CLASSIFICATION OF EMAIL MESSAGES INTO TOPICS USING LATENT DIRICHLET ALLOCATION

By

Cagri Ozcaglar

A Thesis Submitted to the Graduate

Faculty of Rensselaer Polytechnic Institute

in Partial Fulfillment of the

Requirements for the Degree of

MASTER OF SCIENCE

Major Subject: COMPUTER SCIENCE

Approved:

_____

Sibel Adalı, Thesis Adviser

_____

Boleslaw Szymanski, Thesis Adviser

Rensselaer Polytechnic Institute
Troy, New York

April 2008
(For Graduation May 2008)

CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGMENT

I would like to thank my advisor Professor Sibel Adalı for all support and guidance in this work and beyond. I would also like to thank Professor Boleslaw Szymanski for his stimulating suggestions and encouragement during my research and writing of the thesis. I would like to thank Professor Mohammed Zaki for his teachings on data mining, which helped me on my research and thesis.

I would like to thank my parents for all their support. I would also like to thank my brother, Caglar, for his unconditional and continuous company. Finally, thanks to Computer Science Department at RPI, which provided me the environment to pursue my research interests and made this work possible.

ABSTRACT

Latent Dirichlet Allocation(LDA) is generative probabilistic framework to model and summarize large document collections in an unsupervised fashion. Documents in the dataset can be grouped into topics using a three-level hierarchical Bayesian model of LDA. The focus of this thesis is the classification of a dataset from Enron containing email messages into significant topics using LDA. We start by explaining the LDA model. We test LDA with synthetic data and analyze the results based on square error, Kullback-Leibler divergence and average entropy measures in order to assess the true number of topics of datasets and observe that LDA returns more accurate modeling of the dataset if it is clustered into as many number of topics as true number of topics. We model Enron dataset using LDA and use same parameters to find true number of topics of this dataset and present the results. Finally, we conclude that square error is the best measure to find the true number of topics of collection of documents.

CHAPTER 1
INTRODUCTION

Classification of text corpora and collections of discrete data has been a challenge for social network analysis and information retrieval. In the context of text modeling, one of the common models is to assume that each document is relevant to a topic with a certain probability. The contents of a document is modeled with respect to terms that appear in the document. In Latent Dirichlet Allocation, using a three-level hierarchical Bayesian model, each topic is characterized by a distribution over these terms. This model finds the relevance of documents to different topics and cluster text corpora into groups of documents, where each group represents the set of documents that are relevant to a topic. Document modeling assigns each document to a topic with some probability and this preserves essential statistical relationships that are useful for classification, collaborative filtering, summarization of text corpora and relevance judgements of the corpus.

1.1 Motivation

A corpus is a collection of documents. Our goal is to cluster the documents of a corpus into disjoint subsets of documents so that each subset represents documents which are most relevant to a topic. For this purpose, we use Latent Dirichlet Allocation(LDA). Latent Dirichlet Allocation partitions the corpus into groups without losing the relevance of documents with other topics. On the other hand, LDA model assumes that the number of major topics in the corpus is known, takes the number of topics as input and clusters the corpus according to this number. To provide the correct number of topics to LDA, we investigate its performance with a synthetic dataset in which the number of topics is known. After running synthetic data in LDA model with different number of topics, we compare the true number of topics and LDA number of topics of the corpus using different measures such as square error, Kullback-Leibler Divergence and average entropy. This gives an idea of how LDA assigns the topics to the documents of the corpus if the number of topics of

the corpus is overestimated, underestimated or previously known.

1.2 Related Work

Information retrieval researchers have proposed term frequency and inverse document frequency reductions for partitioning documents of a corpus into groups. These reductions manage to identify words which discriminates documents, but no information is extracted from the corpus about intra-document and inter-document statistical structure. For this purpose, Latent Semantic Indexing(LSI) was proposed. LSI finds the semantic structure of a document using singular value decomposition [5]. Then, probabilistic Latent Semantic Indexing(pLSI) was proposed by Hoffmann. pLSI maps documents as well as terms to topics and therefore, each document is reduced to a probability distribution on a fixed set of topics [9]. Methods explained so far uses "bag of words" assumption: the order of words in a document is not considered. Blei proposes a method, Latent Dirichlet Allocation(LDA)[3] which takes the order of words into account and uses a mixture model for exchangability of both words and documents. This generative probabilistic model represents documents as random mixture over topics and topics are represented by a distribution over words. Heinrich presents different parameter estimation methods for text modeling [8]. Dirichlet Multinomial Allocation(DMA) enhances Dirichlet process of LDA by applying efficient variational inference [17]. The following research on the same field are also invaluable for understanding the field and its challenges: [1, 2, 6, 15].

1.3 Organization

This thesis is composed of 5 chapters. Chapter 1 presents a brief introduction to LDA, the motivation for using LDA and the related work. Chapter 2 gives the details of the underlying probabilistic model of LDA. Chapter 3 explains the generation of synthetic data, and presents the sensitivity of LDA parameters to the true number of topics. Chapter 4 presents results of a real data: first, it explains how real data is reduced to a form which LDA accepts as input, and then results of LDA are presented. Final chapter concludes with a discussion of the model and future work.

CHAPTER 2
LATENT DIRICHLET ALLOCATION(LDA)

Latent Dirichlet Allocation (LDA) is a probabilistic generative model of a corpus which uses unsupervised learning. Before going further about LDA, we define the following terms:

- **Word:** An item from a vocabulary indexed by { 1,..,V }.
- **Document:** A sequence of N words denoted by $d = (w_1, w_2, .., w_n)$, where w_i is the $i-th$ word.
- **Corpus:** A collection of M documents denoted by $C = (d_1, d_2, .., d_M)$, where d_i is the $i-th$ document.

The basic idea of Latent Dirichlet Allocation is as follows:

1. Each topic is characterized by a distribution over words.
2. Each document is represented as a random mixture over latent topics.

LDA finds the probabilistic model of a corpus. While doing this, LDA assigns high probability to other similar documents, as well as to the members of the corpus. Models such as *Mixture of Unigrams* model assumes that each document exhibits exactly one topic. However, LDA model lets documents exhibit multiple topics.

