INTERNATIONAL JOURNAL OF COMPUTERS
Issue 3, Volume 1, 2007

A Web 2.0 Tag Recommendation Algorithm Using Hybrid ANN Semantic Structures

Sigma On Kee Lee and Andy Hon Wai Chun

Abstract — This paper presents a novel approach to automatic tag recommendation for weblogs/blogs. It makes use of collective intelligence extracted from Web 2.0 collaborative tagging as well as word semantics to learn how to predict the best set of tags to use, using a hybrid artificial neural network (ANN). Web 2.0 represents the "second generation" of Web applications with new technologies that allow people to work, collaborate and share knowledge in innovative manners. An important characteristic of Web 2.0 is that it embraces the power of the web to harness collective intelligence of its users. In particular, the rise of blogging is one of the most highly touted phenomena of the Web 2.0 era. Weblog or blog is an important innovation that makes it easy to publish information, engage discussion and form communities on the Internet. The use of "tags" has recently become very popular as a mean of annotating and organizing everything on the web, from photos, videos and music to blogs. The use of tags has originally produced a "folksonomy", a system in which the meaning of a tag is determined by its use among the community as a whole. Unfortunately, tagging is a manual process and limited to the users' own knowledge and experience. A blog author might not be aware that there may be more accurate or popular tags to describe his/her content. Collaborative tagging use collective intelligence by observing how different users tag similar content. Our ANN-based algorithm learns this collective intelligence and then reuses it to automatically generate tag suggestions for blog authors based on the semantic content of blog entries.

Keywords—Web 2.0, Blog, Collaborative Tagging, Intelligent Systems, Machine Learning.

1. INTRODUCTION

Web 2.0 represents the "second generation" of Web applications with new technologies that allow people to work, collaborate and share knowledge in innovative manners. An important characteristic of Web 2.0 is that it embraces the power of the web to harness collective intelligence of its users. In particular, the rise of blogging is one of the most highly touted phenomena of the Web 2.0 era. Weblog or blog is an important innovation that makes it easy to publish information, engage discussion and form communities on the Internet. Weblogs or blogs are web sites consisting of content (or "entries") that are dated and displayed in reverse chronological order. Many people think of blogs as online public journals. Its easy-of-use has made it the leading decentralized publishing technology in the Web 2.0 world. Basically anyone with access to the Internet can now publish content, allowing anyone to quickly and easily disseminate their opinions to a very wide audience. The contents of blogs may vary from personal journals, markets or product commentaries, to news and current affairs. In addition, the number of blogs has also grown exponentially to estimated tens of millions to over a hundred million blogs by the end of 2006. Therefore, creating technologies that allow people to easily and quickly find high quality blog content that they are interested in is a very important but difficult task. Our research in automatic tag recommendation is a way to maximize the chances that blog contents will reach those potentially interested in it through more accurate tagging that makes use of collective intelligence of the billion Internet users.

The tens of millions of blogs in the world are interlinked to form what is known as the blogosphere. To support this Web 2.0 phenomenon, special technologies such as custom blog search, analysis engines, and systems that employ specialized information retrieval techniques were invented, all with the aim to make finding information in the gigantic blogosphere easier. In particular, tagging is a popular technique to facilitate the organization and retrieval of blog entries. Tags can be thought of as key words or key phrases attached to documents or objects (blog entries, photos, music, or videos) to help describe those objects. The use of keywords is of course not new. It has been used in categorizing or indexing in the traditional library systems. Keywords provide an easy way to categorize, search, and browse content. Tagging is a term to describe the new set of Web 2.0 technologies to support keywords online, such as collaborative tagging.

One of the characteristics of Web 2.0 collaborative tagging is the ingenious use of "open vocabularies" instead of a formalized ontology. Tags are not selected by professional annotators, but by the average content authors themselves. Although this may sound counter-intuitive, but tags created organically without any centralized control is more interesting that a formalized ontology as it harnesses the collective intelligence of hundred of millions of people! With a rich pool of tags, tags can group documents into broad categories [5] that can solve the problem of synonyms, pluralization and misspelling by using the shared knowledge of other users. The use of tags has organically produced a "folksonomy" [17], [8], short for "folk taxonomy", a system in which the meaning of a tag is determined by its use among the community as a whole. Technorati.com is one of the most popular sites related to the tagging of blogs, while sites like furl.com and del.icio.us help

users collaborate on tagging webpages. Flickr.com is an example of using tags to describe photos.

In our research, we describe a novel approach to automatic tag suggestion that makes use of collective intelligence from collaborative tagging combined with semantic-driven ANN learning to produce a set of most relevant tags for the user to select from. The result of ANN learning is a network that encodes richness and subtleties in mapping content to tags. The results produced will be a list of weighted or prioritized tags that are most relevant to the given blog. In simple terms, our system basically learns how to tag by observing how other humans tag their own blog content. This learned knowledge is then used to automatically generate tag suggestions for new blog entries.

