Bare Demo of IEEEtran.cls for Conferences

Luojie Xiang
Department of Computer Science
Purdue University
West Lafayette, Indiana, USA
Email: xiang7@purdue.edu

Junchao Yan
Department of Computer and Information Technology
Purdue University
West Lafayette, Indiana, USA
Email: yan114@purdue.edu

***Abstract*—The abstract goes here.**

I. Introduction

With the exponential growth of data, it would be efficient for people to understand an area by extracting the topics from millions of documents. Latent Dirichlet Allocation (LDA) is a statistical model that discovers underlying topics from a collection of documents [1]. LDA assumes that the documents are generated from multiple topics, of which a topic is an distribution over a fixed size of words. For each document, it contains the topics with different proportions. Therefore, the words in the document are actually generated from several distributions of the topics. For example, a document might include topics of hadoop and machine learning, therefore it is not reasonable to treat the document as a single topic [2]. However, the large scale data might limit the use of LDA due to the expensive computations. Meanwhile, Hadoop is an open source software for processing large scale data on computer clusters. It provides a programming paradigm called MapReduce that allows researchers easily write applications to run on the clusters. In this project, we extracted the topics from a corpus collected from Stack Overflow using LDA on a hadoop cluster.

II. Latent Dirichlet Allocation

Denote K as the number of topics, V as the size of the vocabulary, $\overrightarrow{\alpha}$ as a positive vector, and η as a scalar. Therefore, for each topic, its distribution over the vocabulary is

$$\overrightarrow{\beta}_k \sim Dir_V(\eta) \tag{1}$$

For each document, its distribution is a mixture of topics, which can be given as

$$\overrightarrow{\theta}_d \sim Dir(\overrightarrow{\alpha}) \tag{2}$$

In addition, for each word,

$$Z_{d,n} \sim Mult(\overrightarrow{\theta}_d), Z_{d,n} \in \{1, \ldots, K\} \tag{3}$$

$$W_{d,n} \sim Mult(\overrightarrow{\beta}_{z_{d,n}}), W_{d,n} \in \{1, \ldots, V\} \tag{4}$$

LDA is a generative model, which provides a joint distribution over observations and hidden variables. Given a collection of documents (D), the posterior distribution of the hidden variables is

$$p(\overrightarrow{\theta}_{1:D}, z_{1:D,1:N}, \overrightarrow{\beta}_{1:K} | w_{1:D,1:N}, \alpha, \eta) = \frac{p(\overrightarrow{\theta}_{1:D}, \overrightarrow{z}_{1:D}, \overrightarrow{\beta}_{1:K} | \overrightarrow{w}_{1:D}, \alpha, \eta)}{\int_{\overrightarrow{\beta}_{1:K}} \int_{\overrightarrow{\theta}_{1:D}} \sum_z p(\overrightarrow{\theta}_{1:D}, \overrightarrow{z}_{1:D}, \overrightarrow{\beta}_{1:K} | \overrightarrow{w}_{1:D}, \alpha, \eta)} \tag{5}$$

Given the posterior distribution, the probability of a word based on topics $\hat{\beta}_{k,v}$, the proportion of topics in a document $\hat{\theta}_{d,k}$, and the topic assignment of a word $\hat{z}_{d,n,k}$ can be calculated as below.

$$\begin{aligned} \hat{\beta}_{k,v} &= E[\beta_{k,v} | w_{1:D,1:N}] \\ \hat{\theta}_{d,k} &= E[\theta_{d,k} | w_{1:D,1:N}] \\ \hat{z}_{d,n,k} &= E[Z_{d,n,k} | w_{1:D,1:N}] \end{aligned} \tag{6}$$

However, the distribution can not be solved in a polynomial time because of the integrals.

III. MapReduce for Latent Dirichlet Allocation

IV. Experiment

A. Dataset

The dataset of this project is obtained from Kaggle (www.kaggle.com), which is a platform for data analysis and prediction competitions. The data that we use are posted by Facebook for a keyword extraction competition. The dataset consists the files both for training and testing, of which the training file contains four columns id, title, body, and tags. In this project, only the title and body are used to extract the topics.