2.1 LDA Model

In LDA model, each document d is assumed to be generated from a K-component mixture model, where the probability θ_d of each component of this mixture is governed by a Dirichlet distribution with parameter α. Then, each of the words w_{di} in the document ($i-th$ word of document d) is generated by a repetitive process of sampling a mixture component z_{di} from θ_d and then sampling the word itself from a multinomial distribution over the entire vocabulary $\beta_{z_{di}}$, associated with the

mixture component. Each of these components is called a *topic*. Therefore, there are K topics and the $i-th$ topic is denoted by z_i. The generative process of LDA is as follows:

- For each document $d = 1, ..., M$ in the corpus C:

  1. Sample mixing probability θ_d ˜ $Dir(\alpha)$
  2. For each of V words w_{di}:

     a) Choose a topic $z_{di} \in \{1, .., K\}$ ˜ $Multinomial(\theta_d)$

     b) Choose a word $w_{di} \in \{1, .., V\}$ ˜ $Multinomial(\beta_{z_{di}})$

In this generative process, V is the vocabulary size and K is the number of topics. The parameter α, given as input to LDA, is the symmetric Dirichlet parameter and $\{\beta_1,..,\beta_K\}$ are the multinomial topic parameters. The other input to LDA is K, the number of topics. Each of K multinomial distributions β_i assigns a high probability to a specific set of words that are semantically consistent. These distributions over the vocabulary are referred to as topics.

2.2 Parameter Estimation

All parameters of LDA can be estimated by approximate variational techniques. One of these approximations, *Variational EM algorithm*, allows efficient unsupervised parameter estimation. Variational EM algorithm is a deterministic approximation method which results in a biased estimate, but it is computationally efficient. In particular, given a corpus of documents $C = \{d_1, d_2, .., d_M\}$, we wish to find parameters α and β that maximize the log likelihood of the data, which is the following:

$$l(\alpha, \beta) = \sum_{i=1}^{M} log\ P(d_i \mid \alpha, \beta)$$

The value of $l(\alpha, \beta)$ can be computed if the value of $P(d_i \mid \alpha, \beta)$ is known for each document d_i. However, $P(d_i \mid \alpha, \beta)$ cannot be computed tractably. Instead, we will use a lower bound on the log likelihood. It can be shown using Jensen's equality

that the log likelihood of the observed data, $log\ P(d_i \mid \alpha, \beta)$ is lower bounded by the sum of expected log likelihood of the entire corpus with respect to the variational distribution $E_Q[log\ P(w, z, \theta \mid \alpha, \beta)]$ and the entropy of the variational distribution $H(Q)$, as follows:

$$P(w \mid \alpha, \beta) \geq E_Q[log\ P(w, z, \theta \mid \alpha, \beta)] + H(Q)$$

Therefore, we can find approximate empirical estimates for the LDA model via an alternating variational EM procedure that maximizes a lower bound with respect to the variational parameters γ and ϕ. The variational EM algorithm proceeds iteratively in two steps:

1. **E-step:** The variational parameters γ and ϕ are estimated by maximizing log likelihood using the lower bound described in the formula above. Since γ_d and ϕ_d for a given document d depend on each other, we estimate them iteratively until the lower bound on the log likelihood of the document converges to a local maximum.

2. **M-step:** The lower bound is maximized with respect to parameters α and β to obtain their new values. This finds the maximum likelihood estimates with sufficient statistics for each document.

The complete procedure of E-step and M-step are in Figure 2.1 and Figure 2.2 respectively.

The time complexity of E-step is $\Theta(MN_{max}^2K)$, where M is the number of documents in the corpus, N_{max} is the maximum document length in the corpus and K is the number of topics. The time complexity of M-step, on the other hand, is $\Theta(VK)$, where V is the observed vocabulary size. Thus, main computational bottleneck in the variational EM algorithm is the E-step.

To overcome this issue, [11] introduces a parallelized Variational EM algorithm. The variational parameters γ_d and ϕ_d in each document d are independent of those in other documents and therefore can be computed independently. In other words, the E-step computations of the documents can be speeded up through par-

Input: Parameters α, β
foreach *document* $d \in 1, .., M$ **do**
  foreach *topic* $k \in 1, .., K$ **do**
    initialize $\gamma_{dk} = N_d/K$
  end
  repeat until convergence
    foreach *position* $i = 1, .., N_d$ **do**
      foreach *topic* $k = 1, .., K$ **do**
        $\phi_{dik} = \beta_{kw_i} \cdot e^{\psi \cdot \gamma_{dk}}$
      end
      Normalize ϕ_{di}
    end
    foreach *topic* $k \in 1, .., K$ **do**
      $\gamma_{dk} = \alpha + \sum_{i=1}^{N_d} \phi_{dik}$
    end
  until;
end
return
Sufficient statistics S where $S_{kv} = \sum_{d=1}^{N_d} \sum_{i=1}^{N_d} \delta_v w_{di} \phi_{dik}$
α-sufficient statistics S_α where $S_\alpha = \sum_{d=1}^{M} \sum_{k=1}^{K} \psi\gamma dk - K\psi \sum_{k=1}^{K} \gamma dk$

Figure 2.1: E-step of Variational EM Algorithm

Input: Sufficient statistics matrices S and S_α
Compute α using Newton-Raphson optimization on S_α
Compute $\beta = S - normalized - row - wise$
return α, β

Figure 2.2: M-step of Variational EM Algorithm

allelization. The results of parallelization verifies that E-step is the main bottlebeck of the algorithm.

2.3 LDA Output Evaluation

LDA takes a data as input in Compact Vector form and produces three types of output files:

- **.beta file:** This file includes `topic` $\times$ `term` matrix. It contains the log of the topic distributions. Each row is a topic and in row k, each entry is the log of probability of a word given the topic, which is log $P(w \mid z = k)$. Since each

topic is defined as a distribution of terms by LDA, .beta files can be used to find most frequently used words for a topic. This file can also be used to find the top words for each topic.

- **.gamma file:** This file includes `document` $\times$ `topic` matrix. It contains variational posterior Dirichlets distributions. This file can be used to find document-based results, such as finding main topics of a document, or finding the top documents that are most related to a specific topic. In the experiments of this thesis, .gamma file is used to find out the distribution of documents to topics.

- **.other file:** This file includes information about the input. The number of topics, the number of terms(vocabulary size) and the parameter α are stored in this file.

CHAPTER 3
SYNTHETIC DATA

LDA distributes a set of documents into topics. LDA accomplishes this distribution given the number of topics, K, of the corpus. It requires another input, α, which is the parameter to Dirichlet distribution used in LDA model. Besides these two parameters, LDA requires a certain data format for input data, which will be mentioned in Chapter 4.

LDA does not know the number of topics of the corpus beforehand. There is no way for LDA to know the number of topics, K, as it takes it as input. The problem is then to determine the correct number of topics from LDA's output. To address this problem, we study the LDA output for a set of synthetic datasets with known properties.

3.1 Setup of Synthetic Data

We define a file in compact vector format that is used by LDA, that is known to contain `n` topics as a `True-n` file. As `True-n` file is the compact vector format of a corpus whose documents are distributed into n major topics, we can use this file as input to LDA and use n as the input to LDA which stands for the number of topics. To create a random `True-n` file, we present `GenerateCompact` algorithm. in Figure 3.1.

