Grenze International Journal of Engineering and Technology, June Issue

Tag Prediction of Stack Overflow Questions using NLP and ML

Mr.P.Yogendra Prasad[1], Mylavarapu Priyanka[2], Shaik Amreen[3], N.Desai Sreenivas Reddy[4] and Muppala Bhanu Teja[5]
[1]Professor, Dept. of CSSE, Mohan Babu University (Erstwhile Sree Vidyanikethan Engineering College), Tirupati.India
yogendraprasad.p@vidyanikethan.edu
[2-5]UG Scholar, Department of Computer Science and Systems Engineering, Sree Vidyanikethan Engineering College, Tirupati, India
Email: priyankasrinivas2126@gmail.com, amreenshaik.2602@gmail.com, sreenivasreddy028@gmail.com, bhanuutejamuppala@gmail.com

***Abstract*—Technology is growing in great leaps and bounds and the number of innovations is increasing every day with major contributor being the advancement in the software development environment. Programming environment includes plethora of tools and support platforms that gives a complete assistance to the programming designers. These platforms can be question- answer based or coding based or tutorial based. Stack Overflow is a very popular online question- answer platform in the current generation and is growing to be the greatest question-answer gathering for programmers and software developers. The amount of information in this forum is so huge that one of the main functionalities of this platform would be tagging i.e., identifying the label or class to which the piece of information belongs to. Tagging helps in easy retrieval and organization of data. We propose a multi label classification system that labels all the questions on the forums with their appropriate tags autonomously. This auto-labelling framework makes it very easy for the users to find answers for the inquiries posted, given significant tags for the same and also helps the site in organizing their questions in a better way.**

***Index Terms*— Tagging, Stack Overflow, Technology, Multi- label, Q&A, forum, autonomous, software, tools, programming, advancements, innovations, platforms.**

I. INTRODUCTION

With the increase in online education, the number of platforms supporting question answer forums are increasing. Out of platforms like Stack Overflow, Coursera, Quora, Stack Overflow is a very popular online question- answer platform and there is a huge amount of information in this forum. So, to make the system efficient, it is very necessary to detect the class to which a particular piece of information belongs to. This detection is called tagging. It helps in easy retrieval and organization of data. The view is to develop a system for autonomously detecting the tags.

The amount of information in online forums like Stack Overflow is rapidly increasing yet it is necessary to increase the efficiency of the forum without allowing the size of the data to be a hindrance to the system. Currently, websites like Quora are expecting users to manually enter the tags related to their queries. But manual

Grenze ID: 01.GIJET.10.2.597

tagging is a difficulty for the end user causing poor user experience. So, an autonomous question tagging system is necessary.

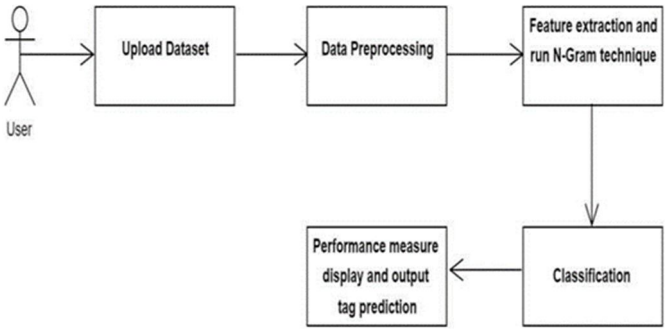

Fig1: Shows the System Architecture Diagram

This system first takes the dataset, then this data undergoes preprocessing involving removal of html tags, text case conversion, removal of punctuation marks, elimination of stop words and stemming of words etc. Then for feature extraction, we use CountVectorizer after which the model is built using the Multi Class Classifier – OneVsRestClassifier for tag prediction. We also analyze the performance metrics of the classification.

II. Literature Survey

Ref. [1] Mihail Eric, Ana Klimovic, Victor Zhong worked on a multi- label classification system that automatically tags users' questions to enhance user experience.
Ref. [2] Vijaya Lakshmi, Vishwa Prabha, Priyadarshini Udayakumar, N Swetha gave the idea of an auto-labelling framework that proposes labels to clients contingent upon the substance of the inquiry.
Ref. [3] Rishabh Bassi, Rohan Piplani, Ravijit Singh Ramana, Jasmeet Singh, Harpreet Singh, Prashant Singh Rana did the research work on a method for question and answer platforms that automatically allocates tags for a given query. And a Document-Term matrix based classification method to autonomously allocate tags to queries posted on any forum.
Ref. [4] Trude ed el in his study examines how tagging acts as a bridge between the social and technical aspects of software development. The author analyzes various factors like tag frequency, popularity, and user adoption to reveal this connection.
Ref. [5] Currently, the users are expected to manually enter the tags (upto 5). But, manual tagging causes poor user experience. Moreover, the performance of the system decreases with increase in data size.

