Stack Overflow Question Analysis using Topic Modeling

Submitted by
Nitish Ganesan (nkg8933@rit.edu)
Rochester Institute of Technology
1) Abstract:
Question Answering sites are getting popular day by day. Questions which does not have a answer in search engines like Google, Bing are posted in these websites. Rather than searching for books, a lot of effort and time can be saved through these websites. Some of these websites have trained professionals to answer the questions and others are open-ended. Any one can answer the questions in the open-ended websites and the answers are rated by other users showing their approval for the posted answer.

Stack Overflow is one such question-answering website specially designed for programmers. Users can post questions on any programming language. Before posting the question, the users can check if a similar question is posted in a search bar provided by the website. These questions have a set of tags which are basically the topics of the question. These tags have to be manually chosen by the user who is posting the question. An ideal question will have a minimum of five tags. These tags help the users who are searching for a particular question or a topic. Certain questions are complicated and they do not have a predefined set of tags. So, the users posting these questions are confused and this introduces errors while defining the tags.

Figure 1. Sample questions in the Stack Overflow website
Topic modeling is a machine learning algorithm specifically used for text mining. Topic modeling can be used to predict the topics for a corpus have a collection of documents. In our context, the collection of documents refer to the set of questions in the Stack Overflow website. Using Topic modeling, the tags in the questions of the Stack Overflow website can be automatically predicted which reduces human efforts and errors.

2) Introduction:
The scope of this project is to predict the tags in the Stack Overflow questions using Topic Modeling and to perform trend analysis on those predicted topics. The dataset has 140272 records and each record has the header of the question and the question itself. Before the model can be built, the questions have to be pre-processed. Pre-processing steps include tokenization, stemming and removing stop words. Topic Modeling algorithms require a bag-of-words model as their input. Hence, the pre-processed data is converted to a bag-of-words model using python.

Three Topic Modeling algorithms will be chosen to predict the set of topics in the corpus. In certain topic modeling algorithms, the number of topics to be predicted should be given as an input. A consistent algorithm which used this method will be used as one of the algorithm. Certain algorithms optimally choose the number of topics by themselves. One algorithm will be chosen in this category. Another category of algorithm will be chosen such that they are good in finding incoherent topics. Since Topic Modeling is an unsupervised algorithm, the algorithms cannot be compared. They do not have a common metric for comparison. So the predicted topics will be compared with the topics already present in the dataset for checking the accuracy and comparing the models.

Trend analysis of the predicted topics will be performed as a final step. Trend analysis has numerous concepts. In this project, trend analysis will focus on finding the topics which have gained popularity over time, topics having more number of posts and analysis based on the open status of questions.

3) Work done by other researchers:
In paper [1], the authors have used LDA which is a Topic Modeling algorithm to detect the topics in the Stack Overflow website. Their model was built using MALLET[2] which a basically a tool built using Java for building the LDA model. For pre-processing the questions, they used a natural language processing tool Apache OpenNLP.

Their train their data using LDA and performed over 2000 iterations. For fine granularity, the authors have chosen 150 as the optimal number of topics. After predicting the topics, the authors try to establish a relationship between the questions by type and concepts.
Their models include the Question concepts, code and text model and Question types model.

In paper [3], the authors use LDA to predict the topics for the questions in the website. Their dataset includes 3474987 posts ranging from the year 2008 to 2010. The average number of topics for each question ranges from 2 to 3. The authors use the MALLET[2] version to build the model. To train the model they perform over 500 iterations.

4) Design :

![Design of Stack Overflow Question Analysis](image)

Figure 2. Design of Stack Overflow Question Analysis

Figure 2 shows the design of the project. The first step is the data extraction part. The data is collected from the data dump section of the Stack Overflow website. Of the different data dumps, the one with 140k records have been chosen. The next step is data pre-processing which includes Tokenization, removing stop words and stemming. Additional filters are also added in this section to improve the accuracy of the prediction. The next step will be converting the pre-processed data to a bag-of-words model as it is a requirement by the topic model. The next step will be to study and select the algorithms that will benefit the application. The model will be trained and the predicted tags will be compared with the tags present in the data dump to check for the accuracy of the
algorithms. The result of the model with the best accuracy will be loaded into tableau and trend analysis will be performed.

