, graph theoretic ideas to extract storylines. It cannot handle overlapping categories, and is not dynamic or incremental. Another recent work analyzes chunks of data over time using static PLSI-like models, but tracking is done using similarity of aspects across time scales [12].

2.2 Handling Streaming Data

To handle streaming data, first we need to understand how much of the new data is already explained by the existing models. We use a "fold-in" approach similar to the one suggested in [5]. For each aspect z_k we keep $p(w|z_k)$ values fixed for all the words that are already seen. We then use the spectral step to estimate the probabilities of the new words (the $p(w|z)$ vectors are normalized as needed), and the document probabilities $p(d|z_k)$. Using the estimated probabilities we compute a new F for all the new data, and use this new F as a starting point to discover new themes as needed.

3 Experiments and Results

We present some preliminary results on news corpora to demonstrate the performance of our algorithm in queryline tracking, indexing and clustering.

3.1 Tracking Storylines Around a Query

We present an interesting application of BAM, called *Queryline Tracking*, which is a mixture of TREC tasks like filtering and novelty detection, but is centered around a query. Essentially, this task involves discovering and tracking themes or storylines [14] based upon a query. As new data comes in, we only wish to surface new themes. We demonstrate this idea on the publicly available RCV1-v2 Reuters news corpus (23,000 documents) [10]. We use data from the first 10 days (Aug 20-30 1996) and run BAM on it incrementally, one day at a time.

Table 1. BAM captures all the storylines around the news alert "Clinton" over a period of 10 days. Days 4 and 6 did not have enough articles

Day 1 Storylines					Day 2	Day 3
Dole	Wage	McDougal	drug	Gingrich	McCurry	Zogby
Powell	Clinton	Whitewater	Hatch	terror	Yeltsin	vote
Clinton	Minimum	Susan	Marijuana	nuclear	Chechnya	gap
convention	bill	Arkansas	coppl	Libya	strenuous	poll
Kemp	legist.	sentence	McCurry	Iraq	Russian	narrow
Day 5	Day 7	Day 8	Day 9	Day 10		
FDA	train	read	Jackson	Morris	litigation	
tobacco	handgun	Jeep	Cuomo	resign	tobacco	
smoke	Brady	liter	Jesse	Dick	lung	
cigarette	Ohio	Americorp	welfare	prostitute	cancer	
advert	Huntington	Cherokee	disagree	tabloid	letter	

We demonstrate the idea using the query "Clinton". Each day we collect the documents that contain the word "Clinton". At the end of each day, we if we have less than 20 documents, we defer the documents to the next day. Otherwise we run BAM on the data to discover new storylines. We can see the results in Table 1.

The first day has stories about the presidential election, Clinton signing a bill to raise minimum wage, The Whitewater case, Senator Hatch complaining to the President about increase in drug use, and Gingrich cautioning the President that the country needs to be preemptively deal with external nuclear threats. On subsequent days the system discovers stories about Chechnya, Zogby's election tracking poll, Clinton asking the FDA to move against illegal practices in tobacco advertisement, etc. Items belonging to the older themes are subsumed by the older models and are available, but not surfaced. We can demonstrate the advantage of the incremental algorithm by showing that for the same number of topics discovered, the static model rediscovers themes from previous days.

We can show similar results using other queries (omitted due to lack of space). The rest of the experiments in the paper are on query independent tasks.

3.2 Indexing Power

Just like LSI, BAM can also be used as an indexing algorithm. We evaluate this by measuring how well it preserves distances between documents in the lower dimensions. We demonstrate this using 1-nearest neighbor (1-NN) comparison on the well known Reuters-21578 set. This dataset has 9063 documents in the training set and 3699 documents in the testing set, with 22226 words and spans 113 different topics. First, we build a K-aspect BAM model using the training data and then take the dot product of the data vector with each $p(w|z)$ vector. Hence the documents are projected from 22226 dimensional space to K dimensions. Now for each projected document d, we choose the nearest neighbor n and compare the labels of n against the true class labels for d. Let A be the number

Table 2. F_1 scores for the 113 class Reuters-21758 dataset using 1-NN. LMDS scores are from [13].

No. of	BAM		LMDS
dimensions	Testing data	Training	Training
10	0.584	0.606	0.625
20	0.677	0.697	0.714
50	0.724	0.757	0.754
100	0.750	0.777	0.768

Table 3. NMI and Rand Index scores for RCV12 and Reuters datasets, for BAM and PLSI. (Higher scores are better) * signifies that these difference of these scores from those of BAM are statistically significant at the level of 0.02.

