Distributed LDA based Topic Modeling and Topic Agglomeration in a Latent Space

Gopi Chand Nutakki
Knowlede Discovery & Web Mining Lab
University of Louisville
g0nuta01@louisville.edu

Olfa Nasraoui
Knowlede Discovery & Web Mining Lab
University of Louisville
olfa.nasraoui@louisville.edu

Behnoush Abdollahi
Knowlede Discovery & Web Mining Lab
University of Louisville
b0abdo03@louisville.edu

Mahsa Badami
Knowlede Discovery & Web Mining Lab
University of Louisville
m0bada01@louisville.edu

Wenlong Sun
Knowlede Discovery & Web Mining Lab
University of Louisville
w0sun005@louisville.edu

Abstract

We describe the methodology that we followed to automatically extract topics corresponding to known events provided by the SNOW 2014 challenge in the context of the SocialSensor project. A data crawling tool and selected filtering terms were provided to all the teams. The crawled data was to be divided in 96 (15-minute) timeslots spanning a 24 hour period and participants were asked to produce a fixed number of topics for the selected timeslots. Our preliminary results are obtained using a methodology that pulls strengths from several machine learning techniques, including Latent Dirichlet Allocation (LDA) for topic modeling and Non-negative Matrix Factorization (NMF) for automated hashtag annotation and for mapping the topics into a latent space where they become less fragmented and can be better related with one another. In addition, we obtain improved topic quality when sentiment detection is performed to partition the tweets based on polarity, prior to topic modeling.

In: S. Papadopoulos, D. Corney, L. Aiello (eds.): Proceedings of the SNOW 2014 Data Challenge, Seoul, Korea, 08-04-2014, published at http://ceur-ws.org

1 Introduction

The SNOW 2014 challenge was organized within the context of the SocialSensor project[1], which works on developing a new framework for enabling real-time multimedia indexing and search in the Social Web. The aim of the challenge was to automatically extract topics corresponding to known events that were prescribed by the challenge organizers. Also provided, was a data crawling tool along with several Twitter filter terms (syria, ukraine, bitcoin, terror). The crawled data was to be divided in a total of 96 (15-minute) timeslots spanning a 24 hour period, with a goal of extracting a fixed number of topics in each timeslot. Only tweets up to the end of the timeslot could be used to extract any topic. In this paper, we focuse on the topic extraction task, instead of input data filtering, or presentation of associated headline, tweets and image URL, because this was one of the activities closest to the ongoing research [AN12, HN12, CBGN12] on multi-domain data stream clustering in the Knowledge Discovery & Web Mining Lab at the University of Louisville. To extract topics from the tweets crawled

[1] SocialSensor: http://www.socialsensor.eu/

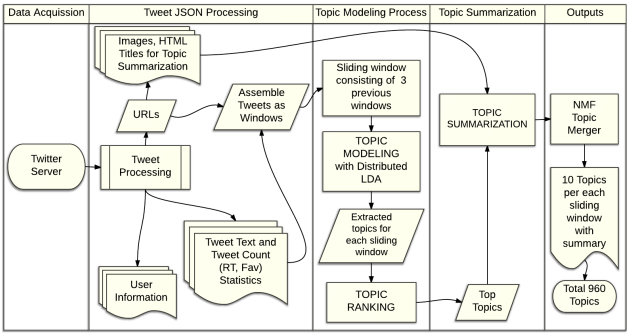

Figure 1: Topic Modeling Framework (sentiment detection and hashtag annotation are not shown).

in each time slot, we use a Latent Dirichlet Allocation (LDA) based technique. We then discover latent concepts using Non-negative Matrix Factorization (NMF) on the resulting topics, and apply hierarchical clustering within the resulting Latent Space (LS) in order to agglomerate these topics into less fragmented themes that can facilitate the visual inspection of how the different topics are inter-related. We have also experimented with adding a sentiment detection step prior to topic modeling in order to obtain a polarity sensitive topic discovery, and automated hashtag annotation to improve the topic extraction.