5) Data Extraction :
The first step of the analysis is the data extraction. The dataset is collected from the data dump site of the Stack Overflow website. There are different data dumps in the website and each data dump has specific attributes and features. This specific data dump contains over 140272 records. Each record has different attributes such as the user ID, question, question header, status of the question, set of tags. The set of tags will be used to verify the accuracy in the Comparison of Accuracy part of the project. The dataset is in the csv format. The dataset is pre-processed and loaded into the model using python.

6) Data Pre-processing :
Data pre-processing consists of four different sections such as removing stop words, stemming, tokenization and additional filters. This is an important step in the project as the accuracy of the model will be based on this step.

Removing Stop words:
This is an important pre-processing step in text mining. Stop words usually denote the commonly used words in a conversation[4]. In Machine Learning, it is important for the model to recognize the important words so that the prediction can be done based on those important words. For example, in the question “What is Machine learning?”, the words “what” and “is” are the stop words.

Stop words should be determined based on the needs of the application. Strict filtering or no filtering will lead to negative results based on the application. For example, in opinion mining the words ‘not’, ‘like’ are important. Removing those words will produce a lot of errors in the application. There are different types of stop words. In this project conjunctions, propositions and the words denoting the noun are considered as stop words. In Topic Modeling, the order of the words does not matter. So strict filtering of stop words is followed in this project. The stop word list used for this application is built by combining different lists and removing the duplicates so that the speed is improved while pre-processing. The language python is used for removing the stop words. Figure 3 shows the sample list of stop words used for this project. The ‘u’ denotes the encoding used by the natural language processor. The list used contains words like ‘a’, ’above’ etc. The stop words are loaded from a csv file are processing using python.
Figure 4 denotes a tokenized version of a sample question used for pre-processing. The question is converted to a set of tokens. The stop words are yet to be removed in this version. Also, the tokens has a lot of noise which is yet to be filtered. They will explained in the additional filters section of data pre-processing.

Fig 4. Tokenized words (Before removing stop words)

Fig 5. Tokenized words (After removing stop words)
Figure 5 shows the list of tokens after the stop words have been removed. Comparing figure 4 and 5, words like ‘i’, ‘with’, ‘and’, ‘but’, ‘my’ have been removed as they are included in the list of stop words.

**Stemming:**

Stemming refers to the converting of a word to its root form. Stemming is performed in a corpus to avoid multiple words having the same meaning to be considered as different words. This process is very important in text mining and helps in improving the accuracy of the model.

Examples:
consign, consigning, consignment, consigned => consign
van, vans, van's, vans' => van

Porter Stemming algorithm:

There are many algorithms developed over the recent years for the stemming process in text mining. One such algorithm is the porter stemmer algorithm. This algorithm is stable and popular compared to other algorithms existing in the internet. Porter Stemmer algorithm considers words as a sequence of consonants and vowels[5]. Any word can be considered in the form:

[C][CV]^[V]

The first C and last V are optional in the listed form. The measure L represents the length of the portion of the word or the entire word. For each word, the algorithm works at three points where L>0, L>1 and L=1.

Fig 6. Tokenized words (Before Stemming)
Figure 6 shows the tokenized words after the stop words have been removed but before stemming has been performed. Words like ‘thanks’ are still in their original form in this figure. In figure 7, stemming has been performed. Words like ‘thanks’ have been converted to their base form ‘thank’.

Tokenization:

Tokenization refers to the splitting of words in a sentence to individual words[6]. These individual words are also referred to as n-grams. This process of splitting words differs for different languages as each language has a different sentence structuring. For this project, splitting of the words is based on the spaces between the words.

