Trending Topic Detection and Automatic Topic Labeling on Twitter

April 14, 2015

1 Type Of Project

Combination

2 Team Members

Arabinda Moni	CSA	arabinda.moni@csa.iisc.ernet.in
Harit Vishwakarma	CSA	harit.vishwakarma@csa.iisc.ernet.in
D.Raghuram Bharadwaj	CSA	raghu.bharadwaj@csa.iisc.ernet.in

3 Introduction

Twitter is one of the most popular social networking platform where people share their thoughts as small text of 140 characters called 'tweets'. Twitter claims to have 500 million users out of which 284 million users are active (source: wikipedia). This huge user base makes twitter data an important resource for understanding public interests,identifying emerging trends and breaking news etc.It can have applications in Defense to get instant alert of some ongoing crime or terror attack.Another application can be to detect and track spread of pandemic.Twitter was used to spread information during many crisis situations like Haiti earthquake, tsunami in Samoa etc. so we can also identify these crisis situations by analyzing live twitter feed.

For the above reasons we selected this project. Our goal was to develop a system that can detect trending topics in twitter and label those topics automatically in human readable format.

3.1 Reading Goals

- We are excited by the idea of finding the topics from which corpus of documents are generated.
- We want to learn in detail, the Machine Learning technique that helps in figuring out the topics.
- We want to investigate, why these techniques give right topics after we apply them.

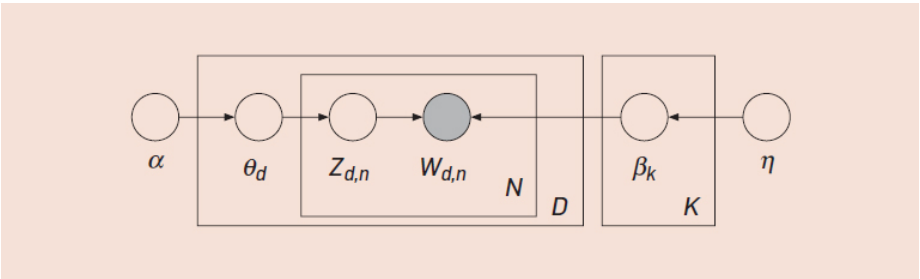
- We want to learn how to extend these techniques to online settings, so that we can apply them to platforms like Twitter to identify trending topics.
- We want to learn labeling the topics with appropriate Label.

3.2 Papers Read

3.2.1 Topic Models - David M.Blei, John D.Lafferty [1]

Rationale: This paper clearly states the idea of Topic Models. It opens with the reason why we apply Topic Models to corpus of documents. It clearly explained Machine Learning Technique "Latent Dirichlet Allocation"which we use to find topics from corpus.

Summary Of the Paper: Finding underlying topics from collection of documents helps us in various applications like, finding semantic similarity between documents, easily accessing documents of interest etc. Generative Process of the Documents can be given by :



(Source : `http://deliveryimages.acm.org/10.1145/2140000/2133826/figs/f4.jpg`)

Consider a Vocabulary V which contains distinct words. Topics are distributions that put probability on the words in Vocabulary. We assume a dirichlet prior on these distributions. We also assume that Topic Proportions, which puts probability on Topics(Distribution over distributions) also come from dirichlet prior.
Now,
1. For each Topic,we pick a distribution over words $\beta_k \sim Dir(\eta)$.
2.For each Document,
a..Pick a vector of Topic Proportions from Dirichlet Distribution. $\theta_d \sim Dir(\alpha)$
3. Now to generate each word in document,
a. Pick a topic assignment $z_{d,n} \sim MULT(\theta_d)$
b. Pick a word from topic picked in (a). $W_{d,n} \sim MULT(\beta_{z_{d,n}})$

