




Mining Tag Relationships in CQA Sites

K. Suryamukhi^(✉) , P. D. Vivekananda, and Manish Singh

Department of Computer Science and Engineering, IIT Hyderabad, Hyderabad, India
{cs17m19p100001,cs17btech11025}@iith.ac.in, msingh@cse.iith.ac.in

Abstract. Community Question Answer (CQA) sites are very popular means for knowledge transfer in the form of questions and answers. They rely on tags to connect the askers with the answerers. Since each CQA site contains information about a wide range of topics, it is difficult for users to navigate through the set of available tags and select the best ones for their question annotation. At present, CQA sites present the tags to the users using simple orderings, such as order by popularity and lexical order. This paper proposes a novel unsupervised method to mine different types of relationships between tags and then create a forest of ontologies to representing those relationships. Extracting the tag relationships will help users to understand the tags meanings. Representing them in a forest of ontologies will help the users in better tag navigation, thereby providing the users a clear understanding of the tag usage for question annotation. Moreover, our method can also be combined with existing tag recommendation systems to improve them. We evaluate our tag relationship mining algorithms and tag ontology construction algorithm with the state-of-the-art baseline methods and the three popular knowledge bases, namely DBpedia, ConceptNet, and WebIsAGraph.

Keywords: Ontology construction · Relationship extraction · Knowledge graphs

1 Introduction

Community Question Answering (CQA) sites, such as Stack Exchange¹, Stack Overflow² and Quora³, are a few among the famous online platforms for sharing knowledge. They allow users to post questions or answer them. In CQA sites, users can quickly get multiple answers to their questions. Moreover, there may not be a single web page that answers the specific question of the user. These sites are enabled with tagging system that helps the askers to route their questions to the experts. These are the main reasons why CQA sites are thriving. For example, Stack Overflow, a Stack Exchange site, has over 13 million users, 20 million questions, 30 million answers, and 59k tags as of October 2020.

¹ <https://stackexchange.com/>.

² <https://stackoverflow.com/>.

³ <https://www.quora.com/>.

In all the CQA sites of Stack Exchange, users are allowed to annotate their questions with at most 5 tags from the available massive number of tags. Thus, it can be challenging for the users to find the best tags to annotate the questions. To aid them, we construct tag ontology, which can show them the relationships between tags. Since a single ontology for all the tags in a CQA site will be too huge for the users to explore and have many weak relationships, we propose automatic creation of a forest of ontologies, where each ontology will be small and will show only the strong tag relationships.

Identifying semantic relationships between tags can help the users to recognise similar tags. Furthermore, distinguishing a tag as either generic or specific can help them to select broader or narrower tags for annotation. In this work, we call the former relationship between tags as sibling relationship and the latter as parent-child relationship.

Most of the existing works [2,3] have focused only on parent-child relationship for ontology construction using either simple co-occurrence measures or textual relationship extraction. In this paper, we show that for CQA sites, these methods cannot properly capture the directed associations between tags. The key contributions of the paper are:

- Extracting three types of tag relationships using tag co-occurrence and tag related meta data.
- An unsupervised algorithm to create forest of tag ontologies.
- A comprehensive evaluation with state of the art baselines and three knowledge bases, namely DBpedia [1], ConceptNet [9], WebIsAGraph [4].

The rest of the paper is organised as follows: We discuss the related work in Sect. 2. In Sect. 3, we detail the algorithms for mining tag relationships and constructing ontology. Section 4 presents experimental setup, evaluations and results. Finally, Sect. 5 concludes the work.

2 Related Work

Our related work can be grouped into three categories, namely generic relation extraction, specific relation extraction and ontology construction. They are presented as follows:

Generic Relationship Extraction. There are many related works [11,12] that focus on extracting any generic relation between entities from raw text. These works rely on the sentences of the text to extract the relationships between entities. For example, in the sentence, *Joe Biden is the president of the USA*, these methods would extract the entities, namely ‘Joe Biden’ and ‘USA’, and then extract the ‘president-of’ relation between them. Since our data does not have tag relationships in the form of text or sentences, we cannot use them.

Specific Relationship Extraction. In these works, the type of relationships to be extracted is pre-defined. Most of them focus on parent-child relation. There are few related works [2,8] that find out whether two entities are semantically

related, but they do not differentiate if they have parent-child, sibling or some other relation. In this paper, we present methods to extract two types of tag relationships, namely parent-child and sibling.

[2, 6] infer parent-child relation using co-occurrence based measures such as support, confidence and overlapping scores. However, these co-occurrence based measures do not accurately capture the semantics of parent-child relation. [3] use text of the resource to mine relationship but as mentioned, the question content is not helpful in CQA sites. In our paper, we extract sibling tags using tag information provided by the CQA site and external resources.

