

TAG GENERALIZATION FOR FACET-BASED SEARCH

by

Tomasz Niewiarowski

Submitted in partial fulfillment of the
requirements for the degree of
Master of Computer Science

at

Dalhousie University
Halifax, Nova Scotia
August 2013

© Copyright by Tomasz Niewiarowski, 2013

Table of Contents

List of Tables	iv
List of Figures	v
Abstract	vi
Acknowledgements	vii
Chapter 1 Introduction	1
1.1 Tagging in Enterprise Content Management Systems	2
1.2 Wikipedia Tag Generalization Approach	4
Chapter 2 Background and Related Work	6
2.1 Tag Recommendation	6
2.2 Tag Ontologies	8
2.3 Wikipedia Based Tag Related methods	9
2.3.1 Building Ontology with Wikipedia	11
Chapter 3 Wikipedia	13
3.1 Preprocessing	16
3.2 General Wikipedia Statistics	17
3.3 Using Wikipedia for Semantic Relatedness	19
3.3.1 Link Based Similarity Measure	23
3.3.2 N-gram Based Similarity	24
Chapter 4 Tag Generalization System	25
4.1 Overview	25
4.2 Tag Disambiguation Module	28
4.2.1 Mapping Tags on Anchor Links	29
4.2.2 Candidate Similarity	33
4.2.3 Selection of the Most Similar Candidates	33
4.3 Link Based Tag Generalization	34
4.3.1 Key Wikipedia Assumption	36

4.3.2	Building a Graph of Related Articles	36
4.3.3	Ranking Articles with the Generality Metric	39
4.4	Concept Unification	43
4.4.1	Closed Dictionary System	44
4.4.2	Open Dictionary System	45
4.4.3	Mixed Approach	45
4.5	Caching and Efficiency	46
Chapter 5	Experiments and System Evaluation	47
5.1	Datasets	48
5.1.1	StackOverflow	48
5.1.2	Delicious	48
5.1.3	PubMed	49
5.1.4	Wordpress	49
5.1.5	Selection of General Tags	49
5.2	Experiment Scenarios	51
5.2.1	Mapping Tags into Wikipedia Articles	51
5.2.2	Overall Performance of the Algorithm	53
5.2.3	The Influence of Tag Frequency	54
5.3	Baseline Methods	54
5.4	Experimental Results	56
5.4.1	Tag into Wikipedia Article Mapping	56
5.4.2	Overall Performance	57
5.4.3	Tests on Infrequent Tags	61
5.5	Demo application	62
Chapter 6	Conclusions and Future Work	66
Bibliography	68

List of Tables

Table 3.1	Number of Wikipedia articles by language.	14
Table 3.2	Wikipedia Statistic	18
Table 3.3	Most popular Wikipedia articles	18
Table 3.4	Wikipedia articles with highest number of links	20
Table 3.5	PageRank value for Wikipedia articles	20
Table 3.6	Top 10 the most ambiguous anchor terms	21
Table 3.7	Top 10 Wikipedia articles with the highest synonyms number .	21
Table 4.1	Example of mapping an anchor to Wikipedia article	32
Table 4.2	Influence of the n most popular meanings for a given anchor. .	32
Table 4.3	Median of number of tags assigned to each post in datasets . .	33
Table 5.1	The most popular tags in Datasets	50
Table 5.2	Dataset Statistics	50
Table 5.3	Set of general tags	52
Table 5.4	Training and testing dataset	54
Table 5.5	Coverage of datasets by Wikipedia	57
Table 5.6	Examples of misclassification	60
Table 5.7	Examples of general tags	64

List of Figures

Figure 1.1	An Example of a tag cloud	1
Figure 1.2	Oris4 - An example of <i>ECM</i> (Enterprise Content Management) system	3
Figure 3.1	Wikipedia growth over the time	15
Figure 3.2	Histogram of incoming links for English Wikipedia articles . .	19
Figure 3.3	Similarity measure between Automobile and Global Warming based on Wikipedia links	23
Figure 4.1	System overview	26
Figure 4.2	Example of a tagged document	27
Figure 4.3	Java Wikipedia disambiguation page example.	30
Figure 4.4	The system disambiguation example.	31
Figure 4.5	An example of tag disambiguation.	35
Figure 4.6	Visualization of Algorithm 1	39
Figure 4.7	Wikipedia concept map	40
Figure 4.8	Wikipedia focused concept map.	41
Figure 4.9	An example calculation of the generality value	42
Figure 5.1	The baseline tag recommendation system scheme	55
Figure 5.2	Overall performance	58
Figure 5.3	An impact of generality level and number of assigned tags . .	59
Figure 5.4	Experiment results without popular tags	61
Figure 5.5	Demo application	63
Figure 5.6	Demo tag cloud	65

Abstract

In this project we address over-specification of tags, a common problem of modern tag-based document management systems. In such systems tags are essential for the document retrieval task. The accuracy of this process depends mainly on the “human factor” i.e. the quality of tags assigned by users. While tagging, users are likely to pick only very specific tags that describe the content of a resource, forgetting about general concepts that represent the resource. Our proposed method to deal with this problem is an automatic tag generalization algorithm which assigns general tags to newly tagged resources. The objective of the algorithm is to provide a layer of tags consisting of general concepts and to provide good support for a system user. The proposed method automatically tags resources with more general and similar tags to user-assigned tags. The method is unsupervised and domain independent. The proposed tag generalization method consists of three stages: (1) the disambiguation and concept mapping stage maps specific tags to Wikipedia articles representing the same concept; (2) link based tag generalization is meant to find similar and more general articles using the Wikipedia link structure; (3) the concept unification stage where the system assigns tags based on the list of general articles. Evaluation on four real-life tag data sets demonstrates that the proposed method is domain independent and outperforms supervised tag recommendation systems for practical training data set sizes.

Acknowledgements

This thesis would not have been possible without the guidance and help of several people. Foremost, I would like to express my gratitude to my advisor Dr. Marek Lipczak for his patience, guidance, encouragement and advice provided through my thesis. I could not have imagined a better advisor and mentor. I would also like to thank my supervisors Dr. Evangelos Milios and Dr. Vlado Keselj for their support and advice. They were always ready to help me and answer my questions. I would also like to thank Dr. Stan Matwin for his comments and challenging questions during my thesis defense.

My Master's study was supported by the Natural Sciences and Engineering Research Council of Canada and the Boeing Company. My thanks and appreciation also go to all my labmates for the two years of working together.

Finally, thanks to my family for their support, without them none of that would be possible.

Chapter 1

Introduction

A *tag* in information systems is a type of *matadata* which is meant to describe various kinds of resources (posts, images, movies) [46]. This description can be obtained in various ways. The most popular are based on the user annotation or results of some automatic tag recommendation techniques.

Tags are a simple but very powerful method to organize content. The main purpose of tags is to improve searching and browsing of accumulated content on the web. One of the most popular techniques is to present *tag clouds* which are usually a set of the most popular tags in the system (Fig. 1.1). Tags are critical especially for multimedia objects like images, videos and music. They provide meaningful descriptors and allow users to organize their content.

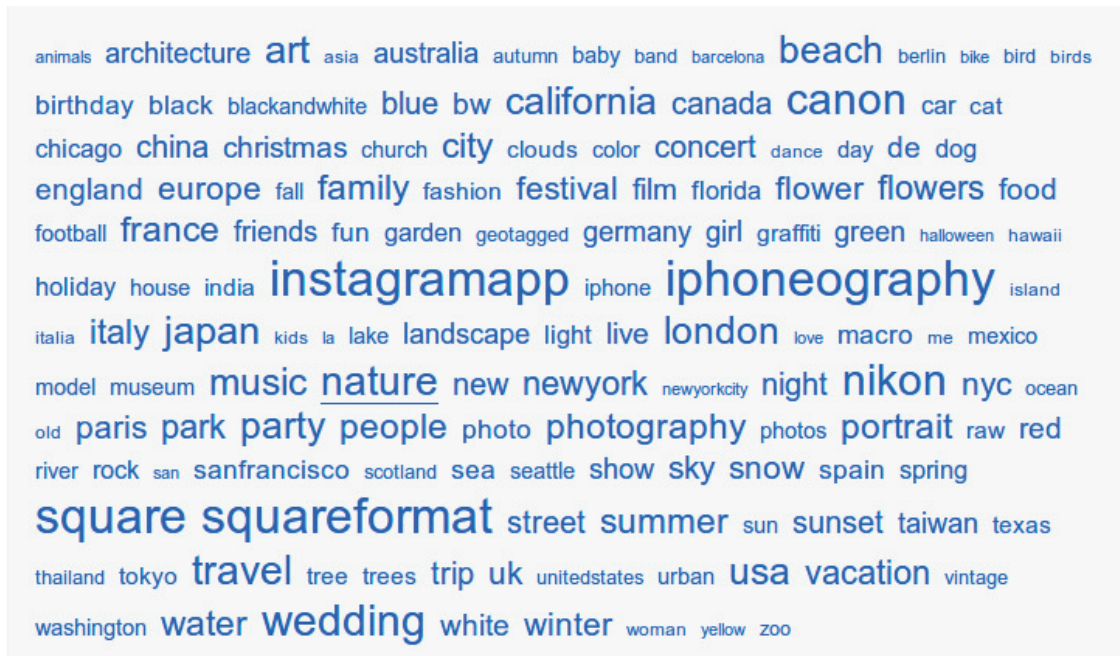


Figure 1.1: An Example of a tag cloud. Tag clouds are one of the most popular techniques to improve browse and search of the online content. Usually they present the most popular tags in the system. [Source: Flickr.com]

There are many online systems which take advantage of the tagging approach. Well known sites like: Flickr¹ or StackOverflow² organize their resources with the help of tags. Both sites are examples of collaborative tagging systems. In *Flickr* users assign tags to their images and in *StackOverflow* users tag their posts. Both systems contain millions of images/questions which we generally refer to as *posts*. Tagging systems in these two examples have the same purpose. They help users to search and browse content. Sometimes it is the only way to effectively browse the content i.e. tags are the main approach to retrieve an image in Flickr system. The popularity of the tagging systems across the Web suggests that it is one of the best methods to retrieve images or videos. That is the main reason why tags become even more popular. As we can see nowadays the most popular Internet social media like *twitter* or *facebook* are introducing new tagging systems. In Twitter users can define specific *hashtags* they want to follow so they will be presented only with the relevant content for them.

1.1 Tagging in Enterprise Content Management Systems

The usage of collaborative tagging is not limited only to big online communities. This technique is very useful and frequently used even by small groups of people who work together. *ECM (Enterprise Content Management)* systems help manage document flow inside the company. Nowadays every company works with digital documents. Many documents are produced every day and some of them are the collaborative work of a group of people. Companies are willing to invest heavily in *ECM* systems to improve the work effectiveness and the quality of the document flow inside the company.

Oris4 (Fig. 1.2), developed by 2nd Act Innovations, is an advanced Enterprise Content Management system. Its objective is to assist employees in managing the document flow in the company. All resources (i.e. documents, e-mails and web-articles) relevant to the company are stored in the system and can be later retrieved. One of the main retrieval approaches used in Oris4 is faceted search based on tags generated by system users and, in the future, automatic tagging algorithms. With

¹<http://flickr.com>

²<http://stackoverflow.com>

faceted search users are presented a list of tags that have been assigned to resources and by picking interesting tags they limit the set of retrieved resources to the ones containing the selected tags.

The screenshot displays the Oris4 interface. At the top, there are tabs for 'Content' and 'Connections', a '2 Tasks' notification, and a user profile for 'Leslie Flemming, Administrator'. The main area is divided into three sections:

- Left Panel:** Contains a search bar, 'Document Type' filters (PDF, DOC, PPT, XLS), 'Collaborators' (user avatars), and a 'More Facets' section with folders for 'My Files', 'Shared Files', 'Dropbox', 'Box', and 'Google Drive'.
- Center Panel:** Lists search results for '715 Results'. Each entry includes a document icon, title, 'Date Modified: 2012-12-21', and 'Uploaded By: Leslie Flemming'. The 'Key Features' document is highlighted.
- Right Panel:** Shows 'Document Details' for the selected 'Key Features' document. It includes a PDF icon, 'Key Features' title, a description: 'We connect all of your information sources - including Salesforce, Box, Google Drive, Gmail, email and shared drives - to allow you to...', 'Uploaded By: Leslie Flemming', 'Date Created: 2012-12-11', 'Folder: Intern', and a set of action icons (email, edit, delete, download, upload, lock). Below this are expandable sections for 'Comments (0)', 'Tags (0)', 'Connections (6)', 'Shares (5)', and 'Versions (1)'. A vertical 'Feedback' button is on the far right.

Figure 1.2: Oris4 - An example of *ECM* (Enterprise Content Management) system, developed by 2nd Act Innovations. It helps employees in managing the document flow in the company. Users can assign tags to all documents and e-mails gathered in the system. They help to organize and browse the gathered content. The key problem in this approach is over specification of tags. Users tend to forget to assign general tags what later can produce sparse dictionaries of tags. It results in the low quality tag cloud, which covers only small subset of documents in the system.

The key problem of tag-based faceted search is over-specification of tags. While tagging, users are likely to pick only very specific tags that describe the content of a resource, forgetting about general concepts that represent the resource. As a result, the system contains a verbose dictionary of tags, each of which is used for a relatively small number of resources. Such tags have low usability as search facets because they do not provide the user a general overview of concepts which could be an easy starting point of the search process (drilling down from general to

specific concepts). To overcome this limitation the company needs an automatic tag generalization algorithm which would assign general tags to newly tagged resources. The objective of the algorithm is to provide a layer of tags consisting of general concepts that can be later used at the beginning of faceted search process.

One of the main limitations of *ECM* systems is the size of the document corpus they cover. Most of the available solutions which aim to solve the problem of sparse dictionaries of tags are based on the supervised algorithms. This means that each supervised system has to be trained to be effective in the tag generalization process. There are two main conditions for a good training process: (1) training set should consist of documents from the same domain as the final system, and (2) amount of training documents should be relatively high. In the perfect conditions supervised tag recommendation method would have to be trained separately for each company on their own data. However, this approach has serious drawback which is the amount of documents in the system. Due to privacy issues, companies are unwilling to share their data with others even for training purposes. The number of documents in *ECM* systems is a few orders of magnitude lower than typical online community systems like Flickr and StackOverflow. Therefore available tag generalization methods which work on the big data sets will not work in the smaller systems.

1.2 Wikipedia Tag Generalization Approach

The main contribution of the thesis is an automatic tag generalization system which overcomes these limitations. We propose a *Wikipedia based tag generalization method* to generalize tags in a system. The main advantage of the proposed system is that it is using Wikipedia as an external source of knowledge about tags. The method is unsupervised and domain independent. Thanks to Wikipedia the proposed method is applicable to a broad range of *ECM* systems and it is independent of the amount of data in the system so even small *ECM* systems can take full advantage of using this tag generalization technique.

The system presented in this thesis consists of 3 modules: (1) tag disambiguation, (2) concept generalization, and (3) tag unification module. The process starts with the set of tags assigned to a given post. In the first module we map tags on the corresponding Wikipedia concepts. The meaning of tags can be ambiguous, therefore

this module assigns the most likely definition within Wikipedia based on the semantic relatedness of the tags. The second module is responsible for creating a ranked list of more general concepts. The list consists of concepts from a graph of Wikipedia articles, which is build based on Wikipedia’s hyperlink structure. The last module is responsible for determination of the most appropriate tags based on the the ranked list of general concepts. Wikipedia titles usually consist of more than one word. However, one-word tags are preferred i.e. “Java (programming language)” and “Java (software platform)” for both of the articles the most appropriate tag would be “java”.

As an output the system produces a new set of general tags generated individually for each post. This additional layer of general tags will be very helpful in exploring the document set.

Experimental results show that our system is better than other tag generalization methods for systems with limited amount of training data like *ECM* systems.

Chapter 2

Background and Related Work

In this chapter we review previous research in related fields. To the best of our knowledge, there is a limited research on the tag generalization and we could not find any previous research which would attempt to solve the problem of tag over-specification. However, a number of research studies have been performed on related research areas like tag recommendation, tag classification, ontology extraction from Wikipedia and Wordnet.

First, we would like to present research related to tag recommendation. After that we would like to discuss semantic relatedness of concepts. In the proposed method we calculate semantic relatedness to disambiguate tags as well as for tag generalization. Semantic relatedness underlies many natural language processing tasks like disambiguation, information retrieval and classification. Another related field of the research is tag classification and extraction of the tag ontology. Prior work in this field mostly focused on classifying tags to predefined categories [4, 32, 35] or creating an ontology based on tag co-occurrence [34]. At the end we will discuss Wikipedia related systems. Recently the idea of using Wikipedia has become very popular. Many systems try to use Wikipedia to improve classification or to create an ontology.

2.1 Tag Recommendation

The fast growing popularity of online communities like *StackOverflow*, *Flickr* or *Delicious* has drawn attention to the problem of tagging online resources. One of the key components of online services is a tag recommendation method. Together with the popularity of online communities it becomes a very active field of research. Tags are especially helpful with organizing online content on social collaborative systems. Tags provide a meaningful overview of the resources, while at the same time help users to organise their content. Sometimes tags are necessary especially to organize multimedia content like photos, music, videos or any non-textual content with no

description. A good tag recommendation system can improve not only managing the digital content but also the interaction with the system.

We can distinguish two major categories in the tag recommendation research. The first class of methods focuses on the content of the resources like titles, descriptions or other metadata. The second class is based on the structure of users, resources, and tags. We can also distinguish systems which take advantage of both approaches.

An example of a popular tag recommendation approach might be system presented in [35]. This method is based on the tag co-occurrence. Methods can identify tags that are frequently occurring together. We can also look at tag recommendation from the perspective of a classification problem. The method [12] builds classifier based on the training data to predict correct tags. The method presented in [1] recommends tags based on the association rules extracted from the Wordnet taxonomy. Another approach presented in [15] proposed a graph-based recommendation system which is based on the modified version of PageRank called FolkRank.