In this way, all documents are generated. But, these Topics, Topic Proportions, and Topic Assignments are not observed (Hidden). We observe only words from the documents. So, our goal is to infer these hidden variables given the word counts in the document. Now, The posterior is :

$$p(\theta_{1:D}, Z_{1:D}, \beta_{1:k} | w_{1:D,1:N}, \alpha, \eta) =$$

$$\frac{p(\theta|\alpha).p(\beta|\eta)\prod_{n=1}^{N}.p(z_n|\theta).p(w_n|z_n,\beta_{1:k})}{\int_\theta \int_\beta \prod_{n=1}^{N}\sum_{z=1}^{K} p(z_n|\theta).p(w_n|z_n,\beta_{1:k})}$$

$p(\theta|\alpha)$ and $p(\beta|\eta)$ can be computed from Dirichlet distribution. $p(z_n|\theta)$ is the probability that θ has put on topic z_n.$p(w_n|z_n, \beta_{1:k})$ is the probability that β_{z_n} has put on word w_n.

So, Numerator can be computed easily. But the Denominator is Hard to compute. Hence, we approximate True Posterior with Mean Field Variational Inference method. We consider a Variational Distribution where the Hidden Variables are considered Independent. Description of variational Distribution that we used is described in the following section.

3.2.2 Online Learning for Latent Dirichlet Allocation - Matthew D.Hoffman, David M.Blei, Francis Bach [2]

Rationale: This paper clearly explains variational Inference for approximate posterior and how to extend idea this idea of LDA to Online Setting. All the mathematical expressions and methods are clearly stated in this paper.

Summary of paper: Online Setting of LDA is needed because (a)In normal LDA, the relative order of documents is not taken into consideration. When we want to learn about emerging Topics, Order of Documents is important.(b).For a very Big corpus, more memory is required. But in Online Setting, we only required a batch of documents at each time slice. In this paper, Batch Variational Bayes method is used as approximate posterior for word distribution over Topics, Topic Proportions and Topic Assignments.

Idea Of variational Bayes: We have already seen that True Posterior is intractable to compute.So, we consider a distribution with variational Parameter for each Latent variable. Here we consider a distribution 'q' as

$$q(z_{d_i} = k) = \phi_{dw_{d_i}k}; q(\theta_d) = Dirichlet(\theta_d; \gamma_d); q(\beta_k) = Dirichlet(\beta_k; \lambda)$$

Now,we minimize the KL Divergence to fit parameters of q.

$$KL(q||p) = E_q[log\frac{q(z, \theta, \beta}{p(z, \theta, \beta|w, \alpha, \eta}]$$

$$= E_q[log(q(z, \theta, \beta))] - E_q[log(p(z, \theta, \beta, w|\alpha, \eta))] + log(p(w))$$

So, this is equivalent to maximizing the term

$$E_q[log(p(z, \theta, \beta, w|\alpha, \eta))] - E_q[log(q(z, \theta, \beta))]$$

Now,by factorizing p and q,and taking summation over all documents 'd' in the corpus(and dividing entire by D(number of documents), we get

$$\sum_d [E_q[log(p(w_d|\theta_d, z_d, \beta))] + E_q[log(p(z_d|\theta_d))] - E_q[log(q(z_d))] + E_q[log(p(\theta_d|\alpha))]$$

$$- E_q[log(q(\theta_d] + E_q[log(p(\beta|\eta)] - E_q[log(q(\beta)]/D]$$

Now we use coordinate Ascent over Variational Parameters.In this method, we find each parameter by keeping other variational parameters fixed.This method is iterated until convergence. In this way,we can update each parameter as :

$$\phi_{dwk} \propto expE_q[log(\theta_{dk})] + E_q[log(\beta_{kw})];$$

$$\gamma_{dk} \propto \alpha + \sum_{w} n_{dw}\phi_{dwk};$$

$$\lambda_{kw} = \eta + \sum_{d} n_{dw}.\phi_{dwk}$$

Extending to Online LDA The same Variational method discussed above is extended to Online setting also. But here, iteration will be ran at each time slice to find variational parameters in that time period.Then, we use these parameters for update in next time period.But we have to fit a good latent parameter λ for Topic distribution over words.At current time period, we compute $\lambda_{current}$,by assuming that documents at current time period constitutes entire corpus and then we update λ_{new} by giving required weights to current value and past value.That is,

$$\lambda_{new} = (1 - \rho_t).\lambda_{old} + \rho_t.\lambda_{computed}$$

In this manner, we find variational Parameters and use this to find approximate values of Topic Proportions,Topic distribution over words and Topic Assignments at each time period.

