CLUSTERING OF DEEP WEBPAGES: A COMPARATIVE STUDY

Muhunthaadithya C[1], Rohit J.V.[1], Sadhana Kesavan[1] and Dr. E. Sivasankar[2]

[1]Department of CSE, NIT Trichy, India
[2]Assistant Professor, Department of CSE, NIT Trichy-620015, Tamilnadu, India

ABSTRACT

The internethas massive amount of information. This information is stored in the form of zillions of webpages. The information that can be retrieved by search engines is huge, and this information constitutes the 'surface web'.But the remaining information, which is not indexed by search engines – the 'deep web', is much bigger in size than the 'surface web', and remains unexploited yet.

Several machine learning techniques have been commonly employed to access deep web content. Under machine learning, topic models provide a simple way to analyze large volumes of unlabeled text. A 'topic'is a cluster of words that frequently occur together and topic models can connect words with similar meanings and distinguish between words with multiple meanings. In this paper, we cluster deep web databases employing several methods, and then perform a comparative study. In the first method, we apply Latent Semantic Analysis (LSA) over the dataset. In the second method, we use a generative probabilistic model called Latent Dirichlet Allocation(LDA) for modeling content representative of deep web databases.Both these techniques are implemented after preprocessing the set of web pages to extract page contents and form contents.Further, we propose another version of Latent Dirichlet Allocation (LDA) to the dataset. Experimental results show that the proposed method outperforms the existing clustering methods.

KEYWORDS

Latent Dirichlet Allocation, Latent Semantic Analysis, Deep Web, Cosine Similarity, Form Content and Page Content.

1. INTRODUCTION

1.1 Deep Web

The internet is a huge repository for an extremely wide range of information. Most of this information is accessed by querying through standard search engines. However, a huge portion of World Wide Web content is not explored by any standard search engines [1], which is referred to as the 'deep web'. This is due to several reasons. One major reason is because this kind of content requires authentication, which the web crawlers cannot get through.

The deep web is significantly gigantic. According to a July 2000 white paper [3], the deep web is 500 times larger than the surface web and indeed, continues to proliferate this magnitude[3].From the surveys presented in [4],[5] we can get a lucid understanding of the surface web. There are links buried far down on sites thriving on the surface web, which direct us to the news and history of the Deep web. It is quite common to come across deep web pages that have links terminating with the extension '.onion'.Such web pages require us to access them through browsers named 'Tor', which connect to the servers containing the repository and get access consent[6].

Today, the 60 largest Deep Web sites contain around 750 terabytes of data, surpassing the size of the entire Surface Web 40 times. 95% of the Deep Web is publically accessible, which is free of cost. Most of this constitutes databases of information that need to be searched directly from the specific website. A small pocket of the deep web is filled with hyper-secret communities who flock there to escape identification from authorities [7].

1.2 Topic Modeling

Topic models provide a simple way to analyze large volumes of unlabeled text. A topic consists of a cluster of words that frequently occur together. Using contextual clues, topic models can connect words with similar meanings and distinguish between uses of words with multiple meanings. Topic models express the semantic information of words and documents by 'topics'[8].

1.3 Clustering

Clustering refers to division of data into groups of similar objects where each group consists of objects that are similar between themselves and dissimilar to objects of other groups.

From the machine learning perspective, clustering can be viewed as unsupervised learning [9] of concepts. We can cluster images, patterns, shopping items, words, documents and many more fields. Clustering has wide scope of application in data mining, text mining, information retrieval, statistical computational linguistics, and corpus based computational lexicography.
Clustering has the following advantages**:**

1. Clustering improves precision and recall in information retrieval.
2. It organizes the results provided by search engines.
3. It generates document taxonomies.
4. It also generates ontologies and helps in classifying collocations.

A good clustering algorithm needs to have the characteristics, mentioned below:

1. Scalability
2. High dimensionality
3. Ability to discover clusters of arbitrary shape

2. DATA ACQUISITION AND PREPROCESSING

2.1 Dataset

In order to evaluate the performance of all the methods and perform a comparative study, we used the TEL-8dataset as mentioned in [8], UIUC Web integration repository [18]. The repository contains web search forms of 447 sources originally classified into 8 domains. The term TEL-8 refers to eight different web source domains belonging to three major categories (Travel, Entertainment and Living). Travel group is related to car rentals, hotels and airfares. Entertainment group has books, movies and music records interfaces. Finally the living group contains jobs and automobiles related query interfaces.