Methods like [21, 30] focus on recommending tags based on the previous user behavior. They collect information about the user behaviors and activities and based on that they can predict that similar users use similar tags. The method presented in [21] is a complex recommendation system, which combines five different modules taking advantage of content, tag co-occurrence, user habits and resource profile. The method is discussed in detail in Section 5.3. Authors of [9] propose a method based on reinforcement learning which takes into account user interests. Lipczak et al. [20] presented a method based on three different tag sources, resource content, resource and user profiles, where resource and user profiles are based on tags assigned for similar resources and previous tags assigned by user.

We believe that a good tag recommendation system can improve greatly the quality of tags in ECM systems. However, we should not require from the user high quality of tags assigned to the content. Tag generalization systems should automatically create an additional layer of assigned general tags. Our objective is a system that is transparent for a user but also benefiting from the external knowledge like Wikipedia.

2.2 Tag Ontologies

Another way to improve the tag experience in online collaboration systems is to use a tag ontology. Tagging is one of main methods to retrieve content in systems like Delicious or Flickr. Introduction of hierarchy in these systems would improve retrieval rate and faceted search of a content. One of the methods was presented by Sigurbjornsson et al. [35] where they mapped Flickr tags to Wordnet categories. Additional layer of Wordnet categories allowed to visualize what categories people are interested in. They found out that more than half of the tags from Flickr can be mapped to Wordnet concepts by simple text matching.

Garcia-Silva et al. [8] distinguished three main categories of methods to enrich folksonomies with semantics. The first category is based on clustering. The clustering of tags creates groups of topics based on inner characteristics of dataset and by using statistical NLP techniques. The second category is based on ontologies which might be taken from external source of knowledge like Wordnet or Wikipedia. The third category consists of a combination of the previous two.

Wordnet is a very popular ontology reference. Djuana et al. [4] proposed a method to create a tag ontology from user tagging behavior and Wordnet lemmas. They claim that user-defined tags provide rich information about item hierarchical classification. They showed that the additional ontology improves tag recommendation. Another solution was proposed by Zhy et al. [52]. They used Wordnet to generate additional atomic entities for tags. Based on the additional entities they were able to teach the system to assign similar tags. However Wordnet has some drawbacks. It is available only in English and it is not updated as fast as Wikipedia. In our system we focused on extracting relationships from Wikipedia which covers many more concepts, especially recent and domain-specific ones. O. Medelyan et al. presented an interesting method to create taxonomy from the set of documents [24]. First they extract named entities from text using NLP tools. Later they map entities to external sources of knowledge like: Freebase, DBpedia and Wikipedia. Later they disambiguate the concepts and at base on categories from external sources they create their taxonomy for the set of documents.

Statistical approach to create an ontology from Flickr tags was presented by Schmitz [34]. The author focused on tag co-occurrence, based on which he created

a graph of possible parent-child relationships. We believe that such approach will not work with our problem. It is a supervised method which needs a lot of training data to deduce a basic ontology. In our opinion external sources like Wikipedia or Wordnet can provide more reliable relationships between concepts.

2.3 Wikipedia Based Tag Related methods

In the literature we can find two main sources of the external knowledge for tag recommendation systems, Wordnet or Wikipedia. In this section we would like to discuss a few examples of Wikipedia based methods.

One of the popular tasks based on Wikipedia is the process of *wikification* [27, 29, 44]. Milne et al. [29] describes how to automatically cross-reference documents with Wikipedia. In this paper they present how to enrich any text with hyperlinks to the Wikipedia concepts. Their method is based on a machine learning approach to identify the most important concepts within the text. They have evaluated this system based on existing Wikipedia articles achieving high accuracy. In their disambiguation task they defined two concepts, *Commonness* and *Relatedness*, where commonness measures the popularity of concept within Wikipedia and relatedness is based on the surrounding content of the probable link. Cucerzan [3] presented a large-scale system for entity recognition and semantic disambiguation based on models extracted from Wikipedia. Disambiguation was performed based on the text context, however text content is not always available. In our system we disambiguate tags based on the tags assigned together and the popularity of concept within Wikipedia.

Mihalcea [26] describes a method for building sense tagged corpora using Wikipedia. They are doing it in a few steps including using anchors and article titles to map them into Wordnet definitions. In their word sense disambiguation process they use machine learning and part-of-speech tagging for a context of an ambiguous word and a Wordnet definition. The method is accurate, however at the same time it is very computationally inefficient. In their approach, they focus on the text context of the ambiguous words.

Wikipedia might be also used to improve classification. Wang et al. [42] presented a method for classification of the text documents. They extract concepts from text, match them with Wikipedia articles. After that they enrich *BOW* (Bag Of Words)

profiles of documents with profiles extracted from matched Wikipedia articles. Finally, they classify documents based on the enriched *BOW* profiles. A Huang et al. presented a method to measure similarity between documents [13]. The method takes into account lexical and semantic properties of documents. In First stage it extracts Bag-of-Words and Bag-of-Concepts based on Wikipedia for each document. In the second stage they calculate the features value and later the system learns from human judgment how to combine these features to compute document similarity.

Another frequent problem which Wikipedia is helpful to solve is computing a semantic relationship between concepts. Semantic relatedness indicates how related two concepts are based on their meaning. This similarity measure is frequently used to solve problems in computational linguistics like concept disambiguation, clustering and information retrieval. In our work we are using it also for the tag generalization task.

In the last few years we have seen many different approaches to computing semantic relatedness. Most of the previous work which deal with concept similarity was based on Wordnet [5]. Wordnet is a well structured taxonomy dataset. However it does not cover a wide range of special domains. Nowadays we can observe a shift to using Wikipedia as a ground truth of the relations. Although, Wikipedia is not as well structured as Wordnet, it has a few advantages over Wordnet. Wikipedia is much larger than Wordnet, it covers many more concepts, it has a large and active contributor community so even recent events are very well covered.

Ponzetto et al. [36] presented a method called Wikirelate! which adapts NLP techniques to compute relatedness based on Wikipedia. First they matched concepts with titles that contain these words. Later based on the text of articles they compute a word based similarity of these concepts. They showed also that Wikipedia outperforms WordNet in a task of computing semantic relationship. Another approach was presented by Gabrilovich et al. [7]. They introduce a concept of *Explicit Semantic Analysis (ESA)* to Wikipedia. Their solution is similar to the vector space model technique popular in the field of information retrieval. In their approach Wikipedia articles are presented as a matrix where words are columns of *tfidf* values across Wikipedia. This method obtains very high accuracy, however, at the same time it is very computational inefficient.

In our project to disambiguate tags and to compute semantic relatedness we have adapted the approach presented by Milne et al. [28]. Authors described an efficient way to obtain semantic relatedness. They use Wikipedia to decide how closely related two concepts are. They do that by using the hyperlink structure of Wikipedia. Their approach is based on the Google distance. It is a very efficient and accurate measure which uses Wikipedia. The method is described in detail in Section 3.3.1

2.3.1 Building Ontology with Wikipedia

Previous research has shown that Wikipedia can be a good source of information. Overell et al. [32] proposed a method for classifying tags using Wikipedia and Wordnet into semantic categories. First they map tags on Wikipedia articles using anchor texts. Later based on the categories of the retrieved Wikipedia articles they classify them into a few predefined categories from Wordnet. They use Wikipedia as the hierarchical corpus and WordNet as the lexicon of classified terms. Wordnet categories can provide a good high level overview of tags. They are using Wikipedia categories templates to classify them into Wordnet categories. A similar approach was presented by Subramaniaswamy et al. [37]. In their method they extract concepts from Wikipedia and based on Wordnet they create the ontology. Later they map extracted topics from textual content of blog to Wikipedia concepts to recommend tags.

Another category of systems includes DBpedia [18], Yago2 [38] or WikiNet [31]. They extract structured content from Wikipedia. They parse information from Wikipedia tables and templates to extract properties for concepts. The extracted information is then available for everyone to query in an SQL-like language. Such systems might be a great source for ontology. However they do not fulfill our needs. The concepts in DBpedia are too well defined and they are too precise. We can easily find properties for some company like date of founding or who is the CEO or what was the revenue for the previous year. However, it is difficult to design a general method which will work with every type of entity with a huge variety of different properties. Moreover, we focus more on the important related concepts, not only on the pure hierarchy. The hierarchy of concepts within DBpedia is taken from Wikipedia. We have done some preliminary experiments which showed that including category concepts from Wikipedia did not increase our accuracy.

An example of another system based on Wikipedia is DBpedia Spotlight [25]. It is a system for enriching a text with DBpedia resources. First DBpedia Spotlight detects and extracts entities from the text, after that maps it to DBpedia resources and disambiguates them. At the end it provides text with important concepts linked to DBpedia.

As we showed Wikipedia is becoming very popular as a source of information. It is popular in tasks like tag recommendation, disambiguation and creating an ontology. In our project we focus on the extraction of general concepts for a given tag. We are not limiting the system to provide only categories but also related concepts which are important for this topic.

Chapter 3

Wikipedia

Wikipedia is a free online encyclopedia. It is one of the biggest collaborative projects engaging millions of people. The English Wikipedia only, has more than 19 millions of registered users and more than 124 thousands users were active within last month.

Wikipedia was launched on January 15, 2001, by Jimmy Wales and Larry Sanger. Soon it became popular and it started to grow very fast (Fig. 3.1). Nowadays Wikipedia has over 30 million articles in 286 languages [47] and this number is increasing every day, including over 4.2 million articles in English. Table 3.1 presents the top 10 languages of Wikipedia. As we can see the English version is more than twice as big as next one in the list. It is surprising that the order of language does not depend on the number of people who speaks this language.

Due to high quality of articles and amount of covered information Wikipedia is one of the most popular websites in the Internet. It is ranked seventh most popular web page in the Internet by *Alexa*¹. It has around 365 million readers around the world and receives more than 2.7 billion page views from the USA every month only.

Over the years Wikipedia has been questioned regarding the quality of its articles. The academic society has many times criticized Wikipedia as a unreliable source and frequently banned it as a resource [14]. On the other hand studies have found that Wikipedia quality is comparable to other traditional encyclopedias [10] especially for popular articles.

Collaborative work of millions of volunteers has positive effect on the quantity as well as on the quality of articles. Many studies [17, 22] focused on examining the impact of adding more contributors to articles and how it improves the overall quality. They also present importance of different coordination techniques to ensure article quality. Wikipedia supports several ways to ensure high quality of encyclopedia articles. Every article is under a version control system similar to systems used

¹<http://www.alexa.com/siteinfo/wikipedia.org>

Language	articles
English	4,259,682
Dutch	1,616,803
German	1,597,096
French	1,397,617
Italian	1,040,993
Spanish	1,023,569
Swedish	1,019,673
Russian	1,015,896
Polish	972,955
Japanese	862,409

Table 3.1: Number of Wikipedia articles by language. *[Source: http://meta.wikimedia.org/wiki/List_of_Wikipedias]*

by developers to manage their code. Each change of an article is saved and easily accessible by a history tab where it can be restored if needed. Moreover, there is also a group of users with special additional permissions. They can block spamming users, revert, monitor and protect articles against any changes. Additionally every change in the article can be discussed by users to reach agreement on the final version of an article. In the end, quality of Wikipedia depends on its users. Frequently people are afraid about vandalism of articles. However, any instance of it is usually fixed within minutes [40]. Due to its smart policies and a large number of editors Wikipedia can grow and ensure high quality. We can simply say that the more popular it becomes, the better it gets.

One of the most important features of Wikipedia aside from its online availability and size is the amount of connections between articles. Internet links guide users to pages which are relevant to the concept mentioned in an article. They are helpful in the navigation between articles and they can greatly improve the learning process. Links are also indicators that two articles connected by link are somehow related to each other.

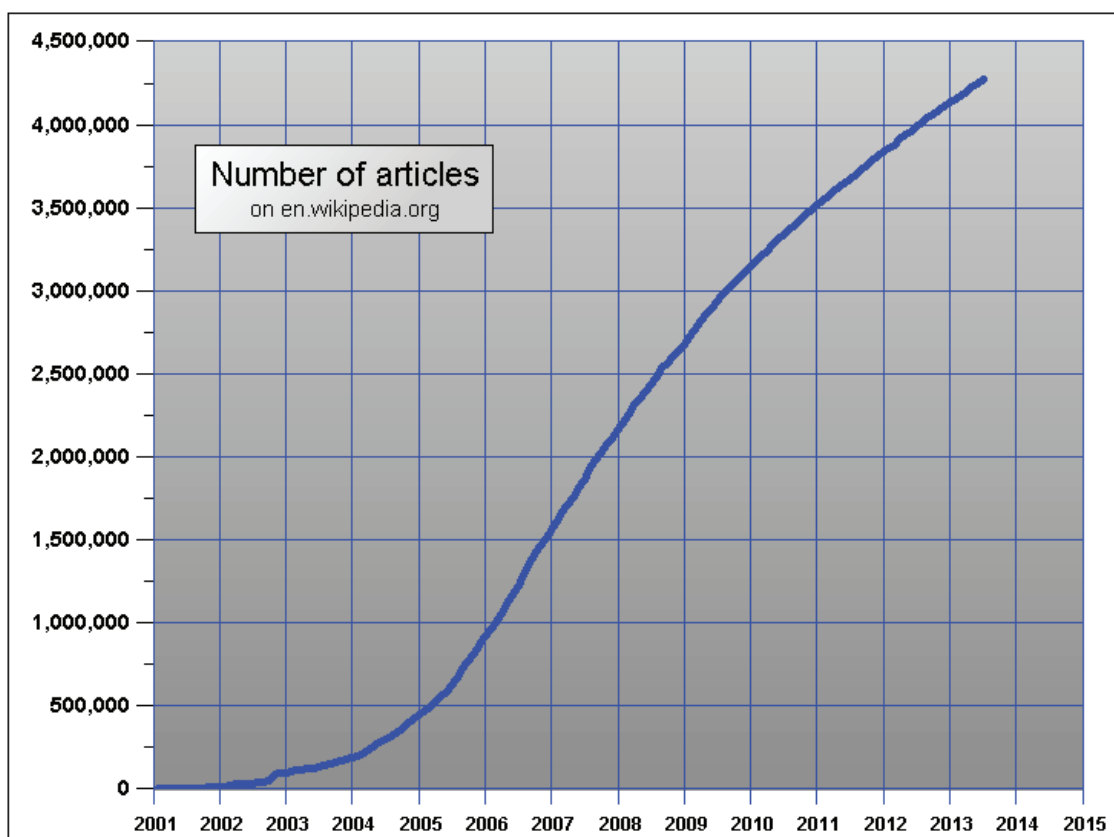


Figure 3.1: English Wikipedia growth over the time. Since Wikipedia launch in 2001 it is constantly growing. We can observe that around 2005 the popularity of Wikipedia exploded resulting in very fast growth. Recently even though the number of Wikipedia users is increasing, the growth of Wikipedia is slowing down. It is most likely that most concepts are already covered. *source: www.en.wikipedia.org/wiki/Wikipedia:Statistics*

3.1 Preprocessing

Wikipedia as an open source projects allows to download its entire content. It is easily available online² in many different versions and formats so researchers can download the most applicable format for their needs. The Wikipedia content is available in the form of XML database dumps which are released periodically, usually every month.

The version used in this study was released in July 2012. At this point Wikipedia contained around 4 million articles. We have downloaded a data set relevant to our needs. This version included full articles with categories, titles and links. It did not include information like editing history, or internal user discussions and statics. The downloaded version was approximately 8.2GB of compressed data, which corresponds to almost 40GB of uncompressed text. The dump consists of XML files. The XML format used by Wikipedia is well documented, however it is rather complex to process. We have decided to use WikiPrep [6, 7] to parse Wikipedia's format of XML. Wikiprep resolves many dependencies and greatly simplifies the structure of the dump files.

In the next step, we had to extract all articles from the dump files. We have created a database where we stored information about each article. We extracted information such as:

- For article:
 - title
 - article body
 - categories
 - hyperlinks
- For each hyperlink:
 - source and destination article
 - position in the article
 - order number of the link on the page
 - anchor text

²<http://dumps.wikimedia.org/>

- paragraph number where the link is
- link context, 3 words before and after or to the end of sentence

After importing the Wikipedia content we have cleaned it. We removed all of the artificial articles like disambiguation pages or pages for multimedia content because all of the images in the Wikipedia are presented as a separate articles. Additionally, we limited the number of links by removing all links pointing to images and saving only the first occurrence of the link in the article.

Finally, as an result we obtained dataset which consist of: 3.8 million articles and 81.5 million hyperlinks. We also mined over 6.8 million distinct anchors. Anchors are words which are linked to other articles. In our system we are using anchors to define the vocabulary of terms by which concepts can be accessed e.g. article “Barack Obama” is frequently linked by: “president obama”, “obama, barack”, “senator obama”, “obama”.

To store this information we use the PostgreSQL database system. The final database size was quite large at 75 GB mostly because of large indexes and few cache tables to improve performance. More detail on caching and efficiency is given in Section 4.5.

3.2 General Wikipedia Statistics

In this section we present general statistics of Wikipedia to demonstrate its size. Researchers, sometimes are not aware of how challenging Wikipedia may be to work with. Basic statistics are presented in Table 3.2.

Wikipedia articles are very densely connected. As we can see on average an article has 21 links. Surprisingly only 16 thousand articles are without any outgoing link. The diameter of Wikipedia is very low, on average it takes only 3.45 steps to go from one article to any other. There are many different ways to measure popularity of articles within Wikipedia. Table 3.3 presents a list of articles with the highest number of incoming links. A histogram of articles and number of incoming links is shown in Figure 3.2.

