Inducing Taxonomy from Tags : An Agglomerative Hierarchical Clustering Framework

Xiang Li[1], Huaimin Wang, Gang Yin, Tao Wang, Cheng Yang, Yue Yu, and Dengqing Tang[2]

[1] National Laboratory for Parallel and Distributed Processing,
School of Computer Science,
National University of Defense Technology, Changsha, China
{shockleylee,jack.nudt,taowang.2005}@gmail.com
[2] College of Mechatronics Engineering and Automation,
National University of Defense Technology, Changsha, China
http://www.nudt.edu.cn

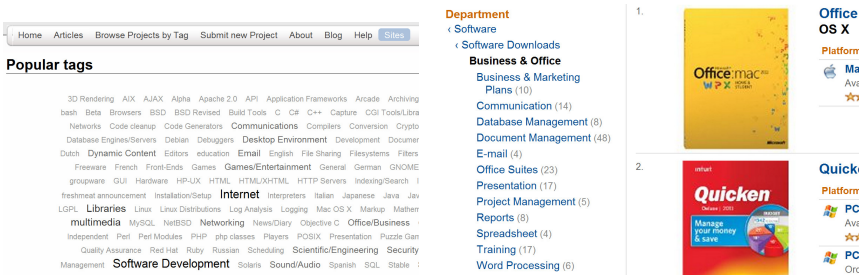
Abstract. By amassing 'wisdom of the crowd', social tagging systems draw more and more academic attention in interpreting Internet folk knowledge. In order to uncover their hidden semantics, several researches have attempted to induce an ontology-like taxonomy from tags. As far as we know, these methods all need to compute an overall or relative generality for each tag, which is difficult and error-prone. In this paper, we propose an agglomerative hierarchical clustering framework which relies only on how similar every two tags are. We enhance our framework by integrating it with a topic model to capture thematic correlations among tags. By experimenting on a designated online tagging system, we show that our method can disclose new semantic structures that supplement the output of previous approaches. Finally, we demonstrate the effectiveness of our method with quantitative evaluations.

Keywords: social tagging; semantics; tag taxonomy; tag generality; agglomerative hierarchical clustering; topic model

1 Introduction

Social tagging websites like Delicious[3] and Flickr[4] are becoming popular, witnessing a soaring click rate during recent years. By annotating web contents with free-form tags that they feel appropriate, users of social tagging websites are enabled to play the dual role of visitors and contributors simultaneously. Specifically, a user is allowed to search and browse resources (documents, images, URLs, etc.) through tags annotated by others and is free to add or delete tags whenever he or she wants.

[3] http://www.delicious.com/
[4] http://www.flickr.com/



(a) tag cloud in Freecode.com (b) taxonomy in Amazon.com

Fig. 1. different navigation mechanisms

For computer programs, tags are merely character strings with no meaningful relationships among them, which makes it hard to organize them into an informative structure. As depicted in Fig.1(a), most tagging websites manage tags in a flat tag cloud, where the font size of a tag is proportional to its frequency of usage, so users can have easy access to buzzword tags. Sometimes however, the desired tags are vague in users' head and may go beyond what is popular. A tag cloud could do little help in this situation, and tags should be better arranged and managed to satisfy the requirement. In retail websites, a user usually navigates through hierarchical taxonomy (as in Fig.1(b)) when her requirement is only a vague notion rather than a concrete keyword. So promisingly, taxonomy of tags will analogously facilitates navigation and searching in the now-booming social tagging websites. Besides, in the view of online data analysts, generating tag taxonomy is crucial for representing and understanding Internet folk knowledge. An ontology-like taxonomy is intrinsically a good knowledge structure which captures semantics of terms in a machine understandable way[1].

Previous literature[2][3] has addressed on how to generate tag taxonomy effectively. By using machine learning techniques, these researches achieved automated taxonomy generation, which could bootstrap and alleviate manual construction. However, new methods[4][5] keep emerging because resultant taxonomy never seemed satisfying enough. In an attempt to detect the bottleneck, we find that to deduce semantic generality of each tag is the most difficult and error prone (see Chapter 2 Related Work). To meet such challenge, we propose a novel approach of tag taxonomy construction. Our innovation lies in: i) By using an *Agglomerative Hierarchical Clustering* (AHC) framework, we only need to compute how similar two tags are and deliberately skip the calculation of tag generality. ii) We seamlessly integrate a *topic model* into the framework, so it is able to capture thematic correlations among tags. iii) We make comparative evaluation with existing approaches, the results demonstrate the usefulness and effectiveness of our method.