2.2 Parsing

Both Latent Dirichlet Allocation (LDA) and Latent Semantic Analysis (LSA) are clustering algorithms for texts. However, the dataset used here is not plain text, but HTML/XML documents. This makes it necessary to parse the given HTML/XML documents into plain text. This process involves extracting only the necessary information:

1. Page Content: Values and textual content visible on the web page.
2. Form Content: The value of the attributes in the form.

Most of the deep web pages are blocked by forms (e.g. login forms). To take advantage of this, we take into consideration the form attribute values, which can reveal important information about the type of deep web pages. Thus, this approach covers a large portion of deep web database for clustering [14].

3. METHODS

A topic consists of a cluster of words that frequently occur together. Topic models can connect words with similar meanings and distinguish between uses of words with multiple meanings. A variety of topic models have been used to analyze the content of documents and classify the documents. In our paper, we have primarily used two algorithms for clustering: Latent Semantic Analysis, and Latent Dirichlet Allocation [12].

3.1 Latent Semantic Analysis

Latent Semantic Analysis (LSA) or Latent Semantic Indexing talks about analyzing documents to find the underlying meaning or concepts of those documents. Latent Semantic Analysis arose from the problem of finding relevant documents from search words. The fundamental difficulty arises when we compare words to find relevant documents, because what we really want to do is compare the meanings or concepts behind the words. LSA attempts to solve this problem by mapping both words and documents into a 'concept space' and doing the comparison in this space.

This mapping of words and documents is implemented using centroid-based clustering. Here, the 'concept space' or cluster, is represented using a central vector, the cluster center, which may not be necessarily one among the document vectors. It is an iterative process that involves trial and failure and the iterations continue until the result achieves the desired properties. The clusters get refined in each iteration, when a new document vector gets added to the dataset and is assigned a cluster, and a new cluster center gets computed after the iteration terminates.

3.1.1 Occurrence matrix

LSA uses a term-document matrix which describes the occurrences of terms in documents - it is a sparse matrix whose rows correspond to terms and whose columns correspond to documents. A typical example of the weighting of the elements of the matrix is *tf-idf* (term frequency–inverse document frequency): the weight of an element of the matrix is proportional to the number of times the terms appear in each document and rare terms are up-weighted to reflect their relative importance.

3.1.2 Dot product

We represent the documents under consideration as vectors. The components of the vectors are the words in the entire concept space. The value of each component represents the weightage of each word in the concept space. The dot product for two vectors: $\vec{\alpha} = (\alpha_1, \alpha_2, \alpha_3 \dots)$ and$\vec{b} = (b_1, b_2, b_3 \dots)$, where $\alpha_n$and b_n are the components of the vector (features of the document, or *tf-idf* values for each word of the document in our example) and *n* is the dimension of the vectors:

$$\vec{a}.\vec{b} = \sum_{i=1}^{n} a_i b_i = a_1 b_1 + a_2 b_2 + \dots + a_n b_n$$

3.1.3 Cosine Similarity

The cosine similarity between two vectors is a measure that calculates the cosine of the angle between them. This metric can be seen as a comparison between documents on a normalized space because we're not taking into the consideration only the magnitude of each word count (tf-idf) of each document, but the angle between the documents. To build the cosine similarity equation, we must solve the equation of the dot product:

$$\vec{a}.\vec{b} = |\vec{a}||\vec{b}|\cos\theta$$

$$\cos\theta = \frac{\vec{a}.\vec{b}}{|\vec{a}||\vec{b}|}$$

Cosine similarity generates a metric that says how related are two documents using angle between the documents represented as vectors.The lesser the angle between the vectors, more related the documents are. Even if there was a vector pointing to a point far from another vector, they still could have a small angle and that is the central point on the use of Cosine Similarity, the measurement tends to ignore the higher term count on documents.

3.2 Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA) is a generative probabilistic topic model for collections of discrete data. The generative model describes that the documents are generated using the following algorithm. For each document, given N topics as input:

1. Firstly, a distribution over N topics is chosen randomly.
2. Next, for each word to be generated in the document,
   a.A topic is chosen from the distribution created in Step 1.
   b. Next, a word is chosen randomly from the corresponding distribution over vocabulary for the chosen topic.

