COMPARISON OF TAG PREDICTION TECHNIQUES FOR STACK EXCHANGE POSTS

Abstract

With the advent of Web 2.0, which allows the user to create dynamic web pages and increasing trend of using social network sites like stack overflow, flicker etc., the concept of tagging is evolved. Tagging is the process by which users annotate their items like question, photo etc. with the relevant keywords called tags. Assigned tags should be such that they explain the subject or context of posted item. These tags are proved to be helpful for the users by suggesting them the items, they are interested in. Also, as the data available on the internet is exponentially increasing, it is becoming a difficult task to organize it. Tagging enables the users to organize their items of interest in a better way. So, the problem that is addressed in this paper is how the tagging process can be automated by recommending most appropriate tags to the user when he posts a new item. Tag recommendation process is categorized into Personalized Tagging and Social Tagging. Recommended approaches for both of these types are presented in the paper.

This research compares different techniques presented for Tag Recommendation. Additionally, topic modelling algorithm named Latent Dirichlet Allocation is also implemented for recommending tags, but experimental evaluations on Stack-Exchange posts revealed that it's not suitable for this dataset. However, results of another technique named Fuzzy Nearest Neighbor Search that is proposed earlier are significantly better for medium to large scale data-set and SVM based method lead all other techniques in small scale data-set containing Stack-Exchange posts.

Table of Contents

List of Tables

Results of 3D-Printing Dataset

Results of Data-Science Dataset

Results of Physics Dataset

Results of Apple Dataset

List of Figures

Figures of 3D-Priting Dataset

Figures of Data-Science Dataset

Figures of Physics Dataset

Figures of Apple Dataset

Chapter 1

Introduction

As per increase in the number of launched sites like stack overflow, flicker, delicious etc., information available on the internet is increasing due to which tagging is becoming popular. Tagging is a process of assigning keyword that describes the topic of post or question. This assigned word is called a tag. Tags help to categorize the posts with similar meaning. Apart from this, tagging also connects professionals by providing them the relevant post or question to which they can answer well. [1]

When we have huge sized data such as large number of posts or webpages, we can keep them in an organized way by making use of keywords. These keywords or tags serve as descriptors for a given resource [2]. Existing Question/Answering sites like stack exchange demands the user to provide appropriate tags for the post they are posting. The provided tags can be a set of words or it can be single word. These assigned words represent the semantic meaning of user's post.

As a result of tagging, we have powerful technique for categorizing the content in top-down manner – based on the assigned tags and their frequencies. Tagging provides flexibility to the users by allowing them to choose tags according to their own view point. Appropriate tags are also useful for retrieving the most suitable documents for user.

Although tags can be assigned and are currently being assigned manually by the user on some sites like stack exchange, but it is tedious and time-consuming task for large number of posts. It

will be more convenient to automate the task or at least provide suggestion to the user at the time of assigning tags.

In tagging system, there is no restriction on users for assigning tags. Users can freely annotate their items with whatever tags they want. Although by freely allowing the users to annotate their items makes the tagging system easy to use and user-friendly but tagging system also have to bear the cost of providing this much flexibility and freedom to users. This cost is a result of extensive vocabulary created as result of tags assigned by users. Furthermore, problems like tag ambiguity and tag redundancy can also occur. Tag ambiguity refers to the problem of assigning tags having different meanings and tag redundancy refers to the problem of assigning tags having similar meanings [3]. Use of redundant tags create the obstacle for algorithms that assign tags to the items on the basis of similarities among items. So, redundant tags complicate the process of identifying similarities among items. While, ambiguous tags falsely represent the items with the dissimilar assigned tags [4]. These problems can be overcome by automating the task of tag prediction.

1.1 Research Problem

The trend of using Question Answering sites is gaining popularity. Thousands of users post on daily basis. It is possible that different users post questions having similar semantic meaning but assign different tags to the questions. Now, although the questions are addressing the same problem, when a new user arrives and he/she searches for that question, then the posts will be retrieved on the basis of tags assigned to the already posted questions. The more relevant the assigned tag is, the more appropriate post will be retrieved to meet the requirement of user.

So, the problem is to organize the similar posts or questions in such a way that user will not need to re-post them if they already exist. The new user will not need to wait for the answer as the similar question posted earlier would already have the answer. In order to solve this problem, the need to automate the tagging task on Question/Answering sites arises, so that the users can easily access their required post.

1.2 Research Question

To find the appropriate or most suitable tags for the posts on Question/Answering sites automatically, we will look into different proposed tagging techniques. In short, the thesis objective is to address the following research question:

"**How to automate the tagging task for posts on social sites?"**

1.3 Structure of Thesis

This thesis is organized as follows:

Chapter 2 provides required background of tag prediction field for the intended audience.

Chapter 3 presents the literature review.

Chapter 4 discusses the experimental setup

Chapter 5 covers result section including the evaluation of explored techniques

Chapter 6 discusses the future work and conclusion.

Chapter 2

Extended Background

In this chapter, all the basic concepts concerned with Tagging will be presented.

2.1 Tagging and Folksonomy

When we talk about tagging, we come across the concept of folksonomy [5]. It refers to a system that enables user to publicly assign tags to the content that are available online. Content can be any webpage, video clips, photos, URLs etc. Folksonomy deals with three units. These units include users, tags and resource that needs to be tagged. Creation of tags is done by user. The created tags are then assigned to resources (contents). The assigned tags are helpful for classifying, managing and summarizing the content [6]. It is reported that folksonomy can be categorized in two forms: [7]

- *Narrow Folksonomy:* Tagging is restricted to some specific number of users.
- *Broad Folksonomy:* Tagging is done mutually or tags are shared among group of users forming a community.

2.2 Subtypes of Tagging

Tagging has two different types: Social Tagging and Personalized Tagging

2.2.1. Social Tagging:

Social tagging is also known as *Collaborative Tagging.* As the name indicates, this type of tagging assigns tags to resources by taking into account the tags assigned by other users to the

same item [8]. Social tagging is considered to be the most popular type of tagging. It allows users to assign tags a in free-form to the resources that are available online. Following this kind of tagging, a loosely bound classifier - for online content – based on feedback of large group of users can be built. In social tagging, tags are shared among all users [11].

Social tagging provides several advantages. It allows the users to connect socially with each other. Users can easily manage and access their relevant information. For participating in this type of tagging, it is not required by user to have any sort of expertise. Apart from it, as things are refined day by day so this type of tagging is much responsive to dynamically occurring changes in terminology and modernization in resources.

It is reported that systems following this kind of tag recommendation fail because of two appropriate reasons. First reason is that every user assigns tags according his/her interest for item's information. Second reason is that each item that is to be tagged can have multiple aspects. For example, we have two users who want to assign tags to pictures available online. One of the users having craze for Mac Systems and other user crazy for fruits, retrieved the images according to their interests. First user retrieved image of Mac System and another retrieved image of apple. Most probably both of them will tag their respective retrieved images as "apple" and after a while if both users want to retrieve their relevant images again, both images i.e. image containing Mac System and apple will be shown to them because of having same assigned tags. So, social tagging leads to this kind of ambiguity [10].

Sites like flicker, delicious make the use of social tagging.

2.2.2. Personalized Tagging:

As the name indicates, personalized tagging does not take into account the tags used by all users which is the case in social tagging. Instead in personalized tagging, tags that have already been used by a specific user or tags used by the users who have the same profile e.g. age group, and vocabulary are considered. At abstract level, we can say that tagging history of user is considered in personalized tagging. This tagging history maintains the vocabulary of each user.

Social tagging leads to too much freedom in choice of tags and it does not take into account the interest of specific user. Every user has his own need depending on the areas in which he is interested. Let say we have three types of users i.e. zoologist, rich man, and a youngster. All of them are interested in resource tagged as Jaguar. As per zoologist, Jaguar corresponds to an animal. According to rich man, Jaguar is car and youngster might take it as a movie "The Jaguar". For fulfilling such user-specific needs, personalize tag recommendation came into being [10]. Tag ambiguity problem that was faced in social tagging can be avoided in personalized tagging. Since it deals with the interest of only a specific user instead of dealing with interests of several users, this type of tagging improves the retrieval system for a specific user. Whenever user assigns tag to any resource, the recommender system suggests tags to a user by analyzing the tags used by him or by analyzing tags used by users having same interest.

2.3 Types of Tags

Different types of tags have also been reported in [11]. These are as follows:

Types of Tags	*Definition*
Attribute tags	Tell the features of the resource
Content Based Tags	Recognize the actual content of the resource
Context Based Tags	Tell the context of the resource i.e. indicates what a resource is about
Purpose Tags	Represent the information gaining task of user e.g. learning some specific tool
Subjective Tags	Use for recommending objects by expressing opinion of user
Ownership Tags	Represent the owner of resource
Organizational Tags	Use for organizing the task that is not completed yet

TABLE 1:*TYPES OF TAGS*

The above-mentioned tags are all the types of tags defined at abstract level. When we talk about these types of tags, we are not taking into account any sort of linguistics. Tags have also been defined on the basis of linguistics. These include functional tags, functional collocation tags, origin collocation tags, function and origin tags, taxonomic tags, adjective tags, verb tags and proper name tags. Gaming specific tags have also been found.

2.4 Characteristics of Tags

There should be some pre-defined criteria for good tags. Otherwise, tagging will be done blindly which may lead to poor tag assignment. In [12], criteria for good tags is reported. According to this criterion, good tags should have the following characteristics:

- Tags should be *generic* so that they would be covering multiple concepts.
- Tags should be *popular* i.e. it should be frequently used so that its chances of being spam would be reduced.
- Each tag should identify only a limited number of resources.

- Tags that are used at personal organization level should not be publicly available. So, *exclusion* of these tags should be done.
- Tags should be *normalized* to avoid any sort of syntactic variance and synonymy problem.

2.5 Motivation of Tagging

It is necessary to present the motivation of tagging. It includes several reasons. Some of these are tags serve as an index for user's content. Tags aid as a descriptor for an item [22]. We can describe the semantic meaning of a resource or we can say that tags help to tell precisely what the resource is about [6]. Tags also work as a classification tool by classifying the user's content according to some characteristics [23]. One of the basic reason is they help the user in improving the tagging skill. At the system level, tags are used to increase the size of set of tags for tagging the untagged resources [24]. Apart from all of these reasons, tags make the retrieval system to be an efficient. Only the items that are relevant to user's requirement are retrieved if annotation is done properly.

Existence of redundant items on the Question/Answering sites can also be overcome by using tagging. Before posting any post, user checks either his required post is already there or not. If user found the content, he/she does not post it. Availability of the content can only be make sure if it is annotated properly so that it will be shown to user whenever he retrieves it.

Other reason for tags motivation include gaining attraction on the sites like Flicker where tags are assigned to photos, opinion expression on sites like Yahoo!, task organization by using tags like "toread", etc. [11].

Tagging is easy to learn and use. It is not required by user to have any sort of experience or knowledge before getting into the process of tagging. It provides flexibility to a user by allowing him/her to add or remove the tags. Tagging process helps in creation of communities having users who have shared preferences [25].

2.6 Tag Recommendation Methods

Tags can be recommended to the user following different techniques. These techniques can be separated for personalized tagging and social tagging. The tag recommendation process might be taking into account the text of the post that is being tagged or it might be considering several similar posts against the post that is ready to be tagged. Tags can also be recommended on the basis of similar interests among users forming a certain group. On the basis of these methods, different methods have been proposed for giving recommendation to the users [13]. The hierarchical representation of these methods is shown below:

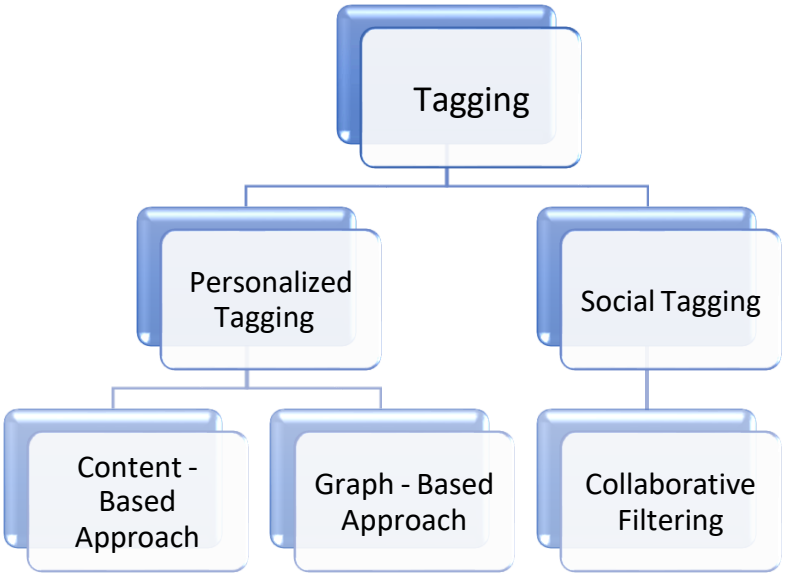

FIGURE 1: *HIERARCHY OF TAGGING SYSTEM*

2.6.1. Personalized Tagging Approaches:

Under Personalized Tagging, following two approaches are proposed: [14]

2.6.1.1. Content-Based Approach:

As the name indicates, content-based approach deals with the textual information of the resource that is being tagged. Tags of those resources are recommended to the user whose textual information or metadata is similar to the current resource and user has already interacted with those resources. For finding the similar resources, similarity-based retrieval is done. One of the simplest method is to calculate cosine similarity between resources [13]. Other techniques for suggesting tags on the basis of contents include Probabilistic based approaches, neighborhood-based approaches and use of classifiers etc. [14] [15]

2.6.1.2. Graph-Based Approach:

Graph-based approach is potentially stronger approach than content-based approach. It considers every similar user and every tagged item. Normally, a threshold is put on the number of tag's occurrences against the posts. While proposing candidate tags, graph based ranking algorithm considers the relevance among the document and user's preferences [14].

2.6.2. Social Tagging Approaches:

For social tagging, collaborative filtering is the suggested technique. Collaborative filtering is proved to be an advantageous in the sense that it can filter any type of items without considering their content. Still it has two major drawbacks. These are cold-start problem and sparsity [16].

Cold-start problem is encountered when we need to tag a new item i.e. cold-start items, or when we encounter a new user i.e. cold-start user. In both of the cases, we would be unable to

recommend high quality tags. Such problem arises when an item or user similar to the currently arrived item or user has never been encountered before.

Second problem is the sparsity that arises when we are available with insufficient data for identifying similar users or similar items. This problem is encountered due to lack of intersection among users or items.