3.2.3 Automatic Labeling of Multinomial Topic Models - Qiaozhu Mei, Xuehua Shen, Chengxiang Zhai [3]

Rationale: Topic Modeling just gives us Topics which are multinomial distributions over vocabulary. By just looking at the weights of each word in a topic it is difficult to interpret the topic. Therefore assigning human readable labels to the topics is an essential step in any topic modeling workflow. This paper proposes method for topic labeling which is principled and intutive unlike previous methods which were based on hueristics, like using top words of a topic as a label for it. Moreover the proposed method can be used in online setting, which is desired for our use case.

Summary of the paper Like a topic model label is also considered as multinomial distribution over vocabublary and then from a set of candidate labels for a topic model t the label l which is maximizing the expected *Pointwise Mutual Information* between t and l which is equivalent to minimizing the *KL-Divergence* between t and l is selected for t. These metrics are encapsulated in a metric called $Score(l, \theta)$. Here $Score : L \times T \rightarrow \mathbb{R}$; $L = V \times \mathbb{R}$ is the Label space and $T = V \times \mathbb{R}$ is the topic space, V is the Vocabulary. This score function captures the semantic distance between the topic θ and a label l. e.g. if there are two labes l_1 and l_2 such that

$$Score(l_1, \theta) > Score(l_2, \theta)$$

then l_1 is a better label than l_2 for topic θ.

The topic labeling task can be decomposed into four subtasks:

1. Finding candidate labels.
2. Deriving a good $Score(l, \theta)$ function.

3. Ranking the labels for each topic based on the score function.

4. Selecting the top ranked labels for each topic.

1. Finding Candidate Labels

For this purpose a separate reference corpus C is used. It could be the same set of documents used for topic modeling or a separate relevant collection of documents.There are following two ways to get good candidate labels. C is also referred as labeling context.

1.1 Chunking/Shallow Parsing:

Frequently occuring phrases are extracted from C using *NLP Chunker*. Advantage of using these phrases as candidate labels is that the labels obtained are meaningful and grammatically correct.

1.2 Ngram Testing:

Word ngrams are extracted from C. To test if an ngram is meaningful or not some statistical test like *Student's T-test*. The advantage of Ngrams is that they can be used on arbitrary C.However they may not be linguistically correct.

The paper compared these two ways and Ngram Testing was found to be better.

2. Deriving Score Function

As mentioned above the topic label l is also represented as a multinomial distribution $\{p(w|l)\}$ and a topic θ is representd by multinomial distribution $\{p(w|\theta)\}$. Let $D(\theta||l)$ denote the KL-divergence between θ and l .

$$\begin{aligned}
Score(l,\theta) &= -D(\theta||l) \\
&= -\textstyle\sum_w p(w|\theta) log \frac{p(w|\theta)}{p(w|l)} \\
&= -\textstyle\sum_w p(w|\theta) log \frac{p(w|C)}{p(w|l,C)} \quad - \sum_w p(w|\theta) log \frac{p(w|\theta)}{p(w|C)} \\
&\qquad\qquad\qquad\qquad - \textstyle\sum_w p(w|\theta) log \frac{p(w|l,C)}{p(w|l)} \\
&= -\textstyle\sum_w p(w|\theta) log \frac{p(w,l|C)}{p(w|C)p(l|C)} \quad - D(\theta||C) \\
&\qquad\qquad\qquad\qquad - \textstyle\sum_w p(w|\theta) log \frac{p(w|l,C)}{p(w|l)} \\
&= -\textstyle\sum_w p(w|\theta) PMI(w,l|C) \quad - D(\theta||C) \quad + Bias(l,C)
\end{aligned}$$

Here $D(\theta||C)$ is independent of label l. so it is same for all labels. $Bias(l,C)$ can be viewed as a bias of using context C to infer the semantic relevance of l and θ.When both the topic models and the candidate labels are generated from the same collection C, it is safe to assume that there is no bias.