2 Related Work

Because of the convention of constructing taxonomies with the general tags at the top and the more specific tags below them[6], a key step in existing taxonomy construction methods is extracting generality of each tag, either by computing a generality score[2][7] or by pair-wise comparison[3][4]. As far as we know, two types of techniques are used to achieve this step.

Some research works[2][7] use *set theory* techniques to compute a tag generality. In such works, each tagged resource is treated as a distinct data item with their textual contents ignored. Each tag is presented as the collection of items it annotates. For example, Heymann et al.[7] proposed a simple yet effective way to learn a tag taxonomy. They first model each tag as a vector, with each entry being the number of times the tag annotates a corresponding resource. *Cosine metric* is then used to measure tag similarity. They come up with the *tag similarity graph* by connecting sufficiently similar (controlled with a threshold) tags. The *graph centrality* metric, which is originally proposed in social network analysis literature, is used to measure the generality of each tag. Finally, tags of higher generality are greedily placed at the upper levels of the resultant taxonomy. Liu et al.[2] used association rule mining which takes each tagged resource as a transaction and tags as items. The rules are in the form of "for an unknown resource X, if tag A appears then probably tag B will also appear", which they call "tag B subsumes tag A". The likelihood of such subsumption is naturally modeled as the confidence and support of the corresponding rule. Based on subsumption likelihood between each pair of tags they calculate an overall generality score for each tag by using a random walk process. Eventually, a taxonomy is constructed using a top-down manner.

However, such set theory based approaches share a common flaw. They only distinguish one resource from another and do not exploit tagged web documents. In light of this, some researches apply topic models to do the job. Such methods learn a latent topic distribution for each tag from web documents they annotate, tag relations are measured based on pair-wise comparison among topic distributions. Tang et al.[3] designed the *Tag-Topic model* based on the classic *LDA(Latent Dirichlet Allocation)* model[8]. They treat each word as a draw from a topic-specific word distribution, and a latent topic is in turn a draw from a tag-specific topic distribution. Based on such a generative model, Gibbs sampling is used to infer distribution parameters. They use Kullback-Leibler (KL) divergence to measure the difference between two distributions, and have proposed a relative generality score metric based on the intuition "if one tag has higher posterior probabilities on the latent topics, then it has a relatively higher generality". Wang et al.[4] merge documents annotated by the same tag (they use 'keyword' instead) as a new document which they believe can explain that tag. They learn a standard LDA model out of the underlying corpus. Those new documents are folded-in thereafter. Thus, topic distribution of each tag (new document) is obtained. Their measurement of relative generality is based on the "surprise" theory[9] plus an intuitive law that "given an anticipated tag A, the appear of a document on a more general tag B will cause less 'surprise' than if

A and B are switched". It is really difficult to come up with a tag generality score that both [3] and [4] resort to intuitions.

3 AHCTC: Agglomerative Hierarchical Clustering for Taxonomy Construction

In order to avoid computing tag generality, we borrow the idea from the classic agglomerative hierarchical clustering which only relies on how similar/distant two points are in building a hierarchy.

The term *agglomerative hierarchical clustering* is not new in the field of taxonomy construction. Brooks and Montanez[10] adopted such methods to organizing tags into a hierarchy, yet their hierarchy is not a taxonomy for lack of supertype-subtype relationships. In Liu et al.[2], authors also mention the use of agglomerative hierarchical clustering. However, their strategy is to iteratively place the most general tag remained, which is different from the classic meaning of the agglomerative hierarchical clustering framework. The brilliance of agglomerative hierarchical clustering is yet to be fully exploited in the field of taxonomy construction.

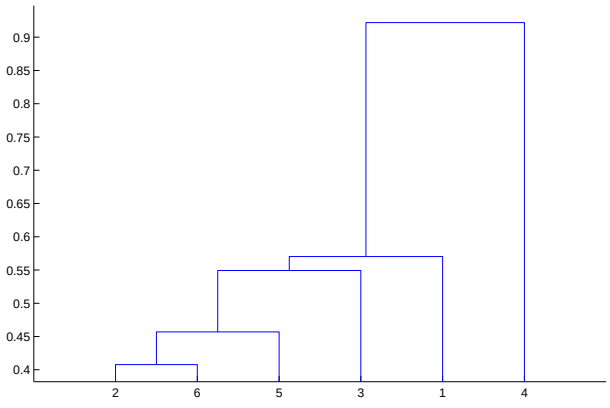


Fig. 2. dendrogram of a typical clustering process

To illustrate the AHCTC framework in a broader picture, we eliminate any presumption on how the proximity scores are calculated, but rather assume they are known a-priori. In the next chapter, we will describe how to get a meaningful proximity score. With each data point being an initial cluster, classic agglomerative hierarchical clustering merges two closest clusters each time until only one all-inclusive cluster is left. The dendrogram of a typical AHC process consisting of 6 points are depicted in Fig.2. Sadly, the result is not a taxonomy for