2. Research Background

Tagging is a way to organize content through labeling. It tries to associate meaning to online content such as blogs, photos, videos and music. Tags are keywords or key phrases that can be associated with content as a simple form of metadata. To a computer, tags serve as a set of atomic symbols that are associated with an object. Unlike the keyword systems used in libraries in which users select keywords from a predefined list, users can choose any string to use as a tag. The idea of using tags to annotate content recently become quite popular within the blogging community. The idea of tagging is not new, photo-organizing tools have used tagging for ages, and HTML has had the ability to allow META keywords to describe a document since HTML 2.0 [4] since 1996.

In a tagging system, an item of content will typically have one or more “tags” associated with it. Tagging software automatically provides links to other items that share the same tag, or even to specified collections of tags (via AI clustering). This allows multiple "browseable paths" through the content to facilitate search and retrieval of related items.

While using tags is flexible and easy, tagging is not without its drawbacks. Tags are just strings without any semantic meaning. For example, the tag “apple” might refer to the fruit, or Apple Computer. The lack of semantic distinction in tags can lead to inappropriate connections between items. In addition, selection of tags is highly dependent on the individual. Different people may use drastically different terms to describe similar content. A case in point, items related to a version of Apple Computer's operating system might be tagged both “OSX,” “Tiger,” and possibly many other terms. Users of tagging systems have to make “intelligent guesses” to determine the most appropriate tag to use or search for.

Collaborative tagging offers an interesting alternative to current efforts. Collaborative tagging is portrayed as a kind of shared knowledge. It allows users to share their tags with other users. It allows users to publicly tag and share content, so that they can categorize information for themselves, and also makes browsing information categorized by others a lot easier.

Tag classification, and the concept of connecting sets of tags between web/blog servers, has lead to the rise of folksonomy classification over the internet. These large-scale folksomonies are formed because knowledgeable or frequent users of tagging systems will have experience in searching and using “tag terms” within these systems. These knowledgeable users tend to find and use “popular” tags so that it will be easier to form connections with other related items. Through this manner, folksomonies evolve organically through a process of group consensus.

In collaborative filtering, patterns in user preferences are mined to make recommendations based on things like users’ opinions — individuals who have shared taste in past will continue to do so. Examples include Ringo [16] and GroupLens [13] as well as e-commerce sites such as Amazon.com. Fab [2] combined content-based and collaborative recommendation. However, collaborative filtering suffers from some well-known limitations [15], such as, the sparsely of user profiles, the latency associated with pre-computing similarity information, and the difficulty in generating predictions about new items. Some of these limitations will also apply to the system presented here.

3. Related Works

Although Web 2.0 is emerged in past few years, some of the researchers have worked with some related research including blog and tag.

3.1. AutoTag

AutoTag[7] from Gilad Mishne, describe a tool which suggests tags for blog posts using collaborative filtering methods. AutoTag generates a small number of tags for a given weblog post. The blogger then reviews the suggestions, selecting those which he/she finds useful. AutoTag also improves its quality. First, by increasing the chance that blog posts will be tagged in the first place, and second by offering relevant tags that may have not been applied.

Once the user supplies a blog post, posts which are similar to it are identified. Next, the tags assigned to these posts are aggregated, creating a ranked list of likely tags. After that, AutoTag filters and reranks this tag list; finally, the top-ranked tags are offered to the user, who selects the tags to attach to the post.

3.2. TagAssist

Another similar research to AutoTag is TagAssist[18]. TagAssist improve the AutoTag system by increase the quality

of suggested tags. It performs lossless compression over existing tag data. AutoTag finds similar tagged posts and suggests some set of the associated tags to a user for selection. While TagAssist uses a similar technique, they have improved on AutoTag's performance by introducing tag compression and case evaluation to filter and rank tag suggestions.

After tag compression, a Tag Suggestion Engine (TSE) is used to suggest a set of tags to a user. TSE operates on the principal of leveraging existing tagged data to provide appropriate tag suggestions for new content. The solutions for new cases are determined by retrieving similar, solved cases from a large corpus of labeled examples and applying those solutions (or transformations of those solutions) to the new problem. Mishne's AutoTag system takes a very similar approach to tag recommendation.

3.3. Yahoo! tag suggestion on URL

Yahoo! also has a similar research on tag suggestion [21]. This system generates tag suggestions given a URL. It uses collaboratively filtering to automatically identify high quality tags for users, leveraging the collective wisdom of Web users. Collaborative tagging techniques suggest tags for an object based on what other users use to tag the object, and a reputation score for each user based on the quality of the tags contributed by the user. Introducing the notion of "virtual" users, the tag suggestion algorithm incorporates not only user-generated tags but also other sources of tags, such as tags auto-generated via content-based or context-based analysis.

4. AUTOMATIC TAG SUGGESTION ALGORITHM

Our AT:tag automatic tag suggestion algorithm, consists of 2 key phases – the Training Phase which involves ANN learning, and the Execution Phase which is responsible for the tag suggestion generation.

4.1. Training Phase

In the Training Phase, we first use robots to crawl the web to collect blogs that have already been manually tagged. Some of these blogs will become part of the training set while others will be used for testing. The main objective of the training phase is to learn how blog content is associated to tags. To keep our experiments manageable, we will limit our robots to focus on subsets of the blogosphere. For example, blogs related to "hiking" only or blogs related to "rock climbing."