We generate disjoint set of words for each topic and each document that is relevant to that topic contains a subset of words in the topic. We generate equal number of documents for each topic combination. In the first two nested for loops of the algorithm, we iterate over 2-subsets of n topics, that is, subsets of n topics which is of size 2. Therefore, these for loops in the algorithm iterates $\binom{n}{2}$ times. After each iteration, a `documentVector`, which is the document vector of a document that is relevant to two topics i and j, is created. This vector is a compact vector for a document of the corpus, so it creates one line of compact vector for each document as output. We will explain how to create compact vector format in Chapter 4.

```
Input: n: Number of topics
       p = 2: Number of topics per document
       s: Number of keywords per topic
       t: Number of keywords per document that has that topic
       k: Number of documents for each configuration
for i ← 1 to n − 1 do
    for j ← i + 1 to n do
        for l ← 1 to k do
            Generate topic i for DocumentVector ;
            Generate topic j for DocumentVector ;
            for w ← 0 to n ∗ s − 1 do
                if DocumentVector [w] == 1 then
                    Write to file w:1 ;
                end
            end
        end
    end
end
return True-n file in compact vector format
```

Figure 3.1: GenerateCompact algorithm

We pose an assumption on this algorithm and assume that each document is relevant to 2 major topics. Therefore, rather than being a variable, $p = 2$ is a constant. Given the number of topics n, $p = 2$ and number of documents for each configuration k, the number of documents in the compact vector generated by `GenerateCompact` algorithm, and therefore the number of documents in a `True-n` file is $\binom{n}{2} \cdot k$.

3.2 Sensitivity to α

One of the inputs to LDA model is α, which is the parameter to Dirichlet distribution in LDA model. If α is set to *fixed*, then it does not change from iteration to iteration in the variational EM. If it is set to *estimate*, then α is estimated along with the topic distributions. We run LDA using α set to estimate.

We measured the sensitivity of LDA results to initial α. For this purpose, we used a `True-n` file as input to LDA, and we ran LDA with $k < n$, $k = n$ and $k > n$ topics. We compare the results by considering underestimation and overestimation

of topics, given the true number of topics. For this purpose, we introduce three parameters to measure the sensitivity of LDA to α. The parameters are fit to LDA model as follows:

- **Square Error:** `True-n` file is a corpus which is composed of documents that are relevant to 2 major topics, no matter what the true number of topics is. This leads us to model Square Error computation to `True-n` file as follows: The relevance of a document to 2 major topics, t_1 and t_2, is 0.5 and the relevance of this document to other topics is 0, which sums up to 1. After running `True-n` file with k topics in LDA, the document will have relevances r_1, r_2,.., r_k to each of k topics as given in the .gamma file. Figure 3.2 displays the relationship between true topics and LDA topics. The rest of the computations are as follows:

  1. Sort the relevances of the document to topics in descending order. Therefore, after sorting, for each document,

  $$r_i \geq r_j \; if \; i < j$$

  2. For each document, we normalize the relevance of that document to topics. At the end of normalization,

  $$\sum_{i=1}^{k} r_i = 1$$

  holds. This normalization corresponds to row-wise normalization of a *document* $\times$ *topic* matrix.

  3. Square Error is computed on sorted and normalized matrix as follows:

  $$\sum_{i=1}^{2} (r_i - 0.5)^2 + \sum_{i=3}^{k} {r_i}^2$$

- **Kullback-Leibler Divergence:** Kullback-Leibler Divergence is a measure which finds the difference between two probability distributions [4]. For two

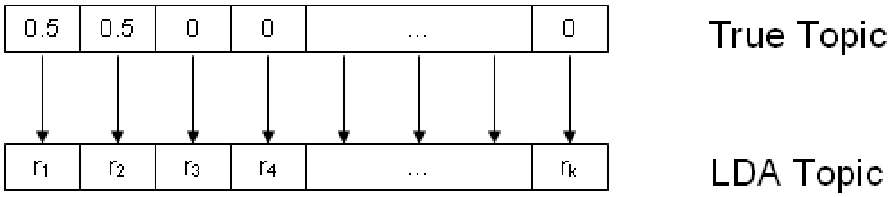

Figure 3.2: Square Error computation for `True-n` file

probability distributions P and Q, Kullback-Leibler Divergence of Q from P is calculated as follows:

$$D_{KL}(P \mid\mid Q) = \sum_i P(i) \log_2 \frac{P(i)}{Q(i)}$$

In our experiments, we want to find the divergence of dataset distribution from uniform distribution for each topic and then average for all topics. Therefore, P is the probability distribution of the dataset and Q is the uniform distribution for our experiments. K-L Divergence for each topic is calculated and then average K-L Divergence is found.

- **Average Entropy Using Buckets:** For each topic, entropy over all documents is calculated and the average of entropies of each topic is found. Average Entropy for each topic is calculated using 100 buckets. Given `document` × `topic` matrix `M` in .gamma file, the following procedure is followed:

  1. Normalize `M` row-wise. This step normalizes the relevance of each document to topics.
  2. Calculate the entropy for each topic using 100 buckets: [0, 0.01], (0.01, 0.02], (0.02, 0.03],..., (0.99, 1]. If n_i is the number of documents which fall into i-th bucket and if $\sum_{i=1}^{100} n_i = d$ is the number of documents in the corpus, average entropy for each topic is computed as follows:

$$H = \sum_{i=1}^{100} -\frac{n_i}{d} \cdot \log_2(\frac{n_i}{d})$$

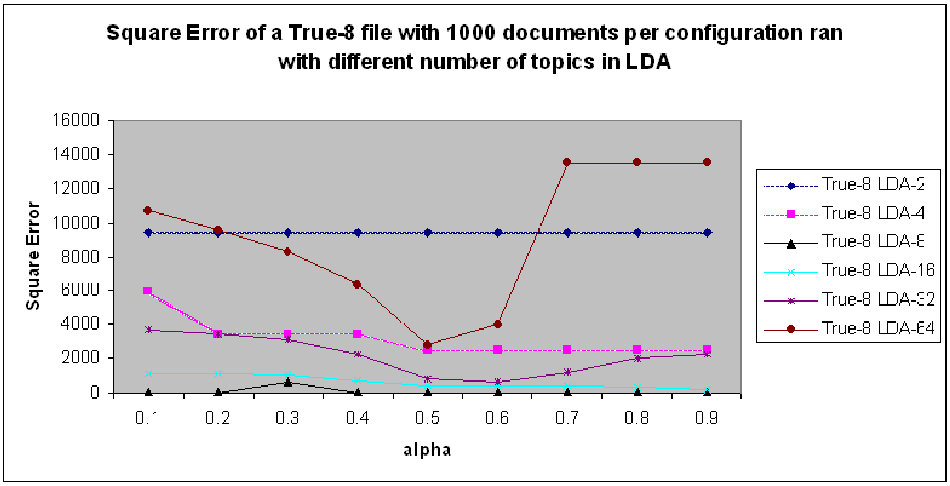

Figure 3.3: Square Error with changing α

3. Find the average entropy of topics.

Using the parameters explained above, we use them to measure sensitivity of LDA results to α. We used different files and we present here the results from a `True-8` file with 1000 configurations for each document. The file includes $\binom{8}{2} \cdot 1000 = 28000$ documents with average length of 50 distinct words.