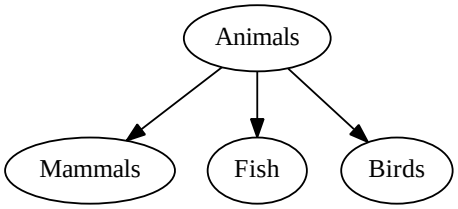


Fig. 3. a hierarchy of the cluster {*Birds*, *Mammals*, *Fish*, *Animals*} after promote

the absence of supertype-subtype relationships. In the dendrogram, for example, a supertype should be there to overarch the cluster consisting of point 2 and 6. Now that data points are tags in our problem, a tag cluster should have a direct supertype (parent tag) or the resultant tag tree is not technically a taxonomy.

Of course, a tag cluster's direct supertype should have high *semantic proximity* to each tag in the cluster. For the cluster {*Birds*, *Mammals*, *Fish Animals*}, a tag like *Animals* is more suitable to be the supertype than a *Living Things* tag. During the clustering process, tags with high proximity are assigned to the same cluster, so the supertype should better be selected from within a given cluster. Specifically, we choose the *medoid* of a cluster, i.e., the most central point. This medoid upgrades as the supertype of all other cluster members. We call this change as a **promote** operation. Now suppose the original cluster is {*Birds*, *Mammals*, *Fish*, *Animals*}, and *Animals* happens to be the medoid according to some proximity measure, the resultant hierarchy after promote is depicted in Fig.3.

In a hierarchical clustering scenario, a new cluster is merged from two old clusters each with a hierarchy itself, it forces the promote operation to be able to combine these two old hierarchies into a new one. With such requirements, we devise a promote mechanism and append it to each merge operation of classic AHC, the new taxonomy construction algorithm is illustrated in Algorithm 1. Two building blocks of the algorithm, $proximity(i, j)$ and $sup(m)$ are assumed known a-priori. We will illustrate how to get them later. The algorithm uses an adjacent matrix T to store resultant taxonomy. Initially, every data point is a distinct cluster. Each iteration in the **while** clause executes a merge and a promote operation. Line 4-7 finds the closest two clusters in the current cluster set to merge.

Line 8-21 is the promote operation, which is also the only operation edits T. Different from the basic promote operation we have described in Fig.3, the promote operation here will be much more intricate since a hierarchical scenario is concerned. For ease of illustration, we call the the two supertypes over the clusters being merged as **old supertypes**. Line 8-11 says if the new supertype *sup(m)* happens to be one of the two old supertypes, an directed edge is added from it to the other old supertype (see Fig.4(a) and Fig.4(b)).

Otherwise, if *sup(m)* comes from *the public*, the procedure will have to go through the intricacies of line 13-20. Fig.4(c) and Fig.4(d) displays them in de-