The algorithm for the Training Phase consists of 3 main stages:

Stage 1: Keyword Extraction
Stage 2: Semantic Processing
Stage 3: ANN Learning

4.2. Stage 1: Keyword Extraction

We use both statistically method and the lexical resources method to perform keyword extraction. This is further divided into 3 steps:

Step 1: extract single keywords using TFIDF score. (statistically based)

Step 2: compute co-occurrence frequency (statistically based)

Step 3: check bigrams using WordNet (lexical resources based)

Step1: extract single keywords using TFIDF score. The TFIDF score [14] is calculated by the following formula:

$$TFIDF(word) = termFreq(word)\,x\log\left(\frac{|corpus|}{DocFreq(word)}\right)$$

where:

termFreq(word) indicates the number of times that a word occurs in the blog entry being processed. It is computed using:

$$termFreq(word) = \frac{n_i}{\sum_k n_k}$$

$|corpus|$ indicates the total number of message in each user.

DocFreq(word) indicates how frequently a word appears in that corpus.

The TFIDF will score individual words within text documents in order to select concepts (represented by keywords) that accurately represent the content of the document. This will cause commonly used words to have a very low TFIDF score, and rare words to have a high TFIDF score. Because the TFIDF score is based purely on how frequent a single word appears in the text, we will need to supplement this with information on a word's relevance in terms of other words.

Step2: compute co-occurrence frequency in the same blog. In our AT:tag algorithm, the keyword extraction stage will also consider bigrams selection where two continuous words are considered as one item. Co-occurrence frequencies are computed for the extracted keywords. In our experiments, we filter out word-pairs that have frequency less than 5. In particular, we try to extract special bigrams that do not appear in our dictionary. Higher frequency bigrams will have higher weightings in our algorithm.

Step3: check bigrams using WordNet. WordNet [11] is a freely available electronic dictionary developed by the Cognitive Science Laboratory at Princeton University. It has been used for text summarization [3] and other natural language processing tasks. In this project, we use WordNet to help with our bigram selection [9]. When bigrams are extracted from the blog, we search WordNet to check if the bigrams are

common phrases or not.

The result of "Stage 1: Keyword Extraction" is a set of keywords or key phrases to represent the blog content. In "Stage 2: Semantic Processing," we further enrich this representation by supplementing the keywords/phrases with semantics.

4.3. Step 2: Semantic Processing

After generating a set of keywords/phrases for a blog, our AT:tag algorithm then use WordNet to extract semantic information. This process helps provide lower-level semantics to our representation and allow us to relate blog with different set of keywords but with similar "meanings."

The design of WordNet was inspired by current psycholinguistic theories of human lexical memory. Words are organized into synonyms sets (synsets) each representing one underlying lexical concept. For example: the set of lexical items {car, automobile auto, machine, motorcar} constitutes one synset representing the concept corresponding to the gloss/definition: "4-wheeled motor vehicle; usually propelled by an internal combustion engine". Different semantic relations link synsets together into different hierarchies (e.g. IS-A and PART-OF relations).

For each keyword/phrase generated from our Stage 1 processing, we select the first synset produced from WordNet. The resulting synset information is used as additional semantics to describe a blog. For example, the keyword "computer" is related to this synset: {computer, computing machine, computing device, data processor, electronic computer, information processing system}. The collection of synset produced from our keywords/phrases is used to represent the semantic content of a blog.

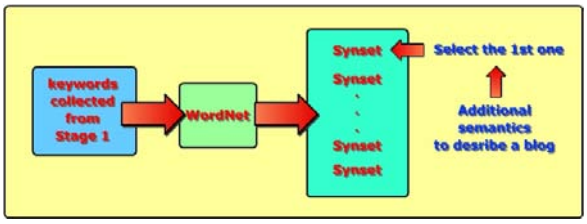

Figure 1 Stage 2: Semantic Processing

4.4. Step 3: ANN Learning

Learning in AT:tag is performed using an artificial neural network (ANN). The structure of the network is shown below:

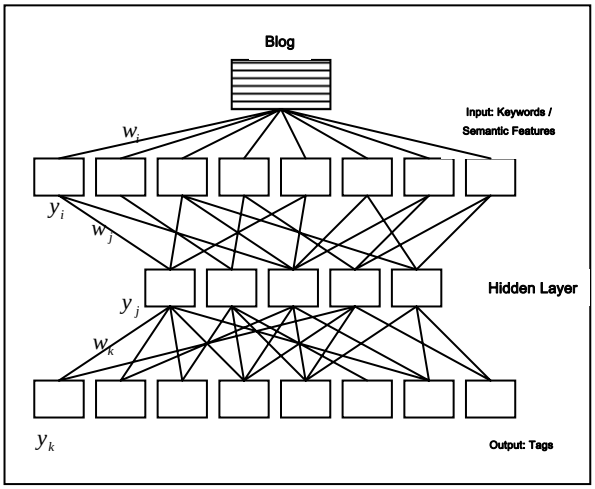

Figure 2. The structure of the ANN used for learning

There are three layers in our ANN - input layer is the feature layer with weighting, one hidden layer, and an output layer which represents the predicted tags. ANN is used to learn the association of the keywords/phrases and semantic features to tags. Learning is needed as the selection of tags can be influenced by several different features. The weights learned through ANN determine the contribution of each feature to the selection of a tag. For the learning algorithm, we use the traditional backpropagation algorithm [19].