3.2.1 Square Error

Square Error is a good measure for finding correct number of topics. Figure 3.3 shows the change in Square Error while α changes. `True-8` file is ran with 2, 4, 8, 16, 32, 64 topics in LDA and when true number of topics is equal to number of topics ran with LDA, Square Error has the minimum value. For the overestimation case in which LDA guesses the number of topics higher and the underestimation case in which LDA guesses the number of topics lower, Square Error is clearly higher than in the right case, where LDA guesses the number of topics correctly. As α changes, there is no significant pattern between α and square error.

3.2.2 Kullback-Leibler Divergence

Figure 3.4 displays the change in Kullback-Leibler Divergence while α changes. For small values of α, Kullback-Leibler Divergence of `True-8` file is proportional to

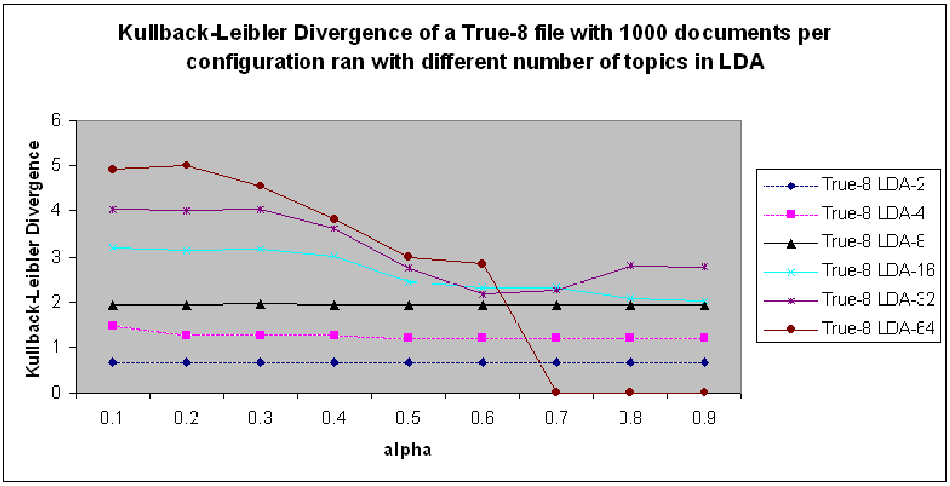

Figure 3.4: Kullback-Leibler Divergence with changing α

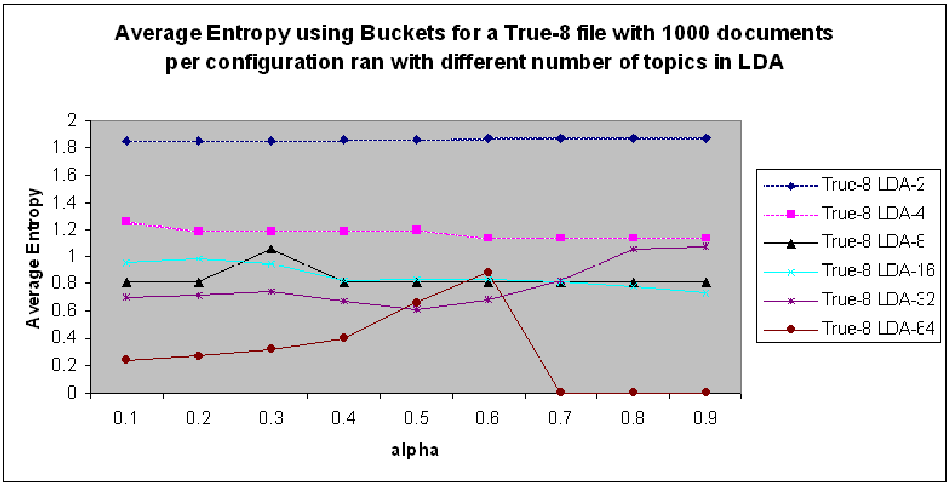

Figure 3.5: Average Entropy using Buckets with changing α

the number of topics in LDA. Moreover, for the underestimation case and right case (`True-8 LDA-n` where n $\leq$ 8), Kullback-Leibler Divergence is almost constant as α changes. This gives an idea about the number of topics: if Kullback-Leibler Divergence is constant with changing α, then we have to guess a higher number of topics for the corpus. Instead, if Kullback-Leibler Divergence is not constant with changing α, then we have to guess a lower number of topics for the corpus. For larger values of α, Kullback-Leibler Divergence of overestimation cases where LDA topic is greater than 8 decreases.

3.2.3 Average Entropy Using Buckets

Change in average entropy using 100 buckets can be seen in Figure 3.5. As the number of LDA topics increase, average entropy decreases. For small values of α, the metric `average entropy` $\times$ `LDA-topic` is constant. For greater values of α, there is no pattern between Average Entropy and LDA number of topics.

3.3 Sensitivity to Number of Topics

Another input to LDA is the number of topics. LDA model doesn't have a prior knowledge about the number of topics and it distributes the documents in the corpus into the number of topics given as input to LDA. However, using `True-n` files, we can measure the correctness of topic distribution of LDA. For this purpose, we keep the other input, $\alpha = 0.1$, constant and use the following parameters to measure the performance of LDA. We used `True-8` files with different number of configurations per document in our tests and we run these files with 2, 4, 8, 16, 32, 64 topics in LDA.

3.3.1 Square Error

Square Error is the best measure for finding correct number of topics. Figure 3.6 displays the Square Error for different number of topics on the `True-8` test files with `10`, `100`, `1000` documents per configuration. When the LDA number of topics is equal to true number of topics, Square Error has its minimum value, it is almost 0. As LDA number of topics diverge from true number of topics in both ways, Square Error increases drastically, letting underestimation and overestimation cases have large Square Error values.

3.3.2 Kullback-Leibler Divergence

The change in Kullback-Leibler Divergence with the LDA number of topics is displayed in Figure 3.7. As the LDA number of topics increase, Kullback-Leibler Divergence increases, independent of true number of topics.