The Dirichlet prior on per document topic distribution is given by

$$p(\theta|\alpha) = \frac{\Gamma(\sum_{i=1}^{k}\alpha_i)}{\prod_{i=1}^{k}\Gamma(\alpha_i)}\theta_1^{\alpha_i-1}....\theta_k^{\alpha_k-1} \qquad (1)$$

Joint distribution of topic mixture θ, a set of N topic z, a set of N words w

$$p(\theta, z, w|\alpha, \beta) = p(\theta|\alpha)\prod_{n=1}^{N} p(z_n|\theta)p(w_n|z_n, \beta) \qquad (2)$$

α -hyper parameter on the mixing proportions.
β -hyper parameter on the mixture components.
θ-the topic mixture proportion for a document.
N-document length
z_n-the topic for the *n*th word in the document.
w_n –the *n*th word in document.

In order to retrieve topics in a corpus, this generative process is reversed. The two distributions: distribution over topics and distribution over vocabulary for the topics are discovered by the reverse process. This process uses Gibbs Sampling [13], which is commonly used as a means of statistical inference. It is a randomized algorithm (i.e. an algorithm that makes use of random numbers, and hence may produce different results each time it is run.

3.2.1 Gibbs sampling

Gibbs sampling is one of the class of sampling methods known as Markov Chain Monte Carlo. We use it to sample from the posterior distribution, p(Z|W), given the training data W represented in the form of

$$W = \begin{bmatrix} \{w_1, \ldots, w_{N_1}\}, & \text{words in the 1st document} \\ \{w_{N_1+1}, \ldots, w_{N_1+N_2}\}, & \text{words in the 2nd document} \\ \ldots & \ldots \\ \{w_{1+\sum_{j=1}^{D-1} N_j}, \ldots, w_{\sum_{j=1}^{D} N_j}\} & \text{words in the } D\text{-th document} \end{bmatrix},$$

Gibbs sampling is commonly used as a means of statistical inference, especially Bayesian inference [15]. It is a randomized algorithm (i.e. an algorithm that makes use of random numbers, and hence may produce different results each time it is run), and is an alternative to deterministic algorithms for statistical inference such as variational Bayes or the expectation-maximization algorithm (EM). Gibbs sampling generates a Markov chain of samples, each of which is correlated with nearby samples.

3.3Latent Dirichlet Allocation – visiting inner links

For some web pages, it is possible that there is little information related to the topic it falls under. However, the webpage may contain links to other related sites, or even its home page. If we were able to visit the inner HTML links in the webpage, we could gather further information about the topic the webpage talks about. For this purpose, we perform parsing individually on all the links inside the webpage. This way, we also gather more relevant test data than what is already available.

However, is also possible to have irrelevant links inside the webpage. For example, there might be a quick link to a search engine website, or advertisements. Since the web pages discovered by visiting the inner links are not so reliable, we must give them lesser weightage than the original data. We achieve this by increasing the frequency of the words in the original webpage, i.e., we repeat the words in the original webpage for a specific number of times.

Once we obtain the modified dataset after visiting the inner links, we apply LDA to obtain the new results.

4. IMPLEMENTATION

We split the implementation into two parts:

1. Parsing
2. The clustering algorithm

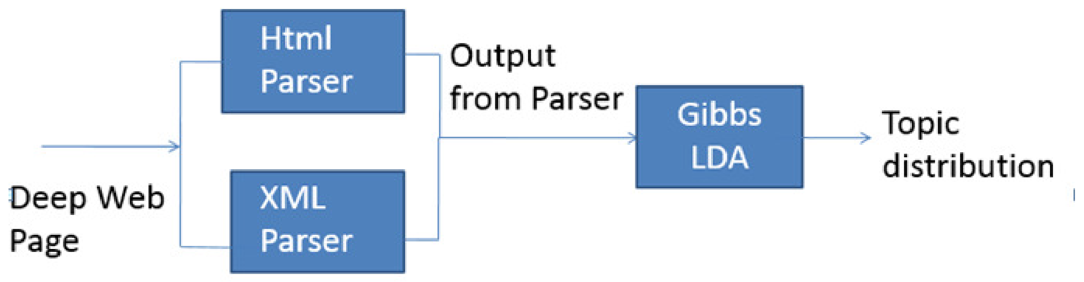

Fig 1: An example Module Diagram (for LDA)

4.1 Parsing

Input: Deep web interface webpage (HTML/XML)
Output: Document with Form Contents and Page Contents as words.