Ontology Construction. There are many related works on ontology construction. [5] constructs a large web scale user-centered ontology having relationships like parent-child, sibling from vast number of user-action logs using graph neural networks. [3] constructed a large ontology in a supervised manner by using classifiers to detect parent-child relation between all the tag pairs. This is very expensive due to the combinatorial explosion of all the tag pairs. In our work, we construct a forest of ontologies in an unsupervised manner where each ontology will have only strong relations between tag pairs.

3 Mining Tag Relationships

In this section, we present our approach to mine pairwise tag relationships and create forest of ontologies to represent them. Section 3.1 presents the problem definition, Sect. 3.3 details the tag relationships mining algorithms and in Sect. 3.4, we present the method to construct ontologies.

3.1 Problem Definition

Our proposed tag ontologies and tag relationships will help users to easily locate related tags and also understand their ontological relationships. We divide our problem into the following 3 sub-problems.

Problem 1 Grouping Related Tags. *Given the set of all tags T in a CQA site and various tag statistics, such as tag count, tag co-occurrence, etc., our task is to group the tags into candidate groups. All relationships will be explored between tags only within each group.*

As it is computationally expensive to compute relationships between all possible pairs of tags and tags belong to specific topics, we cluster the tags into small candidate groups, which are easier to comprehend. We then mine tag relationships only between tags within each group and use the groups to form a forest of ontologies.

We perform tag grouping at a coarser level so that we don't miss tags that do have a relationship but end up in different candidate groups. So, we use only various types of count measures to form the candidate groups. In the following problems, we present methods to extract fine-grained relationships For creating groups, we use standard community detection algorithms exploring various similarity scores between tags in the tag graph as detailed in Sect. 3.2.

Problem 2 Mining Pairwise-tag Relationships. *Given a tag pair (a, b) , where both tags a and b belong to some tag community C , and their descriptions in the CQA site as well as in external knowledge bases, our task is to mine various types of predefined relationships between the tag pair a and b .*

In CQA sites, the user is expected to use a mix of generic and specific tags to annotate their questions. If the user uses only generic tags, many answerers may not be keen on going through the questions text and answering them. While, if the question has only specific tags, the site may not be able to route the question to answerers as only a few of them may have explicitly shown interest in those specific tags. Our parent-child relationship will help users to select the right mix of generic and specific tags. Suggesting similar sibling tags can help the users to understand the better alternative tags.

We use co-occurrence information and along with tag descriptions to extract these tag relationships. In Sect. 3.3, we present the algorithms which scores each tag pair for these relationships.

Problem 3 Forest of Tag Ontology. *Given the tag communities $C_1 \dots C_k$ and the relationship between tags within each community, construct a tag ontology for each tag community.*

Since in a single huge ontology, there can be many weak or wrong relationships, we propose an algorithm to create a forest of ontologies by creating an ontology for each tag community. Our ontologies are directed acyclic graphs that organize the tags in the form of hierarchies using the two tag relationship scores.

The ontology consists of parent-child and sibling relationships as the edge types.

Our ontology construction algorithm is presented in Sect. 3.4.

3.2 Grouping Related Tags

To form tag communities we use a community detection algorithm known as Infomap [7], which is based on MapEquation. We also used Louvain and Walk-Trap, but Infomap performed the best. We build the tag graphs using the following three tag similarity scores as edge weight.

Edge Weight. We use Google Distance to give edge weight between tags. It is a co-occurrence based symmetric measure that can capture the intuitive directed associations between the tags. It is given by:

$$gd(a, b) = \frac{\max(\log N(a), \log N(b)) - \log N(a, b)}{\log N_{total} - \min(\log N(a), \log N(b))} \quad (1)$$

where, $N(a)$ and $N(b)$ are the number of posts containing tags a and b respectively. $N(a, b)$ is the number of posts containing both the tags. N_{total} is the total number of posts in our dataset. We define edge weight as:

$$EW(a, b) = \frac{1}{\exp(gd(a, b))} \quad (2)$$

Edge Weight using only Popular Tags. Since there were many infrequent tags and their relationship with other tags is not so well-defined, including them in the graph resulted in a very low modularity score for all the three community detection algorithms.

We filtered infrequent tags from the graph G and constructed a new tag graph G^{pop} using only the popular tags. Edge weight using only popular tags, $EW P(a, b)$, is then defined same as above.

Edge Weight based on Probabilistic Association. We observed that taking $EW P$ improved modularity score of community detection algorithms. We further improved it by taking into account the probabilistic association between the tags, which quantify the likelihood of their co-occurrence. We model this likelihood using conditional probability. The directed weighted tag graph G^{cp} has edge weight defined as:

$$EWPA(a|b) = \frac{EW P(a, b)}{\sum_c EW P(b, c)} \quad (3)$$

3.3 Mining Pairwise-Tag Relationships

In this section, we present algorithms to mine the two types of tag relationships.