2 Background

2.1 Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA) is a Bayesian probabilistic model for text documents. It assumes a collection of K topics where each topic defines a multinomial over the vocabulary, which is assumed to have been drawn from a Dirichlet process [BNJ03][HBB10]. Given the topics, LDA assumes the generative process for each document d, shown in Algorithm 1, where the notation is listed in Table 1. Equation 1 gives the joint distribution of a topic mixture θ, a set of N topics z, and a set of N words w for parameters α and β.

$$p(\theta, z, w|\alpha, \beta) = p(\theta|\alpha) \prod_{n=1}^{N} p(z_n|\theta)\, p(w_n|z_n, \beta) \quad (1)$$

Integrating over θ and summing over z, we obtain the marginal distribution of a document [BNJ03]:

$$p(w|\alpha, \beta) = \int p(\theta|\alpha) \left(\Pi_{n=1}^{N} \sum_{z_n} p(z_n|\theta)\, p(w_n|z_n, \beta) \right) d\theta$$

Taking the product of the marginal probabilities of single documents, the probability of a corpus D can be obtained:

Table 1: Description of used variables.

Symbol	Description
M	Number of documents in collection
W	Number of distinct words in vocabulary
N	Total number of words in collection
K	Number of topics
x_{di}	i^{th} observed word in document d
z_{di}	Topic assigned to x_{di}
N_{wk}	Count of word assigned to topic
N_{dk}	Count of topic assigned in document
ϕ_k	Probability of word given topic k
θ_d	Probability of topic given document d
α,β	Dirichlet priors

Algorithm 1 Latent Dirichlet Allocation.

Input: *A document collection, hyper-parameters α and β.*

Output: *A list of topics.*

1. Draw a distribution over topics, $\theta_d \sim Dir(\alpha)$
2. **For Each** word i in the document:
3. Draw a topic index $z_{di} \in \{1, \cdots, K\}$ from the topic weights $z_{di} \sim \theta_d$.
4. Draw the observed word w_{di} from the selected topic, $w_{di} \sim \beta_{z_{di}}$

$$p(D|\alpha, \beta) = \Pi_{d=1}^{M} \int p(\theta_d|\alpha) \left(\Pi_{n=1}^{N_d} \sum_{z_{dn}} p(z_{dn}|\theta_d)\, p(w_{dn}|z_{dn}, \beta) \right) d\theta_d$$

The posterior is usually approximated using Markov Chain Monte Carlo (MCMC) methods or variational inference. Both methods are effective, but face significant computational challenges in the face of massive data sets. For this reason, we concentrated on a distributed version of LDA which is summarized in the next section.

2.2 Distributed Algorithms for LDA

It is possible to distribute non-collapsed Gibbs sampling, because sampling of z_{di} can happen independently given θ_d and ϕ_k, and thus can be done concurrently. In a non-collapsed Gibbs sampler, one samples z_{di} given θ_d and ϕ_k, and then θ_d and ϕ_k given z_{di}. If individual documents are not spread across different processors, one can marginalize over just θ_d, since θ_d is processor-specific. In this partially collapsed scheme,

the latent variables z_{di} on each processor can be concurrently sampled where the concurrency is over processors. The slow convergence of partially collapsed and non-collapsed Gibbs samplers (due to the strong dependencies between the parameters and latent variables) has led to devising distributed algorithms for fully collapsed Gibbs samplers [NASW09][YMM09].

Given M documents and P processors, with approximately $M_P = \frac{M}{P}$ documents, distributed on each processor p, the M documents are partitioned into $x = \{x_1, \cdots, x_p, \cdots, x_P\}$ and $z = \{z_1, \cdots, z_p, \cdots, z_P\}$ being the corresponding topic assignments, where processor p stores x_p, the words from documents $j = (p-1)\,M_P + 1, \cdots, pM_P$ and z_P, the corresponding topic assignments. Topic-document counts N_{dk} are likewise distributed as N_{dkp}. The word-topic counts N_{wk} are also distributed, with each processor p keeping a separate local copy N_{wkp}.

Algorithm 2 Standard Collapsed Gibbs Sampling.