A number of issues raised when loading the question from the csv file. In figure 8, you can see a lot of noise associated with the question. This noise was found in every question when loaded form the corpus. These will addressed in the section ‘Additional Filter’.
Figure 9 shows a tokenized version of figure 8. Tokenization was performed using python.

**Bag-of-words:**

The term Bag of words refers to the meaning ‘bag of its words’ i,e the count of unique words in every sentence[7]. This is an important step in natural language processing, especially in topic modeling. This model does not worry about the order in which the words are arranged in a sentence. Since topic modeling gives importance to the unique words and not the sentence sequence, this model has gained popularity. Every question in the corpus has been converted into this model so that they can be given as an input to the topic modeling algorithms.

**Additional filters:**

Additional filters have been used before the bag-of-words model step to improve the accuracy of the models. Numbers excluding the version number of the languages have been removed. Also, individual letters and symbols which do not have a separate meaning have been removed.

Noise was generated when the questions were loaded form the corpus using python. Newlines and additional spaces were creating garbage values such as ‘\r’ and other values such as ‘\’. Filters were created to remove this noise.

After building the model, noise was found in the topics generated, for example ‘tt’,’tttt’ and other words which do have any meaning were also suggested as topics. Individual filters were generated to remove these words so that the accuracy can be improved. The filters generated will check every word and remove them if a match is found.
To reduce the time taken by the models, all the processing steps were created as a separate python script and the output was loaded into a csv file.

Fig 9. Csv file after data pre-processing question (before tokenization and bag-of-words)

7) Implementation of Topic Modeling:

Three algorithms were chosen for topic modeling based on the nature of the dataset: LDA, HDP and NMF. The data pre-processing steps and the initial steps in the algorithms were implemented using python. For the implementation of the core part of models, gensim[8] a python library was used. Gensim is very popular python version in comparison to MALLET[2] used by the previous researchers which is in java.

**LDA(Latent Dirichlet allocation):**

LDA is a generative probabilistic topic model for generating topics in a corpus contains a large set of documents[9]. This model works well for dataset having a huge volume of data as it works on probability. The probability and accuracy will more as the volume increases. The topics generated by the model are a set of topics. Each topic has a cluster and each cluster has a number of words optimally ranging from 5 to 10 words for this project. The model is statistical and initially achieves a random distribution over the topics. This process is generative. In the first step, a random topic is chosen form the number of topics achieved statistically. In the second step, a word is randomly chosen from the distribution of the number of topics.

The number of topics has to be mentioned before the model is run. Also, the number of passes have to mentioned in the model. The number of passes are the number of iterations the model performs to achieve the best accuracy. The model was run several times with different number of passes ranging from 5 to 30 and the number of topics ranging from 50 to 150.
Fig 10. Perplexity vs Number of passes

Fig 11. Perplexity vs Number of topics
For LDA, perplexity is the log-likelihood of the function[10]. To measure the performance of the LDA model, it is divided into training and test data. Perplexity measures the log-likelihood of questions present in the test dataset. This is a decreasing value, so better models have lower values. In figure 10, as we increase the number of passes to increase the accuracy, perplexity reduces steadily. We should also consider the time taken, as the time increases with the number of passes. Since the value becomes steady with the number of passes at 15, it is considered as the optimal value for the model. Figure 11 shows the perplexity according to the number of topics. It is seen that the perplexity value reaches its peak with the number of topics at 100. The value drops after 100. So, the LDA model is built with the number of topics at 100 and the number of passes at 15.

Fig 12. Sample topic clusters generated with the LDA model

LDA generated 100 topics with clusters having an average of 7 words. Figure 12 shows 5 sample clusters taken from the total number of cluster. Each chart shows the top words in each cluster with their corresponding probability in the cluster.
**HDP (Hierarchical Dirichlet Process):**

HDP is a topic model which does not have any parameters and it follows the bayesian approach[11]. The advantage with HDP model is that, we need not specify the number of topics to be generated from the corpus. HDP automatically selects the optimal number of topics from the corpus. This model allows the topics to share the components. This model also uses a probabilistic metric in its approach. It has two probabilistic metrics, one for each document in the corpus and one global probability for the entire corpus.