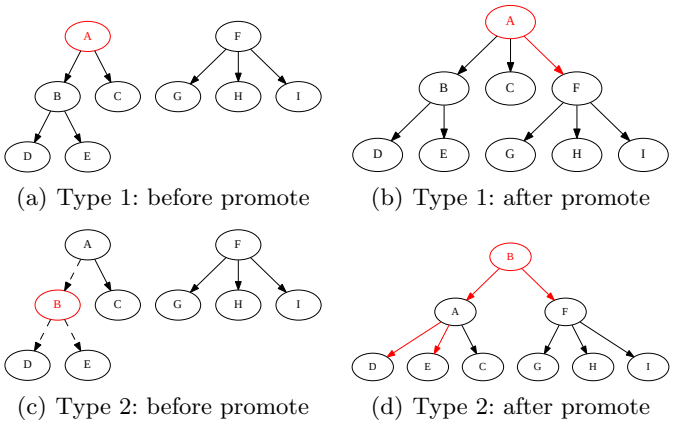
Algorithm 1 AHCTC

Require: Data points $D = \{d_1, d_2, ..., d_n\}$.
Require: $T = [t_{ij}]_{n \times n}$ the adjacent matrix for the resultant taxonomy, $t_{ij} = 1$ when d_i is a direct supertype of d_j.
Require: $M = \{m_1, m_2, ...\}$, the set of all remaining clusters.
Require: $sup(m)$, index of the supertype data point for a given cluster m.
Require: $proximity(i, j)$, the proximity score between data points d_i and d_j.
Ensure: construct the resultant taxonomy T.

```
1: T ← (0)_{n×n}
2: M ← D
3: while |M| > 1 do
4:    find m_i and m_j in M with maximum proximity(sup(m_i), sup(m_j))
5:    merge m_i and m_j as m
6:    add m to M
7:    delete m_i and m_j from M
8:    if sup(m) equals sup(m_i) then
9:       t_{sup(m),sup(m_j)} ← 1
10:   else if sup(m) equals sup(m_j) then
11:      t_{sup(m),sup(m_i)} ← 1
12:   else
13:      find k that t_{k,sup(m)} equals 1
14:      t_{k,sup(m)} ← 0
15:      for all g that t_{sup(m),g} equals 1 do
16:         t_{sup(m),g} ← 0
17:         t_{k,g} ← 1
18:      end for
19:      t_{sup(m),sup(m_i)} ← 1
20:      t_{sup(m),sup(m_j)} ← 1
21:   end if
22: end while
```

tails. Simply speaking, former subtypes of point B (the newly elected supertype), i.e. D and E, should now be A's subtypes after B's promote operation. This is because A is formerly their nearest ancestor besides B, and we want to maintain the relation that A be more general than either D or E after B's promote. To implement this process, the algorithm detaches *sup(m)* from its direct supertype k (line 13-14), attaches all children of *sup(m)* to k (line 15-18) and finally the two old supertypes are attached to *sup(m)* (line 19-20).

If n is the number of data points, the basic AHC algorithm requires $O(n^2 \log n)$ time[11]. The only extra step of our algorithm is the promote operation (line 8-21), which at worst induces $O(n^2)$. The **while** loop will iterates for $n-1$ times, because after each iteration the number of clusters only reduces one. At the worst case, all $n-1$ iterations undergo line 13-20, each needs to look up and edit at most $n-1$ edges in T. Totally speaking, our algorithm has the same time complexity as the basic AHC, $O(n^2 \log n)$.



(a) Type 1: before promote (b) Type 1: after promote

(c) Type 2: before promote (d) Type 2: after promote

Fig. 4. Two types of promote operation during hierarchical clustering. The selected supertype and newly added edges are in red, edges to be removed are in dashed lines.

4 Integration with Topic Models

In order to get AHCTC running, we have to mount its assemblies. Notice that $sup(m)$ finds the most central points based on $proximity(i, j)$. So essentially, we only have to implement $proximity(i, j)$ which quantifies the similarity measure between tags.

Web documents annotated by a certain tag usually contain semantic information of that tag. It will be a great loss for a tag taxonomy construction method to overlook latent semantics in tagged web documents. Recent years have witnessed drastic methodology evolution in the field of information retrieval, towards the unchanging goal of uncovering semantic information in unstructured text data. Since the publish of LDA[8], a family of algorithms[12][13] known as Probabilistic Topic Models emerge and quickly win massive adoptions. Probabilistic Topic Models are skilled at discovering hidden thematic structure of text documents. In a real corpus like New York Times, a document may be tightly related to topics like *Foreign Policy* and *Economics*, while only mentions a little on *Sports*. Topic models answer what themes or topics a document relates to and quantify how strong such relations are. Thus, a thematic similarity measure could be induced for each pair of documents. Wang et al.[4] and Tang et al.[3] each design a variant of the basic topic model able to learn thematic structure of tags from tagged corpus. Our $proximity(i, j)$ measure is built on top of Wang's model. Chapter 4.1 briefly introduces stantard LDA and Wang's topic model.

4.1 Probabilistic Topic Model

LDA is the basic Probabilistic Topic Model. In LDA, a latent topic $z = j$ is modeled as an unlabeled corpus-wide word distribution $\phi^{(j)} = P(w|z = j)$, which was drawn from a dirichlet prior distribution $Dirichlet(\beta)$. The number

of topics T is specified beforehand to adjust the granularity. Each document d is a mixture of topics $\theta^{(d)} = P(z)$ with a dirichlet prior distribution $Dirichlet(\alpha)$. The generative process of each word in d is essentially a draw from the joint distribution $P(w_i) = \sum_{j=1}^{T} P(w_i|z_i = j)P(z_i = j)$. Given the observed documents, Gibbs Sampling algorithm[14] is usually used for posterior inference.