There are many “hub” articles which consist mostly of links to other articles. An example of such articles might be any “List of...” article like “List of towns

articles	3,808,192
links	81,452,864
articles with links	3,792,188
articles with incoming links	2,320,212
avg. links per article	21
articles without links	16,004
articles without incoming links	1,487,980
diameter of Wikipedia	3.45
number of unique anchors	6,803,966

Table 3.2: General statistics of the version of Wikipedia from July 2012 used in our experiments

article	incoming links
United States	249,671
geographic coordinate system	121,600
association football	112,669
international standard book number	92,932
France	84,616
England	80,501
Germany	71,363
village	70,732
United Kingdom	67,824
India	60,151

Table 3.3: The most popular Wikipedia articles according to hyperlink structure of Wikipedia

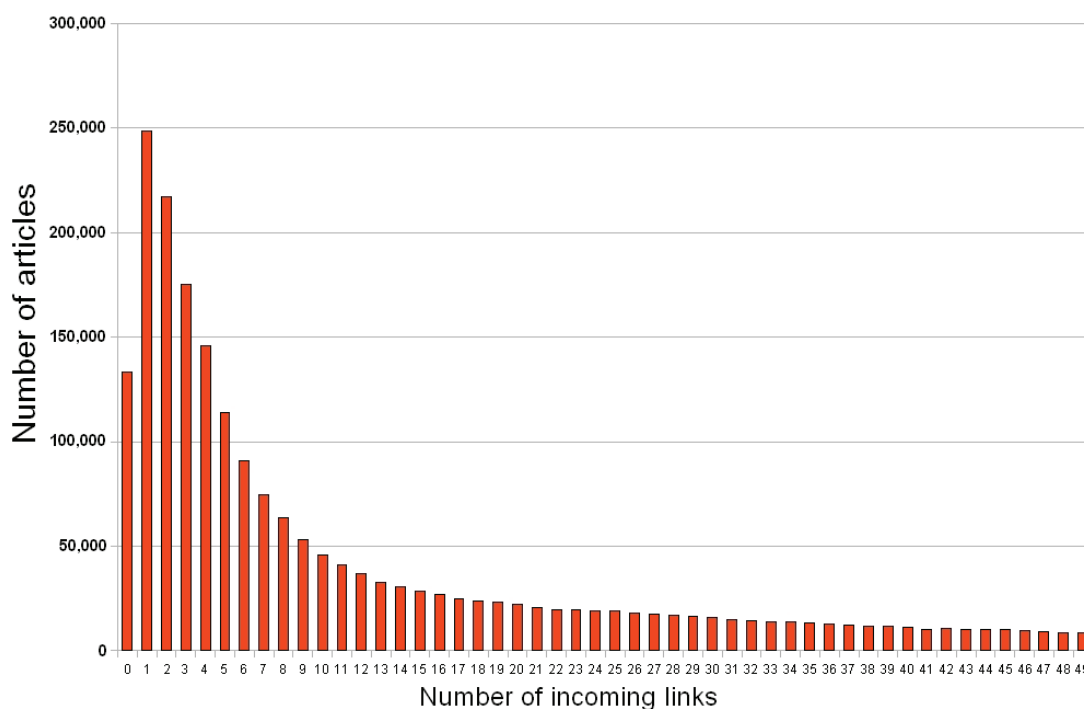


Figure 3.2: Histogram of incoming links for English Wikipedia articles *source: <http://commons.wikimedia.org/>*

in Ontario”. In our project we need to disregard such articles. Usually they can introduce a lot of noise to our results.

To find the most important articles in Wikipedia we have decided to adapt well known method *PageRank* to weight nodes in hyperlinked set of documents. PageRank is one of the most popular algorithms to measure the importance of web page in the Internet. Table 3.5 presents top 10 articles with the highest PageRank value. We used Hadoop cluster to calculate PageRank value for every node in the graph. Unfortunately, experiment showed that PageRank value is not suitable for our needs. It depends mostly on the popularity of the article across Wikipedia ignoring important but not popular concepts.

3.3 Using Wikipedia for Semantic Relatedness

In our work we use the information extracted from Wikipedia to calculate the semantic relatedness of concepts represented by Wikipedia articles. We need to decide for example how closely related are concept like “Automobile” and “global warming”.

article	outgoing links
list of Russian people	8,103
list of film score composers	5,174
list of people from Texas	4,793
index of philosophy articles (rz)	4,697
index of philosophy articles (iq)	4,652
index of u.s. counties	4,620
list of dialling codes in Germany	4,455
list of Asian birds	4,375
IUCN red list endangered animal species	4,291
list of composers by name	3,993

Table 3.4: Wikipedia articles with highest number of links

#	title
1	geographic coordinate system
2	United States
3	village
4	association football
5	moth
6	communes of France
7	genus
8	France
9	Germany
10	England

Table 3.5: PageRank value for Wikipedia articles

word	number of possible meanings
details	17,087
men	5,364
women	3,944
logo	3,818
report	2,922
list of episodes	2,636
singles	2,366
discography	2,332
state leaders	2132
history	1,868

Table 3.6: Top 10 most ambiguous anchor terms in Wikipedia

article	number of possible synonyms
professional wrestling attacks	863
professional wrestling match types	612
glossary of professional wrestling terms	594
United States	558
professional wrestling throws	522
professional wrestling aerial techniques	500
glossary of cue sports terms	496
digimon	479
hangul	465
billboard charts	440

Table 3.7: Top 10 Wikipedia articles with the highest synonyms number. The list is based on anchors from incoming links (skipped meaningless articles like: “postal codes in France” where we can find over 6 thousands distinct anchor text usually numbers)

This is a well known problem of semantic relatedness. This task can be solved by many different automatic approaches proposed in the literature [11, 33, 51]. Automatic approaches are usually based on some external source of knowledge like taxonomies and thesauri. Usually the quality of such a system depends mostly and on the size and quality of the taxonomy used. An example of such background knowledge might be Wordnet³ or Wikipedia, which are frequently used by researchers. Wordnet is a great source of relations between different concepts and words. It was a very popular source of knowledge. However nowadays we can observe that Wikipedia is taking its place in computing semantic relatedness. Strube et al. [33] in 2007 were the first ones who used Wikipedia for this task. Their approach was based on previous methods used with Wordnet. The idea of using Wikipedia became very popular and soon Gábrilovich and Markovitch [7] proposed their solution. They proposed a method based on *ESA Explicit Semantic Analysis* of Wikipedia where the words are represented in columns of a *tfidf* matrix similar to the vector space model technique which is popular in the field of information retrieval. A. Kennedy et al. presented a method to compute semantic relatedness of words across different languages [16]. They assume that words related in English will be also related in other languages. Their method focuses on a set of known translations of the context words. They used POS tagger and focused only on nouns and later based on the context words co-occurrence they computed relatedness between two concepts from different languages. A completely different method was presented by R. West et al. [43]. They presented a method called “Wikispeedia”, a method for computing a semantic distance between concepts based on an online game. Players need to find the shortest path between two Wikipedia concepts by clicking links. It is a novel approach which adapts crowdsourcing to compute semantic relatedness. Another more practical approach to compute semantic relatedness was presented by Milne et al. [28]. The accuracy of the proposed method is slightly lower than *ESA*, however the solution is much simpler and faster, and there is no need for the extensive precomputation required by *ESA*. The following section presents this method in detail.

³<http://wordnet.princeton.edu/>

3.3.1 Link Based Similarity Measure

The method proposed by Milne et al. is based on the hyperlink structure of Wikipedia. A link between two concepts indicates a meaningful connection, i.e. it is most likely that these concepts are relevant to each other. Wikipedia in its structure embeds millions of such connections. The proposed relatedness metric can be considered as an adaptation of Google distance [2] to the Wikipedia graph structure.

Google distance together with Wikipedia allowed them to measure semantic relatedness in a very efficient way. Lets consider the example of 2 words: Automobile and global warming. As we can see in the Figure 3.3 these two concepts are mentioned together in many articles. The large number of overlapping articles suggests high relatedness of the concepts.

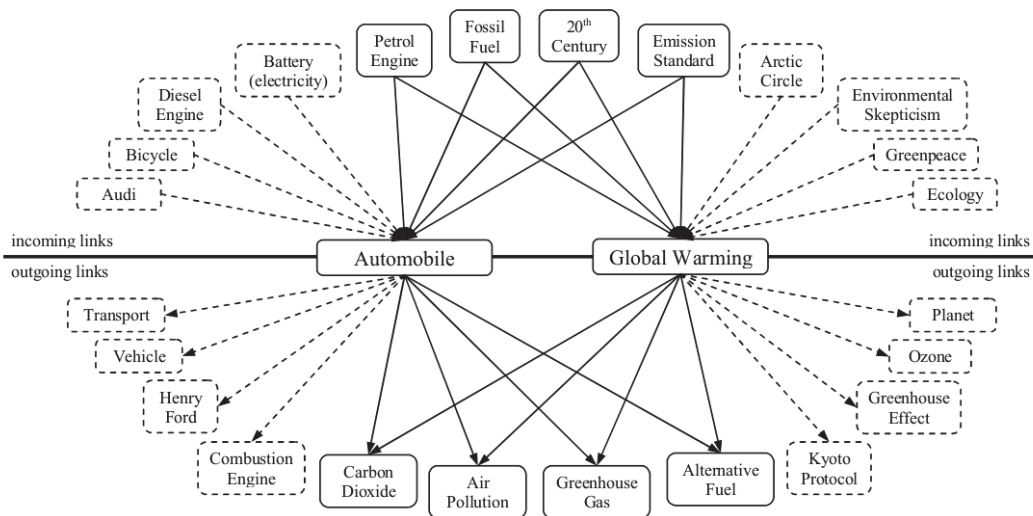


Figure 3.3: Similarity measure between Automobile and Global Warming based on Wikipedia links *source*: [28]

The equation 3.1 is the Google distance adapted to Wikipedia terminology and is calculated as follows: a and b are two Wikipedia concepts. A and B are sets of all articles which are linking respectively to concept a and b . W is a set of all the articles in the Wikipedia

$$sr(a, b) = \frac{\log(\max(|A|, |B|)) - \log(|A \cap B|)}{\log(|W|) - \log(\min(|A|, |B|))} \quad (3.1)$$

As we can see this measure is fairly simple and efficient. To improve the performance we can simply cache sets of incoming links for each article. Value $|W|$ is a constant and there is no need to compute it. The results of this method are very good and comparable to judgment of experts.

3.3.2 N-gram Based Similarity

While calculating the similarity between Wikipedia articles we can disregard the link structure and use text-based relatedness measures. In our experiments we used *CNG* distance measure [41]. This measure is based on character *N-grams* which are a sequence of N letters from the text. Usually the most common *N-grams* consist of 3 to 6 letters, we call them respectively *trigrams* and *six-grams*.

The frequencies of a set of N-grams from the text creates the N-gram profile of the text. Based on the text profiles we can compute similarity of text. Usually similar text will consist of similar amount of the same N-grams.

CNG [41] (Common N-gram) similarity measure 3.2 compares the N-gram profiles of 2 documents. Where P_1 and P_2 are documents, $f_{P_i(x)}$ is the normalized frequency of an N-gram x in profile P_i . The similarity measure ranges from 0 to 1 and the lower the value the more similar the documents are.

$$d_x(P_1, P_2) = \left(\frac{f_{P_1}(x) - f_{P_2}(x)}{\frac{f_{P_1}(x) + f_{P_2}(x)}{2}} \right)^2 \quad (3.2)$$

To create N-gram profile we are using the context of all of the links leading to Wikipedia concept. Using a combined link neighborhood of all the links leading to the concept allows us to take advantage of word context.

Chapter 4

Tag Generalization System

4.1 Overview

The objective of the project is the design and implementation of a tag generalization algorithm, which assigns additional general tags to an already tagged resource, based on the resource content and community-based knowledge.

Wikipedia articles create a rich graph of connections between concepts represented as *wiki links*. Wiki links are the internal Wikipedia hyperlinks that are used in wiki systems to explain a non-trivial concept, by providing a reference to an article that describes that concept. The proposed method maps the tag vocabulary from any tag-based system into the concept graph (Def. 2). The concept graph is then used to extract the concepts which will be a good generalization of the document tags.

The proposed method can be divided into 3 main phases: (1) disambiguation and concept mapping, (2) link based tag generalization, and (3) concept unification (Fig. 4.1). The first phase involves the task of connecting a tag with a proper Wikipedia article (e.g., assigning tag 'java' to an article about programming language or the island depending on the context). We are addressing this concept disambiguation problem by considering the context of tags that co-occur with the mapped tag and their semantic relatedness based on the Wikipedia link structure. The second phase is meant to find similar and more general tags. It is based on various textual features (e.g., general links are more likely to be used at the beginning of an article) and graph based features (e.g., centrality of the concept in the Wikipedia graph). The generalization procedure is applied to all tags assigned to the document by searching for a common ancestor among concepts from Wikipedia. The third phase unifies the concepts extracted from Wikipedia. It determines the most common synonyms of the extracted concepts to improve the quality of the proposed tags. Moreover this step takes into account the characteristics of a tagging system. It can recommend new tags or use tags which are already in the system (to reduce sparseness of tag dictionary)

based on the concepts extracted previously.

The output of the system is a ranked list of general tags generated individually for each post. Assigning general tags for each document will improve search and browsing techniques. An example of an image from *Flickr* is shown in Figure 4.2. The image presents The Empire State Building. User has tagged it with “Empire State Building”. This picture will be visible only for people who search exactly for that specific tag. Our method will assign additional tags like “building”, “USA”, “New York”, “Manhattan” which will significantly improve retrieval rate of this image.

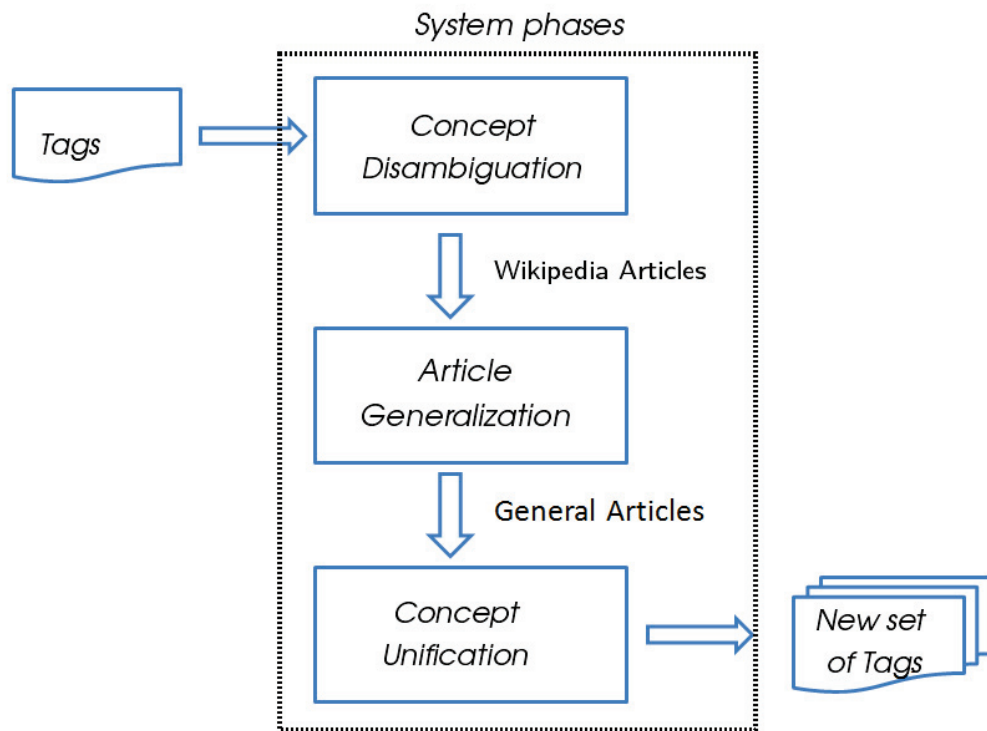
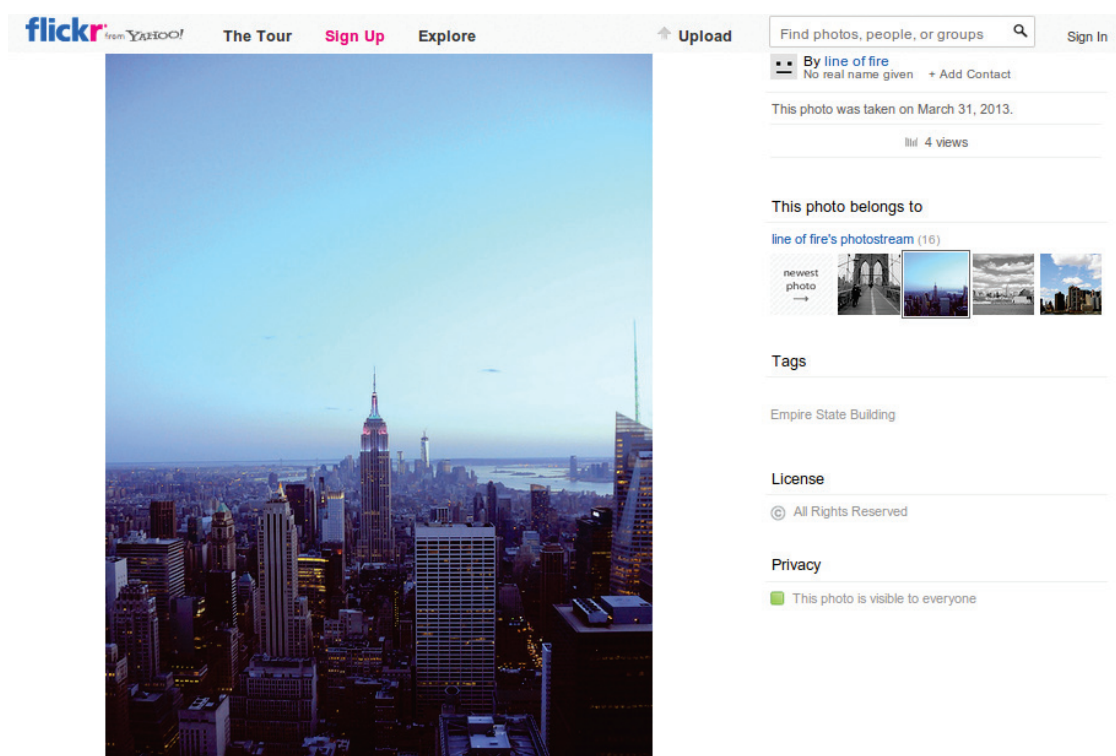


Figure 4.1: System overview. The presented method can be divided into 3 main phases. (1) tag disambiguation and concept mapping (2) link based tag generalization and (3) concept unification. Each one has a specific role in the process of tag generalization. The first phase is responsible for mapping given resource tags on Wikipedia concepts. It addresses disambiguation and synonymous problems. The second phase searches for common more general ancestors among Wikipedia concepts based on the textual and graph based features. The third phase unifies the concepts extracted from Wikipedia to improve quality of the recommended general tags.