LDAGibbsItr($|x_p|$, z_p, N_{dkp}, N_{wkp}, α, β):

1. **For Each** $d \in \{1, \cdots, M\}$
2. **For Each** $i \in \{1, \cdots, N_{dkp}\}$
3. $v \leftarrow x_{dpi}, T_{dpi} \leftarrow N_{dkpi}$
4. **For Each** $j \in \{1, \cdots, T_{dkpi}\}$
5. $\hat{k} \leftarrow z_{dpij}$
6. $N_{dkp} \leftarrow N_{dkp} - 1,\ N_{wkp} \leftarrow N_{wkp} - 1$
7. **For** $k = 1$ to K
8. $\rho_k \leftarrow \rho_{k-1} + (N_{dkp} + \alpha) \times (N_{kwp} + \beta) \,/\, (\sum_{w'} N_{w'k}) + N\beta$
9. $x \sim UniformDistribution(0, \rho_k)$
10. $\hat{k} \leftarrow BinarySearch\left(\hat{k} : \rho_{\hat{k}-1} < x < \rho_{\hat{k}}\right)$
11. $N_{d\hat{k}p} \leftarrow N_{d\hat{k}p} + 1, N_{w\hat{k}p} \leftarrow N_{w\hat{k}p} + 1$
12. $z_{dpij} \leftarrow \hat{k}$

Although Gibbs sampling is a sequential process, given the typically large number of word tokens compared to the number of processors, the dependence of z_{ij} on the update of any other topic assignment $z_{i'j'}$ is likely to be weak, thus relaxing the sequential sampling constraint. If two processors are concurrently sampling, but with different words in different documents, then concurrent sampling will approximate sequential sampling. This is because the only term affecting the order of the update operations is the total word-topic counts $\sum_w N_{wk}$. Algorithm 3 shows the pseudocode

Algorithm 3 Approximate Distributed LDA [NASW09].

Input: *A list of M documents,* $x = \{x_1, \cdots, x_p, \cdots, x_P\}$

Output: $z = \{z_1, \cdots, z_p, \cdots, z_P\}$

1. **Repeat**
2. **For** each processor p in parallel **do**
3. Copy global counts: $N_{wkp} \leftarrow N_{wk}$
4. Sample z_p locally: LDAGibbsItr(x_p,z_p,N_{dkp},N_{wkp},α,β) // Alg: 2
5. Synchronize
6. Update global counts: $N_{wk} \leftarrow N_{wk} + \sum_p (N_{wkp} - N_{wk})$
7. **Until** termination criterion is satisfied

of the AD-LDA algorithm which can terminate after a fixed number of iterations, or based on a suitable MCMC convergence metric. The AD-LDA algorithm samples from an approximation to the posterior distribution by allowing different processors to concurrently sample topic assignments on their local subsets of the data. AD-LDA works well empirically and accelerates the topic modeling process.

3 Topic Extraction Methodology

3.1 Data Preprocessing

The dataset consists of tweets that were acquired from the Twitter servers by continuous querying using a wrapper for the Twitter API over a period of 24 hours. The batch of tweets are acquired in raw JSON[2] format. Various properties of the tweet such as the hashtags, URLs, creation time, counts for retweets and favorites, and other user information including the encoding and language are extracted. The hashtags can provide a good source for creating discriminating features and they were folded as terms into the bag of words model for each tweet where they were present (without the '#' prefix). The URLs can also later provide a method to achieve topic summarization.

3.2 Topic Extraction Stages

The technique assumes a real time streaming data input and is replicated using process calls to the storage records containing the tweets. For AD-LDA, each

[2]JSON: JavaScript Object Notation, is a text-based open standard designed for human-readable data interchange

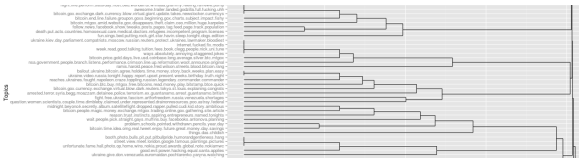

Figure 2: Dendrogram depicting few clusters' hierarchy of topics from the initial window. Agglomeration is based on the cosine similarity. Average-Linkage Agglomerative Hierarchical Clustering was used. Distance is computed as $(1 - similarity)$. Refer to the electronic version of the paper for clarity

tweet is considered as a single document. Figure 1 shows the steps performed to extract the topics in each window or time slot. The procedure starts with the extraction of key information from the Twitter JSON, then the tweet text and other properties are used to extract topics. The topic extraction is performed using the following steps:

1. The documents are stripped of non-English characters and are converted to lowercase. The stop words are retained for the context information (especially for sentiment detection).

2. Groups of documents are assembled into windows based on their *timestamp*. A sliding window's width is equal to three consecutive time slots ending in the current time slot.

3. AD-LDA technique is performed on a 20 node cluster. From each sliding window iteration a total of 1000 topics are extracted, this higher value help in extracting finer topics.