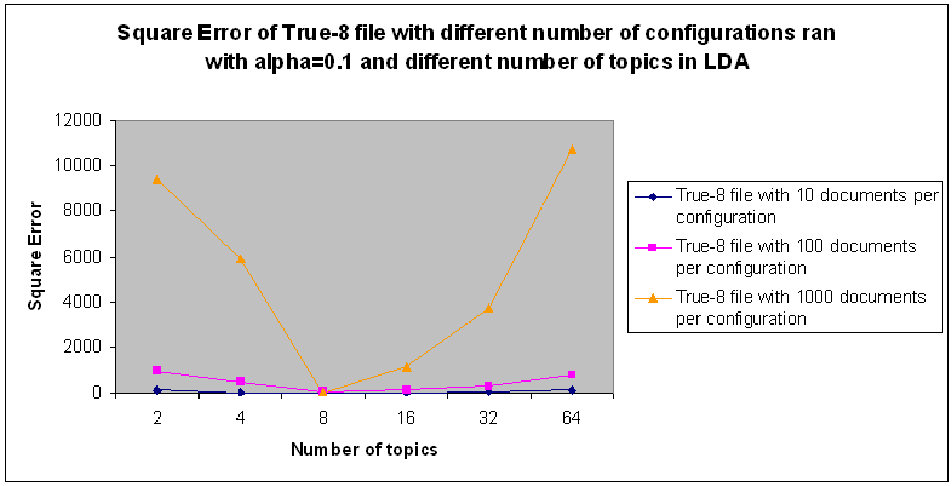

Figure 3.6: Square Error with changing number of topics in LDA

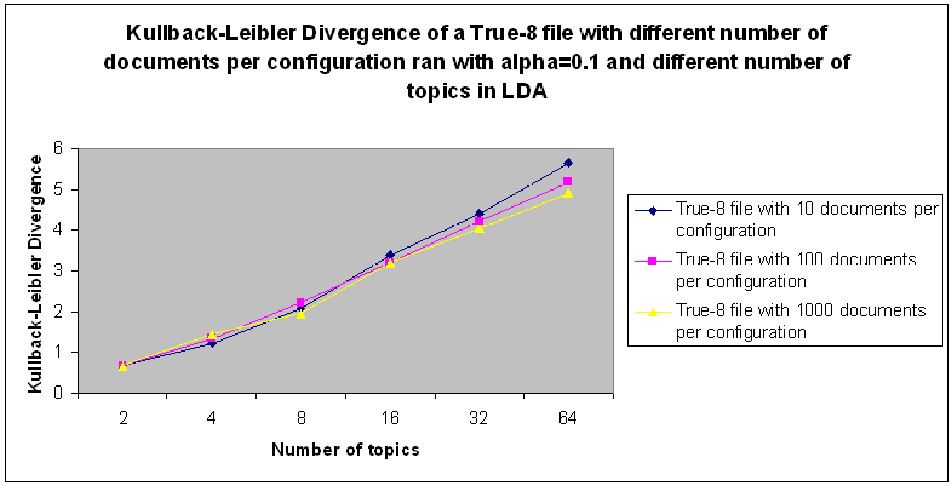

Figure 3.7: Kullback-Leibler Divergence with changing number of topics in LDA

3.3.3 Average Entropy Using Buckets

The change in Average Entropy using 100 buckets is displayed in Figure 3.8. As the number of LDA topics increase, Average Entropy decreases, independent of true number of topics.

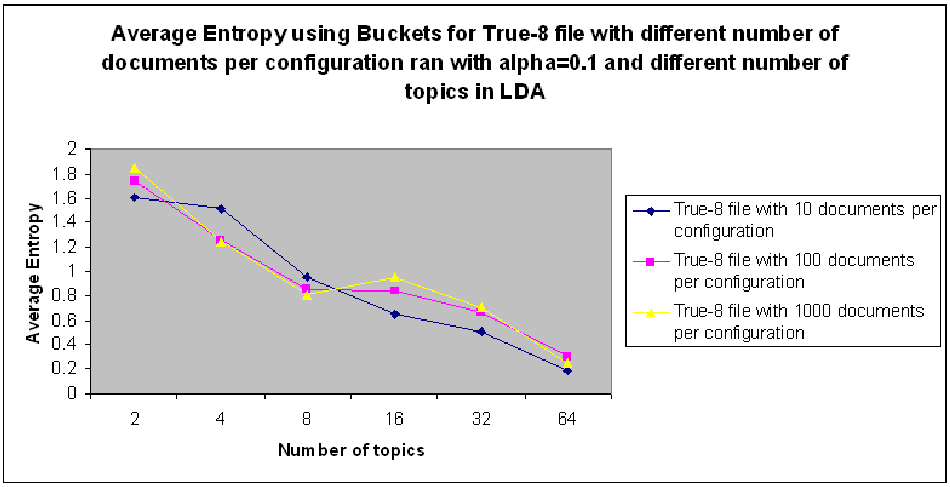

Figure 3.8: Average Entropy using Buckets with changing number of topics in LDA

CHAPTER 4
REAL DATA

In the previous chapter, we tested LDA with synthetic data, where we knew true number of topics apriori. We also tested Latent Dirichlet Allocation with a real dataset, Enron dataset. Enron was one of the leading companies in the world in the energy industry, until it bankrupted in 2001. After Bankruptcy, the emails within the company were collected in a dataset and made public by Federal Energy Regulatory Commission during the investigation. This dataset has been a good resource for email classification and Machine Learning research. Enron dataset is presented and analyzed in more depth in [10].

Enron dataset is a very noisy dataset. For this reason, we preprocess the data using our Text Processor. Text Processor takes a data file and converts it into the Compact Vector format, which is the only data format LDA accepts as input.

4.1 Text Processor

Text Processor applies six algorithms to the dataset. The data is preprocessed first, then stop words are removed, then stemming algorithm is applied. In the meantime, document vector is created and Term frequency, Inverse document frequency and Compact vector are found using this document vector. The algorithms used in Text Processor are as follows:

1. **Preprocess:** This step splits the messages in the dataset using an end of message indicator. It removes punctuation at the beginning or end of words. It places message number at the beginning of each message and performs non-standard processing, such as converting all words into lowercase. It also creates metadata of each message in the file.

2. **Stop Words:** Using a list of stop words for English, this step removes all the words that are in stop words list. Stop words are the words so commonly used

in English that they can not be an indicator of a certain topic for a message including the word.

3. **Stemming:** Martin Porter's Stemming algorithm finds the stems of the words. It is a linguistic method of determining the common stem of morphologic variants of a word. The algorithm is described by Martin Porter in [12]. This algorithm is optional, and if it is used, it decreases the number of words to a big extent. For example, the following three words have the same stem:

   plays $\Longrightarrow$ play
   player $\Longrightarrow$ play
   playing $\Longrightarrow$ play

   On the other hand, some words may be distorted after stemming, therefore this algorithm is not recommended if the results are expected to comprise meaningful words or if they will be used for semantic analysis of the dataset.

   - **Document Vector:** In this intermediate step, document vector of the data file is created. The format of the document vector is:

     $$d_i = (w_{i1} \;\; w_{i2} \;\; ... \;\; w_{ik})$$

     where d_i is document i and w_{ij} is the number of times the keyword at index $j = 1, ..., k$ exists in document d_i. This document vector will be used to find Term frequency, Inverse document frequency and Compact Vector.

