Discriminative Phrase Discovery for Effective Text Classification

Chandana Dasari
Department of Computer Science and Automation
Indian Institute of Science
Bangalore, Karnataka 560012
Email: chandana.d@csa.iisc.ernet.in

***Abstract*—Text classification has received a sustained interest in Data Mining and Machine Learning community with various applications ranging from Spam filtering to sentiment analysis. The conventional approaches for text mining involve representing a document using Bag of Words (BoW) model. BoW model is very likely to ignore semantics, as order of words is not taken into account. Still BoW is widely used because of the ease of representation. It has been well established that going beyond this uni-gram model and finding out relevant bi-gram and trigram phrases in corpus turns out to be beneficial. However, finding exact(right set of) phrases from BoW representation of a corpus is a hard problem. In this paper we propose and discuss various approximations for discovering phrases in a corpus using BoW representation only. To determine the effectiveness of discovered phrases we study their role in text classification problem, which is also a widely studied among text mining practitioners. We further present a comparative study of these approaches in text classification domain using different state-of-the-art classification techniques. Principal contributions of our work include: (i) Approaches for discovering phrases in a corpus, (ii) A comparative study of role of phrases in supervised learning. For validating effectiveness of our approaches we took 2 real world datasets namely Wikipedia, 20 Newsgroup Experimentally we found Bag of Phrases(BoP) model to significantly outperforms BoW in almost every case.**

I. Introduction

In recent years with ease in availability and access to platforms for publishing and sharing content, a large volume is getting generated in forms of blogs and micro-blogs. As of now around 80-90% of the worlds data is held in unstructured formats like web pages, Emails, Technical documents, Digital libraries. Such huge volume makes manual analysis, inference, and retrieval of these documents a tedious and close to impossible task. This has stressed upon tools and techniques for automated analysis of such large text repositories. Text mining is the area which largely deals with extracting useful information out of these unstructured text documents for efficient retrieval, organization, understanding and analysis. It includes techniques like Natural Language Processing (NLP), Information Retrieval, Document Classification, Topic models etc. Text Mining finds applications in various fields like Spam filtering in mails, ranking of blogs on social blogging sites, sentiment analysis of user reviews on e-commerce sites etc. A conventional way of representing documents followed by majority in literature is using vector space model. In the vector space model each document is represented as a d-dimensional vector where d is the vocabulary size. Each component in the vector corresponds to importance of respective term in the document which may be based on term-occurrence, term-frequency, inverse-document-frequency, etc. This way of representing documents is often termed as Bag-of-Words, as it does not respects the order in which terms occur in a document. The reason for Bag-of-Words model being popular can be attributed to its ease of representation which makes other inference and analysis tasks on top of it simpler. On the other hand a complex lossless representation retaining every information about the document will add unnecessary complexity to the problem. However, it has been established that a trade-off between these two yields a better performance [] compared to Bag-of-Words approach with comparatively similar computational costs. The trade-off is to look beyond unigrams and discover bi grams and trigrams (phrases) to be included in the vector space representation. Exploring and representing documents as collection of phrases helps in understanding the document semantically in a better way.

However, discovering phrases from unstructured text is not an easy problem as with the increase in vocabulary size, the number of possible phrases grows super linearly. Finding exact phrases from Bag-of-Words representation of a document is a hard problem, so we try to solve an approximation to this problem where phrase and its frequency is defined as:

Definition A phrase $\mathcal{P}$ with length r is an unordered set of r terms: $\mathcal{P} = \{w_1, w_2, \ldots, w_r | w_i \in \mathcal{W}\}$, where $\mathcal{W}$ is the set of all unique terms in a content-representative document collection. The frequency $f(\mathcal{P})$ of a phrase is the number of documents in the collection that contain all of the r terms[1].

We in our work view the document term matrix corresponding to a given corpus as a network wherein we have two type of nodes "Document" and "Words" . This forms a bipartite graph, where edges between entities corresponds to a relationship indicating that a word *"occurs in"* document. We further formulate the problem of discovering phrases as a link prediction problem between word nodes. The possibility of link between two words indicates their possible involvement in same phrase. However, given d number of word nodes the number of links to be predicted are $O(d^2)$. To overcome this inefficiency we propose some extensions to this to reduce the space over which links are to predicted.

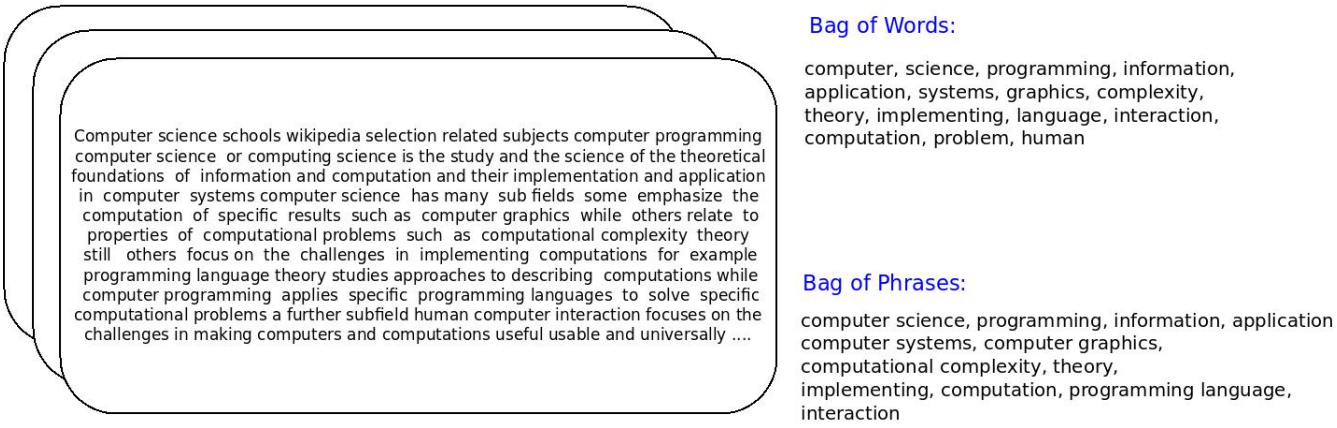


Fig. 1. An illustration of Bag-of-Words vs Bag-of-Phrases

To evaluate the effectiveness of discovered phrases we determine their role in text classification problem. Text classification is task of assigning documents to set of pre-defined classes. Formally it can be described as:

Definition Given a set of m documents $\mathcal{T} = \{x_1, x_2, \ldots, x_m\}$ and their corresponding labels $\mathcal{L} = \{l_1, l_2, \ldots, l_m\}$, where $x_i \in \mathbb{R}^d$ is vector space representation of i^{th} document and $l_i \in \mathcal{C}$ is the label of i^{th} document. The task is to learn a function $f : \mathbb{R}^d \rightarrow \mathcal{C}$, for mapping unseen documents to set of k predefined classes $\mathcal{C} = \{c_1, c_2, \ldots, c_k\}$.