Collaborative filtering is further categorized as following: [17] [18]

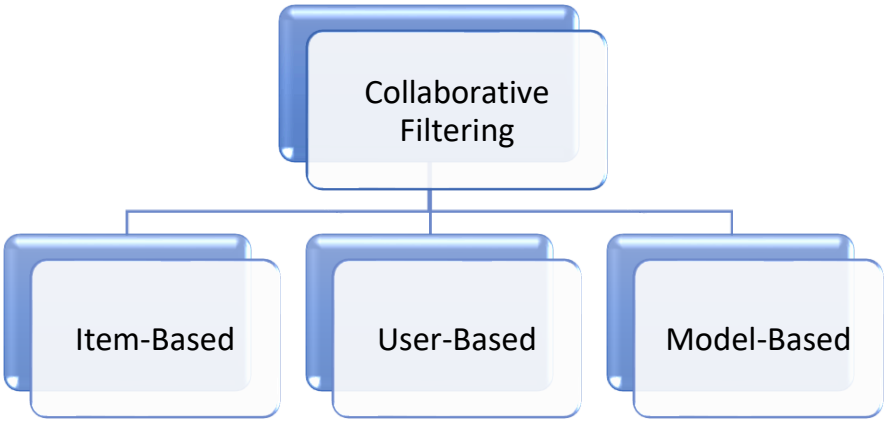

FIGURE 2: *TYPES OF COLLABORATIVE FILTERING*

2.6.2.1. Item-Based Approach:

Item-based collaborative filtering is a neighborhood-based approach and task is to find the similar items to the current item that needs to be tagged. Items similar to the new item are discovered from the items that have already been tagged by user. Item having highest similarity score, its corresponding tags are proposed as candidate tags for new item.

2.6.2.2. User-Based Approach:

User-based collaborative filtering is also a neighborhood-based approach which tries to find the users with the similar interests. It is assumed that users who have common interests will assign

same tags. Again, in this approach, tags used by the highest similar user will be recommended as candidate tags. For finding similar user commonly used measure is Pearson correlation.

2.6.2.3. Model Based-Approach:

In Model-Based collaborative filtering, model is trained on the basis of tags assigned to the items. After getting a trained model, ranked set of candidate tags are obtained for the new item. This type of collaborative filtering makes the use of clustering models [19], ranking model [20], latent factor model [21] etc.

2.6.3. Proposed Approach:

On the linear processing, the technique that we proposed for the tag prediction for stack exchange posts is Fuzzy Nearest Neighbor Search for medium to large scale data-sets. We also tried SVM based approach but its training took considerable amount of time so we couldn't implement it for large datasets. So, for smaller data-sets, results show that SVM is superior than Fuzzy Nearest Neighbor Search. Apart from this, topic modelling-based approaches are also tried but due to sparseness in topics against short texts, it didn't prove to be useful for our data-sets.

2.7 Disadvantages of Tagging

Apart from all of the previously mentioned advantages, tagging also has few obvious disadvantages [11]. These include:

- *Spamming:* It arises when user assign non-suitable tags to the posts that are of his/her interest and he/she just want to promote that post. Also, when a user assigns same tags to all the posts, issue of spamming arises.

- *Ambiguity:* It arises because each user assigns tag to the post according his/her understanding. Every user has his/her own point of regarding each post. So, due to this varying understanding, different tags having different contexts can be assigned by different users to a same post.

Chapter 3

Literature Review

With the increasing trend of using Question/Answering and other similar sites, automating the task of assigning tags on these sites is gaining attraction of researchers. Muchwork has been done in this area and different methods have been proposed. In our literature review, we organized these techniques as the hierarchy presented earlier. It includes content-based and graph-based approaches for personalized tagging. While under social tagging, we categorize the proposed techniques as item-based, content-based and user-based. Few Hybrid techniques are also presented. So, we tried to present the techniques covering both types tagging i.e. social tagging and personalized tagging.

3.1 Personalized Tagging

3.1.1. Content-Based Approaches:

As described earlier, content-based approaches deal with the textual information of the post. Tags are recommended on the basis of mutual occurrence of important keywords of post and tag. Tags having highest co-occurrence count with significant keywords are considered as candidate tags for the post.

Stanely, et al. [26] proposed ACT-R declarative memory retrieval mechanism based on Bayesian probabilistic model. Proposed method is based on the hypothesis that tags are suggested by observing their usage history and their correlation with the content of post i.e. its title and body.

Against every post, an activation score depending on the co-occurrence statistics of words and tags is calculated. Finally, potential tags for that particular posts are the tags having highest activation record.

Yin, et al. [14] proposed a probabilistic model for personalized tag prediction and incorporating the concept of ego-centric effect, item content and environmental effect. According to this method, unigram language model is built for the post and probability of tags that have already been used by user is calculated for the words of post. Building unigram language model and the use of past tags helped in incorporating the concept of item content and ego-centric effect respectively. For incorporating concept of environmental effect, probability of tags used by the user's neighbor (i.e. a user similar to the current user) is calculated. Tags are then ranked according to these probabilities. Now when a new user is encountered, we will not have tags used by him. For predicting the tags used for this new post, we have to use the environmental effect and item content. Items similar to this new post will be retrieved and similarity of new post and retrieved post is calculated. This calculation is referred as user-lan-user similarity. It is claimed that this method has improved the F-Score by 30% on the dataset that is publicly available.

Sood, et al. [27] proposed algorithm named TagAssist which is claimed to be used effectively and efficiently in extensive and real-time data processing. It is a two-step process that has partitioned the tags in two parts i.e. Tag-in (TI) and Tag-Out (TO). Tag-In are considered to be the words that are there in the body of weblog and can be used as tags, while Tag-Out are the abstract words representing the summary of weblog content and are not there in weblog. In the first step, Tag-In are extracted by taking into account the probabilities of term frequency, paragraph frequency and 1st occurrence paragraph position of each word. For calculating these

probabilities, Naïve Bayes is used. Words are then ranked according to their probabilities and top M (mentioned by user) words are selected as tags for Tag-In. In order to find Tag-Out, weighted adjacency matrix based on the co-occurrence count of Tag-In and Tag-Out is built. This matrix is then passed through Latent Semantic Index (LSI) for finding TO tags. System is evaluated using top-k accuracy, tag-recall and tag-precision.

Byde, et al. [28] proposed a simple personalized tagging technique. It says that by having a set of URLs which have already been tagged by user, tags for the new URL requested by that user can be suggested. This can be done by calculating the similarity among content of the current URL and the tagged URLs. For calculating the similarity between URLs, two techniques are suggested named Tagging-based similarity – preserving the frequencies of common tags, and Content-based similarity – preserving the frequencies of words present in URL's content. On the basis of calculated similarity, tags are ranked and top N ranked tags are considered to be the candidate tags for a new URL.

Kim, et al. [29] proposed Automatic In-text keyword tagging system. It is a two-step process and consists of candidate keywords selection and keyword tagging. Given a set of keywords, candidate keywords can be selected from the extracted tokens from the text by using any of the three approaches i.e. string::find() function of C++, Boyer-Moore-Horspool string matching algorithm and by building the inverted index of all terms. Inverted index is supposed to be the most efficient one. If we are using string matching algorithm, token will be selected as candidate tag if it is there in a set of keywords. However, if we are using inverted index, tokens will be stemmed and top certain frequent tokens will be selected as candidate tags. Then, similarity will be calculated between these candidate tags and inverted index terms. Candidate tags will then be sorted according to descending order of byte lengths of keywords.

Wang, et al. [30] proposed a system that assigns multiple tags to a web page. SVM binary classifier is used for the training of each tag. For each tag, positive training examples and negative training examples are collected. Before passing training data to the classifier, it is first preprocessed. Preprocessing involves the extraction of text from web page, and removal of by SMART Information Retrieval System [31]. Size of text should be greater than 800 bytes to provide enough information for classification. On passing the test document to each of the trained classifier, all suggested tags are retained. The results are evaluated by using three approaches i.e. exact word match, string distance and string distance with min3i.e. string distance is used for tag comparison with web pages having greater than or equal to 3 tags.

Gonz´alez, et al. [15] proposed multi-class multi-tag classifier system for stack overflow posts. By multi-class multi-tag, it is meant that a post can have multiple tags and can belong to multiple topics. Two pre-processing methods are proposed. One of them is the Traditional Preprocessing involving the traditional preprocessing steps. Resultant preprocessed questions are then used for making their BoW vectors. These vectors are stored in some iterable data structure. To further reduce the feature space, it is suggested to remove the words that have a normalized document frequency less than threshold value 0.01. The second proposed preprocessing technique named Lazy Preprocessing avoids the use of data structure by reading questions and tags directly from .csv file. Linear Support Vector Classifier (LSVM) was used to reduce the feature space. Proposed Classification System made the use of two classifiers - Naïve Bayes and SVM - and has three steps. First step is of Data Preparation which retrieves the i^{th} (position against which classifier will be trained for tag prediction) tag for all the questions. Value of i can be from 1-5 as it is mentioned in the paper that every post can have five tags. The retrieved tags are then used in second step which is of classifier training. In his step too, value of i (1-5) which corresponds to

i^{th} tag, needs to be mentioned for doing the training of classifier against that particular i^{th} tag. Last step is of Making Prediction. In this step, each i^{th} classifier predicts the i^{th}tag for the post. To cope up with the problem of redundant tags (different classifiers may predict the same tags), Filter algorithm is proposed. Precision, recall and F1 value are the adopted accuracy measures for measuring the effectiveness of proposed approach. According to the results, SVM provides good recall value while Naïve Bayes provides lower recall which results in poor F1 value.

Sriharee [32] and Rattanpanich, et al. [33] proposed a technique which is a two-step process. First step is of pre-processing the articles in which terms from text are extracted. Tf-idf matrix of these terms is built and on the basis of these extracted terms, ontology is built manually. Second step is Auto-Tagging process. This step involves the Article Classification and Tag Selection phases in it. In Article Classification phase, article is classified into relevant domain on the basis of cosine similarity. In Tag Selection phase, ontology weight of extracted terms that are matched with the ontology tags of relevant domain is calculated. The tags are then ranked on the basis of their calculated ontology weights. Edge-based method [34] is used for calculating this ontology weight.

Lu, et al. [35] proposed that webpages having similar content can have similar tags, so untagged pages can be assigned those tags that are assigned to similar webpages. Proposed technique is based on idea of Tag/Term coverage representing how well the tags/terms represent the tagged document. The rare tags/terms are considered to be more important as compared to frequent tags/terms, providing more coverage about the document. Apart from it, webpage that has more tags is assumed to have appropriate set of tags as compared to the webpage having fewer tags. Then, calculation of similarity between two webpages is based on four cosine similarities between tag's vector and term's vector of both webpages. Each cosine similarity is added

according to the weight assigned for tag/term coverage. Finally, for assigning tags to similar webpages probability is calculated which takes into account the similarity of webpages, probability of tag appearing in webpage and trustworthiness of webpage i.e. webpage having more tags is considered to be trustworthy. All tags are then ranked according to calculated probability.

Chirita, et al. [36] proposed a technique named "P-Tag" for suggesting personalized tags for web pages. Basic idea is to extract keywords from user's desktop documents that are relevant to the current document, rank those keywords and the top k words can then be suggested as tags for the current document. Following this basic idea, three approaches have been proposed. All approaches make the use of relevant keywords representing the webpage context.

i) Document oriented extraction: For a given webpage relevant keywords can be extracted from the similar documents placed on user's desktop using any of the three methods i.e. term-document frequency of terms, lexical compounds as discussed in[37]and sentence selection technique which is based on generating summarized sentences for the document. Top k-relevant words are then selected as tag according to their confidence scores.

ii) Keyword oriented extraction: After extracting relevant keywords from the given webpage, related keywords are then selected from the documents placed on user's desktop for each of the extracted relevant keyword. For extracting these related keywords, two techniques are suggested. These include term co-occurrence statistics and thesaurus based extraction. Again top-k words are suggested as tags according to their calculated confidence scores.

iii) Hybrid Extraction: It extracts relevant keywords for a given webpage. Then for each keyword, find the most relevant document from the documents placed on user's desktop. For each selected document, its relevant keywords are selected using any of the methodssuggested in

first technique. Top-k words are suggested as tags according to their calculated confidence scores.

Schuster, et al. [38] proposed a system that suggests tags for the posted programming questions. It is a hybrid system based on Multinomial Naïve Bayes classifier and SVM. Code snippets encapsulated in HTML-markup are extracted and tokenized. Tokens that are present less than 20 times are considered to be the variable names providing not much information about language. So, they won't be considered useful for training. Remaining tokens i.e. tokens having count greater than 20 in corpus are considered to be the tokens having information about programming language and are used for training of Naïve Bayes classifier. Now, while doing the training of SVM, feature vectors are computed for each set of positive examples (documents with their actual tags) and set of negative training examples (documents with randomly assigned tags). This feature vector is composed of six features for each document-tag pair. These include exact title, exact body, relaxed title and relaxed body, title pointwise mutual information (PMI) and body pointwise mutual information (PMI). It is tested that all of these features are important and none of them can't be excluded. For detecting tag for a test document, tag-document feature vector will be calculated for each tag and classifier will determine the most suitable tag for the given document. Finally, the two classifiers are combined for having hybrid system. Its output is union of both systems. The system is tested using F1 Score and the claimed F1 score is 0.41.

3.1.2. Graph-Based Approaches:

In graph-based tagging, graph notation is used i.e. we have connected vertices of users, items to be tagged and tags. Edges of graph are traversed to find the post similar to the current post which

is to be tagged. Tags of similar posts are most likely to be the candidate tags for the current post. These tags are then ranked by using certain ranking algorithms.

Song, et al. [39] proposed real-time automatic tag recommendation for document search engines. It is a two-state framework. Training data consists of triplet having words, docs and tags in it. Fist stage is Offline Learning stage in which weighted adjacency matrix is built having frequencies of words in a document. Normalization of this matrix is done using Laplacian to eliminate the bias. For extracting correlations among tags and words, reducing computational cost and noise removal, eigen values of normalized weighted adjacency matrix are found using Lancoz algorithm. Documents are then partitioned using Spectral Recursive Embedding (SRE) algorithm into K clusters and within each cluster tags are ranked according to proposed ranking algorithm. Then a Poison Mixture Model is proposed to estimate the distribution of words within documents of each cluster. Second stage is of Online Recommendation for tags in which joint probability of given document words with the tags are calculated. Tags are sorted in descending order on the basis of calculate joint probability.

Symeonidis, et al. [8] proposed tensor-based approach which identifies the association among users, items and tags. Basic concern is to predict tags by having user and an item. As a first step, tensor (multi-dimensional array) representing the relationship among user, item and tag is built. Tensor matrix is then unfolded into three matrices for user, item and tag. In order to expose the latent semantic association between users, items and tags, dimensionality reduction using higher order singular value decomposition is performed on three matrices. Application of SVD leads to the development of core tensor that reduces the dimensionality. Each element of this tensor is quadruplet representing the likeliness that a particular user will assign specific tag to a particular item. The tensor is then used in online tag recommendation phase to suggest tags according to

the weights assigned to (user, item) pair. Accuracy of proposed method is calculated by doing the comparison with different algorithms on the basis of precision and recall values.

Guan, et al. [24] proposed Graph-based ranking of multi-type interrelated data objects (GRoMO) for personalized tag recommendation. Given a set of users, set of documents, tag vocabulary and annotation history of all users, offline training is done. In offline training, we construct two affinity matrices. One of the matrices is based on the tagging history of user, and the other matrix maintaining cosine similarity between two documents having one at the index denoting the most similar document within k documents. Both of the matrices are then normalized according to defined equation. Then ranking factor is computed for the tags. During online recommendation, two ranking factors are calculated against the user provided document. Finally, after ranking the tags, top ranked tags are recommended to the user. The concept of personalization is incorporated by being biased towards the frequently used tags by the user.