4. **Term Frequency(TF):** This step finds frequencies of the terms based on the fact that more frequent terms in a message are more important. Term frequency of each term in a document is the frequency of the words normalized by the maximum frequency of this word in all documents. Since document vector contains frequencies of words in documents, given that f_{ij} is the frequency

of word w_i in document d_j, term frequency of this word in this document is computed as follows:

$$tf_{ij} = f_{ij} / \max_i(f_{ij})$$

5. **Inverse Document Frequency(IDF):** This step finds frequencies of the terms based on the fact that terms that appear in many different messages are less likely to be indicative of overall topic of each document. Given f_{ij}, the frequency of word w_i in document d_j, Inverse document frequency of a word in a document is computed as follows:

$$idf_{ij} = f_{ij} * log_2(N/c_i)$$

where N is the total number of documents in the corpus and c_i is the number of documents that contain word w_i.

In the Text Processor algorithm, if both TF and IDF is found then another measure is found by combining the results of TF and IDF, which is TF-IDF. TF-IDF of the data file is computed as follows:

$$tfidf_{ij} = tf_{ij} * idf_{ij}$$

where tf_{ij} is the term frequency of word w_i in document d_j and idf_{ij} is the inverse document frequency of word w_i in document d_j.

6. **Compact Vector:** Document vector is a sparse vector. For this reason, we create a compact version of this vector by displaying a word and its number of occurrences in the document only if this word appears in the document at least once. This step creates another representation of document vector, which has the format LDA requires. The format of compact vector for each document is as follows:

$$[M] \quad [word_1] : [count_1] \quad [word_2] : [count_2] \ldots [word_n] : [count_n]$$

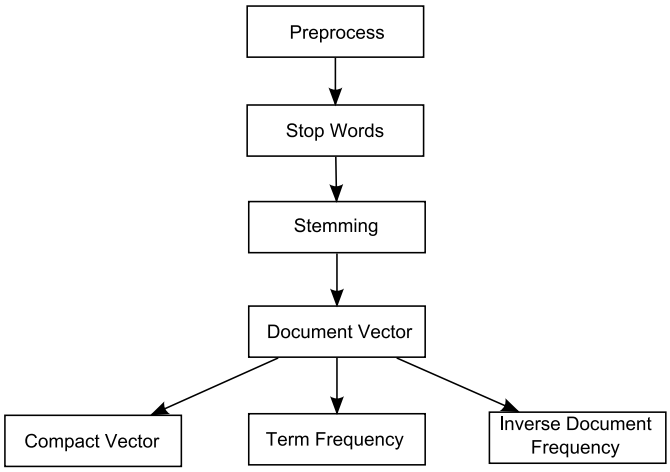

Figure 4.1: Steps of Text Processor

where M is the number of unique terms in the document. $word_i$ is an integer which indexes a word in the document and $count_i$ is the number of occurrences of $word_i$ in this document. The first occurrence of the word at index $word_i$ in the document is before the first occurrence of the word at index $word_j$, given that $i < j$. $count_i$ is greater than 0 for each $word_i$, as compact vector only displays the words that occur at least once in the document.

Steps of Text Processor are shown in Figure 4.1. After data is processed in Text Processor algorithm, compact vector of the data file is created and the file is ready to be used as an input to LDA.

4.2 Results

The experiments on Enron dataset is designed as follows: We first perform the Principal Component Analysis of Enron dataset to give an idea of the clustering of documents to topics. Then we use parameters Square Error, Kullback-Leibler Divergence and Average Entropy Using Buckets measures to find the exact number of topics in the Enron dataset.

4.2.1 Principal Component Analysis

The goal of Principal Component Analysis (PCA) is to find a new set of dimensions that captures the variability of the data best [7]. For this purpose, we

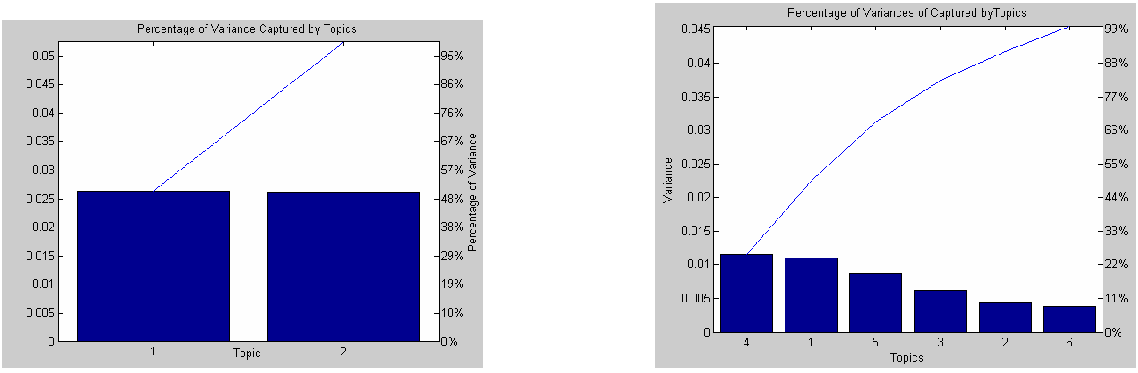

Figure 4.2: Percentage of Variance captured by topics with LDA runs on Enron dataset with 2 topics and 6 topics

used Enron dataset as input to LDA and ran with 2 topics and 6 topics respectively. The percentage of variance captured by topics for LDA runs with 2 topics and 6 topics are displayed in Figure 4.2.

PCA of these two tests show that the percentage of variance captured by topics when LDA is ran with 2 topics are the same and both 50 %. PCA of Enron dataset ran with 6 topics in LDA show that some topics, namely topics 4 and 1, captures the variance of the data most.

4.2.2 Parameters

We tested Enron dataset with the same parameters we used for testing synthetic data. Square Error remains an indicator of the correct number of topics. Kullback-Leibler Divergence and Average Entropy changes directly and inversely proportional to LDA number of topics, respectively.

4.2.2.1 Square Error

We ran Enron dataset with 2, 4, 8, 16, 32, 64 topics in LDA with $\alpha = 0.1$ and the result is shown in Figure 4.3. Square Error is minimum when LDA number of topics is 8. This information from Square Error measure does not imply that true number of topics of Enron dataset is 8. However, it implies that true number of topics of Enron dataset is in the range (4, 16), assuming each document contains 2 topics with equal proportions. This is because, the optimum value of Square Error can be anywhere in this range, including 8 as a candidate for true number of topics.