To determine the goodness of set of discovered phrases, we use the performance of resulting model as an evaluation measure. Contents of the rest of the paper are organized as follows: Section II describes some of the notations and symbols used in this text. A brief overview of related concepts required to understand proposed approach is mentioned in section III

II. Preliminaries and Notation

A. Notation

We introduce some of the related concepts and definitions used in this report here. A set of collection of documents (corpus) is represented as $\mathcal{T}$, which is represented in vector space model as $\mathcal{T} \in \mathbb{R}^{m \times n}$. Unless specified calligraphic symbols refers to a set of objects, uppercase bolds is used for denoting matrices, and lowercase bold is used for denoting a vector. For evaluation purpose we split the available corpus in training and test denoted by $\mathcal{T}_{train}$ and $\mathcal{T}_{test}$ respectively. Similarly label set $\mathcal{L}$ is also split into $\mathcal{L}_{train}$ and $\mathcal{L}_{test}$ accordingly. $|.|$ is used for obtaining the cardinality of sets. An exhaustive list of notations used is provide in table II-A.

TABLE I
Symbol Table

SYMBOL	DESCRIPTION
$\mathcal{T}$	corpus(set of documents)
$\mathcal{C}$	Set of classes
$\mathcal{T}_{train}$	Train Documents
$\mathcal{T}_{test}$	Test Documents
$\mathcal{L}_{train}$	Train Lable
$\mathcal{L}_{test}$	Test Lable
$\mathcal{P}$	Phrase of length r
$\mathbf{V}$	Vocabulary
$\mathbf{D}$	Document Term Matrix
$\mathbf{DP}$	Document Phrase Matrix
$\mathbf{TP}$	Term Phrase Matrix

B. Bag-of-Words

In text mining the most commonly used model for representing text is Bag-of-Words. It treats a document as a multi set of terms occurring in it, ignoring syntax of underlying language and ordering among terms. This assumption in representation makes analysis easy and highly efficient in terms of computation.

Given a document $\mathcal{D}$, and an ordered list of terms $\mathcal{V}$, where $|\mathcal{V}| = n$. The vector space representation for document $\mathcal{D}$ using bag-of-words model is,

$$d_{\mathcal{D}} = (f_{\mathcal{D}1}, f_{\mathcal{D}2}, \ldots, f_{\mathcal{D}n})^{\mathsf{T}}$$

where $f_{\mathcal{D}i}$ is the importance of i^{th} term in document $\mathcal{D}$ Various term weighing techniques can be combined with bag-of-words; some of the popular ones are *(1)* binary weighing, *(2)* term-frequency, *(3)* inverse document frequency, *(4)* term-frequency inverse-document-frequency.

For a corpus $\mathcal{T} = \{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_m\}$, the vector space representation $\mathbf{D}$ is shown in table 2. Element f_{ij} in $\mathbf{D}$ corresponds to weight of j^{th} term in i^{th} document.

C. Bag of Phrases

A phrase is a set of words as defined in section I. A term can occur in altogether different contexts in different documents. For instance, the term *nuclear* when occurring in *nuclear weapon* is related to war and defense while same term when used in *nuclear family* has an orthogonal context related to social life. Term *nuclear* does not have enough discriminative

Fig. 2. Document-Term Matrix

	w_1	w_2	$\cdots$	w_n
d_1	f_{11}	f_{12}	$\cdots$	f_{1n}
d_2	f_{21}	f_{22}	$\cdots$	f_{2n}
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
d_m	f_{m1}	f_{m2}	$\cdots$	f_{mn}

power here in this example but, if entire phrases are used a better discrimination is being guaranteed. Some similar examples are: *data mining* and *coal mining*, *bill gates* and *logic gates* etc. In Bag-of-Phrases representation in addition to terms phrases also form a part of vocabulary.

For a corpus $\mathcal{T} = \{\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_m\}$, the vector space representation using bag-of-phrases $\mathbf{D}$ is shown in table 3. Element f_{ij} in $\mathbf{D}$ corresponds to weight of j^{th} term in i^{th} document, and $\overline{f}_{ij}$ corresponds to weight of j^{th} phrase in i^{th} document. The size of extended vocabulary becomes $n+p$, n being the number of terms and p number of phrases.

Fig. 3. Document-Phrase Matrix

	w_1	w_2	$\cdots$	w_n	p_1	p_2	$\cdots$	p_p
d_1	f_{11}	f_{12}	$\cdots$	f_{1n}	$\overline{f}_{11}$	$\overline{f}_{12}$	$\cdots$	$\overline{f}_{1p}$
d_2	f_{21}	f_{22}	$\cdots$	f_{2n}	$\overline{f}_{21}$	$\overline{f}_{22}$	$\cdots$	$\overline{f}_{2p}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
d_m	f_{m1}	f_{m2}	$\cdots$	f_{mn}	$\overline{f}_{m1}$	$\overline{f}_{m2}$	$\cdots$	$\overline{f}_{mp}$

III. Background

A. Zipf's Law

Feature space for text analysis consists of Unique terms(words or phrases) that occur in documents, which can easily run into millions for even a moderate sized text collection. A major difficulty in text mining is to handle this high dimensionality of feature space. In accordance with the Heap's Law the size of vocabulary increases with the number of tokens present in a corpus. The relationship between the two is given by,

$$|\mathcal{V}| = k \cdot T^b$$

where $|\mathcal{V}|$ is size of vocabulary, T is the total number of tokens in corpus and $k(30 \leqslant k \leqslant 100)$ and $b(\approx 0.5)$ are parameters for the corpus.

It is desirable to reduce the dimensionality of feature space without losing much information about documents. It is observed the terms which are highly rare and highly common does not help in inferring much about the documents. Word frequency distribution in a corpus follows a heavy tail distribution. Number of low frequency(e.g. "accordion", "catamaran", "ravioli") terms is sufficiently high(compared to terms with moderate(e.g. "book","school","science" etc.) or high frequency (e.g. "a", "the", "I", etc.). Zipfs law [2] empirically models the distribution of terms in a large corpus.