[4] modifies LDA to deal with tags (keywords). Their assumption is that documents annotated by a tag usually have thematic information of that tag. For a given tag, they merge documents annotated with it into a *new document*, removing those words occurred only once. After the standard LDA training, new documents are folded-in to the trained model by an extra run of Gibbs Sampling algorithm. Finally, a *tag-topic distribution* for each new document is estimated. Jenson-Shannon divergence or cosine similarity measure could be used to calculate the divergence between any two tag-topic distributions.

4.2 Tag Proximity Measure

Based on the above model, our tag proximity measure is chosen just as the divergence between tag-topic distributions: if we think a tag-topic distribution over T topics as the tag's coordinate in a T-dimensional *thematic space*. The divergence between two tag-topic distributions can be understood as the *thematic space coordinate distance* between the two tags. So it becomes clear that such a proximity measure exploits semantic (thematic) correlation among tags. Based on that, $sup(m)$ is defined as the most central point of cluster m in the thematic space, which ensures that a supertype can thematically represents the whole cluster. In other words, $sup(m)$ finds the data point index $\arg\min_i \{\sum_{d_j \in m} proximity(i, j)\}$. After integrating with the topic model, AHCTC can hierarchically cluster tags that are thematically similar and pick the most central tag in the thematic space as the supertype for each cluster. Notice the difference between our method and [3][4] is that we don't have to tell which tag is more general from the topic model, which is not what a topic model good at.

5 Experiment

Ohloh[5] is a popular online open source software directory and community platform whose user number has exceeded 1,500,000. It provides information on more than 500,000 open source software projects. In the profile of each project, there is a brief description along with other valuable information like development history and technical features. Being a social tagging website, its users are given the freedom to edit this information and to tag the projects. A typical project profile in Ohloh is depicted in Fig.5. To browse projects by tags, users can either pick from the flat tag list or use keyword search for sifting. However, such a flat list does not give the conceptual scope of each tag nor the possible

[5] http://www.ohloh.net/

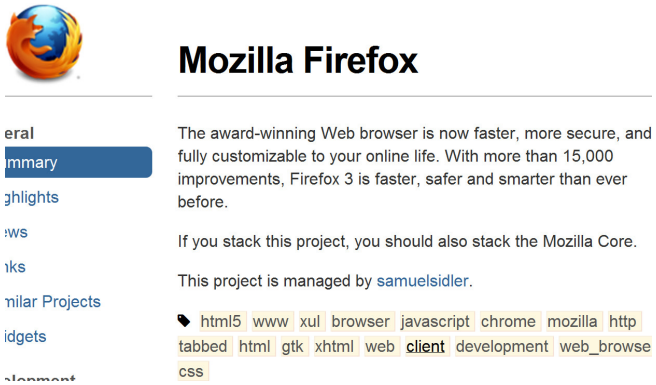


Fig. 5. Part of project profile of the Mozilla Firefox in Ohloh

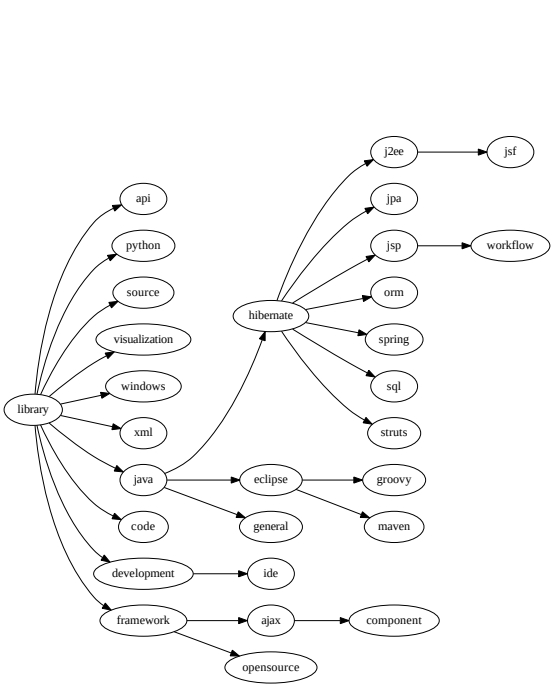
relationships among several tags. A tag taxonomy will apparently do a better job. Based on such a requirement, we choose Ohloh as our dataset to validate our tag taxonomy construction algorithm.