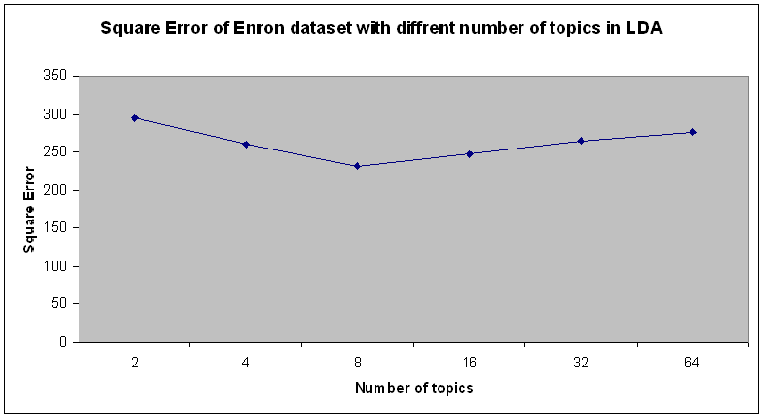

Figure 4.3: Square Error of Enron dataset ran with different LDA number of topics

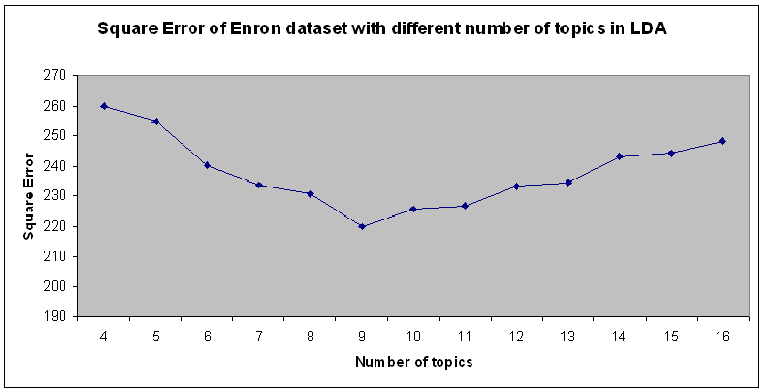

Figure 4.4: Square Error of Enron dataset ran with different LDA number of topics of smaller range

For this reason, we tested Enron dataset with $K \in [4, 16]$ number of topics to find the exact number of topics of Enron dataset. Figure 4.4 shows the change in Square Error while the number of topics changes in the interval [4,16]. As the figure shows, Square Error reaches its minimum when LDA number of topics is 9. Therefore, we conclude that true number of topics of Enron dataset is 9, not 8. This observation gives us a good choice for number of topics which is feeded to LDA as its input. As a result, classification of emails in the Enron dataset will be most accurate when the documents of the dataset are classified into 9 distinct topics.

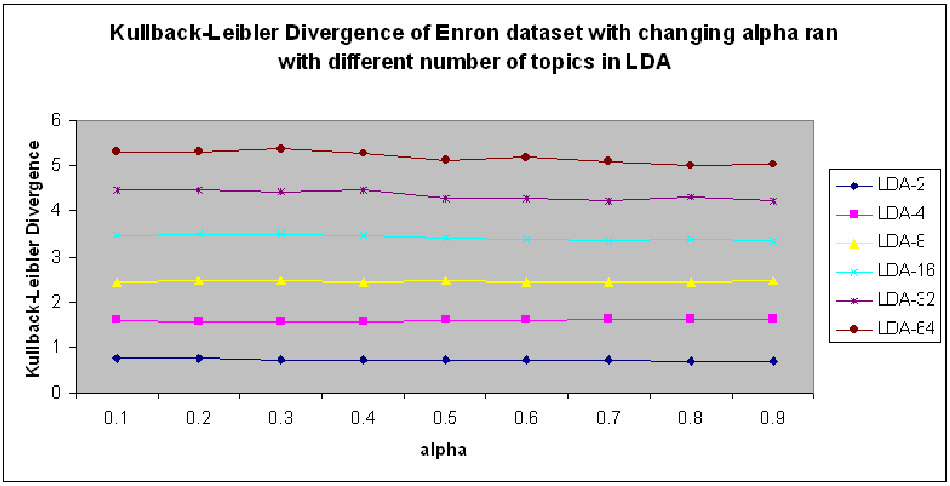

Figure 4.5: Kullback-Leibler Divergence of Enron dataset with changing α

4.2.2.2 Kullback-Leibler Divergence

Kullback-Leibler Divergence of Enron dataset for LDA runs with different number of topics are displayed in Figure 4.5. As we concluded for synthetic data, Kullback-Leibler Divergence of Enron dataset is almost constant for 2, 4, 8 topics. Starting with 16 topics, Kullback-Leibler divergence is not constant for different values of α. Therefore, we conclude that true number of topics of Enron dataset is in the range [8,16). This result justifies our previous result concluded from Square Error that true number of topics of Enron dataset is 9. However, the difference is not significant.

4.2.2.3 Average Entropy using Buckets

Another measure we used to test Enron dataset is Average Entropy using 100 buckets. Figure 4.6 proves for this real dataset that Average Entropy is inversely proportional to number of topics used for LDA. The same figure also displays the change in Kullback-Leibler Divergence which is directly proportional to number of topics, in contrast to Average Entropy. Table 4.1 summarizes the results of LDA runs with $\alpha = 0.1$ and different number of topics on Enron dataset.

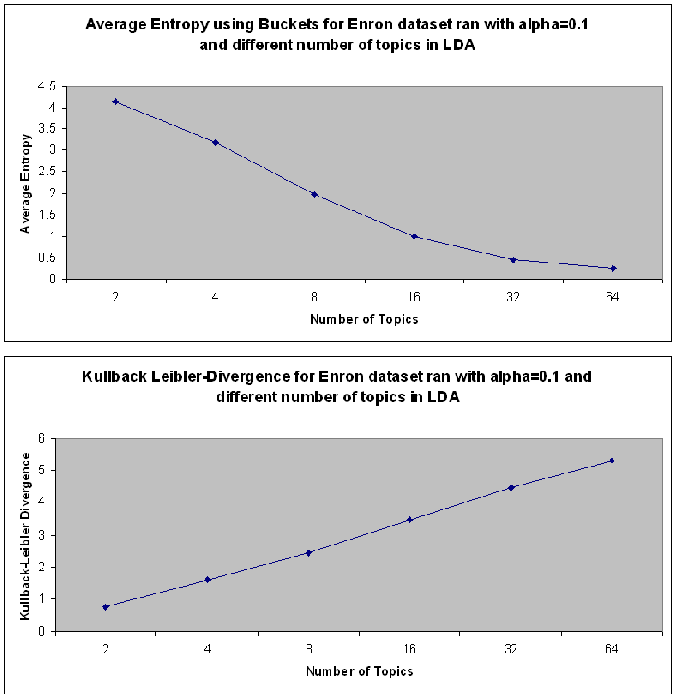

Figure 4.6: Average Entropy using Buckets and Kullback-Leibler Divergence of Enron dataset for different number of Topics in LDA