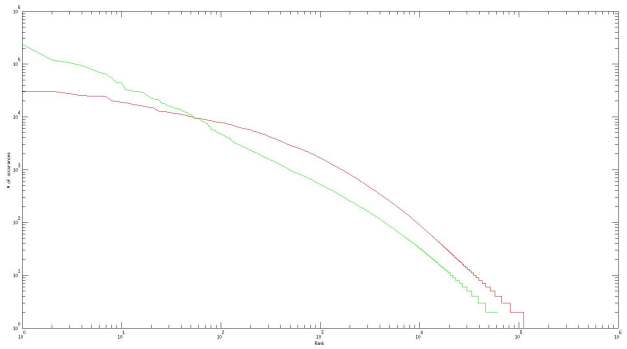


Fig. 4. Zip Curve

It states that : the r^{th} most frequent word has a frequency $f(r)$ that scales according to

$$f(r) \propto \frac{1}{r^\alpha} \quad ; \qquad \alpha \approx 1$$

where r denotes "frequency rank" of a word and $f(r)$ denotes its corresponding frequency in the corpus.

The idea behind Zipfs Law is that the frequency of a term decreases very rapidly with its rank. Based on rank Zipf's law can partition terms into three disjoint sets namely low frequency, moderate frequency and high frequency. Thus can be used to filter out terms in moderate frequency zone which are expected to be useful for analysis.

B. tf-idf weighting

tf-idf is significant and popular weighting technique as this value gives importance to the discriminative features which are not very frequent and not very rare. and it is a combination of the term frequency and inverse document frequency, to produce a composite weight for each term in each document.

The tf-idf weighting scheme assigns to term t a weight in document d given by

$$\text{tf-idf}_{t,d} = \text{tf}_{t,d} \times \text{idf}_t. \tag{1}$$

where

$$\text{idf} = -logP(t/d) = log\,\frac{N}{|d \in D : t \in d|} \tag{2}$$

The tf-idf value varies as follows with the term frequency.

- highest when t occurs many times within a small number of documents (thus lending high discriminating power to those documents);
- lower when the term occurs fewer times in a document, or occurs in many documents (thus offering a less relevance of a term);
- lowest when the term occurs in virtually all documents (less discriminating power).

At this point, we may view each document as a vector with one component corresponding to each term in the dictionary,

together with a weight for each component that is given by above equation (1)

C. Link prediction

Link prediction is an important task for analyzing social networks which also has applications in other domains like, information retrieval, bioinformatics and e-commerce. There exist a variety of techniques for link prediction, ranging from feature-based classification and kernel based method to matrix factorization and probabilistic graphical models. These methods differ from each other with respect to model complexity, prediction performance, scalability, and generalization ability.

Formally Link Prediction can be defines as follows :

Given a network $G < V, E >$ in which an edge $e = (u, v) \in E$ represents some form of interactions between its vertices u and v at a particular time $t(e)$.
Link prediction is a task to predict the edges at time t^1 where $t \leq t^1$ i.e $G[t, t^1]$ is asubgraph of G restricted to the edges with time stamps between t and t^1.

In a supervised training setup for link prediction, we can choose a training interval $[t_0, {t_0}^1]$ and a test interval $[t_1, {t_1}^1]$ where $t_0 < t_1$. Now the link prediction task is to output a list of edges not present in $G[t_0, {t_0}^1]$ but are predicted to appear in the network $G[t_0, {t_0}^1]$.

In any network, as the network evolves, new connections might develop. Link prediction measures [3] try to infer the likelihood of a link being formed between a pair of nodes. Typically link between two nodes is predicted if the two nodes have high similarity. We are using link prediction methods to get the phrases based on the strength of word links.

As we are visualizing our data as a graph, Where a vertex corresponds to a word and an edge represents a some form of association between the corresponding words. The associations among vertices are based on mutual presence of words in the documents.

We use link prediction techniques to get the phrases based on the strength of word links in our network formed by words.

Some of the existing link prediction techniques are:

- Common neighbors
- Jaccard's coefficient
- Adamic Adar
- Resourse Allocation Index

Let Γ(x) denote the set of neighbors of x

1) Common neighbors: This is based on the idea that links are formed between nodes which share many common neighbors. For two nodes x and y, the number of their common neighbors is defined as
Common-Neighbor (x, y)=$|\Gamma(x) \cap \Gamma(y)|$

2) Jaccard's Coefficient: This measure is a normalized version of common neighbors.
It measures how likely a neighbors of x is to be a neighbors of y and vice-versa. It defines the probability that a neighbor of x or y is a common neighbor of both. So for large number of common neighbors, the score would be higher.
$JC(x, y)$= $|\Gamma(x) \cap \Gamma(y)|/ |\Gamma(x) \cup \Gamma(y)|$

3) Adamic-Adar: It assigns large weight to common neighbors z of x and y, which themselves have few neighbors. This measure weighs the common neighbors with smaller degree more heavily. This works better than the previous measures.
Adamic/adar (x, y)= $\Sigma_{z \epsilon \Gamma(x) \cap \Gamma(y)} \frac{1}{log|\Gamma(z)|}$

4) Resource Allocation Index: Score between nodes x and y is calculated as the sum of inverse of the degree of each of the common neighbors z between x and y.
Resource Allocation Index(x,y)=$\Sigma_{z \epsilon \Gamma(x) \cap \Gamma(y)} \frac{1}{|\Gamma(z)|}$

D. Normalized Mutual Information

Mutual information is a method commonly used in statistical language modeling of word associations and related applications.
This feature selection method is to compute $MI(t, c)$ as the expected mutual information (MI) of term t and class c. MI measures how much information the presence or absence of a term contributes to making the correct classification decision on c.
If one considers the two way contingency table of a term t and a category c, where A is the number of times t and c co-occur, B is the number of time the t occurs without c, C is number of times c occurs without t, and N is the total number of documents, then the mutual information criterion between t and c is defined to be

$$I(t, c) = \log(\frac{(A*N)}{((A+C)*(A+B))})$$

This measure used in order to determine the correlation of word(w) and category(c).
and

Formally we can define MI of two random variables X and Y as:
$MI(W; C) = \sum_j \sum_k P(W_j \cap C_k) \frac{P(W_j \cap C_k)}{P(W_j)P(C_k)}$

where $P(W_j, C_k)$ is the joint probability distribution function of W and C
$P(W_k)$ and $P(C_k)$ are the marginal probability distribution functions of W and C respectively.
Intuitively, mutual information measures the information that W and C share. Mutual information can be also defined as

$$MI(W,C) = H(W) + H(C) - H(W,C)$$

as MI varies with the change in overlap, we are moving to NMI which is invariantand NMI defines as :

$$NMI(W,C) = \frac{MI(W,C)}{[H(W)+H(C)]/2}$$

and $H(W)$ and $H(C)$ are entropies associated with random variables X and Y respectively.

and the entropy $H(W)$ defines as:

$$H(W) = -\sum_k P(w_k) \log P(w_k)$$

E. F-measure

Accuracy is a traditional statistical performance measure. and it is defined as

$$accuracy = \frac{tp+tn}{tp+fp+tn+fn}$$

where
tp: no of samples truly classified as positive
tn: no samples truly classified as negative
fp: no of samples falsely classified as positives
fn: no of samples falsely classified as negatives
But this measure does badly when data is skewed (unbalanced).

for example: Consider samples in which 90 belongs to class1 and remaining 10 belongs to class2. Even if our classification model wrongly classifies the entire 10 samples belongs to the class2, the accuracy will be still 90% which is a very good accuracy.
Hence accuracy is not a good measure when data is unbalanced. so we are moving to the other performance measures like precision and recall based.