Sigurbjörnsson, et al. [22] proposed tag prediction approach for the Flicker photos. It starts by making co-occurrence matrix by having a photo with user-defined tags. Normalized counts of tags w.r.t. frequency of all tags is maintained by using two techniques. These include: i) Symmetric measure – uses Jaccard co-efficient to decide which two tags have similar meaning, and ii) Asymmetric measure – maintains the tag co-occurrence matrix by calculating probability of a tag against tags assigned to a photo. This tag co-occurrence matrix will give a set of candidate tags which is used in phase of tag aggregation to produce the ranked list of tags. Two aggregation strategies are proposed named as voting and summing. Now, for recommending the most suitable tags, promotion function is defined which is based on three weighting functions: i) Stability promotion – having user defined tags, it assigns weights to the candidate tags, ii) Descriptiveness-promotion – considering frequently occurring candidate tags be general, it

reduces their contribution in candidate tags, and iii) Rank promotion – having user defined tags, it deals with the position of candidate tag instead of co-occurrence value. Promotion function with suggested tag aggregation techniques calculates the sore for each candidate tag. Tags are then suggested according to this score.

Garg, et al. [40] suggested different approaches for tags prediction. These approaches make the use of user's history by maintaining a binary matrix showing specific picture is tagged with specific tag, a global history of all users' i.e. tags used by users for tagging pictures and a global co-occurrence matrix defining how many times two tags co-occurred. First suggested approach is local scheme making the use of Naïve Bayes of approach in which binary matrix of user's history is used. Based on user's history, probability of new tag is calculated given tag already used by a user. Second suggested approach is incorporating the concept of tf-idf. It is a global scheme that makes the use of global co-occurrence matrix. Third suggested scheme is the hybrid of both local and global scheme. It basically deals with the coverage concept of tags by saying that both co-occurrence information and overall frequency of tags are important. So, for having both of these concepts, global and local scheme should be used together. Lastly, WWW scheme is suggested in which global co-occurrence matrix along with number of pictures tagged with specific tags is used. By doing so, penalty is imposed on the frequent and rare tags.

3.2 Social Tagging

3.2.1. Collaborative Filtering:

3.2.1.1. Item-Based Approach:

As the name indicates, tags are assigned to the post on the basis of similar posts that have already been tagged by the user. It is assumed that most probably user will re-use the tags that he/she have already used for similar posts. Tags that have been assigned to highest similar post will be considered as candidate tags for the current post.

Mishne [41] proposed a simple technique named AutoTag. In this method, we have a collection of posts stored by search engine. Whenever user enters a post, system retrieves the top most similar posts from the collection. Then tag model searches for the top most frequent tag associated with the retrieved similar posts. These frequent tags are then passed through filtering and reranking phase. In this phase, similarity score of any of frequent tags for a post is increased by constant factor if that tag has already been used by the user. Three tags are suggested for each post. The technique is evaluated on the dataset of Intelliseek for 3rd Weblogging Workshop. On comparing predicted tags with the actual tags, calculated precision and recall values were 0.40 and 0.49 respectively.

Xu, et al. [12] proposed technique which takes into account the basic characteristics of tags like uniformity, exclusion of certain tags, high popularity etc. Given a set of tags assigned by all users, set of excluded tags, make a set of tags by taking difference of set of all tags with set of excluded tags. Then for each tag in this new set, its contribution is calculated which is based on the defined authority scores of all users. This authority score basically defines how much user's tagging is consistent with other users. Tags having highest authority scores are penalized

according to defined method for reducing the information redundancy and tag will be rewarded if it co-occurs with the tags already used by user. After all these steps, we will have a set of candidate tags from which top-k tags can be suggested to the user.

Budura, et al. [42] proposed neighborhood-based tag prediction technique. It maintains a graph of documents cited by each document and it is assumed that all the neighborhood documents have the same tags. Proposed technique calculates the relevance score for each for a document by merging the four key principles. These include i) tag occurrence – number of times a tag occurs in neighborhood documents, ii) tag co-occurrence – some specific tags appear with specific documents, iii) document similarity – cosine similarity between the documents and iv) tag distance – shortest distance (in terms of edges) between the closely related documents, relevance score of tag decreases as this distance increases. Now, for deducing tags for the new document, its neighborhood documents are traversed in such a way that total number of visited documents should be minimized. Score for each tag found in neighborhood documents is calculates and pair of (tag, score) is maintained. Candidate tags are then maintained on the basis of worst score and best score calculated for the neighborhood tags. The tag with worst score is assumed to be at k-rank. All the candidate tags having score less than the worst score are eliminated.

3.2.1.2. User-Based Approach:

In user-based approach, similar users are found instead of finding similar items. Most probably, users having similar interest will be using same tags. Tags that have been assigned by highest similar user will be considered as candidate tags for the current post.

Kim, et al. [16] proposed a technique in which those tags will be suggested that have already been by user or by similar users. Three matrices have been built for being used in the technique.

User-item binary matrix (representing either item is selected by user or not), user-tag frequency matrix (having number of items that are tagged as particular tag by user) and tag-item frequency matrix (representing number of users who have tagged each item with particular tag). Best neighbors are selected by calculating cosine similarity of target user with every other user. Tags frequency have also been considered. Rarely occurring tags used by users are considered to be more important. Then, k best neighbors will be selected for suggesting tag i.e. tag used by those neighbors will be recommended.

3.2.1.3. Model-Based Approach:

In Model-based approach, model is made to learn the tags that have been assigned to posts. After getting a learned or trained model, it is used to predict tags for the new posts.

Parikh [43] and Saha, et al. [44] made use of Vowpal Wabbit (VW) and dataset of Stack Exchange. Vowpal Wabbit (VW) is learning system library that has notable features for machine learning tasks such as classification, regression etc. It has multiple loss functions in it representing the cost of particular event. The proposed technique made the use of this library for getting trained classifier against each tag. These classifiers will then be used for predicting the tag of test question. After getting done with data preprocessing phase Like in all the other techniques, first traditional data preprocessing is done. For tag prediction, it is suggested either to use One vs All method or use discriminant classifier for each tag and then select the most suitable tags. Since objective was to predict multiple tags for each question so discriminant classifier is adopted. For prepare the training and testing data, proper method has been proposed which is based on occurrence count of each tag. In the training phase, VB's hinge SVM function is used for getting the trained classifier against each tag. While predicting the tag for the test question, pass it through the trained SVM classifiers of top 500 tags. Against each tag, save the

calculated SVM value in the list. The list will be sorted according to calculated SVM value and top 5 tags will be selected. It is claimed that Mean F1 Score above 0.71132 is achieved using the proposed technique.

3.3 Hybrid Approaches

Boudaer, et al. [45] proposed hybrid technique based on Natural Language Processing and Collaborative filtering. Method is enriched with user histories i.e. the tags that have already been used by user in the past. It is suggested that while predicting the tags, context of document words should be taken into account. Words are ambiguous and if this fact is ignored, unsuitable tags for the particular document would be predicted. Latent Dirichlet Allocation (LDA) [46]is the preferred approach for dealing with the word ambiguity. The proposed technique made the use of Labelled-Latent Dirichlet Allocation (LLDA) which is an extension of LDA. Proposed method has two phases. In the first one, we have the N number of Binary Relevance classifiers based on LLDA for making the predictions about N number of tags against each text fragment. Tag correlation is modeled among the predicted tags for dealing with the words' ambiguity. In the second phase, an argument is made i.e. there are the chances of re-using the tags if user will post the questions on the same topics. Having list of frequently used tags by user, probability of assigning each frequent tag is calculated for the current question. Now, the set of predicted correlated tags and predicted frequent tags is passed to multi-label classification system. This system generates the most suitable tags for the document. Social tag profiles i.e. tag used by the user's peer have also been considered but this lead to the reduced accuracy. Justification for this reduced accuracy can be that tags are highly personal with respect to each user. It is claimed that proposed method provides the accuracy up to 73.8%.

Wetzker, et al. [47] proposed novel user centric model that assigns tag by deducing the meaning of tags already assigned by the user. It is basically a tag translation approach which translates the user assigned tags to global folksonomy and then on the basis of this translation, accurate meaning of tag is predicted before assigning it to any item. It constructs a folksonomy tensor based on items, tags and users. With the help of this tensor, we can have desired co-occurrence matrix of items-tags which is needed for translation. Each entry of this matrix represents the number of users that assigns particular tag to specific item. On the basis of tag co-occurrence with the items, we can have tensor for user's tag vocabulary which maintains the co-occurrence counts of normalized item and tag. This tensor denotes the tag distribution of all items that user has tagged with same specific meaning. The user's personomy tensor is then used for folksonomy translation. For doing so, vector multiplication of user's personomy tensor and tag vector is done. The resultant vector has the weights for the corresponding meaning for each tag. It is noted that tags can be item specific so in order to overcome it, overall tags for the item are also included. Now, while predicting tags for new item, user personomy and overall tags of item are considered. Tags having highest weight will be suggested for the item.

3.5 Other Approaches

It is found in literature that tag recommendation problem can be solved by using the famous topic modelling approach named as Latent Dirichlet Allocation (LDA). LDA divides the document into different topics and each topic has the distribution of words with probabilities. These probabilities represent how much a term is relevant to that particular topic. According to this method, each document is a mixture of different topics or we can say that every document is created by collecting words from different underlying topics [48].

Basically, it is non-deterministic and unsupervised Bayesian probabilistic learning method and can be used to measure similarity between the documents on the basis of their underlying topic distribution. If we look into its details, we come to know that this method starts by traversing all documents and random assignment of its words to the decided number of K topics. After this step, we will have the topic probability distribution of our documents as well as the topic-word probability distribution. Topic probability distribution represents the probability of topic for the document and topic-word probability distribution represents the probability of word for that particular topic. Since it is a totally random distribution, we need to improve it so that we can get more refined topics. This improvement can be done by traversing each word of each document and against each topic calculate two things:

i. $(T|\ D) = \frac{No.\ of\ words\ in\ T + \beta}{total\ words\ in\ T + \beta} * (No.\ of\ words\ in\ D\ that\ belongs\ to\ T\ +\ \alpha\ )$

   Where,

   T represents particular topic,

   D represents document,

   α and β are hyper-parameters to assign some probability even if that word did not appear in the topic before.

   Basically, this probability represents the proportion of document that already belongs to this topic.

ii. $(Word|Topic)$ which represents how much this topic is probable in generating this word

   So, this is a re-assignment step for current word in which we are trying to make its assignment to correct topic by repeating this step several times.

Once we get stable assignment of words, we will get a trained LDA model.

Wu, et al. [49] proposed a tag-content co-occurrence method by making the use of LDA and its extension LLDA. Proposed method has three phases. In the first phase, documents are passed to LDA to generate the underlying topic-word probability distribution. Second phase deals with the incorporation of LLDA. Briefly this step says that latent topic can be can be determined by the topic of document and underlying topics of each topic will be equal to its assigned tags. Third step makes the use of co-occurrence content of tag and calculates the probability of each word of document by keeping in view the two things: 1) either the current word is tag itself, or 2) it is one of the co-occurred words against tag. While creating topic-word probability distribution this method makes sure that words of documents would be assigned to the topics that are correspondent to document's assigned tags. Finally, the created model can then be used to make predictions for test posts.

Krestel, et al. [50] proposed two approaches making the use of LDA. Proposed approach suggests that instead of dividing the documents into topics composed of terms, divide the documents into topics composed of tags assigned to document. Now, we will have individual probability for each tag against document. Final probability of each tag for a document can be calculated by combining the probability of topic for a document and probability of tag belonging to that topic. Then we can pick the top K tags with highest probability as candidate tags.

Chapter 4

Experimental Setup

Methods for tag prediction that are proposed by Hong, et al. [51] are implemented. In this section, the results obtained after implementing those methods are presented and compared.

4.1 Methodology

Following baseline methods are tried for the selection of suitable tags for each post:

4.1.1. Greedy Method:

This method says that first collect the list of commonly used tags. Then scan each post, and find which tags from the collected list are there in the text of the post. The tags that are there as the words in the post's text will be referred to as the candidate tag for that post. These candidate tags will serve as the predicted tags. We already have the list of actual tags against each post. Now, using list of actual tags and predicted tags F-Score can easily be calculated.

4.1.2. Nearest Neighbor Search:

This method says that having list of all tags, take each tag one by one and concatenate all the posts having that particular tag. For these concatenated posts, calculate TF-IDF to be used for calculating cosine similarity of test posts with each tag. These concatenated posts will serve as a centroid for that particular tag as they refer to the context in which that particular tag has been used. Now, take the test posts. For each test post, after calculating its TF-IDF, calculate its cosine

similarity with each tag centroid. Pick the k-top most tags by arranging the tags according to the calculated cosine similarity with the query.

$$Cosine\ (query, ConcatenatedPost) = \frac{\sum_{i=1}^{|V|} q_i d_i}{\sqrt{\sum_{i=1}^{|V|} q_i^2}\sqrt{\sum_{i=1}^{|V|} 2}}$$

Where,

- q_i and d_i is the tf-idf of i^{th} word of test post and concatenated post respectively
- |V| represents size of common words in numerator and vocabulary size of query and concatenated post in denominator

4.1.2.1. Diagrammatic Representation:

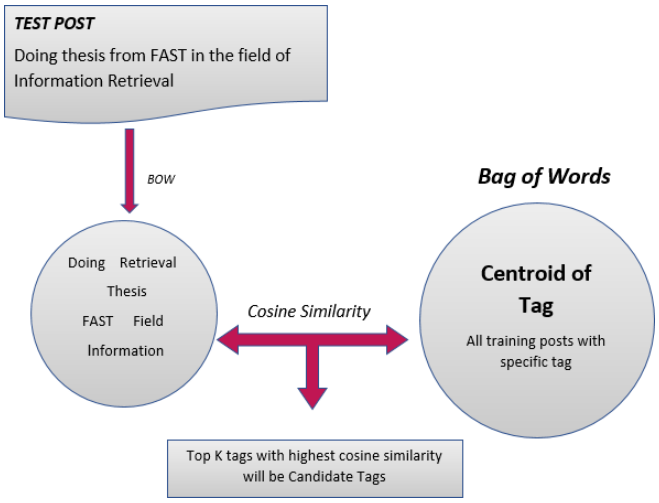

FIGURE 3: *WORKING OF NEAREST NEIGHBOR SEARCH METHOD*

In the above figure, we have test post for which we need to predict tags. After converting to its corresponding bag of words model and having its tf-idf values, the cosine similarity of this BoW model will be calculated with the centroids of all tags. After this step, we will have cosine

similarity scores for all tags and top k tags with highest cosine similarity value will be selected as candidate tags for this test post.