The First tirm can be written as Expectation of Pointwise Mutual Information between l and terms in the topic θ given the Context C.

$$E_\theta(PMI(w,l|C)) = \textstyle\sum_w p(w|\theta) PMI(w,l|C) \qquad .....(1)$$

Thus minimizing KL-Divergence between $p(w|\theta)$ and $p(w|l)$ is equivalent to maximizing the Expected Pointwise Mutual Information between l and terms in topic θ.

We want to label multiple topics simulataneously so it is desirable to find labels which discriminate between different topics. Hence we would like a label l to have high semantic score $Score(l,\theta_i)$ for its target topic θ_i and low score for other topcis θ_{-i}. It can be achieved by the following modification to the Score function.

$$Score'(l, \theta_i) = Score(l, \theta_i) - \mu Score(l, \theta_{-i})$$

where,

$$\begin{aligned} Score(l, \theta_{-i}) &= -D(\theta_{-i}||l) \\ &\approx E_{\theta_{-i}}(PMI(w, l|C)) \\ &\approx \frac{1}{k-1} \sum_{j=\theta_{1..i-1,i+1..k}} \sum_w p(w|\theta_j) PMI(w, l|C) \\ &= \frac{1}{k-1} \sum_{j=1..k} E_{\theta_j}(PMI(w, l|C)) - \frac{1}{k-1} E_{\theta_i}(PMI(w, l|C)) \end{aligned}$$

Hence,

$$Score'(l, \theta_i) \approx (1 + \frac{\mu}{k-1}) E_{\theta_i}(PMI(w, l|C)) - \frac{\mu}{k-1} \sum_{j=1..k} E_{\theta_j}(PMI(w, l|C))$$

Where $E_{\theta_i}(PMI(w, l|C))$ can be computed using equation (1).

Step 3 and 4:

Calculate $Score'(l, \theta)$ for all candidate labels and select the top ranking labels for each topic θ.

The labels obtained by this method were subjectively evaluated by humans. Although the labels were found to be satisfactory but they were not the best labels and it was observed that there is room for improvement.

3.2.4 Probabilistic Topic Models - Review Article by David M.Blei [4]

This is Review article introducing Topic Models and LDA. This paper was very helpful at beginning, for getting idea of Topic models.It has created interest in working on this project.

3.2.5 Observations from Papers

We observe that Posterior is directly proportional to term $p(w_n|z_n, \beta_{1:k})$. So, for a word, this term is maximized for a topic that puts high probability on this word. So, we get topics that put high probabilities on some words rather than putting same probability on all the words as "Top Topics". Dirichlet Priors also plays a role in Topic Distributions. If the prior is less than 1, we get very sparse Topics. But here we assumed that a document is generated from mixture of Topics (Mixed Membership Model), so it's good idea to have large prior. Important step that differentiates LDA and Online LDA is that, after each times slice, Variational word distribution parameter λ on topic will be

$$\lambda_{new} = (1 - \rho_t).\lambda_{old} + \rho_t.\lambda_{computed}$$

We can see that, we can control the parameter for weight it puts on old values. This is justified as we can assume that words in a topic can vary over time.Ex: word "Desktop" will be used more in 2000 in a Topic related to Computer Machinery . But it will be eventually replaced by the word "Laptop". So, By this Online LDA model, we can also draw logical conclusions on how buzz words has changed over time in a particular topic.

4 Implementation

4.1 Data Collection

Twitter gives data at free of cost.We have used twitter api to collect twitter public stream [5].To do so we have used a Java library called Twitter4j[7] which works as a wrapper for twitter api.Twitter needs authentication before giving feeds.Details of how did we authenticate our app is given in appendix.Our implementation has option to filter language and location specific feeds.Our current setting fetches english feed from all over the world, but can be easily changed to fetch feed from any part of the world (see appendix).