Where precision and recall are defined as

$$precision = \frac{tp}{tp+fp}$$

for the above example precision of class1 is 90% and precision of class2 is 0%

$$recall = \frac{tp}{tp+fn}$$

for the above example recall of class1 is 100% and of class2 is 0%
The above discussion points out the significance of both precision and recall, and signifies the importance of giving proper weight to both precision and recall. one such measure is F-measure. In general F-measure is defined as follows

$$F_\alpha = (1+\alpha^2) * (\frac{precision*recall}{precision+recall})$$

TABLE II
CONFUSION MATRIX

		p(positive)	n(negative)
Predicted Values	Y	True Positives(tp)	False Positives (fp)
	N	False Negatives(fn)	True Negatives(tn)

It is a weighted measure of precision and recall. When $weight(\alpha) = \frac{1}{2}$ we call it as a F1-measure.

$$F1 = 2\frac{precision * recall}{precision + recall}$$

F. ROC Curve

To measure the Classifier Performance we use the ROC curve[4]. Since accuracy measure will fail if the data we are dealing has class imbalance. This measure is useful when dealing with highly skewed datasets. Formally, each instance I is mapped to one element of the set p, n of positive and negative class labels.
To distinguish between the actual class and the predicted class we use the labels Y, N for the class predictions produced by a model.

Given a classifier and Instance(I): The decision made by the classifier can be represented in a structure known as a confusion matrix or contingency table. The confusion matrix has four categories:
True Positives(TP) are examples of correctly labeled as positives. False Positives(FP) refer to negative examples incorrectly labeled as positives. True Negatives(TN) corresponds to negatives correctly labeled as negatives. Finally False Negatives(FN) refer to positive examples incorrectly labeled as negatives.
A confusion matrix shown in Figure . The confusion matrix can be used to construct a point in ROC space. In ROC space, one plots the FPR on x- axis and TPR on y- axis. The FPR measures the fraction of negative examples that are misclassified as positives. The TPR measures the fraction of positive examples that are correctly labeled.

In the above table numbers along the major diagonal represent the correct decisions made, and the numbers off the diagonal represent the errors.
Now we will see some of the measures for judging classifier model
tpr= $\frac{tp}{p}$ = Recall
fpr=$\frac{fp}{n}$
Precision=$\frac{tp}{(tp+fp)}$
Accuracy=$\frac{(tp+tn)}{(p+n)}$
Sensitivity=Recall
Specificity=1-fpr
we construct ROC by taking FPR on X-Axis and TPR on Y-axis.

In a Receiver Operating Characteristic (ROC) curve the true positive rate (Sensitivity) is plotted in function of the false positive rate (100-Specificity) for different cut-off points. Each point on the ROC curve represents a sensitivity/specificity pair corresponding to a particular decision threshold. A test with perfect discrimination (no overlap in the two distributions) has a ROC curve that passes through the upper left corner (100% sensitivity, 100% specificity). Therefore the closer the ROC curve is to the upper left corner, the higher the overall accuracy of the test (Zweig and Campbell, 1993).

1) Area Under The Curve: The area measures discrimination, that is, the ability of the test to correctly classify

IV. State-of-the-Art

Phrases have been found to have significant role in text summarization[5], news filtering[6], classification[] and many other applications. Some of the earliest works establishing importance of phrases includes work of Song et.al [7], in which authors consider the term pairs and expand the unigram model of a document with the bi-gram model by using statistical language model. However, the approach considers all word pairs in the documents. This doesn't sound intuitive particularly for pairs which that cross sentence boundaries; and also the approach doesn't scale well.

Han et.al. [8] have shown that frequent patterns have more discriminative power compared to their disjoint counterparts. The work considers "**minimum support**" criteria for feature selection and theoretically establishes that features with low minimum support and high minimum support have limited discriminative power. This can be attributed to their limited coverage in the corpus and commonness in data respectively. Authors further use MMRSFS(Maximal Marginal Relevance Feature Selection) for feature selection. On the similar lines Zhang et. al. [9] presented the concept of multi-word expression for classification instead of individual terms. They apply an algorithmic approach to extract multiwords using lexical tools.

[10][11] observed that if adding new bi-gram features with high discriminative power to existing feature set, makes data more separable. Wallach et. al. proposed a bi-gram topic model[12] which utilizes the hierarchical Dirichlet language model, by incorporating the concept of bi grams into topic models. Bigram Topic Model is a hierarchical generative probabilistic model that extends unigram topic model and incorporates both n-gram statistics and latent topic variables. The model hyper parameters are inferred using a Gibbs EM algorithm. Prediction of phrases is based on conditional probabilities conditioned on the immediately preceding word. However, proposed bi-gram model is not effective and does not scale well beyond few hundreds of documents. Considering scalability issues, Wang et.al proposed a topical n-grams model(TNG)[13], which captures the topics and as well as topical phrases. This model overcomes the problem in traditional collocation approaches where a discovered phrase is always considered as a collocation regardless of the context. The probabilistic approach adopted by TNG not only uses phrases, but also guarantees that meaningful phrases are obtained in right context.

For finding interesting phrases in an ad-hoc document collection, [14] [15] proposed Phrase Inverted Indexing and Sequence Pattern Indexing respectively. [16] proposed a frame work for text analytics and emphasized the significance of top-k most interesting phrases in ad-hoc subsets of the corpus. Where in chosen ad-hoc subsets terms are frequent but are relatively infrequent in the overall corpus. Authors restrict their phrase search to only phrases that satisfy the minimum support(5 or 10) criteria. The approach proposes "forward index method" for extracting the top-k phrases.

Frequent pattern mining is often used to find the discriminative patterns for tasks like classification or clustering. For example,[17] tries to select rules with the top-k highest confidence values on each class from all mined frequent patterns. Hu et. al. [5] discusses the application of mined phrases in opinion mining from customer reviews. Authors aim to summarize the customer reviews of a product based on feature-based opinion summarization.

[6][1] discusses the topical hierarchy from collection of text (CATHY) which is a recursive clustering and ranking approach for topic hierarchy generation. Yet another application [6] discusses about the Automatic Construction and Exploration of News Information Networks based on the CATHY discussed[1] for mainly extracting the typed entities from the plain text.