Based on the previously described 3 phases our system can properly create rules to



New York at Sunset

Figure 4.2: Example of a tagged document. The image from *Flickr* presenting the Empire Sate Building. Each image in *Flickr* has tags assigned by users. Tags are the most popular way to browse images on that website. The image has one tag assigned. Our method can extend this set of tags by adding new terms like “USA” “New York” or “building”. The new set of tags will significantly improve the retrieval rate based on tag search. It also increases a chance of getting this image by drilling down from general to specific concepts e.g. from USA->New York->Empire State Building

meaningfully assign general tags to resources. It has many advantages over different supervised tag generalization systems like:

- Domain independent. The proposed system works effectively in every domain.
- Unsupervised. Works equally good whether system has 0 or 10,000 posts.
- Based on external source of information. Wikipedia is the richest available community-based source of information. It covers most of human knowledge. That way we can apply proposed method in different fields.
- Flexible. System can generalize based on any kind of tags. Tags in the system may come from users or from automatic key term extraction methods like *kea* [50].

4.2 Tag Disambiguation Module

This section describes the first module of the proposed system. It explains the process of tag disambiguation and the process of mapping it to corresponding Wikipedia article. This module takes as an input the set of tags assigned to the document and as an output it presents the most suitable Wikipedia concept assigned to each tag.

Word sense disambiguation is the process of identifying the correct meaning of an ambiguous word; e.g., “java” might mean the island or programming language. The most popular solutions to perform the disambiguation are based on the context of ambiguous words. A context can provide clarification of the tag meaning. Usually contexts consist of words in the same sentence, however, in our problem only tags are available to determine the meaning.

Having tagged resources (i.e. documents, e-mails or web-articles) with at least one assigned tag helps understand what the document presents. Tags can be human annotated or assigned by any automatic key term extractor. In open dictionary systems users are free to tag resources however they like. Vocabulary of tags is not limited by anything. In such systems even similar documents can be tagged in many different ways [19, 45]. It is very difficult to keep control over the way resources are tagged. Usually such systems have a very sparse dictionary of tags. In such case, it

is very hard to map tags to Wikipedia articles by using simple title or text matching techniques.

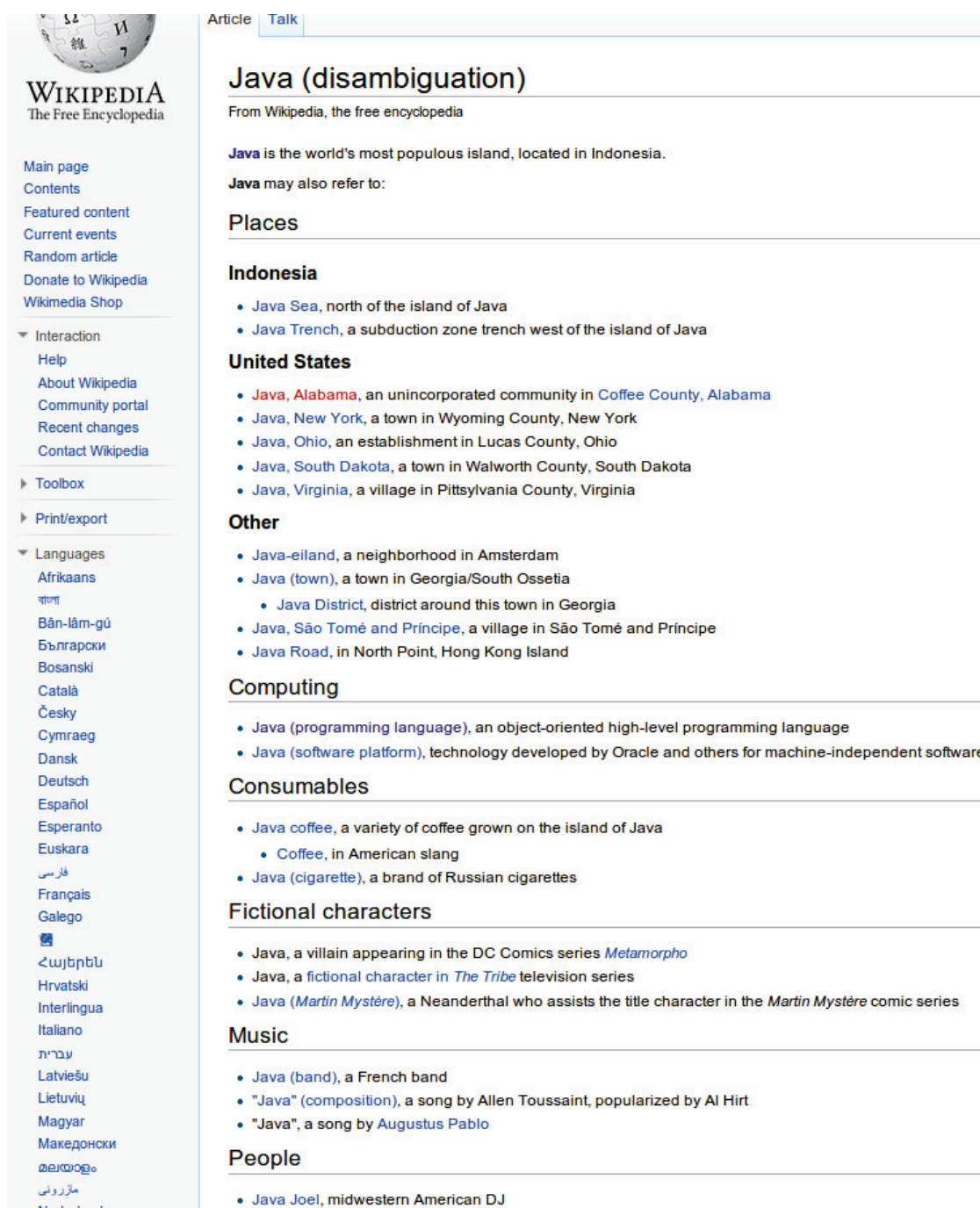
Basically in the process of tag mapping we deal with 2 problems: tag ambiguity and article synonymy. Let us consider an example set of tags: (java, eclipse, sun, database) (Fig. 4.4). Each of the tags considered independently is ambiguous (e.g., *java* can refer to programming language, island or a coffee and others to sun eclipse, software program, star or a company) However, when we consider all tags together it becomes clear which meaning should be assigned to each tag. At the same time, Wikipedia articles can be available under many other names. Each Wiki article title is unique across Wikipedia (Fig. 4.3). However the most popular name for that concept is different (e.g. java for java programming language) and sometimes the name used depends also on the context of using it.

To overcome these 2 main problems of tag we have decided to adapt the approach presented in [29]. The authors deal with a problem of text *wikification*. Wikification is a process of enriching a text document by adding hyperlinks to Wikipedia for important entities in a text. In their method they are using anchor texts from links between articles in Wikipedia. Each article can be linked from many different places and it could be described by many different anchor texts. As a result, using anchors increases the number of synonymous expressions for an article. Anchor text usually depends on the context and it can be more informative than an article title which has to be unique. As we have described in Section 5.1 the coverage for the tags from datasets is very high using this approach.

The process of disambiguation can be divided into 3 steps: (1) mapping tags to anchor links; (2) calculation of the similarity metric for concept candidates; (3) selection of the most similar candidates for a given set of tags. The following sections describe each step in detail.

4.2.1 Mapping Tags on Anchor Links

The first step in the process of tag disambiguation is to find the best candidate Wikipedia concepts for each tag. First we do simple text matching between the tag and anchor texts of Wikipedia articles. Then we select all of the articles which have at least one incoming link with an anchor that is the same as the tag. A few examples



The image shows a screenshot of the Wikipedia disambiguation page for the word "Java". The page is titled "Java (disambiguation)" and is part of the "Article" tab. The main content is organized into several categories, each with a list of links to related articles:

- From Wikipedia, the free encyclopedia**: A brief introductory sentence stating "Java is the world's most populous island, located in Indonesia."
- Java may also refer to:** A general heading for the disambiguation list.
- Places**: A sub-heading for geographical locations.
 - Indonesia**:
 - Java Sea, north of the island of Java
 - Java Trench, a subduction zone trench west of the island of Java
 - United States**:
 - Java, Alabama, an unincorporated community in Coffee County, Alabama
 - Java, New York, a town in Wyoming County, New York
 - Java, Ohio, an establishment in Lucas County, Ohio
 - Java, South Dakota, a town in Walworth County, South Dakota
 - Java, Virginia, a village in Pittsylvania County, Virginia
- Other**: A sub-heading for non-geographical items.
 - Java-eiland, a neighborhood in Amsterdam
 - Java (town), a town in Georgia/South Ossetia
 - Java District, district around this town in Georgia
 - Java, São Tomé and Príncipe, a village in São Tomé and Príncipe
 - Java Road, in North Point, Hong Kong Island
- Computing**: A sub-heading for technology-related items.
 - Java (programming language), an object-oriented high-level programming language
 - Java (software platform), technology developed by Oracle and others for machine-independent software
- Consumables**: A sub-heading for items that can be consumed.
 - Java coffee, a variety of coffee grown on the island of Java
 - Coffee, in American slang
 - Java (cigarette), a brand of Russian cigarettes
- Fictional characters**: A sub-heading for characters from fiction.
 - Java, a villain appearing in the DC Comics series *Metamorpho*
 - Java, a fictional character in *The Tribe* television series
 - Java (*Martin Mystère*), a Neanderthal who assists the title character in the *Martin Mystère* comic series
- Music**: A sub-heading for musical works.
 - Java (band), a French band
 - "Java" (composition), a song by Allen Toussaint, popularized by Al Hirt
 - "Java", a song by Augustus Pablo
- People**: A sub-heading for individuals.
 - Java Joel, midwestern American DJ

The left sidebar of the page contains the standard Wikipedia navigation menu, including links to the Main page, Contents, Featured content, Current events, Random article, Donate to Wikipedia, Wikimedia Shop, and various interaction and language options.

Figure 4.3: Java Wikipedia disambiguation page example. In case of ambiguous meaning of a concept Wikipedia let users decide what they are looking for by presenting “disambiguation” pages. Users get presented a list of concepts divided into categories. Across Wikipedia word “java” has 61 different meanings. It covers a wide variety of concepts from very popular like: programming language, Java island to unpopular like town in Virginia or a model of Chrysler car.

1

Sun; Java; Eclipse;

Sun Microsystems

From Wikipedia, the free encyclopedia

Sun Microsystems, Inc. was a company that sold computers, computer components, computer software, and information technology services and that created the Java programming language, and the Network File System (NFS). Sun significantly evolved several key computing technologies, among them Unix, RISC Processors, Thin Client Computing, and virtualized

Java (programming language)

From Wikipedia, the free encyclopedia

"Java language" redirects here. For the natural language from the Indonesian island of Java, see .
Not to be confused with JavaScript.

Java is a general-purpose, concurrent, class-based, object-oriented computer programming language that is specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that code that

Eclipse (software)

From Wikipedia, the free encyclopedia

For other uses of "Eclipse", see *Eclipse (disambiguation)*.

In computer programming, **Eclipse** is a multi-language Integrated development environment (IDE) comprising a base workspace and an extensible plug-in system for customizing the environment. It is written mostly in Java. It can be used to develop applications in Java and, by means of various plug-ins, other programming languages including Ada, C, C++, COBOL, Fortran, Haskell.

2

Sun; Java; Eclipse; Moon;

Moon

From Wikipedia, the free encyclopedia

*This article is about Earth's Moon. For moons in general, see *Natural satellite*. For other u*

The **Moon** is the only natural satellite of the Earth,^{[d][7]} and the fifth largest satellite in the Solar System. It is the largest natural satellite of a planet in the Solar System relative to the size of its primary,^[6] having 27% the diameter and 60% the density of Earth, resulting in ¹/₈₁ its mass. The Moon is the second densest satellite after Io, a satellite of Jupiter.

Sun

From Wikipedia, the free encyclopedia

*This article is about the star. For other uses, see *Sun (disambiguation)*.*

The **Sun** is the star at the center of the Solar System. It is almost perfectly spherical and consists of hot plasma interwoven with magnetic fields.^{[12][13]} It has a diameter of about 1,392,684 km,^[9] about 109 times that of Earth, and its mass (about 2 × 10³⁰ kilograms, 330,000 times that of Earth) accounts for about 99.86% of the total mass of the Solar

Java

From Wikipedia, the free encyclopedia

*This article is about the Indonesian island. For other uses, see *Java (disambiguation)*.*

Java (Indonesian: **Jawa**) is an island of Indonesia. With a population of 135 million (excluding the 3.6 million on the island of Madura which is administered as part of the provinces of Java), Java is the world's *most populous island*, and one of the most densely-populated places on the globe. Java is the home of 60 percent of the Indonesian population.

Eclipse

From Wikipedia, the free encyclopedia

*This article is about astronomical eclipses. For other uses, see *Eclipse (disambiguation)*.*

"Total eclipse" redirects here. For other uses, see *Total Eclipse (disambiguation)*.

An **eclipse** is an astronomical event that occurs when an astronomical object is temporarily obscured, either by passing into the shadow of another body or by having another body pass between it and the viewer. An eclipse is a type of *syzygy*.^[1]

Figure 4.4: Disambiguation example of the proposed system. Based on the context, each tag can have a different meaning. In the first example based on the similarity between tags it is more likely that the meaning for these tags will be associated with concepts connected with computer science. It is more likely that these 3 concepts will occur together in a document related to computer science than to geographical concepts. However, the second example shows what happens when context changes. An additional tag was added which is not connected to computer science. Now it is more likely that this set of tags describes geographical concepts. Our method to solve the problem of tag ambiguity takes into account the context of tags and their similarity. It tries to choose the most similar meanings for all tags assigned to the resource.

java		eclipse		sun	
java programming language	5053	eclipse software	1047	sun	5731
java	4104	eclipse	212	sun microsystems	345
java software platform	818	mitsubishi eclipse	165	sun bowl	170
java, virginia	128	eclipse novel	148	gwr sun class	141
java town	114	eclipse comics	132	cun unit	135
java district	76	eclipse horse	123	sunderland	119
invasion of java	58	eclipse class cruiser	122	sun motorcycle	99

Table 4.1: Example of mapping an anchor to Wikipedia article. Table presents the most popular meanings for three words “java”, “eclipse” and “sun”. Each meaning is a Wikipedia concept with a number of references in other articles using this tag as an anchor to the article.

First n popular meanings	Coverage of articles
1	89%
2	96%
3	97%
4	98%
5	98,8%
10	99.5%

Table 4.2: Table presents the influence of taking only the n most popular meanings for a given anchor. It clearly shows that the most significant are the 3 or 4 most popular concepts and furthermore we deal with a long tail. For efficiency reasons in the proposed method we have limited candidates to the 10 most popular meanings.

of mapping tags to articles are shown in Table 4.1. It presents the title of Wikipedia article and the number of links for a given tag. Basic preprocessing has been done to improve the matching rate and the efficiency of the algorithm. All the anchors and tags are lower case and all [“-”, “_”] within tags and anchors have been replaced by space character so we consider them as separate words. For the efficiency reasons we have limited candidates to the 10 most popular meanings for each tag (10 most popular articles for a given anchor). This assumption covers around 99.5% of the cases and greatly improves the speed of the algorithm (Table 4.2).

Solving a disambiguation problem based only on the popularity of concepts within Wikipedia works pretty well [26], which means that usually the most popular meaning

Data set	Stackoverflow	Wordpress	Delicious
Median of assigned tags	3	3	5

Table 4.3: Table presents Median of number of tags assigned to each post in different datasets. Most of the posts in each data set have more than one tag assigned. Tags together can create a context to improve disambiguation process.

is the correct one. However, it might depend on the dataset. It has been demonstrated that taking only the most popular meaning for a tag is in average 72.5% correct [26]. We have decided to use this approach whenever tags are missing a context (document has assigned only one tag). If documents have more than a one tag we can improve disambiguation by doing additional steps, described in the following sections.

4.2.2 Candidate Similarity

This is the second step of the tag disambiguation process. The main purpose of this step is to determine the most similar meaning for the set of tags. It is done by computing the similarity between each candidate Wikipedia article.

Usually documents are tagged with more than one tag. The median of the number of tag assigned for each dataset is presented in Table 4.3. The assigned tags together create the context, which can be used to improve disambiguation. Let us consider an example where there is a document with the set of tags (java, eclipse, sun and moon) each of which has multiple meanings, starting from obvious ones like (island, sun eclipse, star, moon) and ending with (programming language, software program, company Sun Microsystems, baseball player). The meaning of the given set of tags becomes obvious if we consider all the tags together.

To decide which Wikipedia concept we should choose from the candidates we compute similarity between each pair of candidates. We are using similarity metrics introduced by [28] and described in detail in Section 3.3.1.

4.2.3 Selection of the Most Similar Candidates

The list of candidates for each tag and the similarity value between them allow us to select the best matching set of articles. In Fig. 4.5 we can see 3 clusters of nodes

corresponding to articles. Each cluster represents different meanings for the corresponding tag. Each element in the cluster represents one Wikipedia concept. Edge thickness represents the similarity value between concepts. From each cluster we need to select one concept. To assign the best candidate Wikipedia article to each tag we have to choose the article which will maximize the overall similarity measure for the set of all tags assigned to the document (Eq. 4.1). This is a combinatorial selection problem where for each cluster of nodes we need to select one node so that the sum of the pairwise similarities of the selected nodes will be maximal. We solve this problem by exhaustive search by limiting the number of tags and concepts per tag. A selection process is shown in Equation 4.1.

Optimization problem 1. To determine the best meaning for the given set of tags we need to find the optimal vector $(j_1, j_2, \dots, j_N) \in P^1 \times P^2 \times \dots \times P^N$ which will maximize the Eq. 4.1 where P^i is the set of concepts associated with a tag i , N corresponds to the number of tags, j_i is one concept candidate from the set of concepts P^i . Similarity is a reflexive relation, therefore we apply a condition $k < l$ to check only half of relations.

$$\underset{\substack{(j_1, j_2, \dots, j_N) \\ j_i \in P^i}}{\max} \sum_{l=1}^N \sum_{\substack{k=1 \\ k < l}}^N sim(j_l, j_k) \quad (4.1)$$

where :

$$(j_1, j_2, \dots, j_N) \in P^1 \times P^2 \times \dots \times P^N$$

4.3 Link Based Tag Generalization

This section describes the second module of the proposed system. It explains the process of generating a ranked list of the more general Wikipedia concepts for a given core concepts from previous module.