The "Stage 3: ANN Learning" is further divided into 4 main steps:

Step 1: Initialize Network
Step 2: Compute Errors
Step 3: Back propagate the errors
Step 4: Adjust weightings (learning)

Step 1: Initialize Network. The following are the ANN initialization procedures. The learning algorithm is described after that.

Procedure 1: w_i initial value: normalized feature occurrence frequency

keyword/key phrase node:

$$\begin{cases} 0 : b\log\_does\_not\_have\_feature \\ 1 : b\log\_have\_that\_feature \end{cases}$$

$$y_i = g(w_i) = \begin{cases} 0 : no\_feature \\ w_i : have\_feature \end{cases}$$

Procedure 2: w_j initial value: random number value [0, 1]

$$y_j = f\left(\sum w_j y_i\right)$$

Procedure 3: w_k initial value: random number value [0, 1]

$$y_k = f\left(\sum w_k y_j\right)$$

where f(x) = Sigmoid function = $\frac{1}{1+e^{-x}}$

In our AT:tag algorithm, the output of our ANN will generate a set of suggested tags, each with a priority weighting. Since there are more than one output nodes, we modify the standard backpropagation algorithm using a hybrid approach.

To produce a prioritized list of tags to suggest, the output is not only a single node. After evaluating the activation function, each output node (each representing one tag) will have an activation value. The higher the value, the higher the ranking will be for a tag in our prioritized suggestion list. Since there are multiple outputs, the backpropagation error calculation will be different. We map the multiple errors to a single error using a regression function. The number of actual tags present in a blog is of course fixed. However, the predicted outputs from the ANN consist of the entire set of tags stored in AT:tag. Therefore, we need to select the same amount of outputs for both the predicted output and actual output. We select the highest N predicted outputs for the N actual outputs for comparison. We then normalize using the highest activation value node in the predicted output, because when there is single actual output, no weighting need to be changed. For multiple outputs, except the highest value node, other nodes must below the activation level of 1. Therefore, although the number of predicted outputs is same as the actual, an error may still exist since the activation level may be too low. If so, the network will continue with its training cycle.

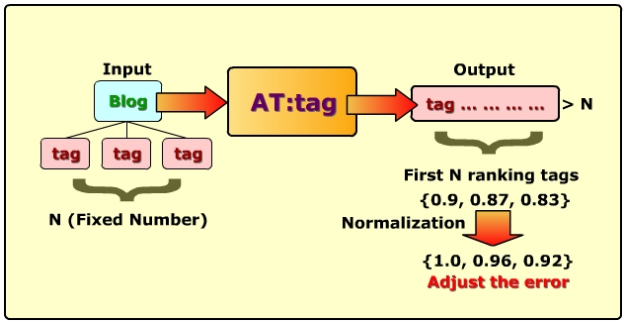

Figure 3. Compare the same number of node with the input tag.

Therefore, we have some basis for matching predicted output versus actual. If the predicted output exists in the actual output, then the error is positive. Otherwise, if the predicted output does not exist in the actual output, then the error is negative. The reason for having different sign of error is to adjust the learning point to move to a more reasonable direction. We then add up all the N error to produce the overall error. Using this regression function, if all tags matched, the error will be small. We will skip the learning progress if the error is below a pre-defined threshold. Only significant errors will trigger learning and the changing of link weights.

Step 2: Compute Errors. The following are the procedures involved in computing the errors during back propagation:

Procedure 1: top $\vec{x}$ predicted outputs nearest to 1 (highest ranking nodes).

Procedure 2: compare to original $\vec{z}$ tags for that blog

Procedure 3: select same number (N) of output nodes $\vec{x}$ as actual $\vec{z}$.

Procedure 4: normalize output node value (using highest value as 1)

- to avoid changing weighting if only have 1 actual output, i.e. N = 1

Procedure 5: actual output of the tag

- $= \begin{cases} 1 : if\ \_tag\_exists \\ 0 : if\ \_tag\_not\_exists \end{cases}$

Procedure 6: error is then calculated by: $\delta_k = \vec{\gamma}(\vec{x} - \vec{z})$

- where $\gamma = \begin{cases} +1 : if\ \_desire\_tag\_match\_actual\_tag \\ -1 : if\ \_deaire\_tag\_not\_match\_actual\_tag \end{cases}$

Procedure 7: if error is less than a threshold T, learning procedure will be skipped.