V. Phrase Discovery using Link Prediction

In this section we discuss the proposed approach for discovering phrases in a corpus from bag-of-words representation. Phrase as defined earlier is an unordered set of terms. It is intuitive to think of phrases as frequently co-occurring patterns in documents which also have some semantics. We begin our work by exploring co-occurrence matrix formed out of a corpus. We further use this approach as a baseline for comparison against all other approaches. For computing co-occurrence matrix we represent corpus as a Document-Term matrix with binary weights. Each element of co-occurrence matrix quantifies the number of documents in which the involved terms are co-occurring. To determine quality phrases we decide some threshold over the minimum number of documents in which the term pair should be present. The pseudo code for the proposed approach is described in **Algorithm 1**.

Although the above approach can find phrases which are highly co-occurring; but it seems that the co-occurrence is not the only criteria to get meaningful phrases. For instance consider a corpus consisting of documents related to information technology. It is expected that word *computer* will be present in almost every document. Now consider a proper phrase *hard disk*, term *disk* will have as many co-occurrences with term *hard* as it is having with term *computer*. So based on chosen threshold the above approach will either select both *computer disk* and *hard disk* as phrase or none.

Algorithm 1: COC based Phrase Extraction algorithm

Input: **D**(Document Term Matrix) of dimension $m * n$, **V**(vocabulary) of dimension $n * 1$
Output: Top scored Phrases
$P \leftarrow null$;
$\mathbf{COC}_{n*n} \leftarrow \mathbf{D}^T * \mathbf{D}$;
sort matrix COC and return indices in I and J
extract top t word pair list;
for $i = 1 \rightarrow t$ **do**
  add $< \mathbf{V}(I(i)), \mathbf{V}(J(i)) >$ to the list of phrases P
end
return list P of phrases generated

One way of overcoming this drawback is to give different weights to different terms based on their document frequency. Inversed document frequency as described in III can be used to avoid such cases. However, again not all documents are of same length and co-occurrence in a larger document should not be treated same as the one in smaller document. Network analysis provides rich literature in this direction. We leverage link prediction techniques from network analysis for phrase discovery.

A. Link Prediction framework

The corpus can be viewed as a heterogeneous networks consisting of document and term nodes, wherein documents are linked to terms occurring in it. Phrase discovery in such a network can viewed as a problem of predicting link between terms. In given network connections between terms is only through the documents containing them; and, the quality of predicted link between terms depends only on the number of shared documents and the strength of links between corresponding document and terms. The common approaches used for link prediction are summarized in section III-C. Link prediction based approach can be thought of as a generic framework for phrase discovery (bi grams in particular). For example, the trivial approach of co-occurrence based phrase discovery mentioned above corresponds to using common neighbor for link prediction. The document-term network in our case is sufficiently dense (≈ 0.02). We in our experiments have used Adamic-Adar (AA) and Resource Allocation Index (RAI) because of their superior performance in dense networks. Figure 5 illustrates link prediction in a document term network. The underlying corpus in example consists of four documents:

D1 : *Machine learning is sometimes conflated with data mining.*
D2 : *Data mining focuses more on exploratory data analysis.*
D3 : *Machine learning focuses on prediction, based on training data.*
D4 : *Machine learning and statistics are closely related fields.*

In figure red and green nodes represents documents and terms respectively. The Grey colored edge between document and term indicates *"is contained in"* relationship between document and terms. The predicted links which corresponds to possible involvement of terms in a phrase is represented by blue edges. The pseudo code for phrase detection using RAI for link prediction is described in **Algorithm 2**. However, the same framework can be used for any link prediction technique.

Algorithm 2: LP(RAI) based Phrase Extraction algorithm

Input: **D**(Document Term Matrix) of dimension $m * n$, **V**(vocabulary) of dimension $n * 1$
Output: Top score Phrases
$P \leftarrow null$;
$L \leftarrow Sum(\mathbf{D}, 2)$;
  ▷ *L vocabulary sum for each document*
Compute $R \leftarrow \mathbf{D}^T * diag(L)^1 * \mathbf{D}$;
sort matrix R and return indices in I and J extract top t word pair list;
for $i = 1 \rightarrow t$ **do**
  add $< \mathbf{V}(I(i)), \mathbf{V}(J(i)) >$ to the list of phrases P
end
return list P of phrases generated

Although very effective technique for finding phrases, we found this method to be not scalable. The complexity of link prediction for the algorithms we considered is $O(mn^2)$. The number of documents and terms both can easily run into millions for a reasonably small sized corpus. However, predicting links between terms which are reasonably isolated from one another is not required. For example, computing strength of link between *island* and *disk* will be a wasted effort. The phrases which are of interest in analysis are generally discriminative in nature. Based on this notion we propose an extension to proposed framework for making it computationally efficient.

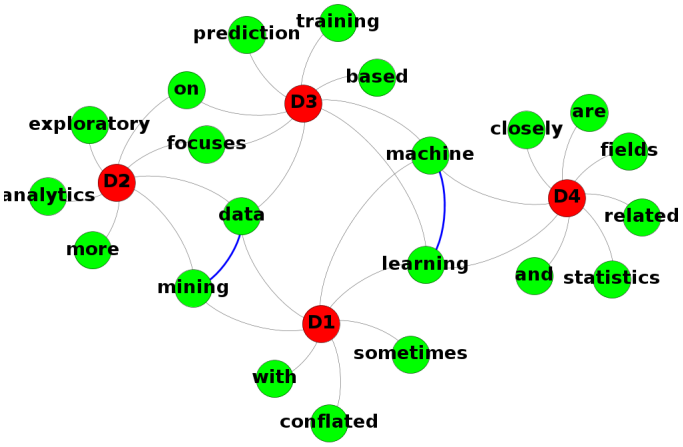


Fig. 5. Link Prediction in Heterogeneous Document-Term Network.