Example:

> **id**: 1
> **title**: How to check if an uploaded file is an image without mime type?
> **content**:
> I'd like to check if an uploaded file is an image file (e.g png, jpg, jpeg, gif, bmp) or another file. The problem is that I'm using Uploadify to upload the files, which changes the mime type and gives a 'text/octal' or something as the mime type, no matter which file type you upload.
> Is there a way to check if the uploaded file is an image apart from checking the file extension using PHP?
> **tags**: php image-processing file-upload upload mime-types

B. Hadoop Setup

The hadoop cluster contains six nodes including one masternode and five slavenodes. To set up the hadoop cluster, we first configured the hosts file as shown below.

```
192.168.65.70 masternode
192.168.65.71 slavenode1
192.168.65.72 slavenode2
192.168.65.75 slavenode3
192.168.65.76 slavenode4
192.168.65.77 slavenode5
```

To configure the hadoop accordingly, we updated the `conf/masters` and `conf/slaves` files on the master node as shown below.

`conf/masters` on the master node:

```
masternode
```

`conf/slaves` one the master node:

```
slavenode1
slavenode2
slavenode3
slavenode4
slavenode5
```

In addition, configuration files `conf/core-site.xml`, `conf/mapred-site.xml`, and `conf/hdfs-site.xml` were modified on all the nodes as shown below.

`conf/core-site.xml` on all the nodes:

```
<configuration>
<property>
<name>fs.default.name</name>
<value>hdfs://masternode:9000</value>
<description>Enter your NameNode hostname
</description>
</property>
<property>
<name>fs.checkpoint.dir</name>
<value>/home/student/DAT500/fs/hdfs/snn
</value>
<description>A comma separated list of paths.
 Use the list of directories</description>
</property>
<property>
<name>hadoop.tmp.dir</name>
<value>/home/student/DAT500/fs/tmp</value>
<description>Comma separated list of paths
</description>
</property>
</configuration>
```

`conf/mapred-site.xml` on all the nodes:

```
<configuration>
<property>
<name>mapred.job.tracker</name>
<value>masternode:9001</value>
<description>Enter your JobTracker hostname
</description>
</property>
<property>
<name>mapred.local.dir</name>
<value>/home/student/DAT500/fs/tmp/mapred/
local</value>
<description>Comma separated list of paths
</description>
</property>
</configuration>
```

`conf/hdfs-site.xml` on all the nodes:

```
<configuration>
<property>
<name>dfs.name.dir</name>
<value>/home/student/DAT500/fs/hdfs/nn
</value>
<description>Comma separated list of paths
</description>
</property>
<property>
<name>dfs.data.dir</name>
<value>/home/student/DAT500/fs/hdfs/dn
</value>
<description>Comma separated list of paths
</description>
</property>
<property>
<name>dfs.replication</name>
<value>2</value>
</property>
</configuration>
```

C. Data Preprocessing

Due to the nature of the stack overflow platform, the posts come in various forms. Most posts has weird characters, huge amount of numbers and texts that's not natural language, representing mathematical formulations, program codes and also program or compiling outputs. For example, a lot of questions are asked about a certain compilation error or run time error, which very commonly contains a very long sequence of error report such as stack trace. These text do includes huge amount of text however, they're machine generated output and can be very confusing to the LDA training process. Therefore, we perform a certain steps of preprocessing before we actually run LDA. This not only prevent the unneccesary alements of the posts from confusing the training process but also reduce the data size tremendously.

- **Retrieve related fields.** The original data comes in a csv file with four fields for each record: id, title, body, tag. Since we want to find out the popular topics among the posts, we want to use the text-rich segments (title and body). Tag is essentially the topic in a sense (tags are mostly about what technology the post is asking like a certain programming language). This can be used as the gold standard later to evaluate the topics we found. Thus, we eliminate them for the training process.
- **Remove contents in ¡code¿¡/code¿.** The ¡code¿¡/code¿ tag contains a lot of mathematical formulations, machine generated text etc. Since they're not helpful in finding topics, we eliminate them.

- **Remove tags.** Tags like ¡p¿¡/p¿ have a large frequency whereas they have no contribution to any topics. So they're all removed.
- **Remove punctuations.** Removing punctuations helps greatly with reducing the feature space. If punctuations are not removed, "happy", "happy." and "happy," would be regarded as different words whereas they should be the same.
- **Lowercase the text.** Lowercase the entire text also helps with reducing feature space since otherwise "happy" and "hapPy" would be regarded as different.
- **Remove newlines and excessive white spaces.**. Since the preprocessed text will be fed into mahout later and text is segmented into words by white space in mahout. Therefore, this would guarantee mahout segments words correctly.

D. Latent Dirichlet Allocation

E. Results

V. Conclusion

The conclusion goes here.

Acknowledgment

The authors would like to thank...

References

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

[2] Ashok Srivastava and Mehran Sahami. *Text mining: Classification, clustering, and applications*. CRC Press, 2010.