Step 3: Back propagate the error. Based on the above the learning parameters are computed as:

$$\delta_i = \sum w_j \delta_j$$

Step 4: Adjust weightings (learning). The weights of the links are then adjusted according to these formulae for each layer of the ANN:

$$w_i' = w_i + \eta \delta_i \frac{\partial w_i}{\partial \sum w_i} x_i$$

$$w_j' = w_j + \eta \delta_j \frac{\partial y_j}{\partial \left(\sum w_j y_i\right)} y_i$$

4.5. Execution Phase

After the Training Phase has been completed, our AT:tag algorithm then makes use of the resulting ANN to automatically suggest tags during the Execution Phase. During this phase, the user submits a completed blog entry to AT:tag and gets a list of prioritized tag suggestions in return. When a blog entry is received by AT:tag, it first extracts keywords/phrases and semantic features to represent that blog entry. The extraction method is the same as the knowledge extraction in the Training Phase. After that, AT:tag uses the extracted features to activate the ANN. Results from the ANN are presented to the user as prioritized tag suggestion.

5. Results and Comparisons

5.1. Experiments

In our experiment, we first using Technorati API [20] (http://www.technorati.com/) for searching the following keywords:

- {ai, ajax, alone, appletag, apple, art, baby, book, bush, car, card, cat, christmas, comedy, computer, crazy, dairy, dog, dressing, education, environment, fire, fish, friends, games, google, government, happy, health, hiking, home, house, idol, internet, job, kiss, law, life, lonely, love, mobile, money, mountain, mp3, music, nature, news, play, pop, popular, robot, rock, sad, school, science, sleepy, snow, song, sport, star, sweet, tag, tagging, technology, telephone, tools, universe, web2.0, web tag, web, weblog, windows, word, world, youtube}.

The results are filtered so that only blogs written in English are returned. The result is analyzed and the first 500 permalinks from each of the target keywords are selected. Our experiments require full content of the blogs and their corresponding tags. For each of the permalinks, we extract the content part of the blog and its corresponding tags by analysis their HTML code. Out of 35417 links, we found 4401 pages that contained blogs with tags. We further divide the 4401 pages into training dataset and testing dataset with 2187 and 2214 data items respectively. The data files are processed to reduce the "noise" of the dataset. For example, we remove special characters and HTML tag comment from the HTML code. The blog content is then split into a series of keywords. For each of the training and testing data items, we extract keywords using our keyword extraction method. The frequencies of these keywords are computed to prepare them for use as inputs to our ANN. The following is an example of the resulting data item that represents a blog:

```
##Contents:
directx
3d
graphics
apis
madison
lockwood
designed
microsoft
highest
versions
introduction
vista
operating
direct3d
developers
hardware
introduced
windows
games
development
version
sophisticated
video
cards
ati
card
produce
makers
compatibility
computer
processing
unit
apollo
hosting
##Tags: 3d graphics,apis,directx,directx version
```

Figure 4. Sample of the data extracted from blog

```
1.BlogTagRobot
2.HtmlParserContent
3.ProcessDataFile
4.SplitData
    --> Training set, testing set
5.TFIDFList
    --> step1KeywordFile.txt
6.BigramList
    --> wordNetFile.txt
7.TermFreqList
    --> termFreqFile.txt
8.MergeList
    step1KeywordFile.txt + wordNetFile.txt --> keywordFile.txt
9.CalculateFrequency
10.WordNetProcess
11.NormalizeFreq
12.AttagLearning
```

Figure 5. Class flow of the ANN method

To reduce the number of inputs to the ANN, we only select the top 10 keywords represented to each blog for training:

```
##Contents:
love
hunter
documenting
happiness
happy
perceived
autonomy
trumps
determining
helpful
##Tags: happiness

##Contents:
built
pregnancy
tester
iphone
steve
coupled
ibaby
software
functional
03
##Tags: [technology],[software],[news],[apple],apple,news,software,technology

##Contents:
phone
cell
phones
codes
news
mobile
eu
ceil
roaming
charges
mobile phone
##Tags: uncategorized
```

Figure 5. Some of the training data input to the ANN

For our ANN experiments, after analyzing results from initial testing, we finalized the ANN design to have 100 hidden nodes and a learning factor of 2.0 as the training parameter. We use one input layer, one hidden layer and one output layer for the ANN architecture. The training stops when the accuracy is 0.5 or better. In our experiments, all the intermediate data are stored so that we can keep track of any part of the intermediate progress.

5.2. Results

In the training, the weighting between each layer are stored and automatically backup. Since the ANN training progress is time consuming and uses a lot of computational power, this approach allows us to restart the experiment at any point.

Figure 6. Sample of the file storing ANN weighting

The following is an example of tags suggested from the ANN method. We use a blog about “iPhone” as the input to the system.