B. Normalized Mutual Information(NMI) based Phrase Detection

It is expected that the terms forming a phrase will mostly belong to same class and will have high discriminative power. It is wise to first find out discriminative terms for each class and then attempt to predict links between terms belonging to same class. Mutual information (MI) is a method commonly used in statistical language modeling of word associations and related applications. MI measures how much information the presence or absence of a term contributes in making the correct classification decision on c. We in our experiments use normalized version of mutual information described in **??** to generate list of discriminative terms for each class. Once we have these lists, using each list as constrained vocabulary we generate a network and use above described link prediction framework for discovering phrases. If each list on an average is having r number of terms then required effort is reduced from $O(mn^2)$ to $O(mr^2)$, where $r << n$. Pseudo code for the approach is described in **Algorithm 3**

Algorithm 3: NMI based Phrase Extraction algorithm

Input: **D**(Document Term Matrix) of dimension $m * n$, **V**(vocabulary) of dimension $n * 1$
Output: Top score Phrases
$KL \leftarrow null$;
$P \leftarrow null$;
Compute **NMI** of dimension $d * k$;
for $i = 1 \rightarrow k$ **do**
  for $j = 1 \rightarrow d$ **do**
    if ***NMI***$(j, i) > \tau_i$ **then**
      add $< w_j >$ that to Key list $\mathbf{kl}_i$;
    end
  end
end
for $j = 1 \rightarrow k$ **do**
  $\mathbf{coc}_j = \mathbf{kl}_j^T * \mathbf{kl}_j$
  apply LP(RAI) on $\mathbf{coc}_j$ results $\mathbf{r}_j$
  sort matrix $\mathbf{r}_j$ and return indices in I and J
  extract top t word pair list for each topic j;
  for $ii = 1 \rightarrow t$ **do**
    add $< \mathbf{V}(I(i)), \mathbf{V}(J(i)) >$ to the list of phrases P
  end
end
return list P of phrases generated

The proposed approach greatly reduced the space over which link prediction is needed to be performed. However, it was observed that there is still scope of doing well. In fact, the documents in a class as a whole does not form a cohesive group; but, the class itself can be broken into subclasses.

C. Non-negative Matrix Factorization based Phrase Detection

Similar class membership generally does not guarantee strong cohesiveness among involved terms. The number of sub-groups or subclasses having such property is much more than the number of classes. Group of highly coupled terms can be found by performing a clustering over terms. A clustering algorithm like k-means giving hard partitions may not serve the purpose here. A hard partitioning algorithm will exactly assign each term to one cluster. However, for phrase detection some terms may not be significant for any cluster. While some terms may be significant for more than one clusters. We explore Non-negative Matrix Factorization(NMF) for finding a soft clustering over terms. NMF is a technique generally used for dimensionality reduction, which projects the data along latent dimensions. These latent dimensions can be viewed as clustering of documents as well as terms.

Given a document-term matrix $\mathbf{D} \in \mathbb{R}^{m \times n}$ for corpus we use NMF to compute a low rank approximation to it by decomposing as product of two low rank (k) matrices $\mathbf{W} \in \mathbb{R}^{m \times k}$ and $\mathbf{H} \in \mathbb{R}^{k \times n}$. Matrix **W** gives a clustering of documents, while matrix **H** gives clustering of terms. The weight of terms along each dimension in **H**, signifies its importance in that cluster or along that dimension. We normalize the rows of **H**, and use a threshold τ for determining useful discriminative terms in each cluster. Similar to previous approach where NMI was used, here also we generate a network corresponding to terms in each cluster. Link prediction is performed only over terms constituting these network. Details of the approach are summarized as pseudo code in **Algorithm 4**.

Algorithm 4: NMF based Phrase Extraction algorithm

Input: **D**(Document Term Matrix) of dimension $m * n$, **V**(vocabulary) of dimension $n * 1$, Topics size k
Output: Top score Phrases
$P \leftarrow null$
$KL \leftarrow null$
$\mathbf{COC} \leftarrow \mathbf{D}^T * \mathbf{D}$
Factorize COC
$\mathbf{COC}_{n*n} \cong \mathcal{W}_{n*k} * \mathcal{H}_{k*n}$
for $i = 1 \rightarrow k$ **do**
  for $j = 1 \rightarrow n$ **do**
    if $\mathcal{W}(j, i) \geq \tau_i$ **then**
      add that features j to $\mathbf{kl}_i$;
    end
  end
end
for $j = 1 \rightarrow k$ **do**
  $\mathbf{coc}_j = \mathbf{kl}_j^T * \mathbf{kl}_j$
  apply LP(RAI) on $\mathbf{coc}_j$ results $\mathbf{r}_j$
  sort matrix $\mathbf{r}_j$ and return indices in I and J
  extract top t word pair list for each topic j;
  for $ii = 1 \rightarrow t$ **do**
    add $< \mathbf{V}(I(i)), \mathbf{V}(J(i)) >$ to the list of phrases P
  end
end
return list P of phrases generated

VI. Our Approach

Based on the above discussion, we can now define our goals in this project as follows:

1) A new representation model for documents is required which can capture the discriminative power of words as phrases. This model should be simple enough to be adopted by all standard classifiers without any major changes.
2) We want to reduce the effect of dimensionality on the performance of the classifiers.
   There are two possible ways to handle this issue:
   a) To find an appropriate distance metric which works well in high dimensional spaces
   b) To reduce the dimensionality itself.

In our work we adopt the second approach.

In short, our work flow can be divided into 4 phases, as follows

1) Data collection
2) Preprocessing
3) Learning
4) Evaluation

now we will examine each phase briefly

1) *Data collection*:
   In this phase we collect raw data from Wikipedia[1] which belongs to different classes. and some other text datasets available in online. In this context, raw data is the documents themselves.
2) *Preprocessing*:
   After collecting raw data we apply feature selection and feature extraction, which includes removal of stop words, lemmatisation etc, which are natural language processing techniques. and by Zipf's law, we eliminate most frequent and rare words, which anyways don't contribute much in text classification. This will result a set of candidate words which we can call as features. And the last step in pre-processing is expressing each document in terms of these features. and feature weights are assigned using tf-idf weighting technique which gives more weight to the low frequency words and less weight to the frequent words.
3) *Learning*: In this phase we apply classification techniques like Naive Bayes Classifier, support vector Machine, and K-nearest neighbors for learning from the data. We get model as output from this phase, which helps us to predict labels of new data.
4) *Evaluation*: In this phase we will measure the macro F1 and area under the curve(AUC) which are very good measures if the data is unbalanced. This tells us how better our model is for new data.

A. Phrase Extraction

Here, we focus phrases of length two. To find phrases, we need to keep a list of word pairs occurring together.

[1]www.wikipedia.com

TABLE III
Document-Term Matrix

	w_1	w_2	$\cdots$	w_n
d_1	f_{11}	f_{12}	$\cdots$	f_{1n}
d_2	f_{21}	f_{22}	$\cdots$	f_{2n}
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
d_m	f_{m1}	f_{m2}	$\cdots$	f_{mn}

We denote a phrase by $<w_i, w_j>$ where w_i is the first word of the phrase and w_2 is the second word. It is possible that in the corpus, a particular word w_i may be followed by more than one word. To avoid enumerating all phrases, we propose a novel approach for finding discriminative phrases in a corpus.

In table III $d_1, d_2, \cdots, d_m$ are m documents in the corpus. $w_1, w_2, \cdots w_n$ are n words (features) and f_{ij} is frequency of j^{th} word in i^{th} document.