4.1.1 Algorithm

1. From the input HTML/XML, gather all the textual content. Textual content refers to all the *text* that is visible on the webpage.
2. For each <form> element, gather all the values of the attributes for each of the child tags. For example, in the following we wish to extract the *"Search by Departure City"* and *"departure city"*.
```
        <form>
        <attrgroup>
                <attr name="Search by Departure City" ename="departure city">
                …
                …
        </attrgroup>
        </form>
```
3. The words extracted from *Step 1* and *Step 2* constitute the *extracted words*. Write the *extracted words* into the output document.
4. For the proposed approach,
    a. If the webpage is the original webpage interface, repeat the extracted word for a fixed number of times (say N), and write to the output document.
    b. If the webpage is among the ones discovered by visiting the inner links, write them to the document as it is.
5. Remove all the stop-words. Stop-words constitute certain regularly used words which do not convey any meaning, like *as, the, of, a, on,*etc.

4.1.2 Output Format

Google App Engine [16] has been used for creating the web-tool for this project. The parser code written in Python takes in multiple HTML/XML files for input and gives the output after extracting the required words. Further, in order to remove grammatical constructs like ‘a’, ‘the’, ‘on’, we include a file of *stop-words* [17]. All the words in this file are checked against the words obtained from the App Engine (Python code), and removed.

The input format for LDA is as follows:

<Number of documents: N>
<Document 1 - space separated words>
<Document 2 - space separated words>

…
…
<Document N - space separated words>
This output file from the python parser code produces the output in the above mentioned format. This output file is then given as input to the LDA code.

4.2 Latent Semantic Analysis (LSA)

In this method, text vectors out of documents are created and we try to find the cosine similarity between all cluster centers and a new document. If the cosine similarity is found to be greater than a particular threshold "**k"**, which a user could set, then the new document joins the corresponding cluster and the new centroid for the cluster is computed. The cluster centroids are nothing but the topics.

We have implemented LSA in python and then analyzed the results. This technique does not set a good accuracy for large corpus of data (as will be seen in our experimental results.

4.3 Latent Dirichlet Allocation (LDA)

Input:
1. A single file constituting the set of documents received after the Parsing stage. Multiple web pages parsed and given as input to LDA.
2. Number of Topics to be discovered.
Output: Clusters - A set of topics with most occurring words in each topic.

4.3.1 Algorithm

As described earlier, we feed the Gibbs sampled data to our LDA model to extract topics from the given corpus.

5. EXPERIMENTAL RESULTS

5.1 Performance Measure

To evaluate the performance of our method, we calculated Precision of the outputs generated.In a classification task, the precision for a class is the number of true positives divided by the total number of elements labeled as belonging to the positive.

The definition of related terms is shown below:

1. TN / True Negative: case was negative and predicted negative
2. FP / False Positive: case was negative but predicted positive

[19] Precision = True Positives / (True positives + False Positives)

We tried various combinations of these 8 domains, and created 4 small datasets and 2 large datasets. We compared the approach used in [8] and the improved approach over the same dataset, and found that the improved performs well in most cases.

5.2 LSA results

The results of LSA were satisfactory when small amount of data was fed. Below is the output of LSA, which takes as input, four HTML files present in the dataset (TEL-8).
Input: movie.html, airfare.html, railway.html, jobs.html

Output:

Cluster-1	Cluster-2	Cluster-3
movie.html	airfare.html	jobs.html
	railway.html	

Table 1: LSA-Small Dataset-1 Output

However, when a huge dataset was given, the algorithm failed, due do its inability to maintain and store the tern-document matrix. As the size of the dataset increases, the dimension of the term-document matrix also increases. Beyond a threshold, it becomes a costly operation to perform operations of big matrices, if it is even possible to store such matrices.

5.3 Comparison of LDA and Its Improved Version

We performed a detailed analysis on the two methods. Though this analysis (image below) clearly shows that the proposed method performs better, this does not give the complete picture. For a better overall picture of the total topic distribution in each of the datasets, we use overall precision for each dataset.