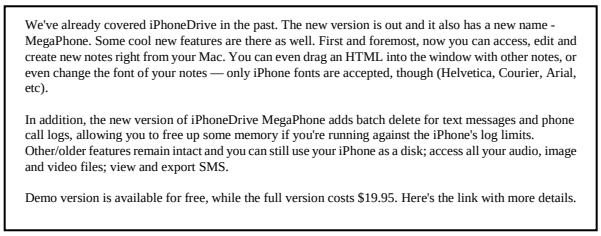

Figure 7. Sample blog data input to ANN

The tags suggested by the system are:

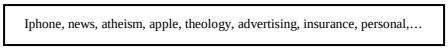

Figure 8. Sample tag generated for the blog

5.3. Analyze

We believe the main reason that the AT:tag algorithm works is that it makes use of collective intelligence provided in Web 2.0 collaborative tagging. Tags suggested are learned from this collective intelligence and will be more acceptable to the user. In addition, we capture subtleties and richness in blog content using semantic information provided in WordNet. The same mechanism allows us to handle differences in how people tag similar blogs as well as how people express similar ideas with different wordings. The collective intelligence of millions of blogs also allows us to reduce the chance of human errors in tagging.

To compare the accuracy of the AT:tag algorithm, we ran some accuracy measurements. The accuracy is computed by comparing original manually created tags for a blog with our automatically generated tags. Quite surprisingly, we found the accuracy in the training phrase to be lower than expected. The calculation of the accuracy is by:

$$\frac{number\_of\_tags\_matches\_the\_original\_blog's\_tag}{total\_number\_of\_tags\_in\_one\_training\_data}$$

We believe there may be several reasons for this discrepancy. Firstly, some blogs may talk about a mixture of several topics at the same time and will confuse the ANN. Another reason is in the keyword extraction process. Spam and advertisements further degrades the ANN learning process. The following explores these issues further.

5.3.1. Mixture of subtopics in data file

By understanding the efficient of the algorithm, we have been analysis the data file that is generated by the robot crawler. Our robot crawler uses the Technorati API (http://technorati.com/) to search for blog contents. The search is restricted to English blogs. The results were analyzed to retrieve the first repeated permalinks for each of the target keywords. It is because our experiment requires full content of the blogs and their corresponding tags. Thus, for each of the permalinks, we extract blog contents and tags by others methods. First, the excerpt and permalink of each blog are given from the Technorati API. Then, we explore the HTML from the permalink and using the excerpt to search the correct starting point of the real content. The search will stop when seeking some cases of the keyword string (e.g. Post Comment, This entry was posted … etc). We extract the content between the starting point and the ending point of the HTML for our real content. Unfortunately, different blog users, different blog server/domain have different style in writing their content. It is very difficult for filtering the noise that affect the true real content. The noises are including the advertisement, HTML coding, JavaScript … etc. This can be partly solved by increasing the number of cases. However, noise from incorrect content identification leads to wrong ANN input and will affect learning quality.

Moreover, blogs are really a mixture of subtopics [10]. The content of weblogs often includes personal experiences, thoughts and concerns. As a result, blog document often contains a mixture of distinct subtopics or themes. In addition, some blog content is spam and not real content. For future research, we plan to investigate algorithms to detect and remove these spam blogs [12]. For example, when we download the content by using Technorati API from the “keyword search” function, we may have the following result returns.

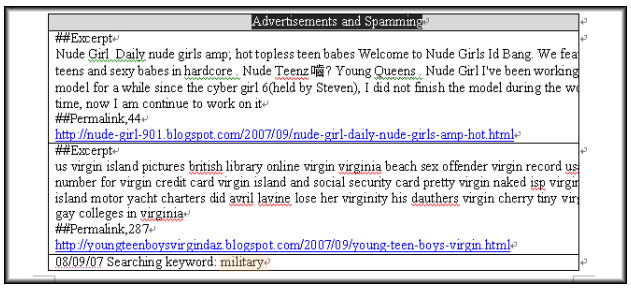

Figure 9. Advertisements and Spamming

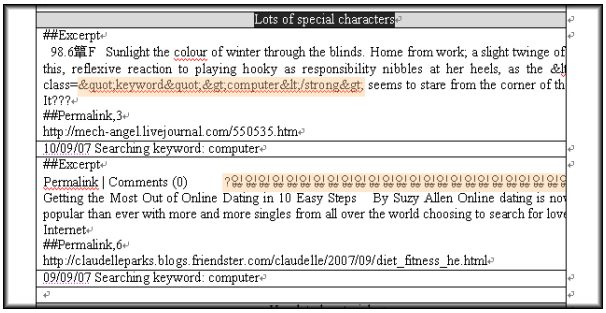

Figure 10. Lots of special characters

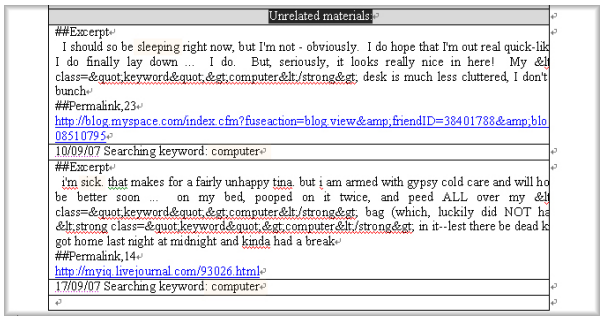

Figure 11. Unrelated materials

The above examples show that there are many noises that affecting the quality of data files – advertisement and spam, special characters, and unrelated materials … etc. Unfortunately, these noises appeared frequently in the blogosphere.