Document-term matrix (**D**) is formed by indexing the words based on their frequency of occurrence in the documents. To increase the weight of low frequency words and to decrease the weight of frequent words, we used a term-frequency-inverse-document frequency **(tf-idf)** weightening technique in forming the document-term matrix.

In constructing the term-phrase matrix, we ordered the phrases based on scores returned by link prediction measures. We define a phrase of length two $P_i <w_{ij}, w_{ik}>$ is non-zero if both w_{ij} and w_{ik} are present, and thus we get the Term-Phrase matrix. we use these phrases as features for our BoP model. We generate the document-Phrase matrix by multiplying the document-term matrix and term-phrase matrix.

DP $=$ **D** $*$ **TP**

where

DP: Document-Phrase Matrix

D: Document-term Matrix

TP:Term-phrase Matrix

The new vector for Bag of Phrases model will have less dimensions (p) where p is the number of phrases. Feature value for a phrase in the vector are computed by using document-term matrix and phrase-term matrix. This new vector can be directly used for classification with any standard classifier. But using this BoP alone is not improving the classification, experiments are tabulated in the subsequent sections. So this is the step to choose a model which takes care of ordering of words and also gives weight to single words thus comes a new model, which is a combination of BoW and BoP models, which has dimension "$n + p$", where

n: number of words

p: number of phrases.

Fig. 6. Resultant Phrases

COC	RAI	NMI	NMF
Game Player	Unexpected operator expression	Literature survey	Football stadium
Soviet Army	Hurricane Storm	Software Developer	Municipal corporation
Catholic Church	Planet Orbit	political party	immunity power
Black Hole	Insulin Diabetes	political voting	electromagnetic wavelength
army battle	Floppy Disk	democratically country	big bang
Party Elections	Graphical Interface	Nation citizen	dog breed
Hurricane Storm	Program Developer	Judiciary principle	nuclear reactor
Team Player	Browser Cookie	corporate banking	element structure
Church Bishop	Mozilla Fire Fox	stock market	Cathedral church
Roman Empire	Software Developer	Rail passenger	Game Player
House Member	Mario Game	God Worship	Insulin Diabetes
Planet Orbit	Cocoa Chocolate	Graphical interface	polar beer
Soviet Union	Milky Galaxy	CPU functionality	Hydrochloric solution
Insulin Diabetics	Differential Equation	eastern railway	Mozilla Firefox
Microsoft user	mail Google	Party Elections	Game Player
Software Developer	Tooth enamel	Insulin diabetics	Game Player
Orthodox church	Helicopter toter	Differential equation	Software program
March April	Jupiter planet	metro passenger	Bank customer
browser programming	Quantum particle	Game Player	blood specimen
February march	Differential equation	program developer	classical thermodynamics

VII. Experimental Results and Evaluation

In this section we start with the introduction of datasets we used and methods for comparison. We then describe the results and evaluation.

A. Data sets

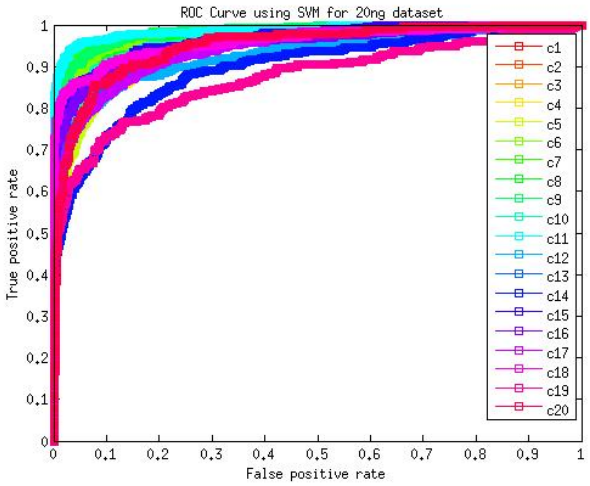
We analyze our performance on two data sets:

- **Wikipedia**: We collected a set of documents from wikipedia which are divided into 13 different classes. We minimally preprocessed the data sets by removing all stop words from the corpus and terms with frequency grater than 10 and less than 10000 in the data set resulting a 4878 document corpus consisting of 29060 unique terms.
- **20ng**: This data set is a collection of 18824 newsgroup documents, partitioned(nearly) evenly across 20 different classes[18].

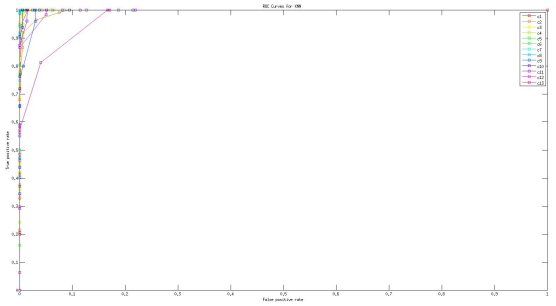
B. Data pre-processing

Text data is high dimensional. Traditional tools like k-NNC and decision trees cannot work in large dimensions. Hence, dimensionality reduction is very important for classification. We used wikipedia and 20ng data sets to evaluate the performance of our BoP methods. The data set is divided into training (60%) and Testing (40%) data are separated in time. This gives us a more realistic scenario for classification.
We cross validated all this experiments for 10 times.

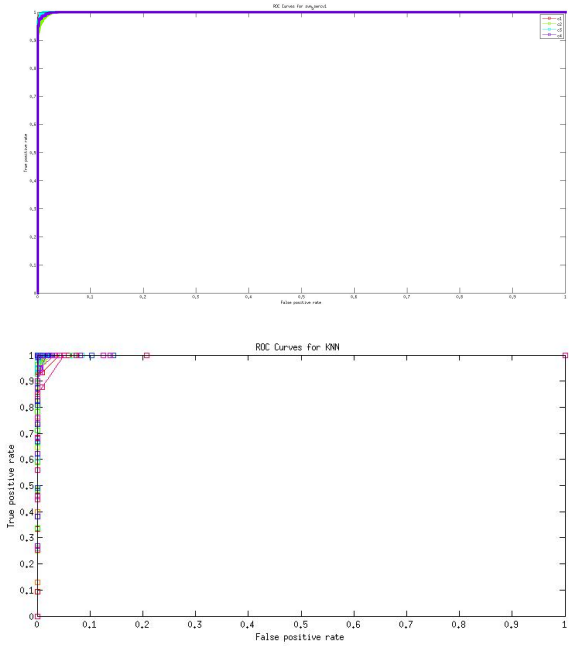
Before any classification task, one of the most fundamental tasks that needs to be accomplished is that of document representation and feature selection. While feature selection is also desirable in other classification tasks, it is especially important in text classification, due to the high dimensionality of text features and the existence of irrelevant (noisy) features. In general, text can be represented in two separate ways. The first is as a bag of words, in which a document is represented as a set of words, together with their associated frequency in the document. Such a representation is essentially independent of the sequence of words in the collection. The second method is to represent text directly as strings, in which each document is a sequence of words. Most text classification methods use the BOW representation because of its simplicity for classification purposes. In this section, we will discuss some of the methods which are used for feature selection in text classification. The most common feature selection which is used in both supervised and unsupervised applications is that of stop-word removal and stemming. In stop-word removal, we determine the common words in the documents which are not specific or discriminatory to the different classes. In stemming, different forms of the same word are consolidated into a single word. For example, singular, plural and different tenses are consolidate into a single word.