The process starts from the set of core articles which are assigned to tags based on mapping described in the previous section. The tag generalization process can be divided into two major parts: (1) creation of a focused Wikipedia concept graph (2) ranking each concept with a generality metric. The following sections describe these two points as well as the key assumption for building the Wikipedia graph.

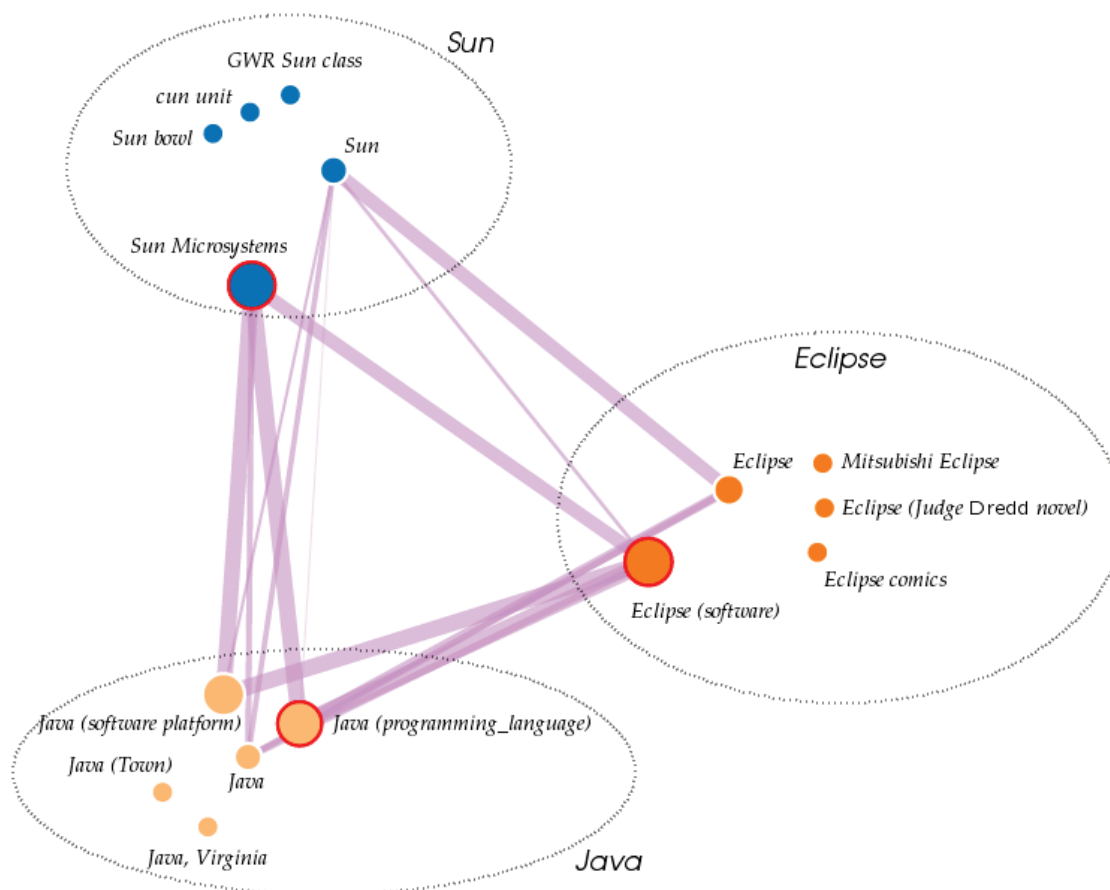


Figure 4.5: An example of tag disambiguation. The Graph presents the disambiguation process for three tags “java”, “sun”, “eclipse”. Each of the tags is represented by a cluster of nodes, where nodes are different concepts. Similarity between concepts is illustrated by thickness of the line. From each cluster only one concept can be selected (node with the red border). Selection is done by maximizing the sum of similarities between nodes (Eq. 4.1). It is a combinatorial selection problem solved by exhaustive search.

4.3.1 Key Wikipedia Assumption

To build a graph using Wikipedia concepts as nodes and Wikipedia links as edges between them is very challenging. While building a meaningful graph we had to deal with three main challenges: (1) very small diameter of Wikipedia; (2) links to irrelevant and to more specific concepts; and (3) size of the Wikipedia.

Wikipedia links create a very dense graph with a small diameter. On average it is possible to get from one article to any other in only 3.45 steps just by following links on the pages. Moreover Wikipedia articles have very rich textual content. Almost all articles have hyperlinks to related concepts. Some of them lead to more general but some to more specific concepts. There are more hyperlinks to more specific pages which in our case is not desirable. We had to find a way to limit links to only more general articles.

To address this challenge we decided to make the assumption that links from the first paragraph tend to lead to the more general concepts than other links. This assumption is based on the Wikipedia policy. According to the official Wikipedia tutorial [48] on writing a good article, opening paragraph of each article should describe the concept in a general way, it should summarize the entire article in a language understandable by a high school student. The following paragraphs describe the concept in more details.

Based on this assumption we create a network of related articles, reducing significantly the number of links and increasing the diameter of Wikipedia.

4.3.2 Building a Graph of Related Articles

This section describes the process of building a graph of the related articles $G_A(V, E)$ of a given A as a *main article* (Def. 1) where V is a set of nodes representing Wikipedia concepts and E is a set of edges between concepts. Each edge corresponds to a Wikipedia link between two articles and it points from the source node to the link destination node. The graph is directed and unweighted.

Definition 1. The *main article* is a concept from which the process of building a graph starts. It is one one of the Wikipedia concepts selected in the optimization problem 1.▲

The graph is constructed separately for each concept selected in Section 4.2.3 and later the graphs are merged to one.

Each node in the graph is pointing to all of the articles linked from its first paragraph. An example of such graph is presented in Figure 4.8. The graph was created from the “Apollo” space program as the main article. As we can see the graph is very large. It contains 344 nodes and 563 edges even though it only includes nodes to the depth of 3 from the main article. This graph is focused on the nodes relevant to the main article.

Definition 2. The graph of related articles $G_A(V, E)$ of a given article A as a *main article* is a directed and unweighted graph where V is a set of Wikipedia concepts and E is a set of edges based on the Wikipedia’s hyperlink structure. The graph is constructed by breadth-first search of the links in the first paragraph of each article using algorithm 1▲

The graph is created in an iterative process. Each step of the algorithm 1 starts with the set of nodes S_q , which is a set of candidate concepts (nodes) to the final graph $G_A(V, E)$. First iteration starts from the set of S_q which only include the main article A . Next we fetch all articles S_a , which are pointed to by links from the first paragraph of the articles in S_q . To reduce the number of collected articles and focus the network on the related topic we decided to fetch only half of the most similar pages in each iteration. We need to determine the semantic relatedness between each article and the main article. Then we take only half of the most similar candidates (S_{top}). Next we add the chosen articles to the set of S_q for the next iteration of the algorithm. Due to the low Wikipedia diameter the iteration number was limited by default to 3. However, it can be adjusted to obtain less general results. A lower value will reduce the chance of getting to general results. A higher value produces results that are too general or graphs which are usually off topic as we see in Fig. 4.7. Figure 4.6 presents the visualization of constructing a graph of related articles.

Two graphs, presented in Fig. 4.8 and 4.7 were created for the “Apollo” space program article. The graph in Fig. 4.8 is focused unlike the one presented in Fig. 4.7 which is unfocused. The main difference between them is that unfocused graph was built based on all links from the first paragraph and focused only on the half most

similar to the starting concept. This approach has many advantages over the unfocused graph. First we have reduced the number of nodes and edges, respectively from 670 to 344 and from 991 to 563. As we can see in the unfocused graph (Fig. 4.7) there are many unnecessary nodes (red circle). Most of them are related to USA, politics, and places in the USA which do not have anything common with the starting concept. It might be caused by the article like “List of” with many links but usually off topic. More importantly the focused graph allowed us to fetch only nodes similar to the main article. It is likely that similar concepts will link to the same articles, at the same time promoting more general concepts.

Algorithm 1 Building a graph of the Wikipedia related concepts $G(V, E)$ where V is a set of Wikipedia concepts and E is set of links between them

Require: Main article A

$V = \{\}$ // Final set of nodes

$S_q = \{A\}$ // set of the concepts candidates for a graph

$S_a = \{\}$ // articles from first paragraph of S_q

$sim = []$ // array of similarity values between retrieved articles and the main article

for $iteration = 0, iteration + + \leq 3$ **do** // adjustable number of iterations

$S_{a+} = get_articles_from_first_paragraphs(S_q)$

for $a \in S_{a+}$ **do**

$sim[a] = similarity(a, A)$ // similarity between a and the main node

end for

$S_{top} = selectTopHalf(sim, S_{a+})$ //take only half of the most similar nodes

$S_{a-} = \{S_{top}\}$ // remove chosen nodes from S_{a+}

$V+ = \{S_{top}\}$ // add chosen nodes to the final set

$S_q = \{S_{top}\}$ // chosen nodes for next iteration

end for

$V+ = \{S_{a-}\}$

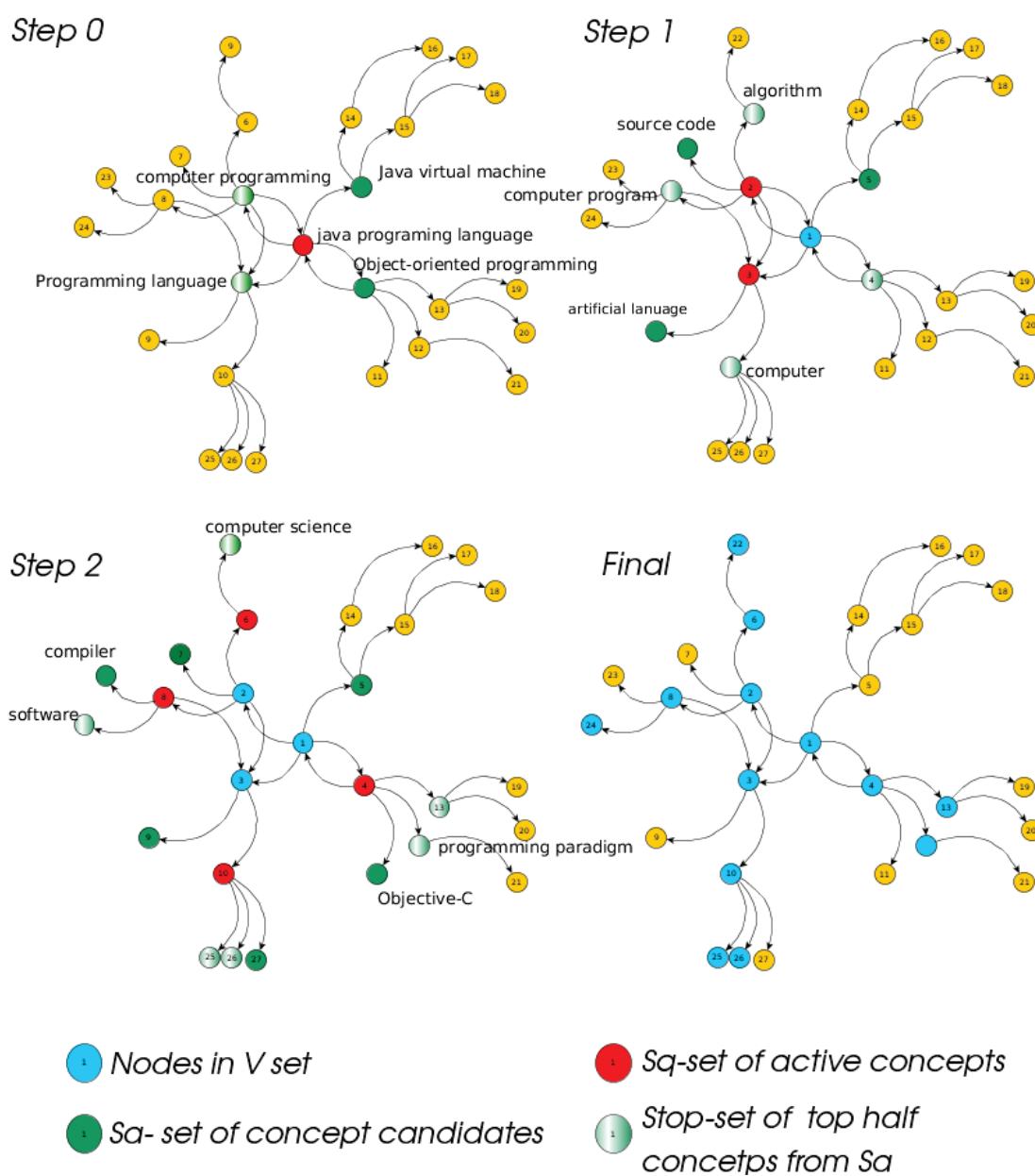


Figure 4.6: Visualization of the algorithm 1 to construct a graph of related articles. Each image presents one of the algorithm iterations. However, graphs present only snapshots of the process in the middle of each iteration, graph was simplified to improve visibility. The graph starts with the “java programming” article.

4.3.3 Ranking Articles with the Generality Metric

The next step after building a graph of Wikipedia articles is to determine the most general articles in the graph. We want to find the nodes which cover the highest number of nodes similar to the main concept. An example calculation of this generality

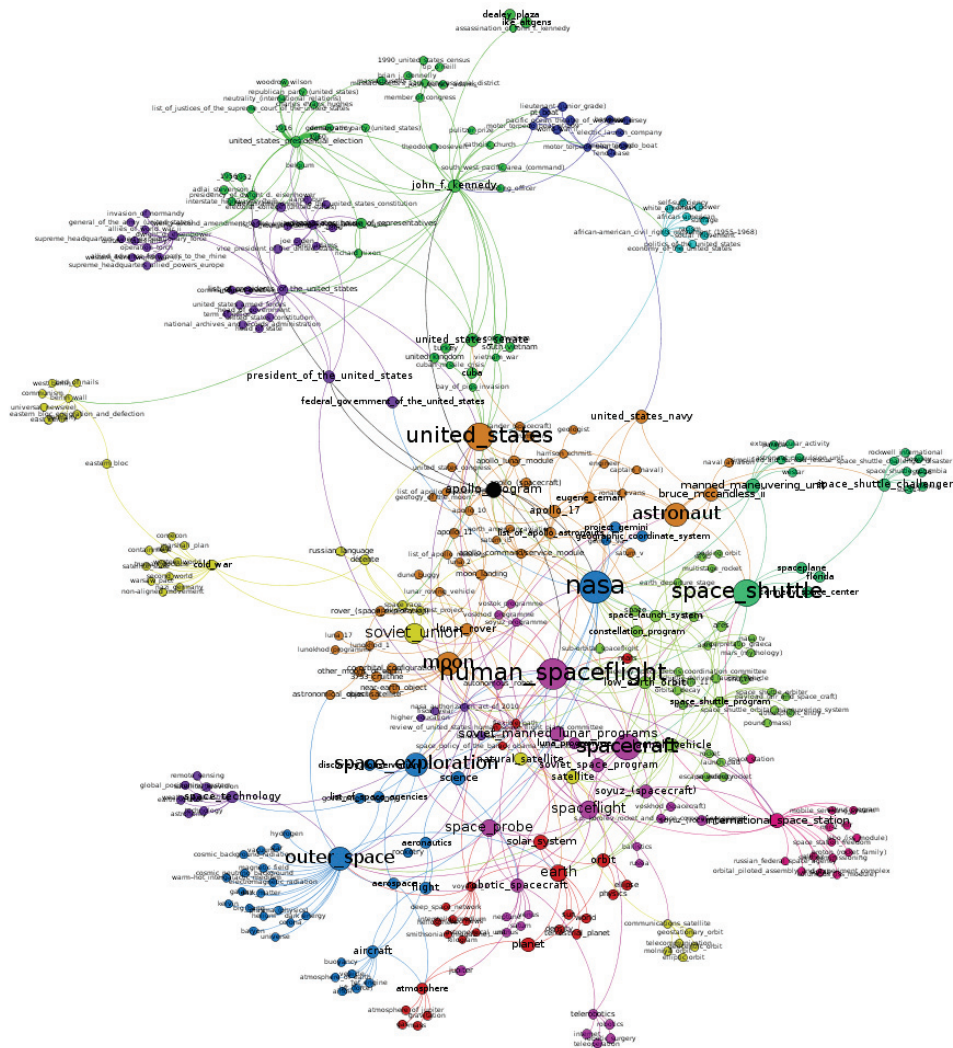


Figure 4.8: Wikipedia focused concept map. The image presents a map of concepts created for the Apollo space program article. The graph is focused only on relevant nodes. Nodes were chosen to build this graph based on their similarity to the starting node. In each step only the half most similar concepts were chosen to follow their links.

metric is presented in Fig. 4.9. This graph presents a simplified version of the graph created for “java programming language” as the main article a_0 . Each node is labeled with the similarity value with “java programming” concept. To compute generality value for “programming language” we need to consider a set T_v which in this case consists of 4 nodes with path to “programming language”: “computer programming”, “objective-c”, “object-oriented programming” and “java programming language”. In general it is a recursive process. For each node in the graph we sum the similarities of the all articles pointing to the node. We also take into account the distance d of the article to the node a_v so that closer articles have greater influence over the final score of the node.