Number of Topics	Square Error	Kullback-Leibler Divergence	Average Entropy
2	294.6098	0.763	4.1391
4	259.7715	1.5844	3.1985
8	230.7499	2.451	1.9707
16	248.003	3.4669	0.9906
32	264.8856	4.4571	0.4609
64	276.6158	5.3131	0.2448

Table 4.1: Square Error, Kullback-Leibler Divergence and Average Entropy of Enron dataset after LDA runs with $\alpha = 0.1$ and different number of topics

4.2.3 Top Words of Topics

Among the output files of LDA, .beta file includes `topic` $\times$ `word` matrix which is relevance matrix of terms to topics. Using this matrix, we can find top keywords of each topic and see if these words can be "named" as in a single topic. For this purpose, we used Text Processor on Enron dataset and cleaned the data to a big extent. We did not use Stemming algorithm of Text Processor for finding top words, because we need to find meaningful English words to collect them in a single topic. After processing on Enron dataset, we obtained 11549 distinct keywords. We ran

"ENERGY"	"BUSINESS"
like	market
think	process
deals	need
price	team
gas	questions
need	think
power	review
trading	unit
know	employees
index	groups
desk	currently
energy	group
day	current
month	business
new	ensure
position	include
group	deal
deal	capacity
want	strategy
transport	performance

Figure 4.7: Top-20 keywords of 2 topics in Enron dataset

LDA on Enron dataset using 6 topics, and found top-20 keywords of these topics by mapping the highest word frequencies of each topic to corresponding words. Among these topics, top-20 keywords of 2 topics could be collected in a single topic. Figure 4.7 shows these top words. The first topic is collected under the topic of "Energy" and the second topic is collected under the topic of "Business".

CHAPTER 5
CONCLUSION

5.1 Conclusion and Discussion

Classification of large datasets into groups can be achieved using Latent Dirichlet Allocation. We modeled and tested both synthetic and real textual data and clustered the documents of these datasets into groups called topics. We found good measures, Square Error and Kullback-Leibler Divergence, for finding the true number of topics of a dataset before using this dataset as an input to LDA. Given this true number of topic for the dataset, LDA produces more accurate results when the dataset is classified into this number of topics using LDA. As a result, the minimum value of Square Error is the point where LDA is ran with true number of topics. Moreover, Kullback-Leibler Divergence is constant if LDA is run with number of topics less than true number of topics. Using these parameters, we found out that Enron dataset has exactly 9 topics.

5.2 Future Work

Latent Dirichlet Allocation relates each document of the dataset with each topic with some frequency value. Therefore, each document is related to each topic to some extent. Clustering the dataset partitions the documents into disjoint group such that each group represents documents that are most relevant to a topic. This maximizes the number of intra-cluster documents, that is, the number of documents in the cluster. Topics are not that strictly grouped though, some documents may be relevant to more than one topic, and some documents may not be even relevant to any topic when the dataset is partitioned into topics with LDA. For this purpose, we can define hubs and outliers. Hubs are the documents that bridge two or more topics. Outliers are documents that are marginally connected to topics. Identifying hubs are useful to find out documents which are relevant to more than one topic. Identifying outliers are useful, because they are irrelevant to all topic and can be treated as noise in the data [16].

Mining social networks from a corpus of email messages is another interesting topic. For this purpose, the best idea to start with is to make use of "From" and "To" fields of email messages to construct the network of email messages as a digraph. In this digraph, nodes are the people and for each email message, there exists a directed edge form the sender of the email to the recipients of the email [13]. After differentiating all edges from a person to any other person in the network by unique identifiers of email messages, we can keep track of activities in the community and mine hidden communities by finding pattern in this directed email network [14]. These patterns which are extracted from information flow of email messages identifies closed groups in large communities.

LITERATURE CITED

[1] Rakesh Agrawal, Sridhar Rajagopalan, Ramakrishnan Srikant, and Yirong Xu. Mining newsgroups using networks arising from social behavior. In *WWW '03: Proceedings of the 12th international conference on World Wide Web*, pages 529–535, New York, NY, USA, 2003. ACM Press.

[2] R. Bekkerman, A. McCallum, and G. Huang. Automatic categorization of email into folders: Benchmark experiments on Enron and SRI corpora. Technical Report IR-418, Center of Intelligent Information Retrieval, UMass Amherst, 2004.

[3] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent Dirichlet Allocation. *J. Mach. Learn. Res.*, 3:993–1022, 2003.

[4] Wikipedia contributors. Kullback-Leibler divergence.

[5] Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, and Richard A. Harshman. Indexing by Latent Semantic Analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.

[6] Pedro Domingos and Matt Richardson. Mining the network value of customers. In *KDD '01: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 57–66, New York, NY, USA, 2001. ACM.

[7] Jiawei Han. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2005.

[8] Gregor Heinrich. *Parameter Estimation for Text Analysis*, 2008. Technical Report.

[9] Thomas Hofmann. Probabilistic Latent Semantic Indexing. In *Proceedings of the 22nd Annual ACM Conference on Research and Development in Information Retrieval*, pages 50–57, Berkeley, California, August 1999.

[10] Bryan Klimt and Yiming Yang. The enron corpus: A new dataset for email classification research. In Jean-Franois Boulicaut, Floriana Esposito, Fosca Giannotti, and Dino Pedreschi, editors, *ECML*, volume 3201 of *Lecture Notes in Computer Science*, pages 217–226. Springer, 2004.

[11] Ramesh Nallapati, William Cohen, and John Lafferty. Parallelized Variational EM for Latent Dirichlet Allocation: An Experimental Evaluation of Speed and Scalability. In *ICDMW '07: Proceedings of the Seventh IEEE International Conference on Data Mining Workshops*, pages 349–354, Washington, DC, USA, 2007. IEEE Computer Society.

[12] Martin Porter. *An algorithm for Suffix Stripping*, 1980.

[13] Paat Rusmevichientong, Shenghuo Zhu, and David Selinger. Identifying early buyers from purchase data. In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 671–677, New York, NY, USA, 2004. ACM.

[14] Xiaodan Song, Belle L. Tseng, Ching-Yung Lin, and Ming-Ting Sun. Personalized recommendation driven by information flow. In *SIGIR '06: Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 509–516, New York, NY, USA, 2006. ACM.

[15] Jaime Teevan, Susan T. Dumais, and Eric Horvitz. Personalizing search via automated analysis of interests and activities. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 449–456, New York, NY, USA, 2005. ACM.

[16] Xiaowei Xu, Nurcan Yuruk, Zhidan Feng, and Thomas A. J. Schweiger. SCAN: A Structural Clustering Algorithm for Networks. In *KDD '07: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 824–833, New York, NY, USA, 2007. ACM.

[17] Kai Yu, Shipeng Yu, and Volker Tresp. Dirichlet enhanced latent semantic analysis. In *LWA*, pages 221–226, 2004.