This is the ROC curve generated by svm classifier using 20 ng dataset.



This is ROC curve generated by knn classifier for the wikipedia dataset using the Link prediction Phrase extraction method



This is ROC curve generated by svm classifier for the wikipedia dataset using the NMI phrase extraction method.

Table 1,2,3 corresponds to the results obtained by KNN,SVM,NBC classifiers respectively. As stated above in the previous sections we can use the proposed model with any well known classifier. In our experiments we selected Naive Bayes classifier(NBC), kNN classifier and Support Vector Machine(SVM).

In Tables 1, 2 and 3 the first column corresponds to the about the dataset we used, and the second column corresponds to the F1 measure reported by BoW and remaining columns corresponds to the F1 measure reported by different BoP methods.

Wikipedia dataset which has 4878 documents divided into 13 different classes. 20ng dataset which has 18824 documents divided into 20 different classes.

Dataset	BoW	BoP COC	BoP NMF	BoP MI	BoP LP
wikipedia	72.33	73.04	76.28	75.54	78.97
20ng	73.55	72.05	77.06	79.68	79.24

Dataset	BoW	BoP COC	BoP NMF	BoP MI	BoP LP
wikipedia	80.09	81.54	86.88	86.24	82.37
20g	78.29	82.36	84.04	85.48	84.24

Dataset	BoW	BoP COC	BoP NMF	BoP MI	BoP LP
wikipedia	78.58	80.03	81.16	82.42	85.25
2ong	78.13	81.24	84.05	81.92	86.98

In table 3, classifiers are performing very well with the combination of BoP and BoW model, compared to BoP and BoW models. Classification accuracy has improved considerably.

VIII. Conclusions and Future Work

We have given various novel and simple schemes for extracting from the text corpus. The proposed Bag of Phrases Model using Link prediction methods captures the discriminative power of two words as an ordered pair. Currently used Bag of words model fails to capture the semantic effect of combination of words. We are able to improve the classification accuracy significantly for classifiers like Naive Bayes, K-nearest neighbor, Support Vector Machine. Still we see a scope for improvement in refining the phrases. Phrases may consist of more than two words which is not captured in the current work. Although we are increasing the dimensionality of the feature space, we are getting better accuracy. We can extend the phrase length using the property of transitivity and clique structure in the networks. The proposed combination of BoP model which captures the semantics of documents present in the form of phrases successfully. and we got considerable improvement in the classification accuracy. Results are tabulated above. The main advantage of this method is it works for all classifiers without any modification in them.

References

[1] Chi Wang, Marina Danilevsky, Nihit Desai, Yinan Zhang, Phuong Nguyen, Thrivikrama Taula, and Jiawei Han. A phrase mining framework for recursive construction of a topical hierarchy. In *Proceedings*

of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, pages 437–445. ACM, 2013.

[2] Ronald E Wyllys. Empirical and theoretical bases of zipfs law. *Library Trends*, 30(1):53–64, 1981.

[3] Mohammad Al Hasan and Mohammed J Zaki. A survey of link prediction in social networks. In *Social network data analytics*, pages 243–275. Springer, 2011.

[4] Tom Fawcett. Roc graphs: Notes and practical considerations for researchers. *Machine learning*, 31:1–38, 2004.

[5] Minqing Hu and Bing Liu. Mining opinion features in customer reviews. In *AAAI*, volume 4, pages 755–760, 2004.

[6] Fangbo Tao, George Brova, Jiawei Han, Heng Ji, Chi Wang, Brandon Norick, Ahmed El-Kishky, Jialu Liu, Xiang Ren, and Yizhou Sun. Newsnetexplorer: automatic construction and exploration of news information networks. In *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pages 1091–1094. ACM, 2014.

[7] Fei Song and W Bruce Croft. A general language model for information retrieval. In *Proceedings of the eighth international conference on Information and knowledge management*, pages 316–321. ACM, 1999.

[8] Hong Cheng, Xifeng Yan, Jiawei Han, and Chih-Wei Hsu. Discriminative frequent pattern analysis for effective classification. In *Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on*, pages 716–725. IEEE, 2007.

[9] Wen Zhang, Taketoshi Yoshida, and Xijin Tang. Text classification based on multi-word with support vector machine. *Knowledge-Based Systems*, 21(8):879–886, 2008.

[10] Deepak Gujraniya and M Narsimha Murty. Efficient classification using phrases generated by topic models. In *Pattern Recognition (ICPR), 2012 21st International Conference on*, pages 2331–2334. IEEE, 2012.

[11] Sreya Dey and M Narasimha Murty. Using discriminative phrases for text categorization. In *Int.Conf. Neural Information Processing(ICONIP)*, pages 273–280. Springer, 2013.

[12] Hanna M Wallach. Topic modeling: beyond bag-of-words. In *Proceedings of the 23rd international conference on Machine learning*, pages 977–984. ACM, 2006.

[13] Xuerui Wang, Andrew McCallum, and Xing Wei. Topical n-grams: Phrase and topic discovery, with an application to information retrieval. In *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*, pages 697–702. IEEE, 2007.

[14] Alkis Simitsis, Akanksha Baid, Yannis Sismanis, and Berthold Reinwald. Multidimensional content exploration. *Proceedings of the VLDB Endowment*, 1(1):660–671, 2008.

[15] Chuancong Gao and Sebastian Michel. Top-k interesting phrase mining in ad-hoc collections using sequence pattern indexing. In *Proceedings of the 15th International Conference on Extending Database Technology*, pages 264–275. ACM, 2012.

[16] Srikanta Bedathur, Klaus Berberich, Jens Dittrich, Nikos Mamoulis, and Gerhard Weikum. Interesting-phrase mining for ad-hoc text analytics. *Proceedings of the VLDB Endowment*, 3(1-2):1348–1357, 2010.

[17] Chengqi Zhang and Shichao Zhang. *Association rule mining: models and algorithms*. Springer-Verlag, 2002.

[18] Ana Cardoso-Cachopo. Improving Methods for Single-label Text Categorization. PdD Thesis, Instituto Superior Tecnico, Universidade Tecnica de Lisboa, 2007.