5.3.2. Failure in keyword extraction

AT:tag uses TFIDF score for keyword extraction. TFIDF will score individual words within text documents in order to select concepts (represented by keywords) that accurately represent the content of the document. This will cause commonly used words to have a very low TFIDF score, and rare words to have a high TFIDF score. The TFIDF score is based purely on how frequent a single word appears in the text. However, TFIDF might not work that well for blogs with similar topics. In our experiments, blogs are retrieved using the same keywords. With the same keywords, similar wordings may appear in different blogs. For example, with keyword "computer," the blog may include other frequently associated keywords such as "program". In the data file set, all blogs are downloaded using the same keywords "computer". Therefore, "program", "computer" are very important keywords and appeared in every blog entry. From the formula for calculating TFIDF:

$$TFIDF(word) = termFreq(word) \times \log\left(\frac{|corpus|}{DocFreq(word)}\right)$$

, these important keywords will have a very low score, since they will be treated as unimportant words like: "is", "am", "a", "were". As a result, keywords extracted by TFIDF from different blogs will be unique and not related to other blogs.

5.3.3. Enormous input/output in the ANN

Because of the TFIDF scores, inputs to the ANN are mostly different unique keywords. As a result, accuracy and quality of learning will be affected.

Besides TFIDF, the ANN is affected by synonyms - multiple tags having the same meaning. This occurs often in the blogosphere. In addition, tags themselves may contain spam words. All these noises affect the resulting generated tags.

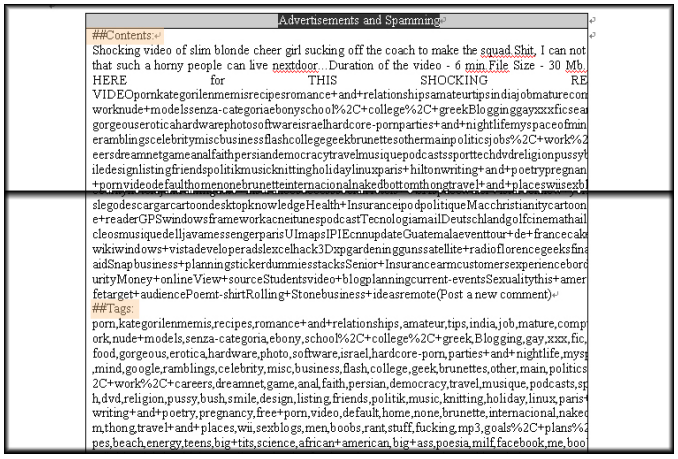

Figure 12. Advertisements and Spamming occur in both blog content and corresponding tags

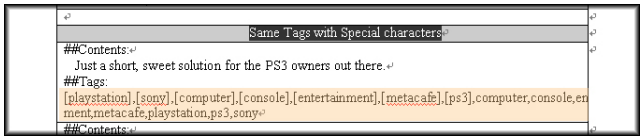

Figure 13. Same Tags with Special Characters

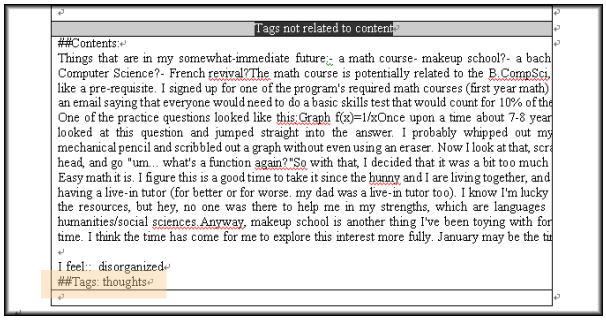

Figure 14. Tags not related to content

All the challenges mentioned above provide new directions to further extend this research.

5.4. Comparisons

In comparison, there is an existing weblog tagging systems called AutoTag [7] which finds the most similar blogs and then collect all the tags in those blogs for ranking and filtering. The disadvantage of this approach is that tags which are not in similar blog entries will not be considered. It cannot suggest new tags if the tags are not already used in one of the similar blogs. In our ANN method, all tags related to the semantic content of the blog will be proposed regardless of whether that set of tags have been used in another blog before or not.

Another related work that analyzes blogs and give tag suggestion is TagAssist [18]. TagAssist improves the AutoTag system by increasing the quality of suggested tags. It performs lossless compression over existing tag data. It is because it uses similar technology for the tag suggestion engine. Their system has the same disadvantage as AutoTag; tags that are not in the similar blog entries will not be considered. However, it improves the quality of tags and eliminates the unnecessary and duplicate tags to user.

Moreover, more related work that parsing each post content for tags are (C H. Brooks and N. Montanez) [6] and Bumpzee [1]. They try to parse for tags by analyzing blog content. In our ANN method, we try to relate blog content to their corresponding tags.