4. The topics are ranked based on the proportion of tweets assigned to the topic in the given window, and then can be clustered/merged together to organize them into more general topic groups.

The jsoup open source HTML parser[3] was used to extract multimedia content such as images and metadata from the URLs extracted from the tweets. The headlines are part of the metadata while the keywords are obtained from the topic modeling itself as the terms with highest probability in the topic.

4 Topic Extraction with AD-LDA and Sentiment Labels

The AD-LDA technique with Gibbs sampling, along with automatically extracted sentiment labels can also be used to extract polarity-sensitive topics. Using sentiment labels may improve the quality of the topics as it results in finer topics. Figure 1 depicts the general flow within the used methodology. A weighted Naive

[3]http://jsoup.org/

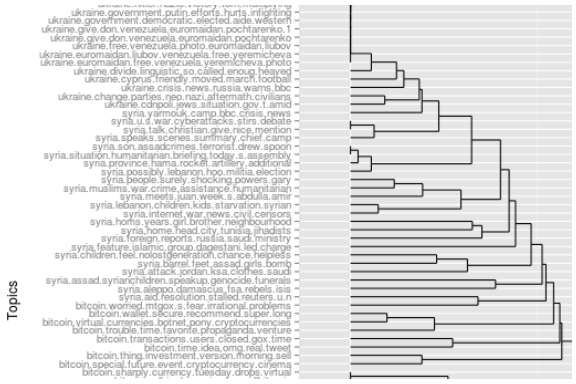

Figure 3: Portion of dendrogram depicting the clusters' hierarchy of topics from the initial window (Number 0). Agglomeration is based on the dot product between the topics' projections on a lower dimensional latent space extracted using NMF with $k_f = 30$ factors. Average-Linkage Agglomerative Hierarchical Clustering was used. Distance is computed as 1 minus similarity. Refer to the electronic version of the paper for clarity.

Bayes classifier [LGD11] trained with labeled tweet samples[4] and a set of labeled tokens[5] with known sentiment polarity can be used to extract the sentiment levels. The tweets are then regrouped based on the sentiment level and the topic modeling is applied on each group, resulting in topics that are confined to one sentiment, as illustrated in Table 2.

5 Topic Agglomeration in a Latent Space

5.1 Discovering Latent Factors Among the Discovered Topics Using Non-negative Matrix Factorization (NMF)

Because initial topic modeling generated a high number of topics (1000 topics per window), that were furthermore very sparse in terms of the descriptive terms within them, these topics were hard to interpret and could benefit from a coarser, less fragmented organization. One way to fix this problem was to merge the topics based on a conceptual similarity by applying Non-negative Matrix Factorization (NMF) [LS99]. Because the topics by words matrix is a very sparse matrix, we used NMF to project the topics onto a common lower-dimensional latent factors' space. NMF takes as input the matrix X of n topics by m words (as binary features) and decomposes it into two factor matrices (A and B) which represent the topics and words, respectively, in a k_f-dimensional latent space, as follows:

$$\mathbf{X}_{n \times m} \simeq \mathbf{A}_{n \times k_f} \mathbf{B^T}_{m \times k_f} \quad (2)$$

[4]https://github.com/ravikiranj/twitter-sentiment-analyzer
[5]https://github.com/linron84/JST

Positive Sentiment	Negative Sentiment
optimistic ukraine antiwar nonintervention	horrible building badge hiding ukraine yanukovych
syria refugees about education children million	syria yarmouk camp crisis food waiting unrest shocking
future technology bitcoins value law accelerating	cnn protocols loss gox bitcoin fault

Table 2: Illustrating a sample of the finer topics extracted after a preliminary sentiment detection phase.

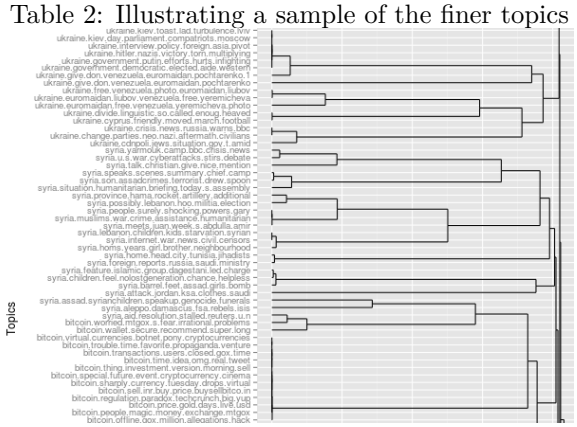

Figure 4: Portion of dendrogram depicting the clusters' hierarchy of topics from the first 6 windows. Agglomeration is based on the dot product between the topics' projections on a lower dimensional latent space extracted using NMF with $k_f = 30$ factors. Average-Linkage Agglomerative Hierarchical Clustering was used. Distance is computed as 1 minus similarity. Refer to the electronic version of the paper for clarity.