4.1.3. Tag-Keyword Co-occurrence Model:

In this method, co-occurrence matrix is built by traversing all training posts. Co-occurrence count of query words' and tag is maintained in this co-occurrence matrix. This is done by maintaining the count of co-occurrence of query words in training posts against each tag. Each time a specific query word and tag is found to be co-occurred, count is updated. These counts are then used in the calculation of word-tag probability. Word-tag probability is calculated as:

$$\text{Probability}(\text{tag} \mid \text{queryWord}) = \frac{No.of\ Posts\ where\ query\ word\ and\ tag\ co{-}occur}{Total\ no.\ of\ posts\ where\ word\ appear}$$

In above equation, denominator part can be considered as document frequency (df) of query word. Now for selecting the most suitable tag, its probabilities are calculated. Probability of each tag will be the highest word-tag co-occurrence probability calculated against it. Now, pick the top K tags according to the sorted probabilities as candidate tags.

4.1.3.1. Diagrammatic Representation:

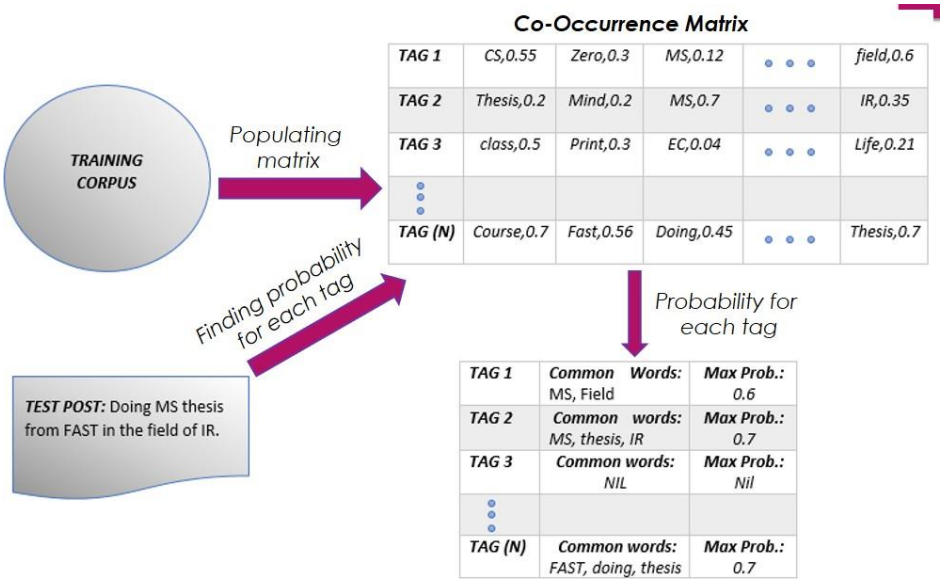

FIGURE 4: *WORKING OF TAG-KEYWORD CO-OCCURRENCE MODEL*

In the above figure, we have created co-occurrence matrix by traversing all the training posts in the training corpus. In co-occurrence matrix, there are calculated probabilities of words that were co-occurred against each tag. Now when we encountered our test post, for predicting its tags we will refer the co-occurrence matrix. From the co-occurrence matrix, we will extract the rows of each tag and will find the common words between the extracted row and our test post. Amongst the found common words, we will look for the maximum probability. Now, this maximum probability will be the probability of that tag for our test post. And the tags against which we won't find any common words, its probability will be zero for current test post.

4.1.4. Support Vector Machine – SVM:

SVM classifier available in Python toolkit named Sklearn is used to predict the candidate tags. Features for the classifier are created as proposed by Schuster, et al. [38]. Author has proposed

six features to be created for the post-tag pair i.e. we need to create these features against each tag of the post. Apart from the actual tags, paper suggested to randomly assign false tags to the post and number of these false tags would be equal to its actual tags. Features that are created for classifier are exact title, exact body, relaxed title, relaxed body, title PMI and body PMI. Exact title and exact body check that is there any tag word in the post's text. Relaxed title and relaxed body check that if we tokenize tag on the basis of hyphen, then all of its tokens are there in the post's text or not. Title PMI and body PMI make the use of co-occurrence matrix. Title-tag and body-tag co-occurrence matrices are populated maintaining the record of words occurring in the title and body against each tag of the post. Based on this matrix, point-wise mutual information is calculated as following:

$$PMI(T, word) = \sum_{i=1}^{N} log \ \frac{(No.of\ posts\ where\ tag\ and\ i^{th}\ word\ co-occurs\ ) * (total\ no.of\ posts)}{(no.of\ posts\ with\ tag\ T) * (no.of\ posts\ with\ i^{th}\ word)}$$

Feature vectors of training posts are passed to SVM for getting trained classifier. The trained model then used to predict tags for testing posts. SVM returns Boolean value (1 for true tag and -1 for false tag) against each tag for every post instead of returning any ranking. So, we predicted that how many tags are truly predicted by our trained model and calculated the accuracy of obtained results.

4.1.4.1. Diagrammatic Representation:

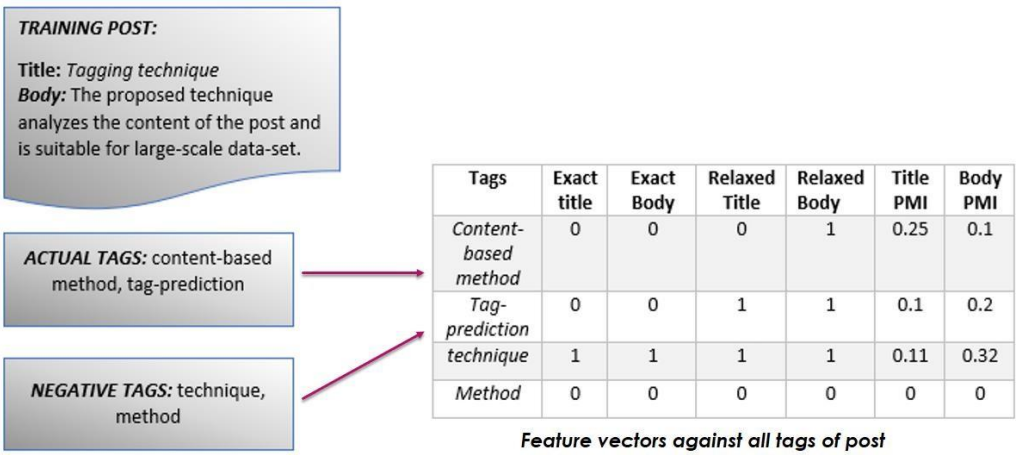

FIGURE 5: *SVM FEATURE VECTORS*

In the above figure, we have training post with its actual tags and negative tags. Now, we need to calculate feature vectors of this training post with all of its actual and negative tags. Feature values calculated against negative tags will serve as negative training examples for this post. In the above example, our tag "content-based method" is not there in the body and title of post. So, values of exact title and exact body are 0 for this tag. For Relaxed Title and Relaxed Body, we tokenize out tag "content-based method" on the basis of hyphen then its tokens will be *content* and *based method*. We found content as a word in the body of post but not in the title so Relaxed Body value is 1 for this tag and Relaxed Title value is 0 for this tag. For Title PMI and Body PMI, co-occurrence matrix maintaining co-occurring count of words against tag is used. On the basis of this matrix, value of title PMI and Body PMI is calculated.

4.1.5. Fuzzy Nearest Neighbor Search:

This method makes the use of Co-Occurrence matrix that was built in Tag-Keyword Co-

Occurrence model. We obtained 10-20 tags predicted by this model and then used these predicted tags in Fuzzy NN. Against each test post, rows of predicted tags are extracted from the co-occurrence matrix. The extracted row will serve as tag centroid and have the probability of co-occurred words against each tag. We then compute the cosine score between the extracted row and query. Here, we will be using probabilities of the words instead of tf-idf as weights of cosine. Pick the k-top most tags as candidate tags by sorting the tags according to their calculated cosine scores. Cosine similarity with probabilities as weights can be calculated as:

$$Cosine\ (query\ ,TagCentroid) = \frac{\sum_{i=1}^{|V|} q_i d_i}{\sqrt{\sum_{i=1}^{|V|} q_i^2}\sqrt{\sum_{i=1}^{|V|} 2}}$$

Where,

- q_i and d_i is probability of i^{th} word of test post and tag centroid (list of words co-occurs against most probable tag) respectively
- |V| represents size of common words in numerator and vocabulary size of query and tag centroid in denominator

Probabilities saved in co-occurrence matrix in Keyword-Tag Co-Occurrence model are used here.

4.1.5.1. Diagrammatic Representation:

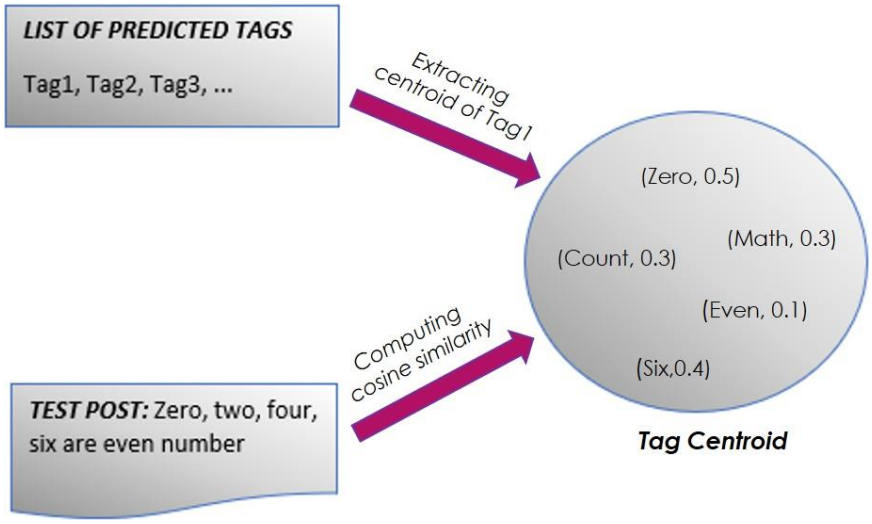

FIGURE 6: *WORKING OF FUZZY NEAREST NEIGHBOR SEARCH METHOD*

In the above figure, we have test and tags predicted for it by Tag-Keyword Co-Occurrence Model. Centroids of all of these predicted tags will be extracted from tag-keyword co-occurrence matrix that was built in Tag-keyword Co-Occurrence Method. Now, cosine similarity will be calculated between the extracted tag centroid and test post. We will have cosine similarities for all of the tags predicted by Tag-Keyword Co-Occurrence Model and top K tags having highest cosine value will be referred as candidate tags for this test post.

4.1.6. Latent Dirichlet Allocation – LDA:

LDA that is implemented in python toolkit named Gensim is used for training and testing purpose

In this basic application of LDA to our problem, we used concatenated posts against each tag for our training phase. These concatenated posts serve as a bag of words for that particular tag. Training is done against all these concatenated posts separately and eventually we will have trained models equal to number of unique tags in our data-set. After training, we will have topic distribution of words for K number of topics against each model. Now, we will test our models

for each test post. Against each trained model of tag, we will get its most probable topics for current test post and we will retain the topics with maximum probability. This maximum probability will be the probability of that tag for current test post. This procedure will be repeated for all test posts. Tags will be sorted according to their probabilities for each test post and top K tags will be referred as candidate tags.

4.1.6.1. Diagrammatic Representation:

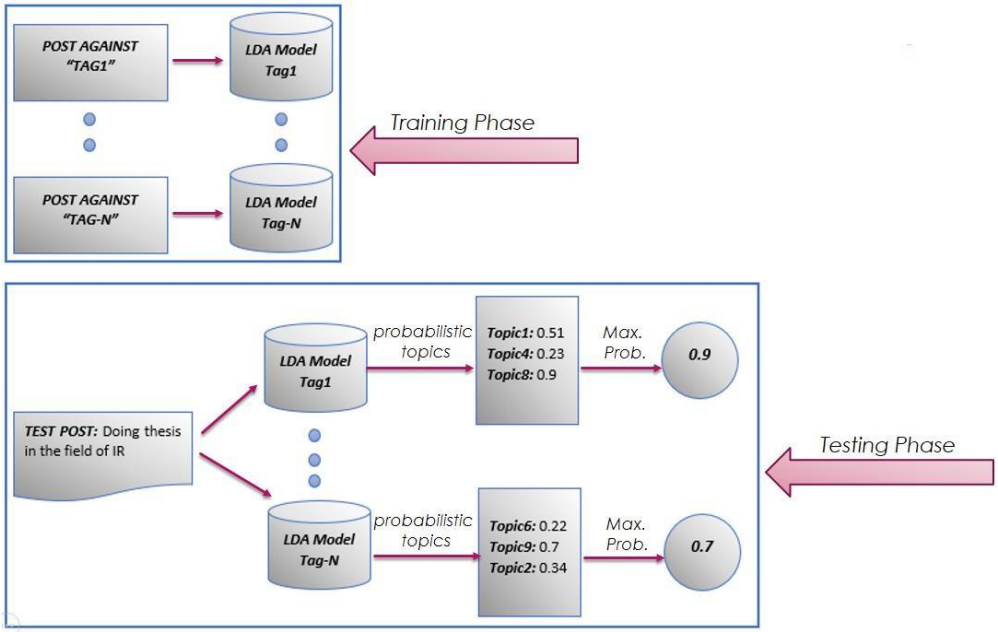

FIGURE 7: *WORKING OF LDA FOR TAG PREDICTION*

In the above figure, training and testing steps are shown. In training phase, we have trained models against all tags. Now, when we start traversing our test posts, each of the test post will be fed to all of trained models. Like in above figure, after feeding test post to trained model against TAG1, model returned the most probable topics for this test post. Amongst the probabilities of the most probable topics we will save the maximum probability. This maximum probability will be the probability of that tag against which we test our test post. Ultimately, we will have

probabilities of all of the tags for current test post and top K tags with highest probabilities will be referred as its candidate tags.

4.1.7. LDA Using Kullback-Leibler (KL) Divergence - Topic Distance Based Approach:

In this approach, Kullback–Leibler (KL) divergence [52] is incorporated as a similarity measure after the inference step of LDA. This method starts with getting trained LDA model against trained data-set. Get the topic probability distribution for training data-set from trained model. Similarly, get the topic probability distribution for testing data-set. Now, KL divergence will be used to find the similarity of each training post with the current test post.

KL divergence basically calculates the difference between the two probability distributions. In our case, these distributions will be topic probability distribution for training post and test post. KL divergence is calculated as:

$$KL - Divergence\ (P||Q) = \sum_{i=1}^{N} P(i) log \frac{P(i)}{Q(i)}$$

Where,

P(i) and Q(i) indicates the probability of i^{th} topic for current training and testing post resp.

According to above formula, all training posts are traversed against each test post and difference of their topic probability distribution is calculated. Tags of corresponding training posts will be sorted in ascending order according to calculated KL divergence. The smaller the KL divergence, more similar the training post is to test post. Top K tags with smaller KL divergence can then be selected as most suitable tags for current test post.