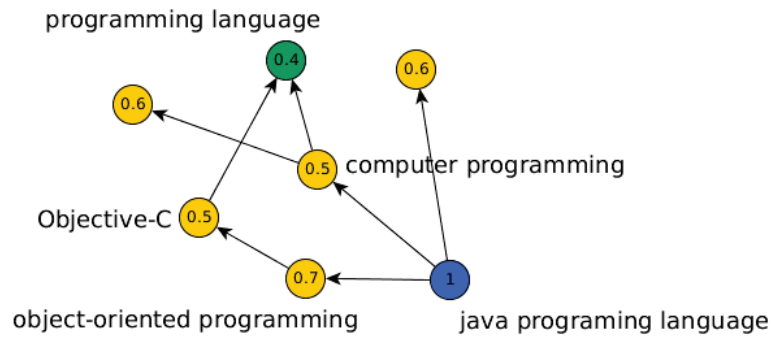


Figure 4.9: An example calculation of the generality value for a concept. The graph presents the simplified graph starting in “java programming language”. Each node is labeled with the similarity value with the main concept (“java programming language”, blue). The generality value for a “programming language” according to equation 4.2 would be: $\frac{0.5}{1} + \frac{1}{2} + \frac{0.5}{1} + \frac{0.7}{2} + \frac{1}{3} = 0.683$

Definition 3. The *generality value* of the concept $v \in V$ in the graph $G_A(V, E)$ defines how general a concept is within the graph of related articles (Def. 2) and it is computed according to Eq. 4.2 where V is a set of all concepts in the graph, a_0 is the main concept, T_v is the set of nodes with a path to a_v and d_t is the distance (number of hops) from the concept a_v to t ▲

$$generality(a_v) = \frac{\sum_{t \in T_n} \frac{sim(t, a_0)}{d_t}}{\sum_{q \in V} sim(q, a_0)} \quad (4.2)$$

We adapted one of the metrics proposed in [39]. In their study the authors proposed a complex method to determine the most important tags in the set of documents. One of the defined metrics was cohesiveness. We adapted the metric of [39] to our problem. Originally this metric was computed for association sets of tags. Our *generality value* is computed according to Eq. 4.2, where T_v is a set of nodes covered by a_v (nodes which have a path to the node a_v), a_0 is the main concept, V is a set of all concepts in the graph. The generality value defines how closely related the covered set of nodes T_v is to tag a_0 . We also consider a distance d_t of the articles to a_v . The denominator of the Eq. 4.2 is responsible for normalization of the result. Each concept may occur only once within a path by this we prevent cycles between articles. We want to find concepts which are linked by many similar concepts. If many similar Wikipedia concepts link to the same concepts it is very likely that that concept will be more general.

At the end of the process every node in the network has assigned the *generality value*. The value will consist of contributions from all the nodes it covers. The higher the value the more general the article should be. As an outcome, this process produces a ranked list of more general Wikipedia concepts based on the generality value.

4.4 Concept Unification

The previously described module generates a ranked list of the most general articles. The list consists of Wikipedia concept titles. Titles however, are not always good candidates for tags. In 85% of the articles the title consists of more than one word. However, based on the *Delicious* dataset users seem to prefer single-word tags. Only 9% of the total number of tags consist of two or more words. Moreover, it is likely that there exists a single word with a similar meaning which is much more popular than the article title e.g. instead of “java (programming language)” the preferred word would be “java”. The additional phrase in the title is added because every article title has to be unique across the Wikipedia.

This module deals with the unification of tags of the recommended articles. It can be considered as the inversion of the first module of the system in which tags were mapped to Wikipedia articles. This time, program has to map articles to the best tag candidates, which in this case are anchors for all incoming links to the article.

It is very likely that one of the anchor texts will be the most popular name for the concept because they are human annotated and depend on the article context.

We have distinguished 3 different approaches based on the tag recommendation system policy: closed dictionary system, open dictionary system and the combination of both. Each system works slightly different. The main difference of those systems is the way they manage tags. It depends if they are allowed to propose a new tag or they prefer to use tags which are already in the system.

4.4.1 Closed Dictionary System

An example of a closed dictionary system is *StackOverflow*. In this system resources can only be tagged by tags which are already in the system therefore assigned general tags should be already in the tag dictionary.

In a closed dictionary system the concept unification is processed in the following steps. First, we need to define a set of general tags S_g and map them on Wikipedia concepts. For each general tag in the system (usually the most frequent ones) our method generates a fixed length vector of more general articles which are called *profile vectors* $\vec{\mathbf{p}}_t$ (Def. 4). These vectors consist of the 50 most general concepts extracted for each tag together with their *generality value* (Def. 3). Having a *profile vector* for each general tag will allow us later to assign the most similar general tag to a post.

Definition 4. *Profile vector* $\vec{\mathbf{p}}_t$ is a vector for each $t \in S_g$ where S_g is a set of predefined general tags. The vector consist of top n *generality values* (Def. 3) returned by link based tag generalization module for a tag t ▲

Definition 5. *Generality vector* $\vec{\mathbf{g}}_p$ is a vector for each tagged resource p consisting of the top n *generality values* (Def. 3) returned by link based tag generalization module (Section 4.3) based on assigned tags for p ▲

Later, at the time of assigning general tags for each post, the system produces a vector of the more general articles for each post which is called *generality vector* $\vec{\mathbf{g}}_p$ (Def. 5). Next, we limit the set of general tags S_g to the intersection of S_g and dimensions of $\vec{\mathbf{g}}_p$ for which generality values are not 0. However, if the intersection is empty then S_g includes all the general tags. In the next step the system compares the *generality vector* $\vec{\mathbf{g}}_p$ for a given post p with all *profile vectors* of the general tags

S_g . Cosine similarity is used to find the most similar *profile vector*. Based on the cosine similarity between these two vectors we can get the ranked list of the general tags from the system and produce the set of general tags.

4.4.2 Open Dictionary System

Some content management systems like blogs or forums allow users to tag their resources without any limitations. The proposed system can assign any set of general tags. Having such a freedom the system can produce the best tags for a given set of articles.

Having a list of more general articles for the given set of tags we could propose the titles as a set of the new more general tags. However as we said before usually it would not be the best choice. Instead we are trying to find the most popular anchors for given concepts.

To generate the ranked list of the new tags we take all of the anchors of the links leading to the more general articles. The final list consists of the most popular anchors. This approach solves the problem of redundant article titles which belong to highly related articles e.g “Java (programming language)” and “Java (software platform)”. For both of the articles the most popular anchor text is simply “java”.

4.4.3 Mixed Approach

This method is a combination of both processes described previously. Sometimes content management systems prefer to assign tags which are already in the system over assigning new ones. This way they will try to reduce the sparseness of the tag dictionary. However it is still possible to introduce a new tag if there is nothing similar enough to the tags available in the system.

In this scenario we take advantage of both methods described above. First we create profile vectors for all the general tags in the tag-based system like StackOverflow where we want to assign general tags to new posts. If cosine similarity between vectors is greater than a predefined threshold we use the closed dictionary approach otherwise we introduce new tags to the system using open dictionary method.

4.5 Caching and Efficiency

The presented system consists of several modules. Each of them is quite computationally demanding. Therefore to improve overall performance we decided to cache results from each module. We can distinguish two layers of cache. The first is provided by DBMS (data base management system) and the second stores precomputed results. The DBMS cache system caches recent system *SQL* queries. Results for each generalization on the Wikipedia concept level are computed and then stored in the DB. The cache grows over time. Each time user makes a query, first the system checks the cache if the answer is already there. If not then the system computes results and save them in the cache. Introducing the cache layer to our system significantly improved speed. The computational time depends on the amount of tags to generalize. However, usually the system is able to provide the answer in real time.

Chapter 5

Experiments and System Evaluation

This chapter describes experiments on the evaluation of the proposed tag generalization method. To evaluate our method and to compare it with other solutions we have decided to test it on well known datasets of tagged resources. From each dataset we have chosen a set of general tags. We took all the resources tagged with those tags. We deleted the general tags from the set of tags. In our experiments we tried to restore the missing tags.

We have conducted three different experiments to evaluate our method. First, we check the coverage of our disambiguation module, to observe how well Wikipedia concepts cover tags from different datasets. In the second experiment we evaluate the overall accuracy of the proposed system and compare it against the baseline methods with the different training dataset size. Baseline methods need a training data to be successful in the task of tag generalization unlike our system which is unsupervised. In the third experiment we check the correlation between the accuracy of systems and tag popularity in the dataset. This experiment aims to minimize the effect of tag co-occurrence for popular tags to highlight the difference between tag correlation and tag generalization process.

In the next section we present datasets of tagged resources used in our experiments. In the following sections we describe each experiment methodology in detail. After that we describe the baseline methods to compare with our system. Finally, we present results as well as a discussion about them. At the end of this chapter we present a demo application developed to illustrate practical aspects of the proposed tag generalization method.

5.1 Datasets

To evaluate the system we have decided to use well known datasets of tagged resources. *StackOverflow*¹ is an example of a controlled dictionary system where tags are managed by administrators. *Delicious*² and *Wordpress blogs* are examples of open dictionary tag systems. Each of them includes different topics. Users are free to tag their resources however they like. *StackOverflow* and *PubMed* are specialized datasets. In *StackOverflow* tag dictionary vocabulary is limited only to computer science topics and in *PubMed* to medical related concepts. In the following sections we will describe the datasets used in more detail.

5.1.1 StackOverflow

One of the most well known tagging datasets is *StackOverflow*. This dataset is a database dump from a popular programming forum. *StackOverflow* is a platform where software developers can post their questions and answers about topics in computer programming. The website is very well organized. Each question has to have tags from a predefined list of tags. Each question and answer can be voted up and down by the community. Users can gain different ranks and popularity points based on the number and quality of answers they provide for questions. The dictionary of available tags is managed by administrators. Users can propose a new tag with the description. System administrators add tags whenever needed. This approach has many advantages. It ensures the quality of tags, prevents redundancy and reduces the sparseness of tag dictionary.

In our experiments we used a database dump from September 2011. Basic statistics of datasets are presented in Table 5.2. StackOverflow dataset is perfectly suited for our needs. It has more than 2 million of tagged posts. The most popular tags are shown in Table 5.1.

5.1.2 Delicious

Delicious is a very popular social bookmarking site. It contains millions of links to websites each of them tagged by user defined tags. Users are free to tag the resources

¹<http://stackoverflow.com>

²<http://delicious.com>

however they like, there is no tag hierarchy. All bookmarks in *Delicious* are public by default but there is also an option to make them private. *Delicious* dataset is not easily available to download. However, researchers crawl the data for their needs. We have used the dataset obtained by Lipczak et al. [21]. Basic statistics are presented in Tables 5.1 and 5.2

5.1.3 PubMed

PubMed ³ is a publicly available dataset of medical literature. It includes bibliographic information for articles published in medical journals. It covers research publications since 1946 until present and contains over 2 million articles. We have crawled over 100 thousands of articles. Tags in this dataset are reliable since every article is tagged by a authors which assign keywords to their publications. It is a great example of a specialized dataset. Usually, tags from this dataset are very meaningful and have straightforward connection with encyclopedic definition.

5.1.4 Wordpress

Wordpress is a one of the most popular open source *CMS* (Content Management) systems in the Internet. It is used by 60 million websites and by over 14% of the top 1 million websites [49] and developed by WordPress Foundation. Usually it is used to create blogs where each post can be tagged to improve their organization and retrieval.

Dataset used in our experiments contains blog posts from the *wordpress.com* domain. The dataset was released by the *ICWSM 2009 Data Challenge*. This dataset is another example of an open dictionary dataset.

5.1.5 Selection of General Tags

To evaluate our methods we need a gold standard for a tag generalization process. Our gold standard is a set of general tags. We have chosen this set based on the tag popularity. The most frequent tags from the datasets are presented in Table 5.1. We assumed that the more frequent a tag is the more general should be. Popular tags like names of programming languages from *StackOverflow* or “news” and “politics”

³<http://www.ncbi.nlm.nih.gov/pubmed>

StackOverflow	Delicious	Wordpress	PubMed
c#	design	life	population
java	blog	news	demographic-factors
php	software	politics	developing-countries
javascript	web	music	research-methodology
iphone	tools	obama	developed-countries
android	reference	family	family-planning
.net	programming	mccain	population-dynamics
c++	music	business	economic-factors
python	video	love	diseases
ruby	webdesign	election	biology

Table 5.1: The most popular tags in datasets

	StackOverflow	Delicious	Wordpress	PubMed
number of posts	2,012,349	26,260,208	1,563,839	112,335
number of unique tags	29,472	6,204,148	351,972	64,852
median number of tags in post	3	5	3	8

Table 5.2: Dataset Statistics

from *Wordpress* blogs dataset are good examples of general tags we would like to assign to posts. As we said in the introduction our goal is to improve user experience in exploring and retrieving documents using tags. One of the standard approaches in such systems is creating a tag cloud from the most popular tags. As many studies [39] have shown a good tag cloud should make it possible to drill down to the largest number of documents, leaving only few uncovered documents. Assigning popular tags whenever it is possible to documents greatly reduces dictionary sparseness. For our experiments we considered popular tags as a good set of general tags. However we have skipped some tags which have a more general meaning equivalent in the dataset like: sql instead mysql, T-sql or .net instead asp.net. We tried also to avoid abstract concepts without clear meaning.

General tags defined for each dataset are presented in Table 5.3. As we can see tags from *StackOverflow* datasets are mostly related to programming languages. Only few of them are related to systems or concepts like “security”.

5.2 Experiment Scenarios

5.2.1 Mapping Tags into Wikipedia Articles

This section describes experiments conducted to test the tag mapping module. Our system is meant to be domain independent, it should work well in any domain without training or gathering any new data. It was shown that Wikipedia articles can be mapped to concepts represented by tags [32]. In this experiment we want to find what percentage of tags can be mapped to Wikipedia articles. The experiment is done for three datasets of tags. The method used for matching a tag with Wikipedia articles is described in Section 4.2.1. For each dataset we consider two meaningful values, absolute and weighted coverage. Absolute coverage is the number of tags we were able match to Wikipedia over the total number of unique tags. Weighted coverage takes also into account the frequency of tags in the dataset. It is more important to cover more popular tags instead of infrequent ones. Results for this experiment are presented in Section 5.4.1.

StackOverflow	Wordpress	Delicious	PubMed
c#	life	design	diseases
java	news	blog	physiology
php	politics	software	behaviour
javascript	music	web	reproduction
iphone	obama	tools	fertility
android	family	programming	viral-diseases
.net	mccain	music	endocrine-system
c++	business	video	genetics-and-reproduction
python	love	art	urogenital-system
ruby	election	linux	hormones
sql	health	news	genitalia
qt	cheats	photography	nuptiality
css	home	business	pregnancy
xml	world	javascript	cancer
regex	blog	google	nutrition
database	money	mac	neoplasms
matlab	web	windows	uterus
linux	media	technology	estrogens
wordpress	travel	politics	surgery
security	online	education	hiv-infections

Table 5.3: Set of general tags used in our experiments. Tags were chosen based on their popularity. However we have removed tags which have some more general equivalent.

5.2.2 Overall Performance of the Algorithm

In this experiment we evaluate the overall accuracy of the proposed system. The biggest advantage of our system is that we are using Wikipedia as an external source of knowledge. Moreover, it does not need to be trained. The performance will be equally good whenever it works with a new system, specific systems designed for a small community of users (e.g., members of an organization) or systems with millions of posts. Existing systems use supervised algorithms. They need to be trained on the specific system to be successful in the task of tag generalization. However, accuracy of such systems depends on the amount of training data which should be relatively high. These algorithms work well in big systems with millions of posts like big online forums but they have problems with systems with relatively small number of users. The need for training data is a serious drawback of such systems. In this experiment we want to investigate when our system outperforms the baseline method and how much training data is needed for the baseline methods to match the performance of our approach.

Experiment scenario consists of several steps:

1. Definition of the set of general tags which are used as a gold standard in the evaluation
2. Selection of training set for the baseline method
3. Construction of test set
4. Training the baseline system on the training set
5. Running both methods on the test set
6. Comparison of results with the gold standard

To test our system, to compare it with the baseline methods we need to define training and test sets. Test set was limited only to posts which have at least two tags including one general. We also removed general tags from the the test set to obtain the gold standard. Systems then are tested on the same test set. However, the training set was used only by the baseline methods. In this experiment training set was changing in each iteration. We have evaluated baseline methods for 16 different

	StackOverflow	Delicious	Wordpress	PubMed
training posts	1,300,000	1,300,000	1,300,000	100,000
testing posts with general tags	10,000	10,000	10,000	10,000

Table 5.4: Training and Testing dataset

training set sizes (1200, 2000, 5500, 7500, 10000, 15000, 20000, 30000, 50000, 75000, 100000, 200000, 500000, 750000, 1000000, 1300000).

5.2.3 The Influence of Tag Frequency

In this experiment we investigate the correlation between popularity of tags and accuracy. Most tag recommendation methods focus on tag co-occurrence. They measure frequency of tags which are present together in one post. Usually such systems do not use any external knowledge to connect tags based on the tag meaning. Their logic is fairly simple: if they see that “php”, “mysql” and “html” frequently appear together, next time they see “mysql” and “html” they assign “php” based on the history that “php” was seen with them very often. However there could be a better tag to recommend e.g. “www” or “Internet”. Both tags are equally correct without additional information.

This experiment aims to examine systems in the situation when only infrequent tags are available. In this experiment we are limiting the test set by excluding posts with the N most popular tags from the dataset.

5.3 Baseline Methods

To evaluate our system we compare it against two other systems: Naive Bayes method and a method presented by Lipczak et al. [21]. These two methods are tag recommendation systems, they do not consider any generality metric, their recommendations are based solely on the examples they have seen before. Naive Bayes approach is a simple method which take into account the co-occurrence of tags and their frequency. Lipczak et al. created created hybrid system for tag recommendation which consists of five separate recommender modules (Fig. 5.1). Each recommender focuses on a different source for tags. *Title recommender* focus on words in title of each post and tries

to extract the best words as tag candidates. *Title-to-tag recommender* extends the set of tags generated from the title by finding the similar and related terms. *Resource profile recommender* tries to extract tags from the content based on the previous tags for the same resource. *User profile recommender* produces the set of tags which is related to the user. Usually users tend to repeat previously assigned tags which may reflect user interests or habits. For the purpose of our evaluation *Tag-to-tag recommender* module is the most important. This module creates a co-occurrence graph of tags that were used together in a post. The weight of an edge in such a graph was equal to the number of posts they occurred together. Later spreading activation algorithm is used to find related elements from the graph.