In this application HDP chose 150 as the optimal number of topics for the entire corpus consisting of 140272 records. HDP also takes the number of passes as the input which specifies the number of iterations. The optimal number of passes found for this model was 15.

**NMF (Non-Negative Matrix Factorization):**

NMF is a topic modeling algorithm widely used in computer vision and clustering of documents[12]. NMF works on matrix which is primarily factored into two matrices. This is done to reduce the dimensionality of the original matrix. Also, the resultant matrix and the other two matrices are all non-negative which helps in the verification of the algorithm. The two matrices are W and H. W represents the cluster of words and each word has a rank, H represents the original questions in the corpus along with the associated rank.

\[ v_i = Wh_i \]

Fig 13. NMF factorization [12]

NMF takes the maximum number of iterations and the number of components as the inputs which correspond to the number of passes and the number of topics in the other two algorithms LDA and HDP. The main goal of this algorithm is to predict the topics in a corpus by approximating the matrices W and H in each iteration according to the function. For this project, the number of iterations were set to 8 and the number of components were set to 100 in NMF.
8) Results:

Figure 14 shows the accuracy achieved and time taken by each algorithm. Accuracy was found by comparing the predicted topics with user given topics present in the corpus. Each word in the topic cluster was compared with each topic in the corpus. A python script was generated for this process. The best accuracy was achieved by LDA. LDA was very stable and consistent when altering the input parameters. Initially the accuracy achieved by the algorithms was ten percent below the shown accuracy. New filters were added to the data pre-processing part to remove the noise generated in topic generation. Even though LDA had the highest accuracy, it took more time compared to the other two algorithms. To reduce the overall time, the pre-processing and finding the accuracy parts were treated as separate sections and processed separately. The time shown in the figure is the time taken to build the model.

HDP had the second best accuracy. This model considered 150 as the optimal number of topics to be generated for the corpus in this project. HDP saved a lot time as we need not experiment with different number of topics to find the optimal number. The time taken was very less compared to LDA but not the best. NMF took the least amount of time to build the model and generate the topics but the accuracy was less compared to the other two algorithms. NMF is good for corpus containing incoherent topics, so it did not achieve the best result with this corpus. Finally, LDA is considered as the best model for this project and the optimal inputs were given to LDA to generate the topics for trend analysis.

Trend Analysis:

Trend analysis was performed with the topics generated by the LDA model. The Tableau tool was used to perform trend analysis of the topic clusters.
Figure 15 shows the status of the questions in the corpus. The highest value corresponds to the number of topics left open. Users of Stack Overflow prefer the topics to be open so that updates can be added from time to time. Another problem with Stack Overflow is the framing of questions. Improper framing of questions leads to topic migration and unsatisfactory results.

Fig 16. Topic Clusters
Figure 17 shows the number of questions asked each year in Stack Overflow. The corpus taken from the data dump has data up to the year 2012. According to the forecast indicator in the tableau tool, around 16k questions will be asked in the year 2013. The number of questions doubles every year.

Figure 18. Trend of topics over time
Figure 18 represents the popularity gained by the topics over the time. Php, java, android and javascript has gained a steady popularity. C++ and c# were equally growing but has lost popularity in the recent years.

9) Conclusion:

In topic modeling, LDA was consistent and was very stable throughout the project. LDA achieved the highest accuracy of 79.94 % for a corpus containing 140272 records. Other models are gaining popularity but they are not as consistent as LDA. Question-answering websites are gaining more and more popularity. According to the trend analysis prediction, 16k questions will be posted in the stack overflow website for the year 2013.

10) Future work:

Further analysis of the questions can be done to find the relationship between questions. With this relationship, existing questions can be suggested to the users when they are posting new questions.

The time taken by the algorithms limited the number of possible iterations in each algorithm. To solve this problem, a distributed architecture like Hadoop can used for parallel processing. This will reduce the time taken and each algorithm can be built with better accuracy by increasing the number of iterations.

References :