4.1.7.1. Diagrammatic Representation:

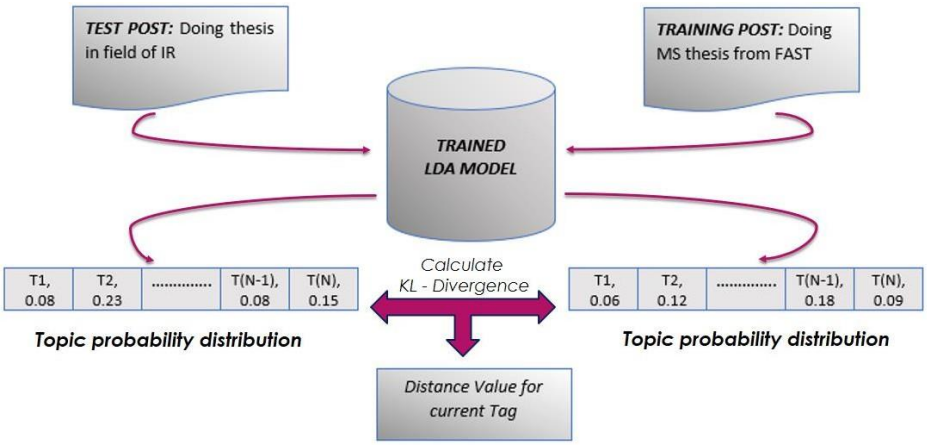

FIGURE 8: *WORKING OF LDA USING KL-DIVERGENCE*

In the above figure, we have trained LDA model against whole training corpus. Now, we will feed each test post to this trained model and against each test post, each training post will also be fed to this model. As a result of this, we will have two probability distribution values and we will calculate the KL-Divergence value between these two distributions. The training distribution against which we will be having minimum KL-Divergence value, its corresponding tag will be referred as candidate tag for current test post.

4.2 System Specifications

Experiments ran on the system with following specifications:

Specs	***Details***
Processor	Intel (R) Core (TM) i7-4500U CPU @ 1.80GHz 2.40GHz
RAM	8.00 GB
System type	64-bit Operating System, x64-based processor

TABLE 2: *SPECS OF SYSTEM USED FOR EXPERIMENTS*

Relatively small datasets take few minutes on running each technique. While large datasets take 4hrs-6hrs for predicting k tags on running Nearest Neighbor Search and Tag-Keyword Co-Occurrence Model.

4.3 Dataset

Dataset used in experiments is obtained from stack overflow dump [53]. Data related to different fields is available there and amongst those datasets few datasets are selected for testing the results. Each dataset has 8 xml files in it. The files that are of our use are Posts.xml and Tags.xml. In posts, we have text of posts that have been posted on the stack overflow related to a field of which that dataset is downloaded. Whereas, Tags.xml is maintaining the record of tags used for the posts of that particular field.

Details of dataset used in experiments are given below:

Dataset	***Total Tags***	***Training Samples***	***Testing Samples***
3-d Printing	165	492	105
Data Science	315	4,105	845
Physics	876	77,979	15,550
Apple	1,048	67,039	13,427

TABLE 3: *DATASETS USED IN EXPERIMENTAL SETUP*

4.2 Data Preprocessing

Before moving towards the implementation of any of the proposed methods, data is pre-processed by following certain steps. For data preprocessing we need to parse our required xml files. Only the required attributes of both of the xml files are parsed. Like in Posts.xml, the

attributes that are parsed are id, postTypeId, body and tags. Apart from this, only the posts that are posted as questions are picked for being used in our experiments. Such posts have their postTypeId=1. While in Tags.xml, attributes that are parsed are "Count" and "TagName".

After the parsing of xml files, they are passed through the pre-processing phase which involves the removal of stop words.

4.3 Evaluation Measure

Methods are evaluated on the basis of following four accuracy measures:

4.4.1. Precision:

Precision deals with the top-ranked results that are mostly relevant to the user's need. It is calculated as:

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

Where,

tp: number of true positive results i.e. results actually required by user.

fp: number of false positive i.e. results that are not actually required by user but still identified as one the required results.

4.4.2. Recall:

Recall deals with the finding of all relevant results from the corpus. It is calculated as:

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

Where,

tp: number of true positive results i.e. results actually required by user.

fn: number of false negative results i.e. results that are actually required by user but still not identified as one the required results.

4.4.3. F-Score:

F-Score takes both precision and recall into account. It is the harmonic mean of precision and recall. Both of these values should be high for F-Score to be higher. It is calculated as:

$$FScore = \frac{2 * p * r}{p + r}$$

Where,

p stands for precision and r stands for recall.

Chapter 5

Results

Results for all of the above-mentioned datasets against each method are reported in this section. Number of considered top most frequent tags are varied in Nearest Neighbor Search Method. Number of predicted tags against which results are reported, are 2,3,5,7 and 9. In SVM, number of predicted tags is not taken into since in this method we don't have any ranking score, so we calculate overall accuracy measures for this method. In LDA, number of iterations are varied because this algorithm converges with the greater number of iterations and we can pick the one giving desired results. In LDA using KL Divergence, number of topics are varied. After presenting results of all techniques against each data-set, comparative graphs of all techniques are presented for all evaluation measures.

5.1 3-d Printing

5.1.1. Greedy Method:

F-Score	Precision	Recall
0.33	0.31	0.35

TABLE 4: *RESULTS OF GREEDY METHOD FOR 3D-PRINTING DATASET*

5.1.2. Nearest Neighbor Search:

The results obtained by varying the number of top most frequent tags are given below:

Top Frequent Tags	No. of Predicted Tags (k)	F-Score	Precision	Recall
60	*2*	0.19	0.21	0.19
	3	0.19	0.18	0.24
	5	0.2	0.15	0.25
	7	0.2	0.14	0.45
	9	0.19	0.12	0.49
90	*2*	0.19	0.22	0.21
	3	0.19	0.18	0.25
	5	0.22	0.17	0.27
	7	0.2	0.14	0.43
	9	0.19	0.13	0.51
110	*2*	0.19	0.22	0.19
	3	0.2	0.19	0.25
	5	0.21	0.17	0.35
	7	0.21	0.14	0.44
	9	0.2	0.13	0.52
140	*2*	0.18	0.21	0.18
	3	0.2	0.19	0.25
	5	0.2	0.16	0.34
	7	0.21	0.15	0.44
	9	0.2	0.13	0.51

TABLE 5: *RESULTS OF NEAREST NEIGHBOR SEARCH FOR 3D-PRINTING DATASET*

5.1.3. Tag-Keyword Co-Occurrence Model:

Since tuned parameter in this approach is number of predicted tags, so on varying it the obtained results are given below:

No. of Predicted Tags (k)	F-Score	Precision	Recall
2	0.14	0.13	0.15
3	0.16	0.13	0.22
5	0.15	0.1	0.28
7	0.16	0.1	0.38
9	0.15	0.09	0.41

TABLE 6: *RESULTS OF TAG-KEYWORD CO-OCCURRENCE MODEL FOR 3D-PRINTING DATASET*

5.1.4. Support Vector Machine:

This method is evaluated by examining how many tags did it predict correctly for all queries. So, the evaluated values are given below:

F-Score	Precision	Recall
0.33	0.28	0.41

TABLE 7: *RESULTS OF SUPPORT VECTOR MACHINE FOR 3D-PRINTING DATASET*

5.1.5 Fuzzy Nearest Neighbor Search:

This method is evaluated by varying the number of top most 10-20 tags predicted for each test post by Tag-Keyword Co-Occurrence Method.

Tags Picked from Co-Occurrence Matric	No. of Predicted Tags (k)	F-Score	Precision	Recall
10	2	0.23	0.24	0.23
	3	0.22	0.19	0.27
	5	0.21	0.15	0.34
	7	0.19	0.12	0.39
	9	0.17	0.1	0.45
20	2	0.21	0.22	0.2
	3	0.24	0.21	0.29
	5	0.24	0.17	0.39
	7	0.22	0.14	0.43
	9	0.13	0.07	0.64

TABLE 8: *RESULTS OF FUZZY NEAREST NEIGHBOR SEARCH FOR 3D-PRINTING DATASET*

5.1.6. Latent Dirichlet Allocation-LDA:

In the basic implementation of LDA based approach, the tuned parameter against which the experiments are carried out is number of iterations. So, the results obtained by varying the number of iterations are given below:

No. of Iterations	No. of Predicted Tags (k)	F-Score	Precision	Recall
50	*2*	0.15	0.17	0.15
	3	0.16	0.15	0.22
	5	0.16	0.13	0.29
	7	0.16	0.11	0.35
	9	0.15	0.09	0.41
500	*2*	0.14	0.17	0.17
	3	0.17	0.16	0.22
	5	0.18	0.14	0.33
	7	0.18	0.12	0.39
	9	0.17	0.11	0.45
1000	*2*	0.15	0.17	0.15
	3	0.18	0.17	0.22
	5	0.19	0.14	0.33
	7	0.18	0.12	0.39
	9	0.16	0.1	0.43

TABLE 9: *RESULTS OF LATENT DIRICHLET ALLOCATION FOR 3D-PRINTING DATASET*

5.1.7. LDA using Kullback–Leibler divergence:

This method is evaluated by varying the number of topics. So the results obtained by varying the number of topics are given below:

No. of Topics	No. of Predicted Tags (k)	F-Score	Precision	Recall
Equal to number of tags - 162	*2*	0.11	0.07	0.3
	3	0.13	0.11	0.17
	5	0.12	0.09	0.22
	7	0.11	0.08	0.26
	9	0.11	0.07	0.3
500	*2*	0.12	0.13	0.12
	3	0.13	0.11	0.18
	5	0.12	0.09	0.22
	7	0.12	0.07	0.27
	9	0.11	0.07	0.31
800	*2*	0.11	0.11	0.12
	3	0.12	0.11	0.16
	5	0.11	0.08	0.21
	7	0.11	0.08	0.27
	9	0.1	0.07	0.29

TABLE 10: *RESULTS OF LDA USING KULLBACK-LEIBLER DIVERGENCE FOR 3D-PRINTING DATASET*

5.1.8. Comparative Analysis of All Evaluation Measures Against All Techniques:

5.1.8.1. F-Score:

Following graph is the comparison of maximum F-Score obtained against different techniques by varying the number of predicted tags. It can be observed from the graph that results of SVM and greedy method are considerably better against this data-set.

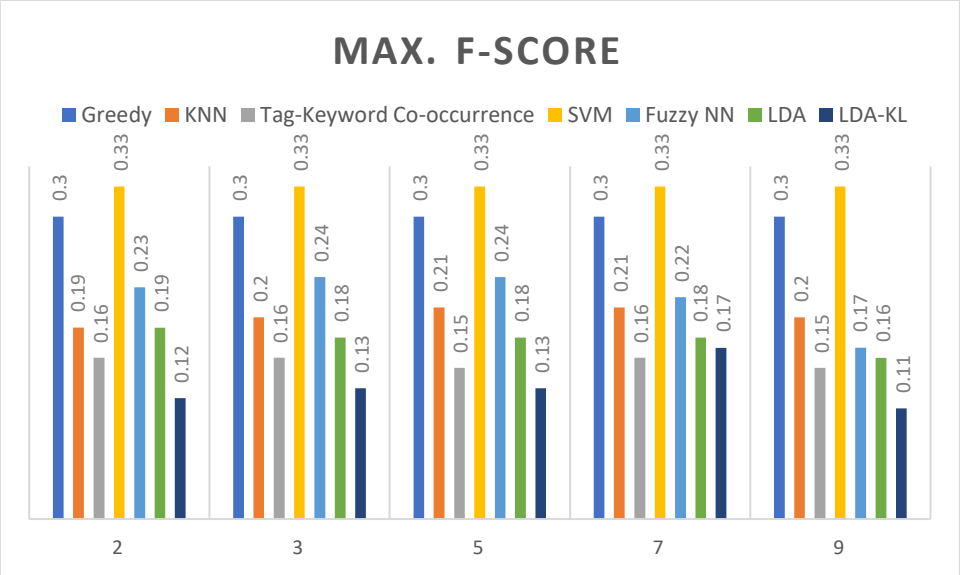

FIGURE 9: *MAX. F-SCORE OBTAINED AGAINST DIFFERENT TECHNIQUES FOR K = 2,3,5,7,9*

5.1.8.2. Precision:

Following graph is the comparison of maximum Precision obtained against different techniques by varying the number of predicted tags. It can be observed from the graph that as we are increasing the number of predicted tags, precision drops down.

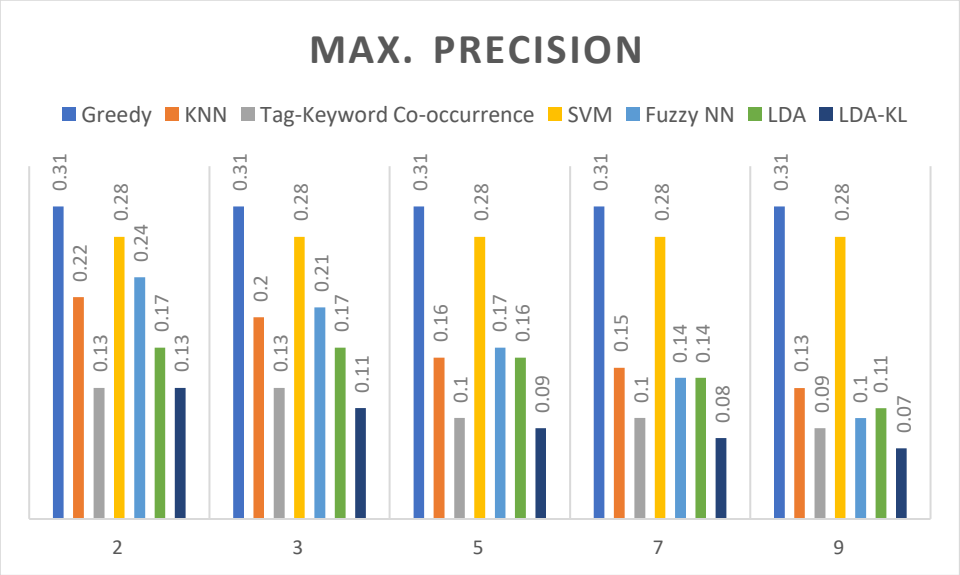

FIGURE 10: *MAX. PRECISION OBTAINED AGAINST DIFFERENT TECHNIQUES FOR K = 2,3,5,7,9*

5.1.8.3. Recall:

Following graph is the comparison of maximum Recall obtained against different techniques by varying the number of predicted tags. It can be observed from the graph that as we are increasing the number of predicted tags, recall increases.