where k_f is the approximated rank of matrices A and B, and is selected such that $k_f < min(m, n)$, so that the number of elements in the decomposition matrices is far less than the number of elements of the original matrix: $nk_f + k_f m \ll nm$.

Topics factor (A) can then be used to find the similarity between the topics in the new latent space instead of using the original space of original terms. the obtained similarity matrix from the NMF factors can finally be used to cluster the topics.

To find A and B, the Frobenius norm of errors between the data and the approximation is optimized, as follows

$$J_{NMF} = ||\mathbf{E}||_F^2 = ||\,\mathbf{X} - \mathbf{AB^T}\,||_F^2 \qquad (3)$$

Several algorithms have been proposed in the literature to minimize this cost. We used an Alternating Least Square (ALS) method [PT94] that iteratively solves for the factors, by assuming that the problem is convex in either one of the factor matrices alone.

Algorithm 4 *Basic Alternating Least Square (ALS) Algorithm for NMF*

Input: Data matrix $\mathbf{X}$*, number of factors* k_f
Output: optimal matrices $\mathbf{A}$ *and* $\mathbf{B}$

1. *Initialize matrix* $\mathbf{A}$ *(for example randomly)*
2. **Repeat**
   (a) *Solve for* $\mathbf{B}$ *in the equation:* $\mathbf{A}^T\mathbf{AB} = \mathbf{A}^T\mathbf{X}$
   (b) *Project solution onto non-negative matrix subspace: set all negative values in* $\mathbf{B}$ *to zeros*
   (c) *Solve for* $\mathbf{A}$ *in the equation:* $\mathbf{BB}^T\mathbf{A}^T = \mathbf{BX}^T$
   (d) *Project solution onto non-negative matrix subspace: set all negative values in* $\mathbf{A}$ *to zeros*
3. **Until** *Cost function decrease is below threshold*

5.2 Topic Organization Stages: Topic Feature Extraction, Latent Space Computation using NMF, Latent Space-based Topic Similarity Computation, and Hierarchical Clustering

In the following, we summarize the steps that are applied post-discovery of the topics, in order to generate a hierarchical organization from the sparse topics.

1. *Preprocessing of the topic vectors*: For each window, the topic-word matrix ($\mathbf{X}_{n\times m}$) is extracted from the final topic modeling results. The features are the top words in a topic, and they are binary (1 if a topic has the word in question and 0 otherwise).
2. *Latent Factor Discovery using NMF*: The topic-word data was normalized before running NMF. The latter produces two factors (A and B), where n, k_f and m are the number of topics, latent factors and words, respectively. Our main goal was to compute the Matrix A, also called the topics basis factor, which transfers the topics to the latent space. Choosing the number of factors), k_f, has an impact on the results,. After trial and error, we chose $k_f = 30$.
3. *Generating the topic-similarity matrix in the la-*

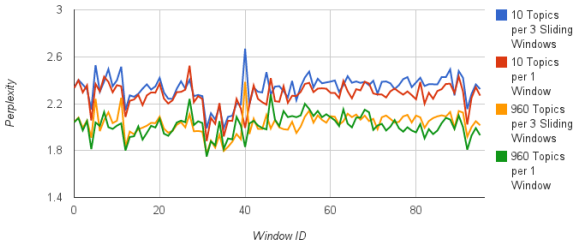

Figure 5: Perplexity trends for each sliding window of width three for various numbers of extracted topics.

tent space: The computed topic basis matrix (A) was used to obtain the similarity in lieu of the original topic vectors. The normalized inner product of the matrix A and its transpose was calculated for this purpose. Normalization of the product is equivalent to computing the Cosine similarity between topic pairs. The resulting matrix contains the pairwise similarity between each pair of topics within the latent space.

4. *Hierarchical Clustering of the latent space-projected topics based on the new pairwise similarity scores computed in Step 3*: we experimented with several linkage strategies such as single and average linkage. The latter was chosen as optimal.