Furthermore, Yahoo! [21] also has a different approach for collaborative tag suggestions. In their method, they are using greedy heuristic approach while we are using an ANN learning approach for the tag suggestion. The algorithm emphasizes the correlation of tags and the "reputation" of the user. While our method uses online dictionary and neighbor documents to enrich the information extracted from blog content. Our AT:tag system will adjust the weighting within the ANN and do not need to store the score of every tag with each object. ANN provides a proven approach for collaborative tag suggestion.

6. Conclusion

We believe our research is insightful because it explores how we leverage on Web 2.0 to create a new feature that benefits the blogosphere – automatic tag suggestion generation. Our novel AT:tag algorithm uses a hybrid ANN to learn subtle mappings between rich semantic features and tags. Our algorithm leverages on the vast amount of collective intelligence that is available in Web 2.0 collaborative tagging to produce results that are in resonance with other users.

The current accuracy rate of AT:tag is affected by many sources of noise. As future research, we plan to add additional filters to reduce the amount of noise as well as enhance the TFIDF computation.

We believe our AT:tag algorithm can also be used in other situations where tags or metadata need to be generated. For example, our technology can also be used to automatically generate metadata for HTML files by analyzing the semantic content of a webpage.

References

[1] A. Beard, "Bumpzee Adds Amazing Feature – Autotagging." Available at http://andybeard.eu/2007/02/bumpzee-adds-amazing-feature-autotagging.html, 2007
[2] Balabanovic, M., and Shoham, Y. "Content-based, collaborative recommendation." Comm. ACM 40(3):67-72, 1997.
[3] Barzilay R., Elhadad M. "Using lexical chains for test summarization.", In Proceedings of Intelligent Scalable Text Summarization Workshop (ISTS), 1997.
[4] Berners-Lee, T., and Connolly, D. "Hypertext markup language specification – 2.0." Technical Report RFC 1866, MIT/W3C, 1996
[5] Christopher H. Brooks and Nancy Montanez. "An Analysis of the Effectiveness of Tagging in Blogs.", American Association for Artificial Intelligence Conference, 2006
[6] Christopher H. Brooks and Nancy Montanez. "Improved Annotation of the Blogosphere via Autotagging and Hierarchical Clustering.", Proc. of the 15th International World Wide Web Conference, Edinburgh, Scotland, 2006
[7] Gilad Mishne, "AutoTag: a collaborative approach to automated tag assignment for weblog posts", Proceedings of the 15th international conference on World Wide Web, 2006
[8] Emanuele Quintarelli, "Folksonomies: power to people.", paper presented at ISKO Italy-UniMIB Meeting, Mi, June 2005
[9] L. Bentivogli and E.Pianta. "Beyond lexical units: Enriching wordnets with phrasets.", In Proceedings of the Research Note Sessions of the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL'03), pages 67-70, Budapest, Hungary, April 2003
[10] Mei, Q., Liu, C., Su, H., and Zhai, C. "A probabilistic approach to spatiotemporal theme pattern mining on weblogs". In Proceedings of the 15th International Conference on World Wide Web (Edinburgh, Scotland, May 23 - 26, 2006). WWW '06. ACM Press, New York, NY, 533-542, 2006
[11] Miller, George A., Richard Beckwith, Christiane Fellbaum, Derek Gross, & Katherine J. Miller, "Wordnet: An on-line lexical database. Int'l Journal of Lexicography", 3(4):235–312, 1990.
[12] Pranam Kolari, Akshay Java, Tim Finin, Tim Oates, and Anupam Joshi, "Detecting Spam Blogs: A Machine Learning Approach", In Proceedings of the 21st National Conference on Artificial Intelligence (AAAI 2006), 2006
[13] Resnick, P.; Iacovou, N.; Suchak, M,; Bergstorm, P.; and Riedl, J. "GroupLens: An Open Architecture for Collaborative Filtering of Netnews.", In Proc. ACM Conf. on Computer Supported Cooperative Work, 175-186., 1994.
[14] Salton, G., and McGill, M. J. "An Introduction to Modern Information Retrieval." New York: McGraw-Hill, Inc., 1983
[15] Sarwar, B. M.; Karypis, G.: Konstan, J.; and Ridel, J. "Analysis of recommender algorithms for e-commerce.", In Proc. 2nd ACM E-Commerce Conf. (EC'00), 2000.
[16] Shardanand, U., and Maes, P. "Social Information Filtering: Algorithms for Automating "Word of Mouth".", In Proc. ACM CHI'95 Conf., 210-217, 1995.
[17] Shirky, C. "Folksonomy" Available at http://www.corante.com/many/archives/2004/08/25/folksonomy.php , 2004.
[18] Sood, S., Owsley, S., Hammond, K., and Burnbaum, L. "TagAssist: Automatic Tag Suggestion for Blog Posts". ICWSM, 2007
[19] Tariq Samad. "Back-propagation is significantly faster if the expected value of the source unit is used for update.", In International Neural Network Society Conference Abstracts, 1988.
[20] Technorati, www.technorati.com, 2006
[21] Xu, Z., Fu, Y., Mao, J., Su, D., "Towards the Semantic Web: Collaborative Tag Suggestions", Proceedings of Collaborative Web

Tagging Workshop at 15th International World Wide Web Conference (WWW 2006), 2006.