There are two main reasons why tweeter data is difficult to deal with compared to wikipedia articles, news articles or research papers upon which OLDA is generally used -

- Size limitation of tweets.In general these articles have thousands of words. Whereas individual tweets have an upper limit of 140 characters. Thus individual tweets are not suitable for topic modeling. So, we clubbed 10000 tweets in a single document and run OLDA on that.Now we could easily run OLDA on this.Thus we solved this problem.
- Spelling errors, non standard words (ex. 'lol','rolf' etc) and use of non english words in english fonts (ex. 'Aam admi'). Tweets contains a reasonable amount of misspelled words which can lead to under performance of OLDA. We tried to use spell correcter to correct misspelled words but spell correction takes very long time and is not reliable enough, it was confused with non english words written in english font and trying to correct them. Moreover tweets contains many english words that are not even in english dictionary. Our another attempt was to stem the words but that also was not helping. So we left this problem for future.

4.2 Data pre-processing

Tweets contains lots of stop words (high frequency words present in every topic in almost same proportions with no contribution to topic modeling. For example: 'the','is','are' etc), images, videos, audio files,ASCII emoticons(sequence of special characters to express emotions) etc. which do not play any role in separating one topic from another.So we needed some preprocessing of the data.Preprocessing has basically the following three steps:

4.2.1 Stop word removal

We have used a list of 157 stop words. [6]

4.2.2 URL removal

We have written our own regular expression to remove url's from tweets.

4.2.3 Special character removal

Special characters are of no interest to us, so we have removed all special characters.

4.3 Implementation of OLDA

We used the code provided at [8]. It is java implementation of the algorithm described in 3.2.2 .

Parameters used:
$Numberoftopics$ $K = 5$
$\tau = 1$; $\kappa = 0.8$; $\alpha = 1/5$; $\eta = 1/5$

4.4 Topic Labeling

We implemented the algorithm described in section 3.2.3 . For Candidate label generation we used Ngram Testing and for this we used [9] which is a popular java package for Ngram Statistic. We implemented two methods for Scoring the candidate labels. $Method1$, which considers the frequency of a label along with its score for ranking and $Method2$ which ranks labels based on just score. $Method1$ serves the purpose of a baseline method, since high frequency candidate labels are more likely to be a good topic.We show the results obtained in the next section.

Parameters used:
$\mu = 0.07$

5 Evaluation

Labeled data for twitter is not freely available as Public.we use Twitter API to fetch data from specific domains like cricket,crime.Then, we run OLDA on this data.It returns us set of top topics.We feed these topics to Automatic labeling module.This returns set of top 2-gram labels.Then,we rate the labels manually on a scale of 1-5 based on relevance to that topic.

6 Results

We collected and clubbed 10,000 tweets on Cricket domain at each time slice.We ran OLDA on 3 such time slices. So, total of 30,000 tweets.Each of the tables contains labeled topics obtained at that time slice.

Top words from Topics at time instance 1.

Topic 1	Topic 2	Topic 3	Topic 4	Topic 5
ipl	ipl	spo	ipl	spo
club	benaud	clubipl	pakistan	ritchie
spo	ritchie	ritchie	ritchie	ipl
benaud	kings	benaud	benaud	pakistan
ritchie	ritchiebenaud	pakistan	ritchiebenaud	ritchiebenaud
ritchiebenaud	match	sri	club	benaud
voice	pakistan	lanka	match	club
match	rajastan	ritchiebenaud	rajastan	team
sri	punjab	match	spo	kings
kings	royals	galle	sri	match
punjab	club	kings	royals	punjab
pakistan	cwc	air	punjab	rajastan
xi	spo	badureliya	XI	sri
england	sri	force	voice	faulkner
cwc	england		test	cwc