Dataset and Parameter Setting. From Ohloh, we have crawled 10,000 open source software project profiles and extracted their descriptions and tags. Suffering from the common flaws of all folk knowledge, tags might be poor phrased or too esoteric and some are even meaningless. We omit tags with less than 300 references and manually delete the meaningless tags '1', '???', '????', '?????' which somehow are widely adopted in Ohloh. The result after preprocessing is a set of 267 tags. For topic modeling, we set the number of topics at 60 and iterate 2000 times in the Gibbs Sampling process.

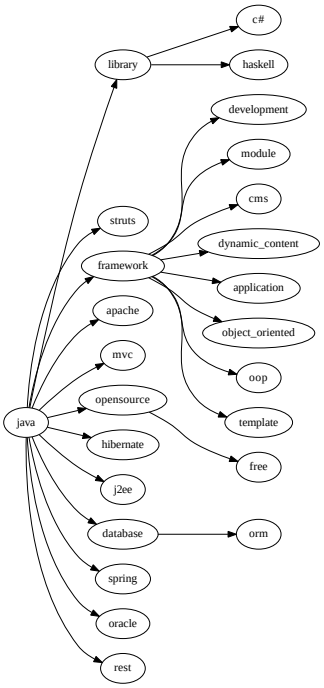
Baseline methods. We compare AHCTC with two of the published methods on the same dataset. For comparison with set theory based methods, we choose the ontology induction algorithm proposed in [2] as our baseline. Since their method is based on tag subsumption, we use the shorthand SUBSUME to stand for their method. Ohloh does not provide author information of each tag, so we use $\mathbb{K}^R = (G = R, M = T, I)$ as the projection from a folksonomy onto a formal context, details are given in the original paper. We set the parameters at the best setting given by the authors, i.e.,$\lambda = 0.95, \theta_s = 0.00001, \theta_c = 0.15$ and the maximum number of random walk iteration equals 1000.

For comparison with topic model based methods, we choose the LSHL algorithm proposed in [4] as the baseline. The other algorithm in [4], GSHL, is just a modified version of LSHL aiming at a different goal. Their experiment use keywords of academic paper abstracts as *Concepts* while here we use the 267 Ohloh tags to take their places. Parameters are also set at what the original paper suggests, $TH_s = 0.6, TH_d = 0.35, TH_n = 0.4$ and the number of topics for topic modeling is set to 60.

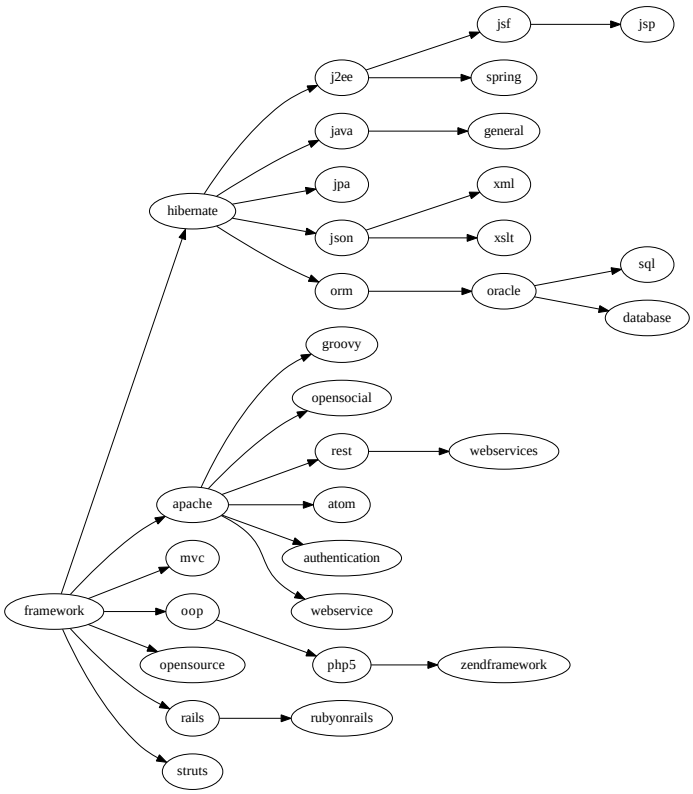
Experiment result. The resultant taxonomies are shown in Fig.6. Since the space is limited, only the *framework* subtree of our result is depicted. For result of LSHL algorithm, the *library* subtree is chosen since it has sufficien-