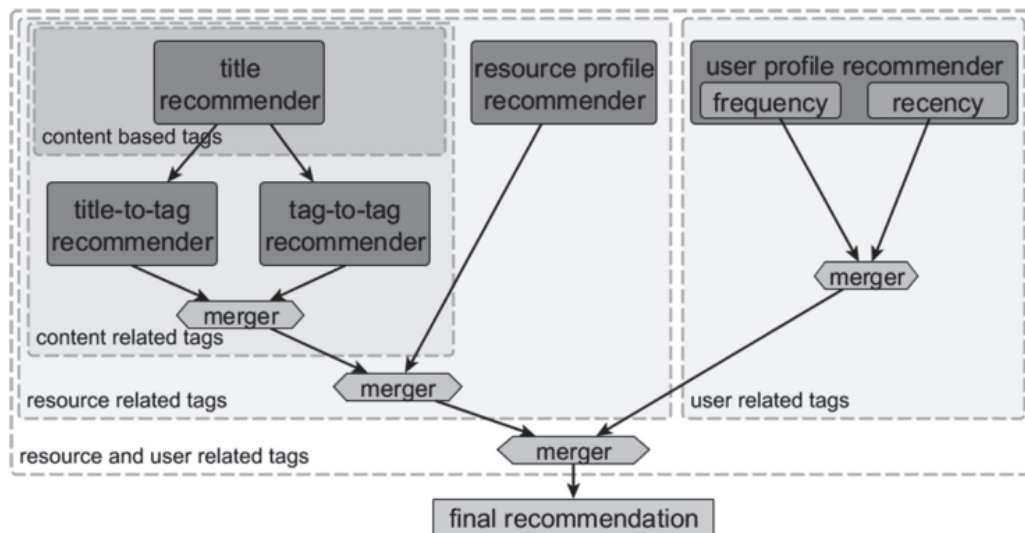


Figure 5.1: The baseline tag recommendation system scheme. System consist of five separate recommenders. In our evaluation we are using only one subsystem, tag-to-tag recommender *source: [21]*

Our *tag generalization* method works only on existing tags. We ignore all other types of potential sources of information like titles or text from the resources. This makes our method independent from the type of content. It can work with images, posts or any kind of binary data which has assigned tags. To compare our method with other solutions we had to apply same restrictions to other methods. We used only the *Tag-to-tag recommender* from the baseline system, all other recommenders were ignored.

Small changes were needed to adapt this method to tag generalization task. We had to limit the set of possible tags to recommend only tags from a predefined general tags set.

5.4 Experimental Results

This section presents results achieved in experiments. We have done preliminary experiments to compare two different techniques of calculating the similarity between Wikipedia articles, which were described in Section 3.3. The link based similarity measure is based on the link structure of Wikipedia. However, N-gram based similarity is a text-based relatedness measure. Link based similarity resulted to be better suited to our needs. Both methods produced comparable results. However, N-gram based method has problems with very similar concepts like “java (programming language)” and “java (software platform)” The vocabularies of these two concepts are very similar. The link based similarity measure has two advantages over the N-gram method in our project: (1) it is a “binary” measure, it operates only on the existing links, it solves the problem of similar word profiles; (2) linked based similarity is much faster, we do not have to generate any profiles and compare them.

5.4.1 Tag into Wikipedia Article Mapping

We have tested tag coverage among three datasets for: StackOverflow, Delicious and Wordpress. Absolute coverage and weighted coverage for each dataset are shown in Table 5.5. Absolute coverage is the number of unique tags matched to articles over the total number of unique tags in the dataset. Weighted coverage also takes into account also the frequency of tags in the dataset. Weighted coverage was considered to reduce the influence of the long tail tags which are not very popular. Tag used a hundred times is one hundred times more important than a tag which is used only once. Therefore weighted coverage shows real usage of tags. It gives more realistic picture of the tag coverage. Moreover, it reduces the influence of misspelled tags. The results show that Wikipedia is covering most of the popular tags. There are many tags which are not covered by this approach, however they are not popular. As we described in Section 4.2.1 we do only basic preprocessing of text. We can distinguish several reasons of having long tail of uncovered tags. In most cases uncovered tags usually

consist of many words connected together by special characters or by *camelCase* notation. Frequently, tags have some sort of misspelling, unpopular abbreviations or words in foreign languages. In *StackOverflow* dataset we could not match tags which are very uncommon e.g. function names from some programming libraries, names of programs or version's numbers.

	StackOverflow	Delicious	Wordpress	PubMed
absolute coverage	37%	16%	71%	63%
weighted coverage	81%	87%	91%	79%

Table 5.5: Results of Tag to Wikipedia article mapping for different datasets

Wikipedia covers very well tags from the datasets. The presented datasets have different characteristics: *Delicious* and *Wordpress* are examples of general purpose datasets with information which is not limited to any field of science. *StackOverflow* and *PubMed* on the other hand are examples of specialized domain specific datasets.

5.4.2 Overall Performance

In this experiment we removed all general tags and later we attempted to restore them. The hidden general tags were our gold standard. We tested our method against Naive Bayes and spreading activation based methods. Results based on different amounts of training data for four different datasets are shown in Figure 5.2.

As we can see the proposed method outperforms baselines for datasets with a small amount of training data. In the case of spreading activation based method it needs at least 50 thousand training posts to learn how to classify posts correctly. The behavior of Naive Bayes method was very predictable from our perspective. Usually it classifies everything to a few most frequent classes. However, for a user it might be misleading when Naive Bayes assigns the same tags to all resources. The results show also that our system is domain independent. We have two kinds of datasets, two specialized in one domain and two general purpose. *StackOverflow* and *PubMed* are specialized in computer science and medical fields. We can observe also that performance of our method is higher in highly specialized datasets. It is the highest in *PubMed* 36%, 26% in *StackOverflow*, 23% *Delicious* and 21% in *Wordpress* blogs.

The reason behind this order might be the process of disambiguation and concept clarity. Concepts in medical field are well defined usually with unique name. What is even more important is that they are very meaningful with a clear encyclopedic definition. In contrast most tags from blogs are abstract and their connection is not clear e.g. “life”, “love” “home”. The supervised method performance improves with the training dataset size. 50 thousand posts/resources is a very high number and it proves that supervised methods can not be used in *ECM* systems for small or mid size companies in the tag generalization task. It would take years to create enough content to train the system in a mid size company. Our unsupervised solution fulfill these goals.

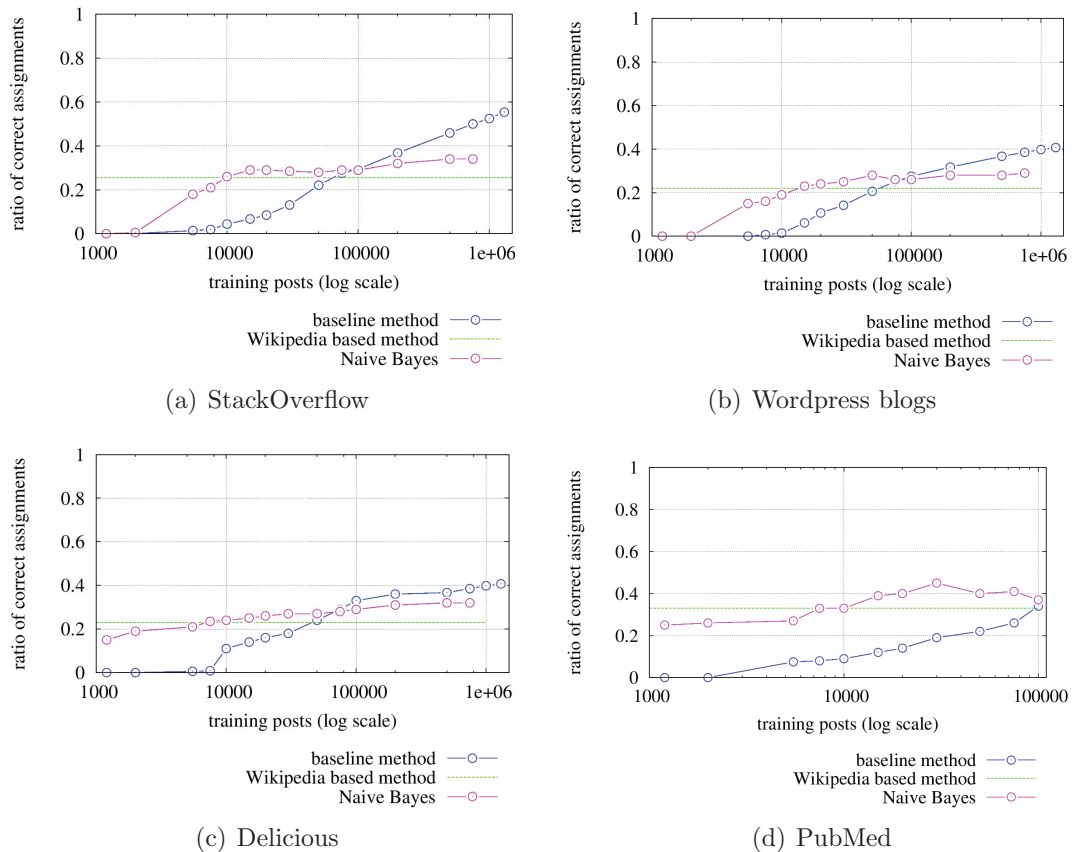


Figure 5.2: Overall performance of the proposed tag generalization Wikipedia approach compared to Naive Bayes and spreading activation methods with different amount of training data. Plots present results from 4 datasets

The influence of the number of assigned tags to posts on the final system accuracy is presented in Figure 5.3(b). We can see that system works best for posts with 3 tags

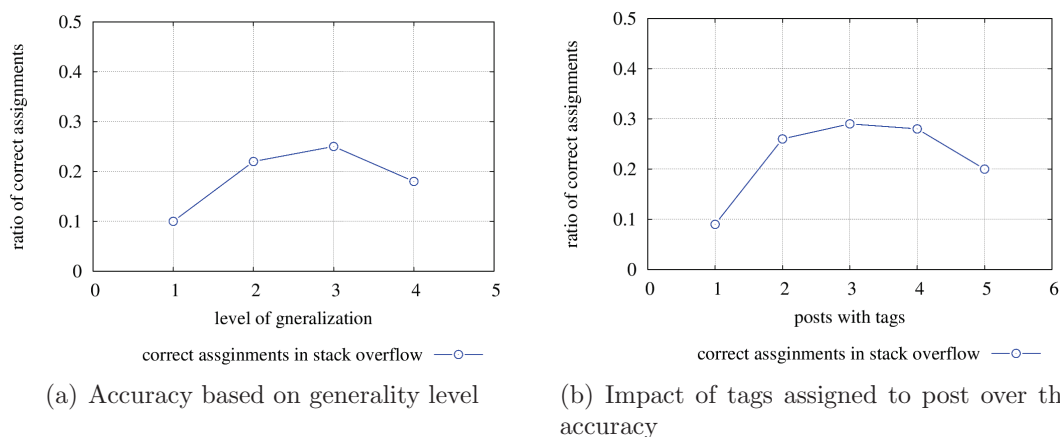


Figure 5.3: An impact of the assigned tags and generality level on the final accuracy. Figure (a) shows the influence of generality level of the proposed tags. It can be adjusted by controlling the iteration number of algorithm 1. Figure (b) shows that system works best for posts assigned with 3 tags.

assigned. It is most likely that our system had a disambiguation problem with posts with one tag assigned. The system did not have enough context to disambiguate this tag correctly. We can see that posts with 4 and more tags were not correctly disambiguated. We can explain that system tries to generalize all the tags together resulting in very general tags. To deal with this limitation we have developed an auto tag selection option. Based on the popularity of a concept within Wikipedia we can decide which tag is too general and ignore it, focusing just on the small set of tags. The system checks the frequency of each tag assigned to the post within Wikipedia selecting the 5 least frequent tags. This process is applied only for posts with more than 6 tags assigned. However, it is not a common situation that posts are tagged with so many tags. Usually resources are tagged with 3 to 5 tags which depends on the dataset 4.3.

Behavior of the baseline systems is not surprising. Systems learn from the previous examples and their accuracy grows together with the training set size. After the experiments we analyzed the misclassified cases of our system. We can distinguish a few causes of errors. Examples of errors are presented in Table 5.6. The first column shows tags which were assigned to the document and based on them the system tries to restore hidden tag. In the second column we can see our generalizations. The last column shows real hidden tag. In most cases it is very difficult even for an expert in

computer science to predict or justify the real hidden tag based on the input set of tags. Usually, the input tags which we generalize were not specific enough to come up with the real tag, in many cases we can say that our assignment is logical based on the input set of tags.

Another type of misclassification was caused by tags which do not have any straight logical connection between them. An example of such case might be set of tags: “php, mysql, html” Php is one of the hidden general tags. By having as input a pair of “mysql, html” it is very hard to predict that the general tag would be php. Most likely we would say that the best general tag should be “www”.

system input	generalization	real tag
operator-overloading, language-design	c++,java, python, objective-c, .net, c#, javascript	c#
api, design	.net, java, c++, objective-c, c#, ruby	java
list-comprehension; letters; filter; distinct	c#, javascript, c++, python	python
visual-studio; assertions	c#,.net,java,c++,objective-c, asp.net, php, ruby, javascript, wpf	c++
encode; percent; character	regex, python, database, c++	java
arrays; associative-array	matlab, javascript, java	php
scheduling; centos	linux, qt, c++, java, php	php

Table 5.6: Examples of misclassification. Systems tries to restore real tag based on the tags from the post

Baseline methods are based on tag co-occurrence. They measure how frequent were co-occurrences of php and mysql before. It does not matter that there might be a more logical concept to connect it. Baseline system works better for posts where there is no clear logical connection between tags. It is using the frequency of tags to decide which tag to assign. In the next experiment we check what happens if we remove posts with frequent tags from the test set. We wanted to reduce the effect of co-occurrence of tags.

5.4.3 Tests on Infrequent Tags

In this experiment we investigate the influence of tag frequency on the accuracy by removing posts with frequent tags. Results are presented in Figure 5.4. We can see that the accuracy of baseline methods decreases. We can explain that by the fact that they are tag co-occurrence based and they have fewer examples to learn from.

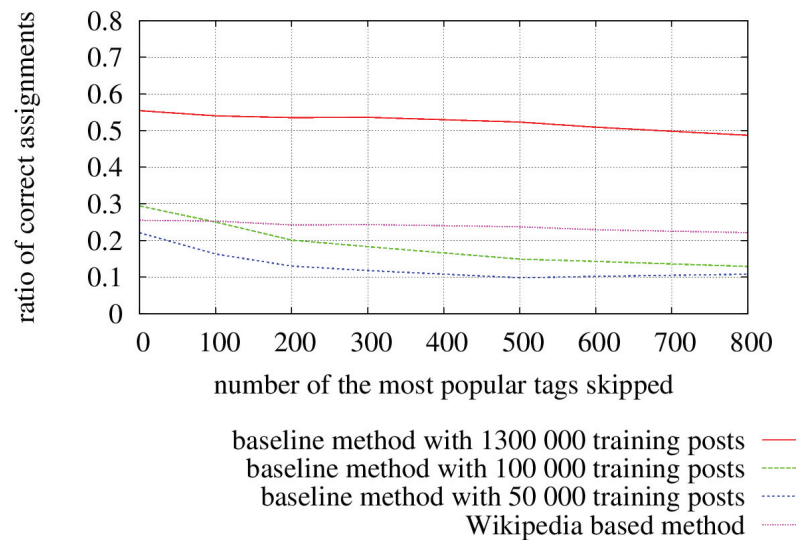


Figure 5.4: Experiment results without frequent tags. Wikipedia based tag generalization method is almost insensitive to the popularity of tags unlike the baseline method

As we can see the proposed method is almost insensitive to the tag popularity changes. Accuracy values are almost constant. Baseline method behaves differently, in the beginning both methods were comparable for 50 thousand training posts. However, baseline accuracy drops quickly. By ignoring only 100 of the most popular tags our method is as good as the baseline with 100 thousand training posts. We can also observe that if baseline system has enough data to learn the popularity of tags has a little effect on the accuracy. It has enough training data to predict even uncommon tags. Results shows clearly how important the large training data is for the baseline system and how unpredictable the accuracy can be for infrequent tags.

5.5 Demo application

To show potential usage of the presented tag generalization method aside from content management systems as we described in introduction, we created a demo application (Fig. 5.5⁴) which is available to public.

The demo automatically assignees tags to any text document. Application combines two systems. First, key term extractor to extract tags from the text. User can choose KEA [50] and Maui [23] as a term extractor. Second, our tag generalization method to propose additional layer of more general concepts. The combination of these systems might greatly improve the process of key word extraction.

Layout of the demo is presented in Fig 5.5. The lower left window presents keywords extracted from the text in the top window. The extracted keywords can be deleted. Next, user has an option by adjusting the scrollbar to choose the generality level for new tags. There are three levels of generality: 1) a little more general than tags extracted from the text; 2) more general but not always 3) general concepts, most important terminology in the field. As we have shown in experiments system works best from 2 to 4 tags. Having many tags will decrees quality of the new general concepts, algorithm will try to generalize all of them together. It is very important that tags extracted by *KEA* would be high quality. Based on them our system generates general tags. To improve this process we provided an automatic tag selection option. Selection is based on the popularity of concept within Wikipedia. If we have more than 7 tags extracted from the text, our system will ignore the most general concepts and focus on the most precise concepts.

A few examples of an input and the output of the generalization system are shown in Table 5.7. We can see examples of one word tags, multiple word tag and inputs which consist from many tags. Tags are separated by comas. Based on the examples we can observe the disambiguation precess. Tag “Samsung” alone has a meaning which is more related to multinational corporation. However, together with the tag “galaxy” which is a very ambiguous concept, the meaning shifts more towards telecommunication.