Table 1: Topics with Labels at time instance 1

Topics	Method 1				Method 2			
	Topic Label	Score	Frequency	Relevancy Score	Topic Label	Score	Frequency	Relevancy Score
Topic 1	spo club	0.015	513	4	colonize rid	1.01	2	3
Topic 2	richie benaud	0.086	373	5	sixes flowing	1.136	2	3
Topic 3	srilanka	0.709	281	5	ssc badureliya	11.39	2	3
Topic 4	kings Xi	0.081	118	5	charming Knowledgeable	1.036	2	2
Topic 5	rajastan royals	0.323	181	5	cwc wining	2.808	2	5

Table 2: Topics with Labels at time instance 2

Topics	Method 1				Method 2			
	Topic Label	Score	Frequency	Relevancy Score	Topic Label	Score	Frequency	Relevancy Score
Topic 1	richie benaud	0.342	1695	5	crazypusa badureliya	7.166	2	3
Topic 2	st kitts	0.046	261	3	chennai super kings delhi daredevils	0.814	2	3
Topic 3	knit hats	0.780	2	3	knit hats	0.780	2	3
Topic 4	kitts invitat-ional	0.013	238	4	armey marvelling	0.760	2	3
Topic 5	spo club	0.035	1659	4	preparation st	1.634	2	3

Table 3: Topics with Labels at time instance 3

Topics	Method 1				Method 2			
	Topic Label	Score	Frequency	Relevancy Score	Topic Label	Score	Frequency	Relevancy Score
Topic 1	richie benaud	0.051	3137	5	mcc universities	4.39	2	5
Topic 2	durham mccu	0.115	157	5	mccu chelmsford	2.207	2	5
Topic 3	suriya_ offl imraina	0.200	7	1	booker msgs	0.695	2	2
Topic 4	rip richie	0.119	775	5	target perform	5.514	4	4
Topic 5	leeds bradford mccu	0.681	119	4	essex cardiff	15.39	2	5

7 Conclusion

- The idea of LDA and Online LDA for Topic modeling has been thoroughly read and understood.
- Applied Online LDA to live Twitter feed to get Trending Topics.
- We observed the topics and labels obtained were quite relevent to the document corpus at the particular time instance. E.g. "richie benaud" the popular cricket commentator, was one of the trending topics due to his sad demise.
- Implemented Automatic Labeling to label Topics in Human-Readable format.We found that both $Method1$ and $Method2$ are having almost same rating for topic labels.

8 Acknowledgments

- We thank Harikrishna Narasimhan for suggesting us this project and giving us idea of LDA.
- We are thnakful to Dr. Shivani Agarwal and Prof. Chiranjib Bhattacharrya for giving us the opportunity to work on this project.
- We are very thankful to Video Lectures on Topic Models delivered by David M.Blei, which enhanced our understanding. It can be found at: `http://videolectures.net/mlss09uk_blei_tm/`
- we are thankful to tutorial on Variational Inference, which helped us in understanding concept and preparing report.Tutorial can be found at : `https://www.cs.princeton.edu/courses/archive/fall11/cos597C/lectures/variational-inference-i.pdf`

References

[1] https://www.cs.princeton.edu/ blei/papers/BleiLafferty2009.pdf

[2] https://www.cs.princeton.edu/ blei/papers/HoffmanBleiBach2010b.pdf

[3] http://sifaka.cs.uiuc.edu/czhai/pub/kdd07-label.pdf

[4] https://www.cs.princeton.edu/ blei/papers/Blei2012.pdf

[5] Here is the link to twitter official public stream api page.
https://dev.twitter.com/streaming/public

[6] List of stop words was downloaded from here.
https://code.google.com/p/twitter-sentiment-analysis/source/browse/trunk/files/stopwor

[7] Twitter4j: Java api wrapper library for twitter is available here.
http://twitter4j.org/en/index.html

[8] jolda: Java library for online LDA is available here:
https://github.com/miberk/jolda

[9] jnsp: Java library for Ngram Statistic:
http://jnsp.sourceforge.net/

A How to fetch stream from twitter?