5.3 Automated Hashtag Annotation

We have also experimented with a simple tag completion or prediction step prior to topic modeling. Annotation for a given tweet is determined by finding the top frequent tags associated with the K_{LS}[6] nearest neighboring tweets in the NMF-computed Latent Space to the given tweet. Once the tags are completed, they are used to enrich the tweets before topic modeling. Of course, only the tweets bag of word descriptions in a given window are used to compute the NMF for that window's topic modeling. The annotation generally resulted in lower Perplexity of the extracted topic models, as shown in Figure 6.

6 Results

6.1 Distributed LDA-based Topic Modeling

Figure 2 shows[7] a sample of the topic clusters' hierarchy extracted from the initial window and without NMF-based latent space projection of the topics. The clusters are of debatable quality. Perplexity is a common metric to evaluate language models [BL06][BNJ03]. It is monotonically decreasing in the likelihood of the test data, with a lower perplexity score indicating better generalization performance. For a test set of T documents $D' = \left\{\vec{w}^{(1)}, \cdots, \vec{w}^{(T)}\right\}$ and N_d being the total number of keywords in d^{th}document, the perplexity given in Equation 4, will be lower for a better topic model. Figure 5 shows the perplexity trends, suggesting that more topics result in lower (thus better) perplexity. Also, irrespective of the number of topics, AD-LDA-based topic modeling can extract topics of good quality.

$$perplexity\,(D') = \exp\left\{-\frac{\sum_{d=1}^{T} \ln p\left(\vec{w}^{(d)}|\vec{\alpha},\beta\right)}{\sum_{d=1}^{T} N_d}\right\} \quad (4)$$

6.2 Sentiment Based Topic Modeling

Table 2 shows a subset of topics, extracted from the positive and negative sentiment groups of tweets, and these tend to be more refined than the standard unsentimental topics. From the initial window, 1000 topics were extracted in the same way as the Distributed LDA, however topic modeling was preceded by a sentiment classifier that classifies the tweets based on their sentiment (positive or negative). Although positive and negative topics still share a few keywords, they are clearly divided by sentiment.

6.3 Topic Clustering in the Latent Space

Figure 3 shows the topic clusters created using the latent space-projected features extracted using NMF. The clusters in Figure 3 seem to have better quality compared to the clusters in Figure 2 because of the more accurate capture of pairwise similarities between topics in the conceptual space. Figure 4 shows the clustering of the top 10 topics for a series of 6 windows, showing how the agglomeration can consolidate the topics discovered at different time slots, helping avoid excessive fragmentation throughout the stream's life.

6.4 Automated Hashtag Annotation

Tweet data is very sparse and not every tweet has valuable tags. To overcome this weakness, we applied an NMF-based automated tweet annotation before topic modeling. Adding the predicted hashtags to the tweets enhanced the topic modeling. The automated tag annotation, described in Section 5.3, generally resulted in lower Perplexity of the extracted topic models, as shown in Figure 6, suggesting that the auto-completed tags did help complete some missing and valuable information in the sparse tweet data, thus helping the topic modeling.

[6] we report results for $K_{LS} = 5$

[7] Refer to the electronic version of the paper for clarity.

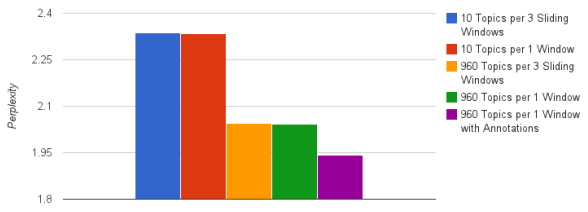

Figure 6: Perplexity for different numbers of topics and varying window length, showing improved results when NMF-based automated tweet annotation is performed before topic modeling.

7 Conclusion

Using Distributed LDA topic modeling, followed by NMF and hierarchical clustering within the resulting Latent Space (LS), helped organize the topics into less fragmented themes. Sentiment detection prior to topic modeling and automated hashtag annotation helped improve the learned topic models, while the agglomeration of topics across several time windows can link the topics discovered at different time windows. Our focus was on the topic modeling and organization using the simplest (bag of words) features. Specialized twitter feature extraction and selection methods, such as the ones surveyed and proposed by Aiello et al. [APM+13], have the potential to improve our results, a direction we will explore in the future. Another direction to explore is the news domain specific, user-centered approach, discussed in [SNT+14] and a more expanded use of automated annotation to support topic extraction and description.