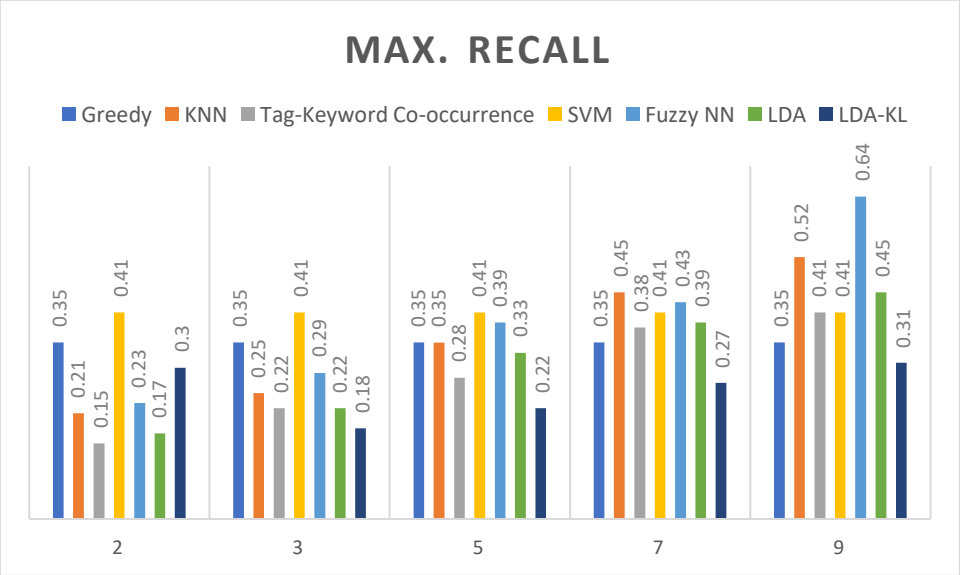

FIGURE 11: *MAX. RECALL OBTAINED AGAINST DIFFERENT TECHNIQUES FOR K = 2,3,5,7,9*

5.2 Data-Science

5.2.1. Greedy Method:

F-Score	Precision	Recall
0.23	0.22	0.30

TABLE 11: *RESULTS OF GREEDY METHOD FOR DATA-SCIENCE DATASET*

5.2.2. Nearest Neighbor Search:

The results obtained by varying the number of top most frequent tags are given below:

Top Frequent Tags	No. of Predicted Tags (k)	F-score	Precision	Recall
50	*2*	0.23	0.27	0.23
	3	0.24	0.23	0.29
	5	0.25	0.2	0.4
	7	0.24	0.17	0.47
	9	0.22	0.15	0.53
100	*2*	0.23	0.26	0.22
	3	0.23	0.22	0.28
	5	0.17	0.17	0.19
	7	0.23	0.16	0.46
	9	0.22	0.15	0.52
150	*2*	0.22	0.24	0.22
	3	0.23	0.22	0.28
	5	0.22	0.17	0.36
	7	0.22	0.15	0.43
	9	0.2	0.14	0.49
250	*2*	0.19	0.21	0.18
	3	0.2	0.19	0.24
	5	0.2	0.16	0.33
	7	0.19	0.14	0.39
	9	0.19	0.12	0.45

TABLE 12: *RESULTS OF NEAREST NEIGHBOR SEARCH FOR DATA-SCIENCE DATASET*

5.2.3. Tag-Keyword Co-Occurrence Model:

Since tuned parameter in this approach is number of predicted tags, so on varying it the obtained results are given below:

No. of Predicted Tags (k)	F-Score	Precision	Recall
2	0.28	0.29	0.26
3	0.31	0.27	0.35
5	0.30	0.23	0.46
7	0.28	0.19	0.54
9	0.27	0.17	0.61

TABLE 13: *RESULTS OF TAG-KEYWORD CO-OCCURRENCE MODEL FOR DATA-SCIENCE DATASET*

5.2.4. Support Vector Machine:

This method is evaluated by examining how many tags did it predict correctly for all queries. So, the evaluated values are given below:

F-Score	Precision	Recall
0.29	0.22	0.49

TABLE 14: *RESULTS OF SUPPORT VECTOR MACHINE FOR DATA-SCIENCE DATASET*

5.2.5 Fuzzy Nearest Neighbor Search:

This method is evaluated by varying the number of top most 10-20 tags predicted for each test post by Tag-Keyword Co-Occurrence Method.

Tags Picked from Co-Occurrence Matric	No. of Predicted Tags (k)	F-Score	Precision	Recall
10	2	0.34	0.37	0.31
	3	0.36	0.32	0.4
	5	0.34	0.26	0.52
	7	0.31	0.21	0.59
	9	0.27	0.18	0.63
20	2	0.32	0.35	0.29
	3	0.34	0.31	0.38
	5	0.34	0.25	0.51
	7	0.3	0.21	0.58
	9	0.28	0.18	0.64

TABLE 15: *RESULTS OF FUZZY NEAREST NEIGHBOR SEARCH FOR DATA-SCIENCE DATASET*

5.2.6. Latent Dirichlet Allocation-LDA:

In the basic implementation of LDA based approach, the tuned parameter against which the experiments are carried out is number of iterations. So the results obtained by varying the number of iterations are given below:

No. of Iterations	No. of Predicted Tags (k)	F-Score	Precision	Recall
50	*2*	0.15	0.17	0.14
	3	0.16	0.16	0.19
	5	0.17	0.13	0.26
	7	0.16	0.12	0.32
	9	0.16	0.1	0.37
500	*2*	0.15	0.18	0.15
	3	0.17	0.17	0.2
	5	0.18	0.14	0.28
	7	0.17	0.12	0.35
	9	0.17	0.11	0.39
1000	*2*	0.16	0.19	0.16
	3	0.17	0.16	0.19
	5	0.18	0.14	0.28
	7	0.18	0.12	0.35
	9	0.17	0.11	0.4

TABLE 16: *RESULTS OF LATENT DIRICHLET ALLOCATION FOR DATA-SCIENCE DATASET*

5.2.7. LDA using Kullback–Leibler divergence:

This method is evaluated by varying the number of topics. So the results obtained by varying the number of topics are given below:

No. of Topics	No. of Predicted Tags (k)	F-Score	Precision	Recall
Equal to number of tags - 162	*2*	0.11	0.07	0.3
	3	0.13	0.11	0.17
	5	0.12	0.09	0.22
	7	0.11	0.08	0.26
	9	0.11	0.07	0.3
500	*2*	0.12	0.13	0.12
	3	0.13	0.11	0.18
	5	0.12	0.09	0.22
	7	0.012	0.07	0.27
	9	0.11	0.07	0.31
800	*2*	0.11	0.11	0.12
	3	0.12	0.11	0.16
	5	0.11	0.08	0.21
	7	0.11	0.08	0.27
	9	0.1	0.07	0.29

TABLE 17: *RESULTS OF LDA USING KULLBACK-LEIBLER DIVERGENCE FOR DATA-SCIENCE DATASET*

5.2.8. Comparative Analysis of All Evaluation Measures Against All Techniques:

5.2.8.1. F-Score:

Following graph is the comparison of maximum F-Score obtained against different techniques by varying the number of predicted tags. It can be observed from the graph that results of Fuzzy Nearest Neighbor Search are relatively better as compared to other techniques for this medium size data-set.

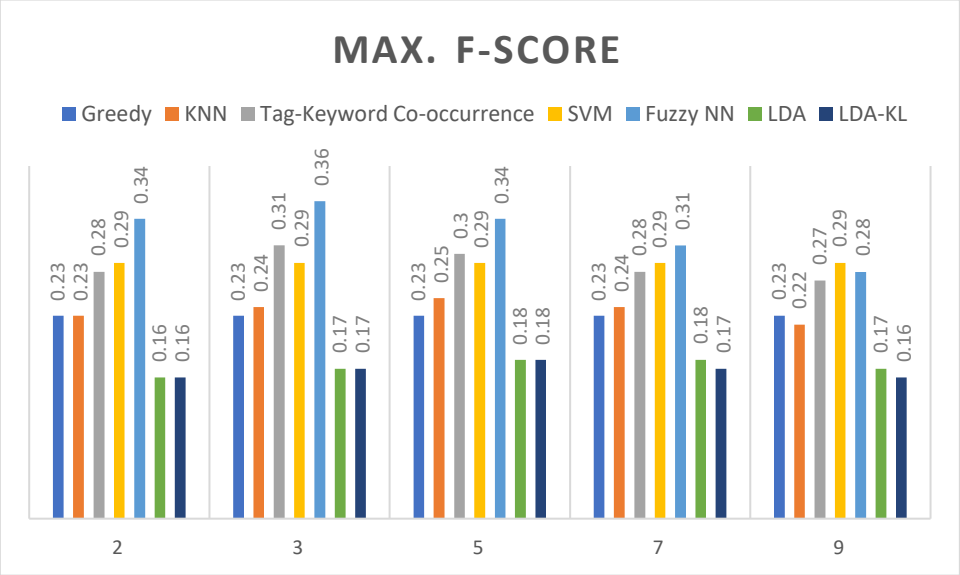

FIGURE 12: *MAX. F-SCORE OBTAINED AGAINST DIFFERENT TECHNIQUES FOR K = 2,3,5,7,9*

5.2.8.2. Precision:

Following graph is the comparison of maximum Precision obtained against different techniques by varying the number of predicted tags. It can be observed from the graph that as we are increasing the number of predicted tags, precision drops down.

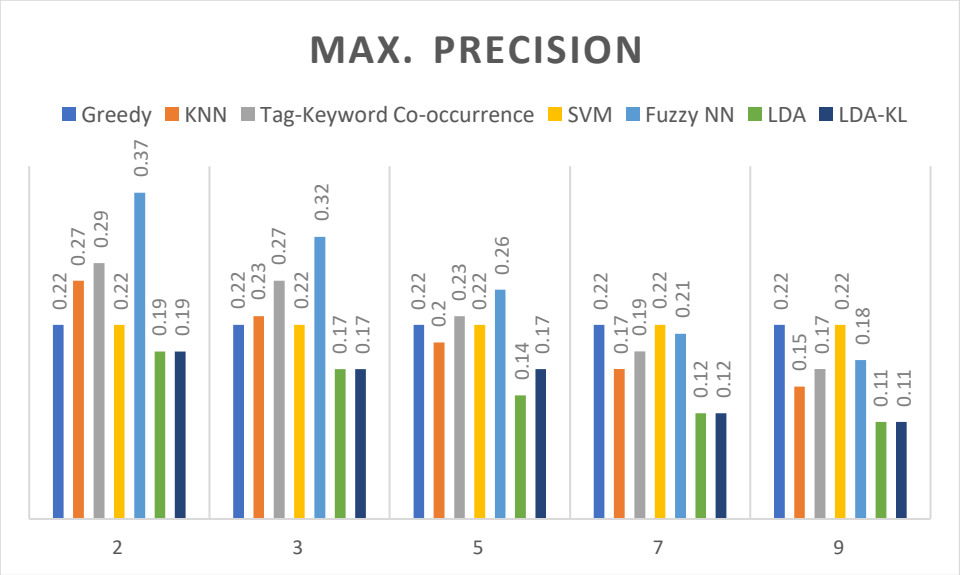

FIGURE 13: *MAX. PRECISION OBTAINED AGAINST DIFFERENT TECHNIQUES FOR K = 2,3,5,7,9*

5.2.8.3. Recall:

Following graph is the comparison of maximum Recall obtained against different techniques by varying the number of predicted tags. It can be observed from the graph that as we are increasing the number of predicted tags, recall increases.

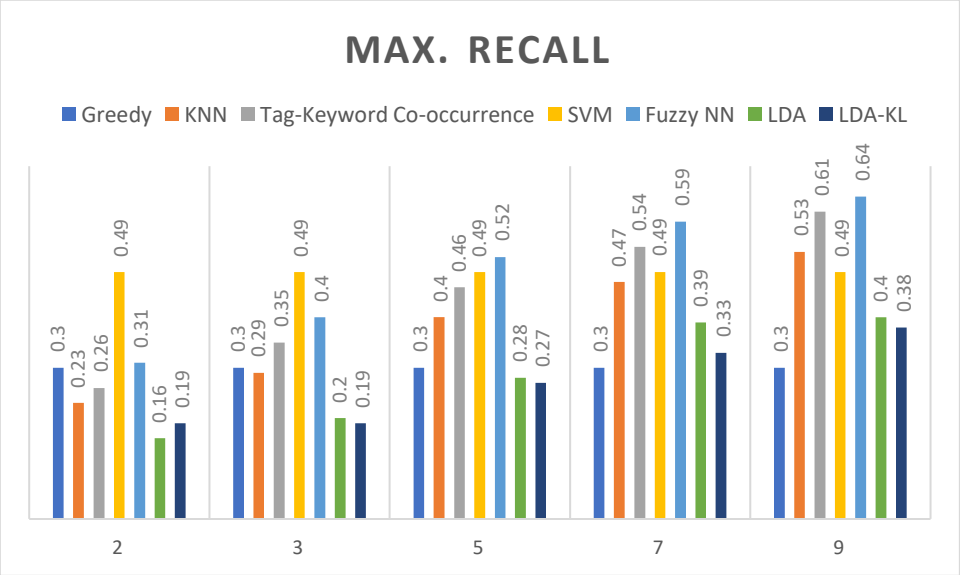

FIGURE 14: *MAX. RECALL OBTAINED AGAINST DIFFERENT TECHNIQUES FOR K = 2,3,5,7,9*

5.3 Physics

5.3.1. Greedy Method:

F-Score	Precision	Recall
0.16	0.18	0.19

TABLE 18: *RESULTS OF GREEDY METHOD FOR PHYSICS DATASET*

5.3.2. Nearest Neighbor Search:

The results obtained by varying the number of top most frequent tags are given below:

Top Frequent Tags	No. of Predicted Tags (k)	F-score	Precision	Recall
200	*2*	0.22	0.27	0.19
	3	0.23	0.23	0.25
	5	0.23	0.19	0.34
	7	0.22	0.16	0.4
	9	0.21	0.14	0.45
300	*2*	0.21	0.26	0.19
	3	0.22	0.23	0.24
	5	0.23	0.19	0.33
	7	0.22	0.16	0.39
	9	0.21	0.14	0.45
450	*2*	0.19	0.25	0.17
	3	0.21	0.22	0.23
	5	0.22	0.18	0.31
	7	0.21	0.15	0.38
	9	0.2	0.14	0.43
500	*2*	0.19	0.24	0.17
	3	0.21	0.21	0.23
	5	0.21	0.17	0.3
	7	0.2	0.15	0.37
	9	0.19	0.13	0.42

TABLE 19: *RESULTS OF NEAREST NEIGHBOR SEARCH FOR PHYSICS DATASET*

5.3.3. Tag-Keyword Co-Occurrence Model:

Since tuned parameter in this approach is number of predicted tags, so on varying it the obtained results are given below:

No. of Predicted Tags (k)	F-Score	Precision	Recall
2	0.32	0.37	0.26
3	0.34	0.32	0.35
5	0.33	0.25	0.46
7	0.3	0.21	0.54
9	0.28	0.18	0.61

TABLE 20: *RESULTS OF TAG-KEYWORD CO-OCCURRENCE MODEL PHYSICS DATASET*

5.3.4. Fuzzy Nearest Neighbor Search:

This method is evaluated by varying the number of top most 10-20 tags predicted for each test post by Tag-Keyword Co-Occurrence Method.