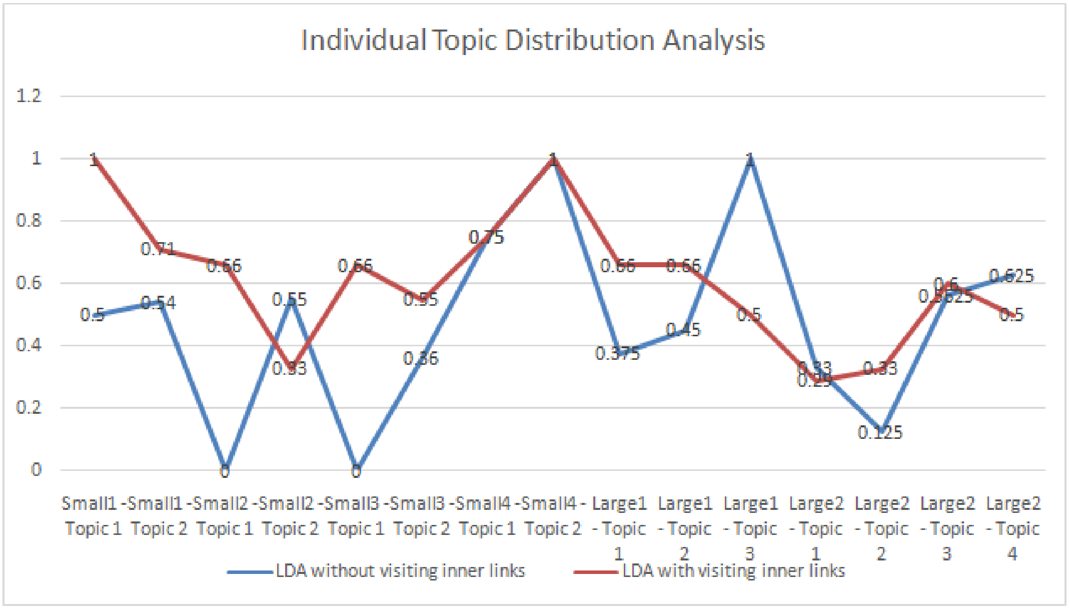

Fig 2: Individual Topic Distribution Analysis. Small refers to small dataset, and Large refers to large dataset. Several topics are discovered within for each dataset.

We use the same formula for precision used above, and average it for the over the number of topics. Therefore, we get an averaged precision value for each dataset.

Averaged Precision = Sum of all precision values / Number of topics discovered

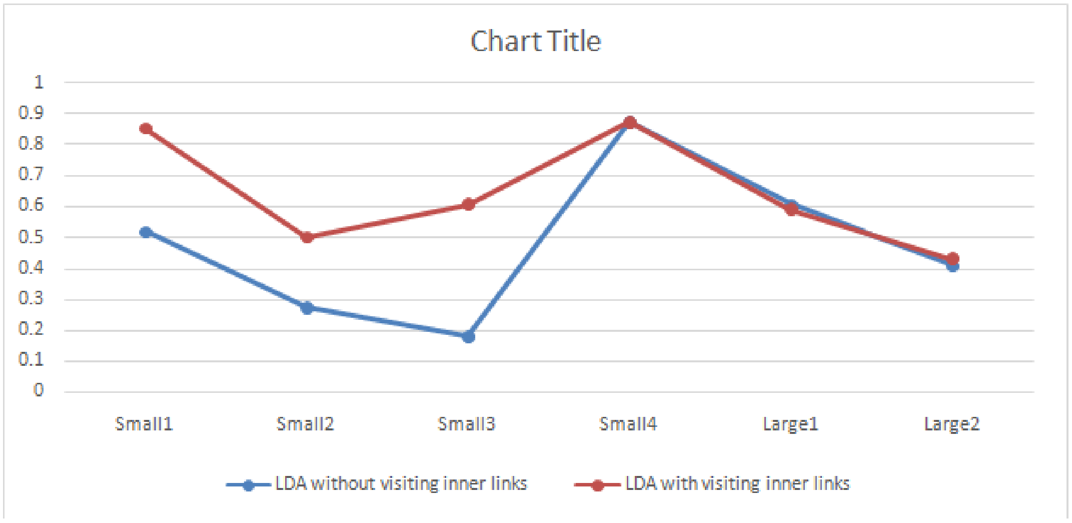

Fig 3: Performance evaluation for datasets

The above chart further strengthens the proof that the new method of visiting inner HTML links works better.