Another practical usage of our system is improving the quality of tag clouds based on tags assigned to documents. Tag clouds (Fig. 5.6) were created based on tags

⁴<http://tomasz.cs.dal.ca/wiki/demo>

Extract tags

In machine learning, support vector machines (SVMs, also support vector networks[1]) are supervised learning models with associated learning algorithms that analyze data and recognize patterns, used for classification and regression analysis. The basic SVM takes a set of input data and predicts, for each given input, which of two possible classes forms the output, making it a non-probabilistic binary linear classifier. Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that assigns new examples into one category or the other. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New examples are then mapped into that same space and predicted to belong to a category based on which side of the gap they fall on

Maui Auto tag selection: Generality level:

Kea tags: support vector, SVM, machine learning

General tags proposed by system: statistics; machine learning; mathematics; computer; artificial intelligence; computer science; data; algorithm; statistical classification; programming language;

Debug Info: tags resolved to: support vector machine; support vector machine; machine learning;

Figure 5.5: Screen shoot of the demo application. Demo assigns new tags to any text document; first we extract terms from the text using key extractor and later based on them our system provides new layer of general tags. There is a couple options to improve process of the generalization: tag generalization level and auto tag selection

input tags	recommended general tags by system
empire state building	new york city; manhattan; united states; borough (new york city); new york city subway; world trade center; lower manhattan; september 11 attacks; rapid transit
higgs boson	particle physics; elementary particle; physics; subatomic particle; quark; strong interaction; weak interaction; mass; fundamental interaction;
dog	animal; mammal; canidae; gray wolf; domestication; jackal; food; species
support vector machine	statistics; machine learning; statistical classification; dependent and independent variables; pattern recognition; regression analysis; algorithm; artificial intelligence; categorical data; data
brain tumor	brain; tumor; cancer; nervous system; cell (biology); spinal cord; neoplasm; neuron; benign tumor; central nervous system
leukaemia	blood; cancer; cell (biology); bone marrow; tumor; immune system; lymphocyte; lymphatic system; organism; tissue (biology)
dalhousie university	canada; nova scotia; halifax regional municipality; provinces and territories of canada; atlantic ocean; maritimes; atlantic canada; halifax harbour; halifax peninsula; prince edward island
samsung	south korea; seoul; multinational corporation; subsidiary; conglomerate (company); samsung; corporation; administrative divisions of south korea; chaebol; seocho-gu
samsung, galaxy	mobile phone; portable media player; south korea; personal digital assistant; digital camera; global positioning system; smartphone; multinational corporation; corporation; subsidiary
europe, apollo, greece	greek mythology; ancient greece; aegean sea; asia; roman mythology; mediterranean sea; turkey; classical antiquity;

Table 5.7: Demo Application: Examples of tag generalization output



Figure 5.6: Tag cloud created based on the PubMed dataset. Left cloud was contains only tags extracted from the dataset. Right cloud, general concepts generated by our system. Bottom cloud is a combination of tags from both clouds

from the PubMed dataset. Size of concepts in the cloud depends on the popularity of tag. The top left cloud consists only of tags from the dataset. We can observe that many tags are very precise, frequently related to internal organ diseases. The top right cloud is based only on tags generated by our system. We can observe that concepts in the second cloud become much more general. We have tags like: “disease”, “organ”, “kidney”, “blood” or “cancer”. We can observe transition from tags like: “liver-transplantation” and “liver-cirrhosis” to general ”liver”. The bottom cloud contains tags from both cloud sets. We can see that such general clouds would improve greatly the process of browsing a document set by drilling down from general to specific concepts.

Chapter 6

Conclusions and Future Work

In this project we aimed to address the problem of over-specification of tags in tag-based document management systems. People tend to over-specific tags which result in sparse dictionaries of tags. Such dictionaries are not very useful in the facet-based retrieval task. We proposed a method for automatic tag generalization which assigns more general tags to previously tagged resources. Current content management systems have two main limitations: (1) the size of document corpus they cover is very small; (2) the huge variety of documents domains they can work with. There is no method especially designed to solve the tag generalization problem. However, usually this problem is addressed by applying tag recommendation methods. Most of the current solutions for tag recommendation are based on supervised methods and they need to be trained on a large corpus of training data. Our Wikipedia based tag generalization method overcomes these limitations. Our system uses Wikipedia as an external source of information to extract more general Wikipedia concepts. Thanks to Wikipedia the proposed method is applicable to a broad variety of ECM systems regardless of post quantity in the system. Until now only big corporations could afford to have content management systems personalized to their own needs. However, nowadays even small companies need to manage their digital documents. The proposed system works on the existing tags and based on Wikipedia enriches them with an additional layer of general tags.

The proposed *Wikipedia based tag generalization method* is unsupervised and domain independent. In system design process we had to solve several problems like tag disambiguation, concept generalization and candidate concept unification. Each of which is a complex problem. Experiment results show that our system is better than other tag generalization methods for systems with limited amount of training data which applies to the majority of *ECM* systems.

The proposed system is not only limited to *ECM* systems. It can work equally

well in a systems like *Flickr*, *StackOverflow* or any other system with tags. Tag generalization system in this case would enrich the set of tags assigned to a picture or any other resource. A picture tagged with a tag “Empire State Building”, would be made retrievable by our system by tags like: “Manhattan”, “USA”, “New York” greatly increasing a retrieval rate of the picture. Another possible application of our system might be a program which would visualize the content of folders. It would scan all the documents or libraries, extract key terms from the content later generalize them and finally present a tag cloud based on the files.

We believe that a new hybrid system that combines our Wikipedia-based tag generalization system with tag co-occurrence based one would be a great extension of this work. The system which would combine the advantages of both methods. It would work well for small systems without enough training data and over the time would learn from the user input.

Another possible extension of the project would be an evaluation on different external sources of knowledge. Wikipedia covers most of the fields of science, however, It would be interesting to evaluate the system together with more specialized dataset like: SNOMED or MeSH. It would be also very interesting to include taxonomic informations from systems like DBpedia [18] or Yago2 [38]. However, the taxonomy in these two systems is based on Wikipedia categories. Our preliminary experiments showed that focusing on Wikipedia categories did not improve the results.

An interesting extension of this project would be additional evaluation based on a user study where users could evaluate the quality of general tags. The goal of our evaluation was to compare our system against existing approaches. It is possible that the user study would provide results which are closer to real world usage of the system. However, execution of a proper user study is challenging. First, our evaluation is based on thousands of posts. To get even close to this number in an user study we would have to find many participants. What is even more important every user study participant needs to be an expert in the dataset domain. For *StackOverflow* it would be computer science and for *PubMed* medicine. That is why it will not be possible to use systems like Amazon Mechanical Turk to perform this task.

Bibliography

- [1] Luca Cagliero, Alessandro Fiori, and Luigi Grimaudo. A rule-based flickr tag recommendation system. In Naeem Ramzan, Roelof Zwol, Jong-Seok Lee, Kai Clver, and Xian-Sheng Hua, editors, *Social Media Retrieval*, Computer Communications and Networks, pages 169–189. Springer London, 2013.
- [2] Rudi L. Cilibrasi and Paul M. B. Vitanyi. The Google similarity distance. *IEEE Trans. on Knowl. and Data Eng.*, 19(3):370–383, March 2007.
- [3] Silviu Cucerzan. Large-scale named entity disambiguation based on Wikipedia data. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 708–716, Prague, Czech Republic, June 2007. Association for Computational Linguistics.
- [4] Endang Djuana, Yue Xu, Yuefeng Li, and Audun Josang. Ontology learning from user tagging for tag recommendation making. In *Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology - Volume 03, WI-IAT '11*, pages 310–313, Washington, DC, USA, 2011. IEEE Computer Society.
- [5] Christiane Fellbaum. *WordNet: An Electronic Lexical Database*. Bradford Books, 1998.
- [6] Evgeniy Gabrilovich and Shaul Markovitch. Overcoming the brittleness bottleneck using Wikipedia: enhancing text categorization with encyclopedic knowledge. In *In Proceedings 21st National Conference on Artificial intelligence - Volume 2, AAAI'06*, pages 1301–1306. AAAI Press, 2006.
- [7] Evgeniy Gabrilovich and Shaul Markovitch. Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In *In Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 1606–1611, 2007.
- [8] Andrés García-silva, Oscar Corcho, Harith Alani, and Asunción Gómez-pérez. Review: review of the state of the art: Discovering and associating semantics to tags in folksonomies. *Knowl. Eng. Rev.*, 27(1):57–85, February 2012.
- [9] Feng Ge, Yi He, Jin Liu, Xiaoming Lv, Wensheng Zhang, and Yiqun Li. A reinforcement learning based tag recommendation. In Yinglin Wang and Tianrui Li, editors, *Practical Applications of Intelligent Systems*, volume 124 of *Advances in Intelligent and Soft Computing*, pages 251–258. Springer Berlin Heidelberg, 2012.

- [10] G. Giles. Internet encyclopaedias go head to head. In *Nature*, 438, 900-901, 2005.
- [11] Jorge Gracia and Eduardo Mena. Web-based measure of semantic relatedness. In *Proceedings of the 9th international conference on Web Information Systems Engineering*, WISE '08, pages 136–150, Berlin, Heidelberg, 2008. Springer-Verlag.
- [12] Paul Heymann, Daniel Ramage, and Hector Garcia-Molina. Social tag prediction. In *31st Annual International ACM SIGIR Conference (SIGIR'08)*, July 2008.
- [13] Lan Huang, David Milne, Eibe Frank, and Ian H. Witten. Learning a concept-based document similarity measure. *J. Am. Soc. Inf. Sci. Technol.*, 63(8):1593–1608, August 2012.
- [14] Scott Jaschik. A stand against Wikipedia. In *Inside Higher Ed.* 2007. <http://www.insidehighered.com/news/2007/01/26/wiki>.
- [15] Robert Jäschke, Leandro Marinho, Andreas Hotho, Lars Schmidt-Thieme, and Gerd Stumme. Tag recommendations in folksonomies. In *Proceedings of the 11th European conference on Principles and Practice of Knowledge Discovery in Databases*, PKDD 2007, pages 506–514, Berlin, Heidelberg, 2007. Springer-Verlag.
- [16] Alistair Kennedy and Graeme Hirst. Measuring semantic relatedness across languages. In *xLiTe: Cross-Lingual Technologies workshop collocated with NIPS 2012*, 2012.
- [17] Aniket Kittur and Robert E. Kraut. Harnessing the wisdom of crowds in Wikipedia: quality through coordination. In *Proceedings of the 2008 ACM Conference on Computer Supported Cooperative Work*, CSCW '08, pages 37–46, New York, NY, USA, 2008. ACM.
- [18] Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N. Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick van Kleef, Sören Auer, and Christian Bizer. Dbpedia - a large-scale, multilingual knowledge base extracted from Wikipedia. *Semantic Web Journal*, 2013. Under review.
- [19] Marek Lipczak. Tag recommendation for folksonomies oriented towards individual users. In *Proceedings of the ECML/PKDD 2008 Discovery Challenge Workshop, part of the European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases*, 2008.
- [20] Marek Lipczak, Yeming Hu, Yael Kollet, and Evangelos Milios. Tag sources for recommendation in collaborative tagging systems. In Folke Eisterlehner, Andreas Hotho, and Robert Jschke, editors, *ECML PKDD Discovery Challenge 2009 (DC09)*, volume 497 of *CEUR-WS.org*, pages 157–172, September 2009.

- [21] Marek Lipczak and Evangelos Milios. Efficient tag recommendation for real-life data. *ACM Trans. Intell. Syst. Technol.*, 3:2:1–2:21, October 2011.
- [22] Jun Liu and Sudha Ram. Who does what: Collaboration patterns in the Wikipedia and their impact on article quality. *ACM Trans. Manage. Inf. Syst.*, 2(2):11:1–11:23, July 2011.
- [23] Olena Medelyan, Eibe Frank, and Ian H. Witten. Human-competitive tagging using automatic keyphrase extraction. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3 - Volume 3*, EMNLP '09, pages 1318–1327, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.
- [24] Olena Medelyan, Steve Manion, Jeen Broekstra, Anna Divoli, Anna-Lan Huang, and IanH. Witten. Constructing a focused taxonomy from a document collection. 7882:367–381, 2013.
- [25] Pablo N. Mendes, Max Jakob, Andrés García-Silva, and Christian Bizer. Dbpedia spotlight: shedding light on the web of documents. In *Proceedings of the 7th International Conference on Semantic Systems*, I-Semantics '11, pages 1–8, New York, NY, USA, 2011. ACM.
- [26] Rada Mihalcea. Using Wikipedia for automatic word sense disambiguation. In *North American Chapter of the Association for Computational Linguistics (NAACL 2007)*, 2007.
- [27] Rada Mihalcea and Andras Csomai. Wikify!: linking documents to encyclopedic knowledge. In *Proceedings of the Sixteenth ACM Conference on information and knowledge management*, CIKM '07, pages 233–242, New York, NY, USA, 2007. ACM.
- [28] David Milne and Ian H. Witten. An effective, low-cost measure of semantic relatedness obtained from Wikipedia links. In *Proceedings of AAAI*, 2008.
- [29] David Milne and Ian H. Witten. Learning to link with Wikipedia. In *Proceedings of the 17th ACM conference on Information and knowledge management*, CIKM '08, pages 509–518, New York, NY, USA, 2008. ACM.
- [30] Gilad Mishne. Autotag: a collaborative approach to automated tag assignment for weblog posts. In *Proceedings of the 15th International Conference on World Wide Web*, WWW '06, pages 953–954, New York, NY, USA, 2006. ACM.
- [31] Vivi Nastase and Michael Strube. Transforming Wikipedia into a large scale multilingual concept network. *Artif. Intell.*, 194:62–85, January 2013.
- [32] Simon Overell, Börkur Sigurbjörnsson, and Roelof van Zwol. Classifying tags using open content resources. In *Proceedings of the Second ACM International Conference on Web Search and Data Mining*, WSDM '09, pages 64–73, New York, NY, USA, 2009. ACM.

- [33] Simone Paolo Ponzetto and Michael Strube. Knowledge derived from Wikipedia for computing semantic relatedness. *J. Artif. Int. Res.*, 30(1):181–212, October 2007.
- [34] P. Schmitz. Inducing ontology from Flickr tags. In *Proc. of the Collaborative Web Tagging Workshop (WWW '06)*, May 2006.
- [35] Börkur Sigurbjörnsson and Roelof van Zwol. Flickr tag recommendation based on collective knowledge. In *Proceedings of the 17th International Conference on World Wide Web, WWW '08*, pages 327–336, New York, NY, USA, 2008. ACM.
- [36] Michael Strube and Simone Paolo Ponzetto. Wikirelate! computing semantic relatedness using Wikipedia. In *Proceedings of the 21st National Conference on Artificial Intelligence*, pages 1419–1424. AAAI Press, 2006.
- [37] V. Subramaniaswamy and S. Chenthur Pandian. Effective tag recommendation system based on topic ontology using Wikipedia and Wordnet. *Int. J. Intell. Syst.*, 27(12):1034–1048, December 2012.
- [38] Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. Yago: A Core of Semantic Knowledge. In *16th international World Wide Web conference (WWW 2007)*, New York, NY, USA, 2007. ACM Press.
- [39] Petros Venetis, Georgia Koutrika, and Hector Garcia-Molina. *On the selection of tags for tag clouds*. WSDM '11. ACM, New York, NY, USA, 2011.
- [40] Fernanda B. Viegas, Martin Wattenberg, Jesse Kriss, and Frank van Ham. Talk before you type: Coordination in Wikipedia. In *Proceedings of the 40th Annual Hawaii International Conference on System Sciences, HICSS '07*, pages 78–, Washington, DC, USA, 2007. IEEE Computer Society.
- [41] Nick Cercone Calvin Thomas Vlado Keselj, Fuchun Peng. N-gram-based author profiles for authorship attribution. In *Association for Computational Linguistics*, 2003.
- [42] Pu Wang, Jian Hu, Hua-Jun Zeng, Lijun Chen, and Zheng Chen. Improving text classification by using encyclopedia knowledge. In *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*, pages 332–341, 2007.
- [43] Robert West, Joelle Pineau, and Doina Precup. Wikispeedia: an online game for inferring semantic distances between concepts. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence, IJCAI'09*, pages 1598–1603, San Francisco, CA, USA, 2009. Morgan Kaufmann Publishers Inc.
- [44] Robert West, Doina Precup, and Joelle Pineau. Completing wikipedia's hyperlink structure through dimensionality reduction. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management, CIKM '09*, pages 1097–1106, New York, NY, USA, 2009. ACM.

- [45] Robert Wetzker, Carsten Zimmermann, Christian Bauckhage, and Sahin Albayrak. I tag, you tag: translating tags for advanced user models. In *WSDM*, pages 71–80, 2010.
- [46] Wikipedia. Tag (metadata) — Wikipedia, the free encyclopedia, 2013. [Online; accessed 22-May-2013] [http://en.wikipedia.org/wiki/Tag_\(metadata\)](http://en.wikipedia.org/wiki/Tag_(metadata)).
- [47] Wikipedia. Wikipedia — Wikipedia, the free encyclopedia, 2013. [Online; accessed 22-May-2013] <https://en.wikipedia.org/wiki/Wikipedia>.
- [48] Wikipedia. Wikipedia:manual of style/lead section— Wikipedia, the free encyclopedia, 2013. [Online; accessed 22-May-2013] http://en.wikipedia.org/wiki/Wikipedia:Lead_section#Opening_paragraph.
- [49] Wikipedia. Wordpress- Wikipedia, the free encyclopedia, 2013. [Online; accessed 05-July-2013] <http://en.wikipedia.org/wiki/WordPress>.
- [50] Ian H. Witten, Gordon W. Paynter, Eibe Frank, Carl Gutwin, and Craig G. Nevill-Manning. Kea: practical automatic keyphrase extraction. In *Proceedings of the fourth ACM conference on Digital libraries, DL '99*, pages 254–255, New York, NY, USA, 1999. ACM.
- [51] Ziqi Zhang, Anna Lisa Gentile, and Fabio Ciravegna. Recent advances in methods of lexical semantic relatedness 2013 a survey. *Natural Language Engineering*, FirstView:1–69, 5 2013.
- [52] Ya-Tao Zhu, Sheng-Hua Liu, Xue-Qi Cheng, Yue Liu, Yuan-Zhuo Wang, and Jin-Gang Liu. Using wordnet-based neighborhood for improving social tag recommendation. In *Proceedings of the 8th International Conference on Intelligent Computing Theories and Applications, ICIC'12*, pages 221–228, Berlin, Heidelberg, 2012. Springer-Verlag.