Tags Picked from Co-Occurrence Matric	No. of Predicted Tags (k)	F-Score	Precision	Recall
10	*2*	0.37	0.43	0.32
	3	0.38	0.36	0.4
	5	0.35	0.27	0.49
	7	0.32	0.22	0.56
	9	0.29	0.19	0.61
20	*2*	0.36	0.42	0.31
	3	0.37	0.35	0.39
	5	0.34	0.27	0.48
	7	0.31	0.22	0.54
	9	0.28	0.19	0.59

TABLE 21: *RESULTS OF FUZZY NEAREST NEIGHBOR SEARCH FOR PHYSICS DATASET*

5.3.5. LDA using Kullback–Leibler divergence:

This method is evaluated by varying the number of topics. So, the results obtained by varying the number of topics is given below:

No. of Topics	No. of Predicted Tags (k)	F-Score	Precision	Recall
500	*2*	0.1	0.12	0.08
	3	0.1	0.1	0.1
	5	0.11	0.09	0.15
	7	0.11	0.08	0.2
	9	0.1	0.07	0.23
Equal to number of tags -	*2*	0.09	0.11	0.07
	3	0.09	0.1	0.1
	5	0.1	0.08	0.15
	7	0.1	0.07	0.19
	9	0.1	0.07	0.22
2000	*2*	0.08	0.11	0.07
	3	0.09	0.09	0.1
	5	0.09	0.08	0.14
	7	0.09	0.07	0.18
	9	0.09	0.06	0.21

TABLE 22: *RESULTS OF LDA USING KULLBACK-LEIBLER DIVERGENCE FOR PHYSICS DATASET*

5.3.8. Comparative Analysis of All Evaluation Measures Against All Techniques:

Training of SVM and LDA based approach was taking considerable amount of time for medium to large-scale data-sets due to which the results of these techniques are not reported for this data-set. So, comparison of evaluation amongst the rest of techniques i.e. techniques except SVM and LDA.

5.3.8.1. F-Score:

Following graph is the comparison of maximum F-Score obtained against different techniques by varying the number of predicted tags. It can be observed from the graph that results of Fuzzy Nearest Neighbor Search are relatively better for large scale data-set.

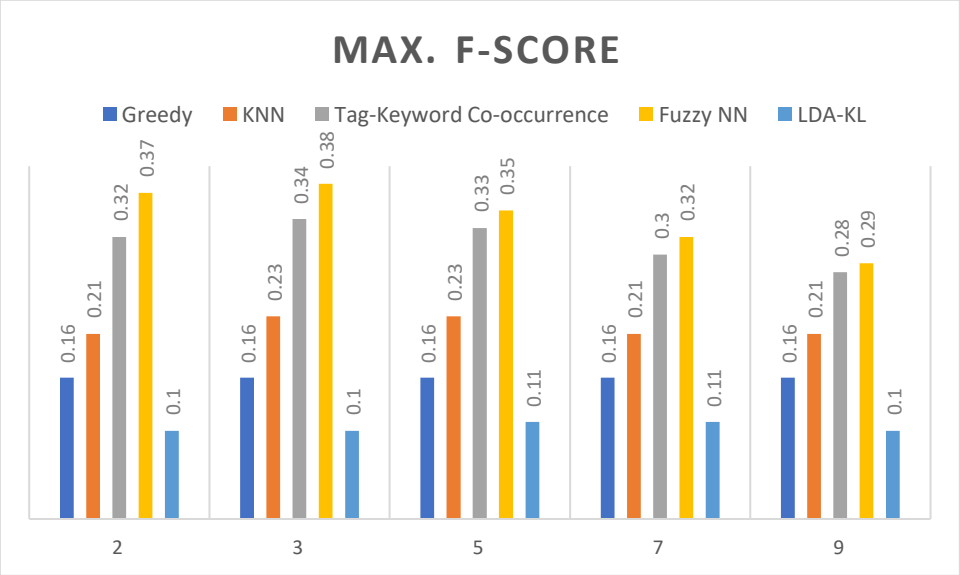

FIGURE 15: *MAX. F-SCORE OBTAINED AGAINST DIFFERENT TECHNIQUES FOR K = 2,3,5,7,9*

5.3.8.2. Precision:

Following graph is the comparison of maximum Precision obtained against different techniques by varying the number of predicted tags. It can be observed from the graph that as we are increasing the number of predicted tags, precision drops down.

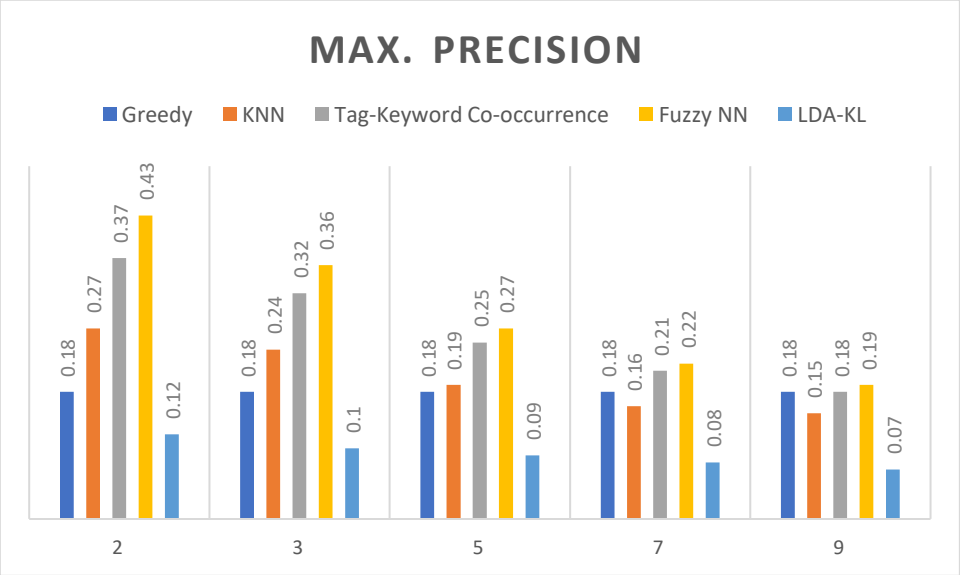

FIGURE 16: *MAX. PRECISION OBTAINED AGAINST DIFFERENT TECHNIQUES FOR K = 2,3,5,7,9*

5.3.8.3. Recall:

Following graph is the comparison of maximum Recall obtained against different techniques by varying the number of predicted tags. It can be observed from the graph that as we are increasing the number of predicted tags, recall increases.

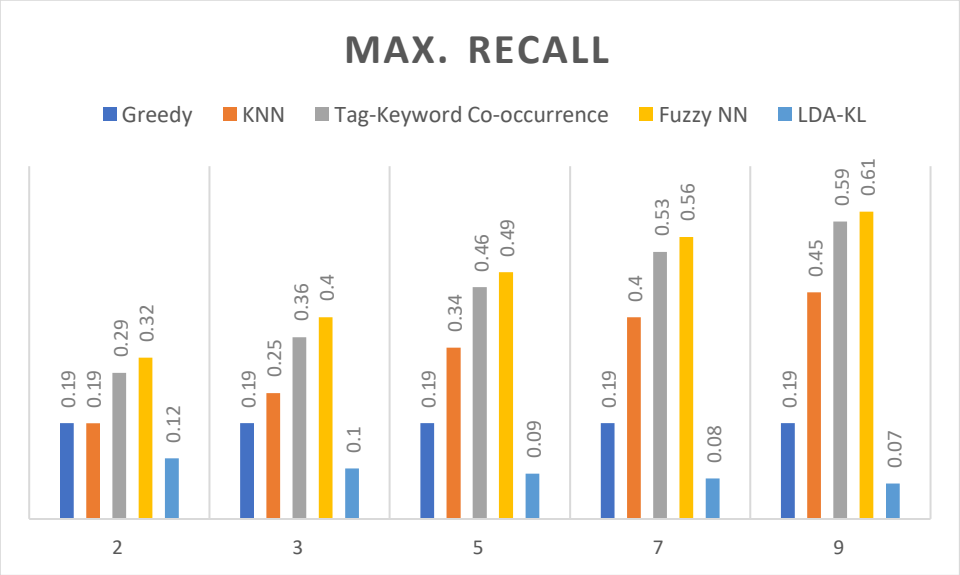

FIGURE 17: *MAX. RECALL OBTAINED AGAINST DIFFERENT TECHNIQUES FOR K = 2,3,5,7,9*

5.4 Apple

5.4.1. Greedy Method:

F-Score	Precision	Recall
0.34	0.29	0.52

TABLE 23: *RESULTS OF GREEDY METHOD FOR APPLE DATASET*

5.4.2. Nearest Neighbor Search:

The results obtained by varying the number of top most frequent tags are given below:

Top Frequent Tags	No. of Predicted Tags (k)	F-score	Precision	Recall
150	*2*	0.18	0.22	0.17
	3	0.19	0.19	0.21
	5	0.19	0.15	0.28
	7	0.18	0.13	0.33
	9	0.17	0.11	0.37
350	*2*	0.17	0.21	0.16
	3	0.18	0.1	0.2
	5	0.18	0.15	0.27
	7	0.17	0.12	0.32
	9	0.16	0.11	0.36
450	*2*	0.16	0.19	0.15
	3	0.17	0.17	0.19
	5	0.17	0.14	0.26
	7	0.15	0.11	0.27
	9	0.14	0.1	0.31
500	*2*	0.16	0.19	0.15
	3	0.17	0.17	0.19
	5	0.17	0.13	0.25
	7	0.16	0.11	0.29
	9	0.15	0.1	0.17

TABLE 24: *RESULTS OF NEAREST NEIGHBOR SEARCH FOR APPLE DATASET*

5.4.3. Tag-Keyword Co-Occurrence Model:

Since tuned parameter in this approach is number of predicted tags, so on varying it the obtained results are given below:

No. of Predicted Tags (k)	F-Score	Precision	Recall
2	0.38	0.43	0.34
3	0.4	0.38	0.44
5	0.39	0.3	0.55
7	0.35	0.24	0.63
9	0.31	0.2	0.67

TABLE 25: *RESULTS OF TAG-KEYWORD CO-OCCURRENCE MODEL FOR APPLE DATASET*

5.4.4. Fuzzy Nearest Neighbor Search:

This method is evaluated by varying the number of top most 10-20 tags predicted for each test post by Tag-Keyword Co-Occurrence Method.

Tags Picked from Co-Occurrence Matric	No. of Predicted Tags (k)	F-Score	Precision	Recall
10	*2*	0.39	0.45	0.35
	3	0.4	0.38	0.43
	5	0.38	0.29	0.54
	7	0.35	0.24	0.63
	9	0.32	0.21	0.68
20	*2*	0.39	0.45	0.34
	3	0.39	0.37	0.42
	5	0.36	0.28	0.52
	7	0.33	0.23	0.59
	9	0.3	0.19	0.65

TABLE 26: *RESULTS OF FUZZY NEAREST NEIGHBOR SEARCH FOR APPLE DATASET*

5.4.5. LDA using Kullback–Leibler divergence:

This method is evaluated by varying the number of topics. So the results obtained by varying the number of topics are given below:

No. of Topics	No. of Predicted Tags (k)	F-Score	Precision	Recall
800	*2*	0.12	0.15	0.11
	3	0.13	0.14	0.15
	5	0.14	0.11	0.2
	7	0.13	0.09	0.23
	9	0.12	0.08	0.26
Equal to number of tags - 1048	*2*	0.12	0.15	0.11
	3	0.13	0.14	0.15
	5	0.13	0.11	0.19
	7	0.12	0.09	0.23
	9	0.12	0.08	0.26
2000	*2*	0.11	0.13	0.09
	3	0.12	0.12	0.13
	5	0.12	0.09	0.18
	7	0.11	0.08	0.21
	9	0.11	0.07	0.24

TABLE 27: *RESULTS OF LDA USING KULLBACK-LEIBLER DIVERGENCE FOR APPLE DATASET*

5.4.8. Comparative Analysis of All Evaluation Measures Against All Techniques:

Training of SVM and LDA based approach was taking considerable amount of time for medium to large-scale data-sets due to which the results of these techniques are not reported for this data-set. So, comparison of evaluation amongst the rest of techniques i.e. techniques except SVM and LDA.

5.4.8.1. F-Score:

Following graph is the comparison of maximum F-Score obtained against different techniques by varying the number of predicted tags. It can be observed from the graph that results of Fuzzy Nearest Neighbor Search are considerably better.

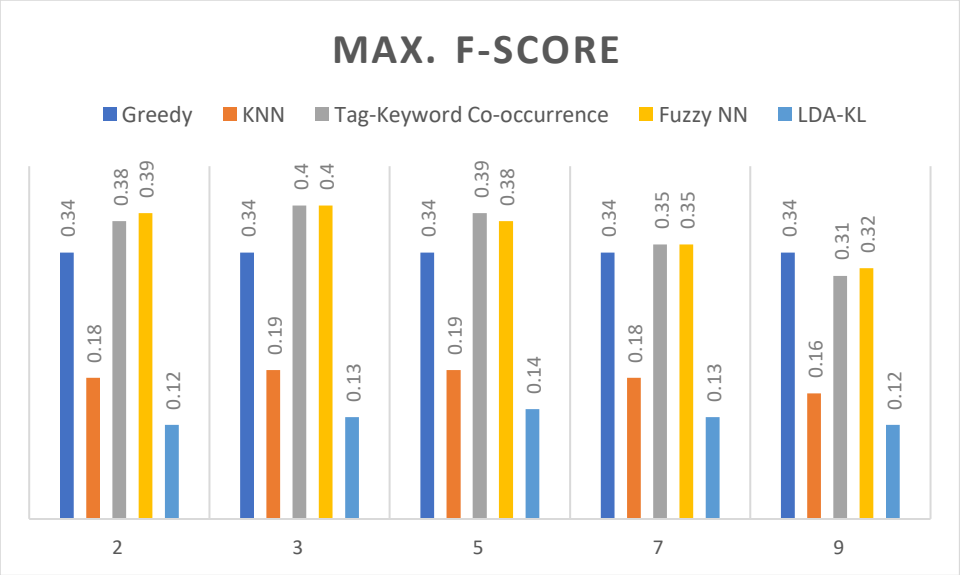

FIGURE 18: *MAX. F-SCORE OBTAINED AGAINST DIFFERENT TECHNIQUES FOR K = 2,3,5,7,9*

5.4.8.2. Precision:

Following graph is the comparison of maximum Precision obtained against different techniques by varying the number of predicted tags. It can be observed from the graph that as we are increasing the number of predicted tags, precision drops down.