III. Proposed Work

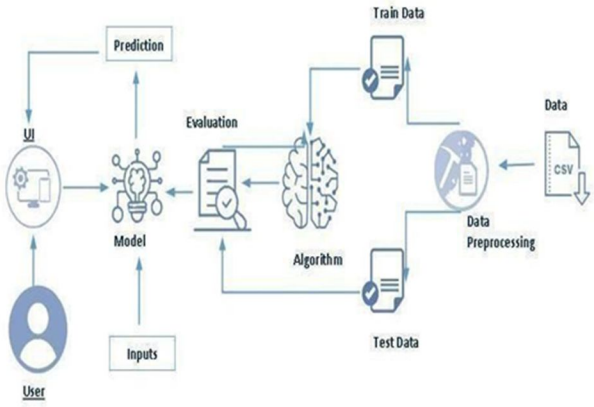

Fig2: Preprocessing Steps

Here we propose a multi label classification system that makes use of Natural Language Processing and good classifiers to label the questions on the forums with their appropriate tags autonomously. This auto- labelling framework makes it very easy for the users to find answers for the queries posted, given significant tags for the

same. This system helps in easy retrieval and proper organization of data. In stage 1, we propose to do the data preprocessing i.e removal of HTML tags, certain code syntaxes, conversion of text to lower case, removal of punctuation marks, elimination of stop words and stemming, use count vectorizer and OneVsRestClassifier for query's titles only and analyze the title based prediction metrics.

A. Preprocessing Method

The few preprocessing steps that have to be performed are as mentioned below:

i. Removal of HTML Tags: Since HTML tags in the data don't provide additional meaning for question comprehension, we utilize Python's Scrapy library to eliminate them.
ii. Text to Lowercase: To improve question understanding, we convert all text to lowercase. Since questions focus on meaning, not capitalization, treating "Apple" and "apple" as equivalent simplifies processing.
iii. Eliminate Punctuation marks: Since punctuation doesn't affect the meaning of questions, we remove it to focus on the important parts.
iv. Tokenization: To analyze text effectively, we break it down into smaller units known as tokens These can be characters, words, sentences, or even paragraphs. Our application specifically uses word tokenization, which means we extract individual words from the text, like picking out puzzle pieces one by one.
v. Elimination of Stop Words: Frequent words like 'a', 'an', 'the', 'and' are called stop words because they carry little meaning for understanding content. While common in language, they don't contribute much to distinguishing different topics. Since our next section uses the most frequent words as features, keeping stop words would bias the analysis towards these common but meaningless terms.
vi. Stemming: To reduce redundancy and focus on core meaning, we use stemming. This collapses words like "trouble" "troubled" and "troubling" into "troubl" grouping similar ideas and making our analysis more efficient.

B. Feature Extraction and N-Gram technique

After cleaning up the questions using techniques like tokenization and stop word removal, the next crucial step is extracting key features. This is like summarizing the questions, focusing on the important bits instead of every detail. For that we use a technique called Stemming, which shrinks words to their basic form. Imagine "walking," "walked," and "walks" becoming "walk." This helps group similar questions and reduces unnecessary complexity. It is important because, by extracting these features, we create a compact and meaningful representation of each question. This makes it easier for the tagging system to understand the questions' core meaning and assign the most relevant tags.

Feature extraction can be done as follows:

i. Identify common tags: The most frequent tags used in questions are identified.
ii. Gather examples: For each tag, 1000 questions with that tag (positive examples) and 800 questions without it (negative examples) are collected. This balances the data to avoid biasing the model.
iii. Extract key words: From each positive example, the 100 most frequent words are extracted. This creates a large set of words associated with each tag.

Overall, this process aims to train a model that can accurately predict a tag for a new question based on its keywords.

i. N-grams in context: N-grams are sequences of n adjacent items, typically words, in a text. They capture word order and context, which helps understand language better than simply analyzing individual words.
ii. Extracting top 100 words: This selects words that frequently co-occur within questions related to the same tag. This captures some level of sequence and context similar to n-grams.
iii. Large set of words: While not explicitly forming n-grams, having many words associated with each tag helps it identify patterns and relationships between co-occurring words.