(a) part of LSHL *java* subtree



(b) SUBSUME *library* subtree



(c) AHCTC *framework* subtree

Fig. 6. Part of the resultant taxonomies

t overlapped tags with AHCTC *framework* subtree. We can see from Fig.6(a) and Fig.6(c) that despite some parts, the two trees are different in their structures. The AHCTC *framework* subtree apparently classifies the tags according to the technical framework they belong. In the LSHL *library* subtree, tags are first classified by different types of programming libraries. For example, in the AHCTC *framework* subtree *hibernate* is classified as a kind of framework but in LSHL it is deemed as a java library. The two taxonomies are of different facades. They complement one another and are both valuable in disclosing tag semantic structure. If these machine-learned taxonomies are to be used as prototypes helping ontology engineers to build a well-defined taxonomy, both the resultant taxonomies will be needed in order to provide a comprehensive picture.

The SUBSUME taxonomy seems rather flat, many tags appear as the root's children. This coincides with experiment results and illustrations given in the original paper. In fact, this taxonomy takes 91 out of 267 tags as the children of root *java*, including most of the tags in AHCTC *framework* subtree. To make a visual comparison with the other two subtrees, we can only render an excerpt of the *java* tree consisting of only pertinent tags. In Fig.6(b) the induced edges seem reasonable individually, but the whole taxonomy is too flat to be informative. We believe the cause is that SUBSUME algorithm does not exploit text of tagged documents.

Quantitative Evaluation. In order to make a convincing quantitative evaluation, we have to do multiple experiments for each of the three algorithms. We conduct 3 AHCTC experiments each time with a different tag similarity measure, Jenson-Shannon divergence (JS), cosine similarity (COS) and KL divergence. For the LSHL taxonomy which also needs to calculate a tag similarity score, JS and COS as the only measures proposed in the original paper are evaluated respectively. We fail to conduct multiple versions of the SUBSUME experiment since the algorithm do not have any replaceable building block.

To evaluate a tag taxonomy is difficult, because several complementary taxonomies might exist. However, several related works have tried in designing effective taxonomy evaluation methods. [2] uses the concept hierarchy from Open Directory Project as the ground truth to validate their learned ontology. In comparing the two hierarchies, both lexical and taxonomical precision and recall are evaluated. [4] asks domain experts to evaluate the correctness of each edge individually, and precision is defined as the proportion of correct edges in all the learned edges.

Our quantitative evaluation considers both these techniques. First, we ask 20 students to judge the correctness of each edge in a taxonomy. They are given 3 days to finish this task, and are encouraged to consult any accessible knowledge source for tags they don't understand. We use the precision measure proposed in [4] to calculate edge correctness. We calculate the average value and standard deviation of the edge correctness. The results are given in TABLE 1. All AHCTC taxonomies have high average edge correctness compared to other results, and the AHCTC-KL taxonomy has the highest. AHCTC-JS and LSHL-COS tax-

onomies have relatively higher standard deviations, which means many of the edges are indeterminate so judges make quite different judgements.

Table 1. Edge Correctness Judged by Human Experts

	Avg.	S.d.
LSHL-JS	0.7319	0.0935
LSHL-COS	0.6888	**0.1313**
SUBSUME	0.6897	0.0639
AHCTC-KL	**0.7940**	0.0432
AHCTC-JS	0.7895	**0.1213**
AHCTC-COS	0.7519	0.0879

To consider the correctness of each edge individually is not fully appropriate. We refer to the taxonomical precision measurement proposed in [2]. The golden truth is chosen as the $software$ subtree of the Open Directory Project concept hierarchy. This subtree covers 179 out of 267 tags we are concerning, which is a 67.29% lexical precision. We also apply the taxonomical metrics defined in [2]. The taxonomical precision, recall and F-measure (TP, TR, TF) for different experiments are given in TABLE 2. AHCTC with KL divergence gets the highest value on all three metrics.

Table 2. Taxonomical Quantitative Evaluation

	TR	TP	TF
LSHL-JS	0.0194	0.0235	0.0212
LSHL-COS	0.0244	0.0296	0.0268
SUBSUME	0.0267	0.0393	0.0318
AHCTC-KL	**0.0525**	**0.0514**	**0.0519**
AHCTC-JS	0.0370	0.0334	0.0351
AHCTC-COS	0.0306	0.0328	0.0317

Discussion The above experiments and evaluations demonstrate the usefulness and effectiveness of AHCTC algorithm, given the existence of similar methods: through qualitative evaluation, we show that AHCTC discovers valuable tag structures that are different from those discovered by any existing algorithm. Quantitative metrics suggest that AHCTC can construct better taxonomies.