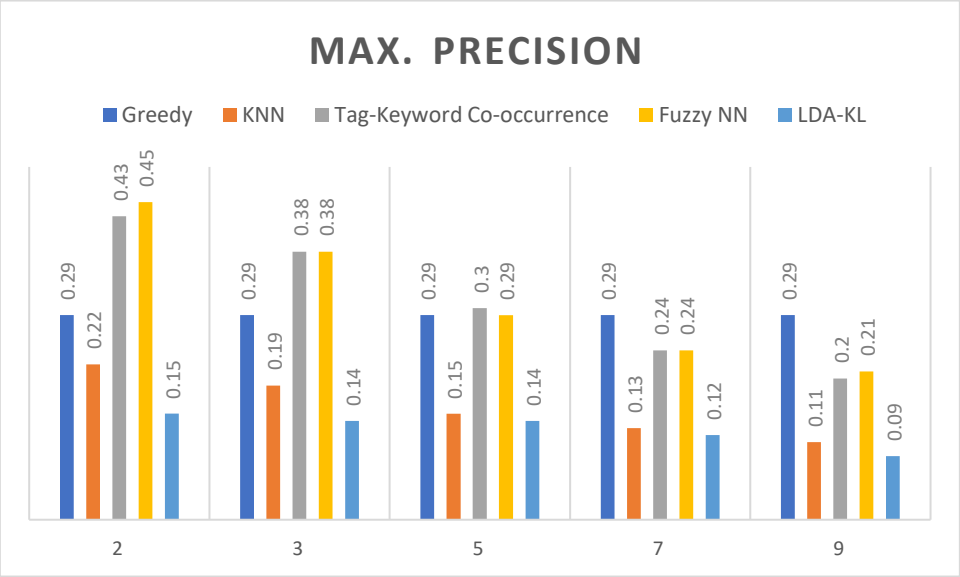

FIGURE 19: *MAX. PRECISION AGAINST DIFFERENT TECHNIQUES FOR K = 2,3,5,7,9*

5.4.8.3. Recall:

Following graph is the comparison of maximum Recall obtained against different techniques by varying the number of predicted tags. It can be observed from the graph that as we are increasing the number of predicted tags, recall increases.

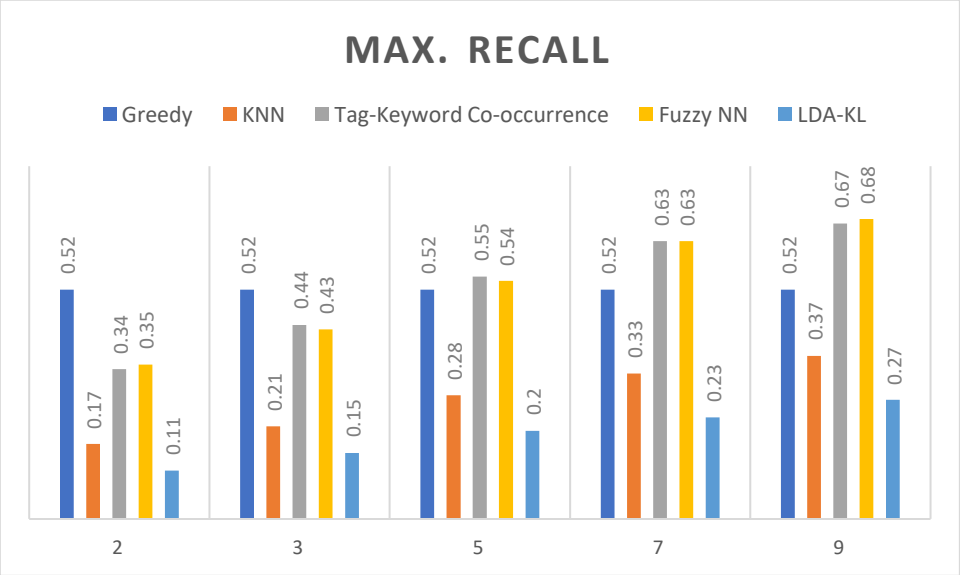

FIGURE 20: *MAX. RECALL OBTAINED AGAINST DIFFERENT TECHNIQUES FOR K = 2,3,5,7,9*

5.5 Analysis of Results

On the basis of above experiments, following observations are made:

- Greedy method is suitable for *smaller datasets*. The reason is that rest of the requires large training set as they are based supervised learning. So, if you have enough training set then you may get better results against these methods.
- On varying the number of *Predicted Tags (K)*, recall increases and precision drops.
- On varying the number of *Top Frequent Tags* in Nearest Neighbor Search Method, again the results almost remain the same. Intuitively recall should be increased on increasing the number of top frequent tags. But apparently, results don't show this behavior. So, we

can say that if you have total tags around 1000, then for predicting candidate tags, considering only the top 200-250 frequent tags will be reasonable. Because increasing the number of top frequent tags is only increasing the time but it is not improving the results.

- Latent Dirichlet Allocation (LDA) based techniques didn't come out to be fruitful for our considered datasets. This thing can be justified by one of the limitations of LDA mentioned in [54], that is LDA is not suitable for multi-labeled data-sets. In our data-set, we have multi-labeled posts. Also, LDA does not consider the correlation of words. If this would have been considered by LDA, then similar words would belong to same topics and we would have better topic distribution.
- In LDA using KL divergence (Topic Distance Based Approach), it is observed that on increasing the *Number of Topics*, F-Score gradually drops down. So, number of topics should be such that topic distribution should neither be too generic nor too specific. Precision and Recall show random behavior on varying the number of topics.
- On varying the number of *Iterations* in LDA, there is no significant improvement in the results. So, we should consider optimum number of iterations because increasing the number of iterations only leads to increased amount of time in training in case of our data-set.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

This research covers the implementation of various methods suggested for Tag Prediction. These include Greedy (Tag in Text) Method which is a baseline method, Nearest Neighbor Search Method, Tag-Keyword Co-occurrence Model, Using SVM for Tag Prediction, Fuzzy Nearest Search and two variations of Latent Dirichlet Allocation. Stack-Exchange dataset is used for experimental setup. The methods are then evaluated on the basis of three evaluation measures. These include Precision, Recall, F-Score. Although reasonable results are obtained against all methods except methods incorporating the use of LDA, but the technique that lead all other techniques for recommending tag for data-set like Stack-Exchange, is SVM in case of smaller data-sets and Fuzzy Nearest Neighbor Search in case of medium to large scale data-set. Since, we didn't work on Parallel Processing and training of SVM took much time so we couldn't test it for larger data-sets.

6.2 Future Work

The future work of this research can be the implementation of suggested techniques on Parallel Processing like Spark, Hadoop. Specially for the methods that involve training and testing phase. Like in SVM, when we move to bigger data-sets it's been above 12hours and still we didn't get a trained model. Same is the case with testing of LDA model. Also, Nearest Neighbor Search took

around 5-6 hours in producing results for larger data-sets. So, we can get sufficient improvement in time by moving these techniques over Parallel Computation.

References

[1] http://stackoverflow.com/help/tagging

[2] Golder, S. A. "Usage Patterns Of Collaborative Tagging Systems". *Journal of Information Science* 32.2 (2006): 198-208. Web.

[3] Gemmell, Jonathan, et al. "Personalization in folksonomies based on tag clustering." *Intelligent techniques for web personalization & recommender systems* 12 (2008).

[4] Shepitsen, Andriy, et al. "Personalized recommendation in social tagging systems using hierarchical clustering." *Proceedings of the 2008 ACM conference on Recommender systems*. ACM, 2008.

[5] Peters, Isabella. *Folksonomies: Indexing and retrieval in Web 2.0*. Vol. 1. Walter de Gruyter, 2009.

[6] Brooks, Christopher H., and Nancy Montanez. "Improved annotation of the blogosphere via autotagging and hierarchical clustering." *Proceedings of the 15th international conference on World Wide Web*. ACM, 2006.

[7] Vander Wal, Thomas. "Explaining and showing broad and narrow folksonomies. Blog post 2005-02-21." *Online: http://www. vanderwal. net/random/category. php* (2005).

[8] Symeonidis, Panagiotis, Alexandros Nanopoulos, and Yannis Manolopoulos. "Tag recommendations based on tensor dimensionality reduction." *Proceedings of the 2008 ACM conference on Recommender systems*. ACM, 2008.

[9] Lu, Caimei, Xiaohua Hu, and Jung-ran Park. "Exploiting the social tagging network for web clustering." *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 41.5 (2011): 840-852.

[10] Jelassi, Mohamed Nader, Sadok Ben Yahia, and Engelbert Mephu Nguifo. "A personalized recommender system based on users' information in folksonomies." *Proceedings of the 22nd International Conference on World Wide Web*. ACM, 2013.

[11] Gupta, Manish, et al. "Survey on social tagging techniques." *ACM Sigkdd Explorations Newsletter* 12.1 (2010): 58-72.

[12] Xu, Zhichen, et al. "Towards the semantic web: Collaborative tag suggestions." *Collaborative web tagging workshop at WWW2006, Edinburgh, Scotland*. 2006.

[13] Asanov, Daniar. "Algorithms and methods in recommender systems." *Berlin Institute of Technology, Berlin, Germany* (2011).

[14] Yin, Dawei, et al. "A probabilistic model for personalized tag prediction." *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2010.

[15] González, José R. Cedeño, et al. "Multi-class multi-tag classifier system for StackOverflow questions." *Power, Electronics and Computing (ROPEC), 2015 IEEE International Autumn Meeting on*. IEEE, 2015.

[16] Kim, Heung-Nam, et al. "Collaborative filtering based on collaborative tagging for enhancing the quality of recommendation." *Electronic Commerce Research and Applications* 9.1 (2010): 73-83.

[17] Ma, Hao, et al. "Recommender systems with social regularization." *Proceedings of the fourth ACM international conference on Web search and data mining*. ACM, 2011.

[18] Gemmell, Jonathan, et al. "Hybrid tag recommendation for social annotation systems." *Proceedings of the 19th ACM international conference on Information and knowledge management*. ACM, 2010.

[19] Merialdo, Arnd Kohrs-Bernard. "Clustering for collaborative filtering applications." *Intelligent Image Processing, Data Analysis & Information Retrieval* 3 (1999): 199.

[20] Liu, Nathan N., and Qiang Yang. "Eigenrank: a ranking-oriented approach to collaborative filtering." *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2008.

[21] J. Canny. "Collaborative filtering with privacy via factor analysis". *In Proc. of SIGIR '02, pages 238–245,*Tampere, Finland, 2002.

[22] Sigurbjörnsson, Börkur, and Roelof Van Zwol. "Flickr tag recommendation based on collective knowledge." *Proceedings of the 17th international conference on World Wide Web*. ACM, 2008.

[23] Strohmaier, Markus, Christian Körner, and Roman Kern. "Why do Users Tag? Detecting Users' Motivation for Tagging in Social Tagging Systems." *ICWSM*. 2010.

[24] Guan, Ziyu, et al. "Personalized tag recommendation using graph-based ranking on multi-type interrelated objects." *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2009.

[25] Quintarelli, Emanuele. "Folksonomies: power to the people. June 2005." *ISKO Italy-UniMIB meeting*. 2005.

[26] Stanley, Clayton. "Predicting Tags For Stackoverflow Posts". 2013

[27] Sood, S. C., Hammond, K. J., Owsley, S. H., & Birnbaum, L. (2007). TagAssist: Automatic tag suggestion for blog posts. In *ICWSM 2007 - International Conference on Weblogs and Social Media*

[28] Byde, Andrew, Hui Wan, and Steve Cayzer. "Personalized Tag Recommendations via Tagging and Content-based Similarity Metrics." (2007).

[29] Kim, Jin-Suk, et al. "Automatic in-text keyword tagging based on Information retrieval." *Journal of Information Processing Systems* 5.3 (2009): 159-166.

[30] Wang, Jian, and Brian D. Davison. "Explorations in tag suggestion and query expansion." *Proceedings of the 2008 ACM workshop on Search in social media*. ACM, 2008.

[31] McCallum, Andrew Kachites. "Bow: A toolkit for statistical language modeling, text retrieval, classification and clustering." *http://www. cs. cmu. edu/~ mccallum/bow* (1996).

[32] Sriharee, Gridaphat. "An ontology-based approach to auto-tagging articles." *Vietnam Journal of Computer Science* 2.2 (2015): 85-94.

[33] Rattanapanich, Rittipol, and Gridaphat Sriharee. "Auto-tagging articles using latent semantic indexing and ontology." *Asian Conference on Intelligent Information and Database Systems*. Springer International Publishing, 2014.

[34] Wu, Zhibiao, and Martha Palmer. "Verbs semantics and lexical selection." *Proceedings of the 32nd annual meeting on Association for Computational Linguistics*. Association for Computational Linguistics, 1994.

[35] Lu, Yu-Ta, et al. "A Content-Based Method to Enhance Tag Recommendation." *IJCAI*. Vol. 9. 2009.

[36] Chirita, Paul-Alexandru, et al. "P-tag: large scale automatic generation of personalized annotation tags for the web." *Proceedings of the 16th international conference on World Wide Web*. ACM, 2007.

[37] Anick, Peter G., and Suresh Tipirneni. "The paraphrase search assistant: terminological feedback for iterative information seeking." *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 1999.

[38] Schuster, Sebastian, Wanying Zhu, and Yiying Cheng. "Predicting tags for stackoverflow questions." (2013).

[39] Song, Yang, et al. "Real-time automatic tag recommendation." *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 2008.

[40] Garg, Nikhil, and Ingmar Weber. "Personalized, interactive tag recommendation for flickr." *Proceedings of the 2008 ACM conference on Recommender systems*. ACM, 2008.

[41] Mishne, Gilad. "Autotag: a collaborative approach to automated tag assignment for weblog posts." *Proceedings of the 15th international conference on World Wide Web*. ACM, 2006.

[42] Budura, Adriana, et al. "Neighborhood-based tag prediction." *European Semantic Web Conference*. Springer Berlin Heidelberg, 2009.

[43] Parikh, Chintan. "Identifying Tags from millions of text question." (2013).

[44] Saha, Avigit K., Ripon K. Saha, and Kevin A. Schneider. "A discriminative model approach for suggesting tags automatically for stack overflow questions." *Proceedings of the 10th Working Conference on Mining Software Repositories*. IEEE Press, 2013.

[45] Boudaer, Glenn, and Johan Loeckx. "Enriching Topic Modelling with Users' Histories for Improving Tag Prediction in Q&A Systems." *Proceedings of the 25th International Conference Companion on World Wide Web*. International World Wide Web Conferences Steering Committee, 2016.

[46] Blei, David M., Andrew Y. Ng, and Michael I. Jordan. "Latent dirichlet allocation." *Journal of machine Learning research* 3. Jan (2003): 993-1022.

[47] Wetzker, Robert, et al. "I tag, you tag: translating tags for advanced user models." *Proceedings of the third ACM international conference on Web search and data mining*. ACM, 2010.

[48] Blei, David M., Andrew Y. Ng, and Michael I. Jordan. "Latent dirichlet allocation." *Journal of machine Learning research*3.Jan (2003): 993-1022

[49] Wu, Yong, et al. "Tag2word: Using tags to generate words for content based tag recommendation." *Proceedings of the 25th ACM international on conference on information and knowledge management*. ACM, 2016Wu, Yong, et al. "Tag2word: Using tags to generate words for content based tag recommendation." *Proceedings of the 25th ACM international on conference on information and knowledge management*. ACM, 2016

[50] Krestel, Ralf, and Peter Fankhauser. "Tag recommendation using probabilistic topic models." *ECML PKDD Discovery Challenge* 2009 (2009): 131.

[51] Hong, James, and Michael Fang. "Keyword extraction and semantic tag prediction (2013).

[52] Choubey, Rahul. "Tag recommendation using Latent Dirichlet Allocation." PhD diss., Kansas State University, 2011.

[53] https://archive.org/download/stackexchange

[54] Ramage, Daniel, et al. "Labeled LDA: A supervised topic model for credit attribution in multi-labeled corpora." *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1-Volume 1*. Association for Computational Linguistics, 2009.