C. Classification

OneVsRestClassifier: One-vs-all (OvA) tackles multi- class problems by training a separate classifier for each class. Each classifier "fights" for its class against all others. This method is simple, efficient, and interpretable thanks to each class having its own dedicated classifier. While not always the best, OvA is a popular and reliable choice for many multi-class classification tasks.

CountVectorizer: CountVectorizer in scikit-learn takes text documents and turns them into numbers suitable for machine learning models. It counts how often each word (or group of words) appears in each document, creating a matrix that captures the word usage patterns. Think of it as a translator, turning words into numbers that machine learning models can understand! We use CountVectorizer and its handy features like removing common words or focusing on specific word combinations, all to get text data ready for powerful analysis.

D. Performance Measure Display and Output Tag prediction

For performance evaluation we utilize few metrics like, F1 Score: The F1 score is a powerful metric for evaluating the performance of machine learning models in multi-class classification tasks. It combines two key metrics: precision (identifying correct examples) and recall (not missing real examples). Unlike accuracy, which can be misleading in imbalanced datasets, F1 score gives equal weight to both precision and recall, making it a balanced measure of overall performance. A high F1 score indicates a model that is both accurate and able to capture all relevant information, making it a valuable tool for assessing the effectiveness of your multi-class model.

TABLE I: TITLE VALUES

#	Data	Values
1	Precision	79.9822
2	Recall	40.9828
3	F1	54.1958
4	Accuracy	66.104
5	HL	3.1171

TABLE II: BODY VALUES

#	Data	Values
1	Precision	62.3776
2	Recall	10.1673
3	F1	17.4847
4	Accuracy	53.976
5	HL	4.3176

TABLE III: TITLE- BODY VALUE

#	Data	Values
1	Precision	77.6463
2	Recall	43.7682
3	F1	55.9808
4	Accuracy	66.3146
5	HL	3.0927

IV. RESULTS

Evaluation metrics and Accuracy:
After testing machine learning algorithms on the Kaggle data set we got the results as follows:
It is providing efficient tag identification for the queries of the Stack Overflow type.
Quite a good accuracy, precision and hamming loss for carefully selected features using one-vs-rest classifier (Linear SVM), that helps in faster data retrieval for other platforms (Quora, Coursera) that are expecting manual tag entry from users.

	Title	Body	Title-Body
Precision	79.98	62.37	77.64
Accuracy	66.10	53.97	66.31
Hamming Loss	3.11	4.31	3.09

V. Conclusion

The proposed system is an efficient way that helps in autonomously detecting or identifying the tags for the queries posted on Stack Overflow. Using one-vs-rest classifier, implemented using the linear Support Vector Machine, system will have the capability to provide a quite a good accuracy and precision for carefully selected features. The tag prediction based on titles has achieved a precision of 79.98%, accuracy of 66.10%. There are other multiple platforms like Quora, Coursera etc that are requiring users to manually enter the tags for a faster retrieval of data. So, this system would help such other websites also to have their questions and answers classified under various appropriate tags autonomously.

Future Work

As our future enhancement, we would work on autonomous detection of tags of questions based on not just their titles, but also their body. Furthermore, to improve the performance of our system, we need to consider or choose certain filtered features. Also, we can use feature selection techniques like Principal Component Analysis (PCA) to improve our filtering process. The hierarchical nature of tags can be identified in order to give a better identification and classification. We can try out non-linear classifiers like the Gaussian kernel SVM's and neural networks so that data is grouped in a better manner.

References

[1] Saha, A. K., Saha, R. K., and Schneider, K.A. (2013). A discriminativemodel approach for suggesting tags automatically for Stack Overflow questions. 2013 10th Working Conference on Mining Software Repositories (MSR).
[2] #ML #NLP: Autonomous Tagging Of Stack Over Ow Questions [PDF Document] (fdokumen.id)
[3] Stack Overflow Tag Prediction Using Deep Learning Algorithm (ijaresm.com)
[4] 377-153025525644-48.pdf (digitalxplore.org)
[5] Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features.
[6] McCallum, A. K. (1999). Multi-label text classificationwith a mixture model trained by em. In AAAI 99 Workshop on Text Learning.
[7] https://scikitlearn.org/stable/modules/generate d/sklear n.svm.LinearSVC.html
[8] https://scikitlearn.org/stable/modules/generate d/sklear n.feature_extractiontext.CountVectorizer. html