6. Summary and Conclusion

The requirement for an efficient and scalable clustering algorithm for deep web pages is fulfilled by our approach. The results show that sampling of data is useful for clustering heterogeneous deep Web sources. Although LSA performs well with small datasets, it proves to be non-efficient with large datasets. Our proposed Gibbs-LDA based technique is more efficient than the existing techniques.

It gives more accurate results when links in the web pages are parsed and their web page content are also taken into account. As LDA produce soft clusters [20] it assigns probability to each document for each cluster. Hence, our tool is suitable for the scenario where the sources are sparsely distributed over the web.

In order to increase the weightage for the content in our base webpage over the one that we parse in our web page, we have increased the frequency of all the texts that appear in base webpage, which increases the space complexity. New innovative methods could be proposed in future which could possibly reduce space.

Acknowledgements

We would like to thank our Project guide, Dr. E. Sivasankar and Ms.Selvi Chandran, Research Scholar, CSE, NIT-Trichy. Their input, guidance and encouragement made this project possible and a success.

REFERENCES

[1] The Deep Web: Surfacing hidden value. Accessible at http://brightplanet.com, (2000).

[2] Madhavan, J., Cohen, S., Dong, X. L., Halevy, A. Y., Jeffery S. R., Ko, D. and Yu, C (2007),"Web scale data integration", Conference on Innovative Data System Research (CIDR), 342–350.

[3] Tor: The Second-Generation Onion Router. Accessible at www.onion-router.net/Publications/tor-design.pdf

[4] Vincenzo Ciancaglini, Marco Balduzzi, Max Goncharov, and Robert McArdle, "Deepweb and Cybercrime: It's Not All About TOR", Trend Micro

[5] David M. Blei, Probabilistic topic models(2012), communications of the acm, Vol.55, No.4

[6] Estivill-Castro, Vladimir (2002), "Why so many clustering algorithms: a position paper", ACM SIGKDD Explorations Newsletter 4.1

[7] Blei, D., M., Ng, Y., A. and Jordan, M., I. (2003)," Latent Dirichlet Allocation", Journal of Machine Learning Research.

[8] Umara Noor, Ali Daud, Ayesha Manzoor (2013)," Latent Dirichlet Allocation based Semantic Clustering of Heterogeneous Deep Web Sources", In: Intelligent Networking and Collaborative Systems (INCoS), 132- 138.

[9] Unsupervised Learning. Accessible at http://mlg.eng.cam.ac.uk/zoubin/papers/ul.pdf, (2004).

[10] Thanaa M. Ghanem and Walid G. Aref. (2004), "Databases deepen the web". IEEE Computer, 37(1), 116–117.

[11] Dennis Fetterly, Mark Manasse, Marc Najork, and Janet Wiener (2004) "A large-scale study of the evolution of web pages." In Proceedings of the 12th International World Wide Web Conference, 669–678.

[12] Steve Lawrence and C. Lee Giles (1999), "Accessibility of information on the web." Nature, 400(6740),107–109

[13] A Theoretical and Practical Implementation Tutorial on Topic Modeling and Gibbs Sampling. Accessible at http://u.cs.biu.ac.il/~89-680/darling-lda.pdf, (2011)

[14] The ElementTree XML API. Accessible at https://docs.python.org/2/library/xml.etree.elementtree.html

[15] The Bayesian Approach. Accessible at https://www.cs.cmu.edu/~scohen/psnlp-lecture6.pdf

[16] Google App Engine: Platform as a service. Accessible at https://cloud.google.com/appengine/docs

[17] Stop Words. Accessible athttp://www.webconfs.com/stop-words.php

[18] The UIUC Web integration repository. Accessible at http://metaquerier.cs.uiuc.edu/repository

[19] Precision and recall .Accessible at en.wikipedia.org/wiki/Precision_and_recall.

[20] Clustering: Overview and K-means algorithm. Accessible at http://www.cs.princeton.edu/courses/archive/spr11/cos435/Notes/clustering_intro_kmeans_topost.pdf