	RCV1-v2 subset		Reuters subset	
	NMI	Rand	NMI	Rand
BAM	0.54	0.49	0.56	0.32
PLSI	0.54	0.49	0.51*	0.26*

of matching labels over all documents, and B be the number of unmatched labels. The microaveraged F_1 score is then $F_1 = A/(A + B/2)$. For test data, we can follow a similar procedure except we choose the nearest neighbor from the *training set* in the reduced space. Table 2 shows the average F_1 numbers as K is varied from 10 to 100.

The F_1 score using the unprojected training data was 0.719 [13]. BAM matches this score when using around 50 aspects. The results on the unseen test data is similar to that on the training data. Training data results for BAM are comparable to Landmark MDS (which is similar to LSI) [13].BAM sacrifices a little bit of indexing power to gain the ability to grow, and to create higher quality topics.

3.3 Correlation of Aspects with Human Labels

The goal of this section is to quantitatively show that the aspects created by BAM correlate well with human labeling. These experiments are done on two selected subsets with 10 topics each - one from Reuters-21578 (8009 documents) and another from RCV1-v2 (14814 documents). We run BAM incrementally to get 10 aspects. Then, we do a hard classification for each document by assigning it to the aspect with highest $p(z|d)$ value. We then compute normalized mutual information (NMI) and the Rand index for the resulting partitions by making use of the topic labels assigned by human experts. We average over 10 runs to get the numbers shown in on Table 3. BAM performs significantly better than PLSI on the Reuters subset and performs just as well on the RCV1-v2 subset. The number of documents per class is similar across classes in the RCV1-v2 subset, and most clustering algorithms tend to do well on such sets. The Reuters subset is very unbalanced, and we see that BAM does better on this set.

4 Summary

In this paper we have introduced a new framework for building incremental aspect models incorporating the strengths of both spectral and probabilistic methods. The main advantage of this method is that it can handle documents arriving in a stream, and the model can grow or shrink as needed. We demonstrated some of the capabilities of the new approach in indexing and clustering. Using the new framework we presented a new application called "queryline tracking".

Acknowledgments. We thank John Platt, Chris Meek and Asela Gunawardana for valuable discussions.

References

1. R. K. Ando and L. Lee. Iterative Residual Rescaling: An Analysis and Generalization of LSI. In *SIGIR*, September 2001.
2. D. M. Blei, T. L. Griffiths, M. I. Jordan, and J. B. Tenenbaum. Hierarchical Topic Models and the Nested Chinese Restaurant Process. In *NIPS*, 2004.
3. D. Böhning. A review of reliable maximum likelihood algorithms for semiparametric mixture models. *J. of Stat. Planning and Inference*, 47:5–28, 1995.
4. I. S. Dhillon. Co-clustering documents and words using bipartite spectral graph partitioning. In *Knowledge Discovery and Data Mining*, pages 269–274, 2001.
5. T. Hofmann. Unsupervised learning by Probabilistic Latent Semantic Analysis. *Machine Learning*, 42, 2001.
6. Z. G. J. Zhang and Y. Yang. A Probabilistic Model for On-line Document Clustering with Application to Novelty Detection. In *NIPS*, 2005.
7. J. Kleinberg. Authoritative sources in a hyperlinked environment. *ACM*, 46(5):604–632, 1999.
8. A. Kontostathis et al. A survey of emerging trends detection in textual data mining. In *Survey of Text Mining*, pages 185–224, 2003.
9. R. Lempel and S. Moran. The stochastic approach for link-structure analysis (salsa). In *ACM Tran. on Info. Sys.*, volume 19, pages 131–160, 2001.
10. D. D. Lewis, Y. Yang, T. Rose, and F. Li. RCV1: A New Benchmark Collection for Text Categorization Research. *JMLR*, 5:361–397, 2004.
11. L. Mason, J. Baxter, P. Bartlett, and M. Frean. Boosting Algorithms as Gradient Descent in Function Space. In *NIPS*, pages 512–518, 2000.
12. Q. Mei and C. Zhai. Discovering evolutionary theme patters from text - an exploration of temporal mining. In *Knowledge Discovery and Data Mining*, pages 198–207, 2005.
13. J. Platt. FastMap, MetricMap, and Landmark MDS are all Nystrom Algorithms. In *10th International Workshop on AI and Statistics*, pages 261–268, 2005.
14. U. M. R. Kumar and D. Sivakumar. A graph-theoretic approach to extracting storylines from search results. In *Knowledge Discovery and Data Mining*, pages 216–225, 2004.
15. G. Ridgeway. Looking for lumps: boosting and bagging for density estimation. *Computational Statistics and Data Analysis*, 38(4):379–392, 1999.
16. S. Rosset and E. Segal. Boosting density estimation. In *NIPS*, 2002.