8 Acknowledgements

We would like to thank the organizers of the SNOW 2014 workshop, in particular the members of the SocialSensor team for their leadership in all the phases of the competition.

References

[AN12] Artur Abdullin and Olfa Nasraoui. Clustering heterogeneous data sets. In *Web Congress (LA-WEB), 2012 Eighth Latin American*, pages 1–8. IEEE, 2012.

[APM+13] Luca Maria Aiello, Georgios Petkos, Carlos Martin, David Corney, Symeon Papadopoulos, Ryan Skraba, Ayse Goker, Ioannis Kompatsiaris, and Alejandro Jaimes. Sensing trending topics in twitter. *IEEE Transactions on Multimedia*, 2013.

[BL06] David Blei and John Lafferty. Correlated topic models. *Advances in neural information processing systems*, 18:147, 2006.

[BM10] David M Blei and Jon D McAuliffe. Supervised topic models. *arXiv preprint arXiv:1003.0783*, 2010.

[BNJ03] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[CBGN12] Juan C Caicedo, Jaafar BenAbdallah, Fabio A González, and Olfa Nasraoui. Multimodal representation, indexing, automated annotation and retrieval of image collections via non-negative matrix factorization. *Neurocomputing*, 76(1):50–60, 2012.

[HBB10] Matthew Hoffman, David M Blei, and Francis Bach. Online learning for latent dirichlet allocation. *Advances in Neural Information Processing Systems*, 23:856–864, 2010.

[HN12] Basheer Hawwash and Olfa Nasraoui. Stream-dashboard: a framework for mining, tracking and validating clusters in a data stream. In *Proceedings of the 1st International Workshop on Big Data, Streams and Heterogeneous Source Mining: Algorithms, Systems, Programming Models and Applications*, pages 109–117. ACM, 2012.

[LGD11] Chang-Hwan Lee, Fernando Gutierrez, and Dejing Dou. Calculating feature weights in naive bayes with kullback-leibler measure. In *Data Mining (ICDM), 2011 IEEE 11th International Conference on*, pages 1146–1151. IEEE, 2011.

[LH09] Chenghua Lin and Yulan He. Joint sentiment/topic model for sentiment analysis. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 375–384. ACM, 2009.

[LHAY07] Yang Liu, Xiangji Huang, Aijun An, and Xiaohui Yu. Arsa: a sentiment-aware model for predicting sales performance using blogs. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 607–614. ACM, 2007.

[LS99] Daniel D Lee and H Sebastian Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

[LZ08] Yue Lu and Chengxiang Zhai. Opinion integration through semi-supervised topic modeling. In *Proceedings of the 17th international conference on World wide web*, pages 121–130. ACM, 2008.

[McC] Mallet: A machine learning for language toolkit. http://www.cs.umass.edu/ mccallum/mallet.

[NASW09] David Newman, Arthur Asuncion, Padhraic Smyth, and Max Welling. Distributed algorithms for topic models. *The Journal of Machine Learning Research*, 10:1801–1828, 2009.

[PCA14] Symeon Papadopoulos, David Corney, and Luca Maria Aiello. Snow 2014 data challenge: Assessing the performance of news topic detection methods in social media. In *Proceedings of the SNOW 2014 Data Challenge*, 2014.

[PT94] Pentti Paatero and Unto Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5(2):111–126, 1994.

[SNT+14] S Schifferes, N. Newman, N. Thurman, D. Corney, A.S. Goker, and C Martin. Identifying and verifying news through social media: Developing a user-centered tool for professional journalists. *Digital Journalism*, 2014.

[TM08] Ivan Titov and Ryan McDonald. Modeling online reviews with multi-grain topic models. In *Proceedings of the 17th international conference on World Wide Web*, pages 111–120. ACM, 2008.

[WWC05] Janyce Wiebe, Theresa Wilson, and Claire Cardie. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2-3):165–210, 2005.

[YMM09] Limin Yao, David Mimno, and Andrew McCallum. Efficient methods for topic model inference on streaming document collections. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 937–946. ACM, 2009.

[ZBG13] Ke Zhai and Jordan Boyd-Graber. Online topic models with infinite vocabulary. In *International Conference on Machine Learning*, 2013.