As we have mentioned, parameters in SUBSUME and LSHL experiment are set at the practical best settings suggested by original papers. However, authors didn't guarantee that those settings are always the best regardless of what

dataset being used. For this reason, we have actually tested several other combinations of parameter values. It turns out that the best settings still practically outrun other settings in our dataset in terms of the quality of resultant taxonomies.

It shall be noted that evidence given by our quantitative evaluations is still weak in measuring how 'good' a taxonomy is: edge correctness considers whether each individual edge is correct and can not measure the correctness of an entire taxonomy. Taxonomical metrics gained by comparing against a gold taxonomy is not persuasive, given that several complementary taxonomies might exist. It is also the reason why metrics in TABLE 2 are quite small. Taxonomy evaluation is still an open issue to be investigated in the future.

6 Conclusion

Many research efforts have been spent on inducing a taxonomy from a set of tags. This paper proposes a novel approach based on agglomerative hierarchical clustering, which can effectively skip the error prone step of calculating each tag's generality. A topic model is integrated into the AHC framework to disclose thematic correlations among tags. The experiment is built on top of data from Ohloh, an online social network software directory. With qualitative and quantitative evaluations, we demonstrate usefulness and effectiveness of the proposed method, after comparing with two representative previous works.

7 Acknowledgement

This research is supported by the National Science Foundation of China (Grant No.60903043), the National High Technology Research and Development Program of China (Grant No. 2012AA010101) and the Postgraduate Innovation Fund of University of Defence Technology (Grant No.120602). Our gratitude goes to authors of [4], Wang Wei and Payam Barnaghi, for their help in providing us the details of LSHL algorithm.

References

1. Staab, S., Studer, R., eds.: Handbook on Ontologies. 2. edn. Springer, Berlin (2009)
2. Liu, K., Fang, B., Zhang, W.: Ontology emergence from folksonomies. In Huang, J., Koudas, N., Jones, G.J.F., Wu, X., Collins-Thompson, K., An, A., eds.: CIKM, ACM (2010) 1109–1118
3. Tang, J., fung Leung, H., Luo, Q., Chen, D., Gong, J.: Towards ontology learning from folksonomies. In Boutilier, C., ed.: IJCAI. (2009) 2089–2094
4. Wang, W., Barnaghi, P.M., Bargiela, A.: Probabilistic topic models for learning terminological ontologies. IEEE Trans. Knowl. Data Eng. **22**(7) (2010) 1028–1040
5. Navigli, R., Velardi, P., Faralli, S.: A graph-based algorithm for inducing lexical taxonomies from scratch. In Walsh, T., ed.: IJCAI, IJCAI/AAAI (2011) 1872–1877

6. Russell, S.J., Norvig, P.: Artificial Intelligence - A Modern Approach (3. internat. ed.). Pearson Education (2010)
7. Heymann, P., Garcia-Molina, H.: Collaborative creation of communal hierarchical taxonomies in social tagging systems. Technical report, Computer Science Department, Standford University (April 2006)
8. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of Machine Learning Research **3** (2003) 993–1022
9. Itti, L., Baldi, P.: Bayesian surprise attracts human attention. In: NIPS. (2005)
10. Brooks, C.H., Montanez, N.: Improved annotation of the blogosphere via autotagging and hierarchical clustering. In Carr, L., Roure, D.D., Iyengar, A., Goble, C.A., Dahlin, M., eds.: WWW, ACM (2006) 625–632
11. Tan, P.N., Steinbach, M., Kumar, V.: Introduction to Data Mining. Addison-Wesley (2005)
12. Blei, D.M., McAuliffe, J.D.: Supervised topic models. In Platt, J.C., Koller, D., Singer, Y., Roweis, S.T., eds.: NIPS, Curran Associates, Inc. (2007)
13. Blei, D.M., Griffiths, T.L., Jordan, M.I.: The nested chinese restaurant process and bayesian nonparametric inference of topic hierarchies. J. ACM **57**(2) (2010)
14. Griffiths, T.L., Steyvers, M.: Finding scientific topics. Proceedings of the National Academy of Science **101** (2004) 5228–5235