- Create a twitter account.
- Go to settings then add mobile number and verify it.
- Go to https://apps.twitter.com/ and create an app.
- There you will find options to generate keys and tokens.
- Paste those keys and tokens in the file twitter4j.properties which is located inside the project directory.
- Now run the program. It will fetch twitter public stream.

B How to fetch stream from specific geographic location?

Open file ttm/TreetReceiver.java, go to the method getTweets(), there you can specify coordinates of a rectangular geographic area as a parameter of the filter.locations() method. For example to get feeds from India (and some nearby parts which comes under the rectangle) we can specify this location {{57,5},{ 101,39}}.

C Twitter API rate limit

Twitter 1.1 api has a rate limit of 180 search queries per 15 minute window.Here are details https://dev.twitter.com/rest/public/rate-limiting.

E0270 Course Project Proposal

Project Title: Identifying and Ranking Trending Topics in Twitter using Online LDA.
Type Of Project : Combination
Team Members :

I) Arabinda Moni CSA arabinda.moni@csa.iisc.ernet.in
II) Harit Vishwakarma CSA harit.vishwakarma@csa.iisc.ernet.in
III) D. Raghuram Bharadwaj CSA raghu.bharadwaj@csa.iisc.ernet.in

Introduction/Motivation:
Twitter is a very popular microblogging platform where people share their thoughts in small tweets. It is a good source of information on trending public interests. Real time topic analysis on twitter live feeds will help in figuring out the trending topics, whereas the current trending based on # tags suffers from the following problems

a) redundant (trending # tags in the same topic)
b) #tags are often crypted, while the topic analysis will give more user(advertiser) friendly outputs.

Here the challenge is that twitter feeds have limited words. However the topic modeling techniques that have been proposed are applied on large documents.

Machine Learning technique(s) :
Latent Dirichlet Allocation(LDA) is widely used technique for topic modeling. It has also been extended for the Online Topic Modeling. It is called Online Latent Dirichlet Allocation (OLDA).

In Online Topic Modeling one receives streaming data (documents) and learns, updates the topic model in an online fashion. It also involves detecting emerging topics.

We want to do online topic modeling on live tweets to find the trending topics and rank them based on popularity. This would be of interest to Advertisers and Campaigning etc.

Also we want to explore this area and learn more about it.

Reading papers

1) On-Line LDA: Adaptive Topic Models for Mining Text Streams with Applications to Topic Detection and Tracking - Loulwah AlSumait, Daniel Barbar´a, Carlotta Domeniconi
   This paper proposes techniques for Online LDA and Emerging topic detection.
2) Automatic Labeling of Multinomial Topic Models - Qiaozhu Mei, Xuehua Shen, Chengxiang Zhai

Doing Topic modeling using OLDA just gives a list of topic IDs however we need a semantic label which is human readable and relevant to the topic. This paper proposes techniques for Topic Labeling problem.

Implementation:

1) Methods : LDA (Latent Dirichlet Allocation) (Online), Topic Labeling (Online)
2) Data sets: Live twitter feeds. We will fetch them using twitter APIs.
   For implementation we will fix on a set of tweets such that there are a few trending topics in it and we will identify the topics manually. We will use this set for testing.
3) Evaluation Criteria:
   Evaluation of the final outcome is largely subjective since it involves topic labels which are generated by machine.
4) Baseline methods to be compared against: If we could come up with a new approach then we will compare it against the existing OLDA.

Expected outcomes:

- Ranked trending topics from live twitter stream,with support to each topic.
- Possible extension or improvement in the existing methods.

References:

- Online Learning for Latent Dirichlet Allocation - Matthew D. Hoffman,David M. Blei & Francis Bach
- Latent dirichlet allocation- Andrew Y. Ng,David M. Blei & Michael I. Jordan
- Empirical Study of Topic Modeling in Twitter- Liangjie Hong and Brian D. Davison