Parent-child Relationship. To detect the parent-child relationship between tags a and b , we use their co-occurrence based probabilistic association score $EWPA(a|b)$, defined in Eq. 3, and their entropy. The $EWPA(a|b)$ tells us how likely the tag a would be mentioned in the post given the tag b . The entropy of a tag is defined as follows:

$$H(a) = - \sum_c EWPA(a|c) \log EWPA(a|c) \quad (4)$$

Tags having higher entropy are expected to have a generic meaning. If $EWPA(a|b)$ and $H(a)$ are significantly greater than $EWPA(b|a)$ and $H(b)$, respectively, then tag a is most likely a parent of tag b . Otherwise, they may not have any relationship or may have sibling relationship, which is described in the next subsection.

Sibling Relationship. Sibling tags occur at the same level in the tag ontology. In the ontology, the siblings are kept in the order of decreasing entropy. In other words, the first tag would be the most popular sibling.

To detect siblings we use three knowledge sources, namely the Tag Excerpt and the Tag Wiki Description, which are available as metadata in the CQA site; and the Tag Abstract available from Wikipedia using the DBpedia SPARQL Endpoint⁴. The Tag Excerpt briefly describes the tags and the Tag Wiki contains a more detailed description. These are written by the CQA site topic experts. Tag Abstract includes the abstract of the tag's Wikipedia article.

We create a corpus by combining the information from the above three information sources and use it to train a Glove word embedding model, where we

⁴ <https://wiki.dbpedia.org/>.

represent each word using 300-dimensional vector. For two tags a and b , we define their sibling score as:

$$sib(a, b) = 1 - WMD(a, b) \quad (5)$$

We say that the tags a and b are siblings if the value of $sib(a, b)$ is greater than some threshold value. WMD is defined as:

$$WMD(a, b) = \min_{T \geq 0} \sum_{i,j} T_{i,j} c(x_i, x_j), \quad (6)$$

where T is the transportation cost to transfer every word i in the description of tag a to every word j in the description of tag b . Finally, the minimum travel or transfer cost is chosen. c is the Euclidean distance between the embedding of word i and word j denoted by x_i and x_j respectively.

Instead of representing the tag embedding vectors with all the words present in the tag descriptions, we can choose only the top- n words that best represent the tag. For this purpose, we use a supervised variant of LDA topic modelling algorithm, Labelled LDA (L-LDA). It defines a one-to-one correspondence between LDA's latent topics and the category links of the tags used as labels. The category links are obtained from Wikipedia tag articles using DBpedia SPARQL Endpoint. We then extract top- n ($n = 10$) words to represent each tag and calculate similarity between them using Eq. 5 defined as $sibLLDA(a, b)$.

3.4 Constructing Tag Ontology

After extracting the relationships between the tag pairs, we propose an hierarchical ontology construction algorithm which, constructs the ontology graph in the form of a DAG. We obtain forest of ontologies for all the tag communities. The ontologies consist of the two relationships. The type of the edge is chosen based on the similarities thresholds in the precedence order of siblings and parent-child. The process of setting these thresholds is detailed in Sect. 4.5.

Given the tag community as input, our algorithm follows a greedy approach to organize the tags in the ontology by ordering them in the decreasing order of their entropies. First, the tag with highest entropy is chosen and then the parent-child scores satisfying the threshold with that tag as the parent are ranked in decreasing order. Next, each tag in those pairs are ranked in the decreasing order of their entropy. Then, each tag pair is checked for the presence of the two relationships. First, the presence of sibling relationship is checked for each tag pair. If it satisfies the sibling threshold, then we add sibling edge to the ontology graph. Otherwise, the parent-child edge is added, as it already satisfies the threshold.

4 Evaluation

In this section, we present the evaluation of our algorithms. We first describe the dataset and then we present the evaluations and results.

4.1 Dataset

We used the SuperUser⁵ data from StackExchange site. It contains over 10 million posts and a total of 5500 unique tags. The posts are from a variety of domains like databases, security, web browsers, programming languages, etc.

We used three users to label a part of our dataset for evaluation. Any ambiguity in labelling was resolved through discussion and mutual agreement. We gave 5150 tag pairs to users and asked them to label them individually as parent child or sibling.

Our dataset had 1378 unique tags after removing the infrequent tags. We obtained 18 communities using the Infomap algorithm on the tag graph G^{cp} . We evaluate our ontology generation algorithm on 5 tag communities.

4.2 Evaluation Metric

Tag relationships are evaluated using Precision, Recall and F1-score against the ground truth human labelled dataset. We also compare our results with the state of the art baselines.

Tag ontologies are evaluated by calculating their similarity with ground truth ontologies. The standard metrics are taxonomic precision (TP), taxonomic recall (TR) and taxonomic F-measure (TF). The idea is to find the similarity between the proposed ontology L and the ground truth ontology G for each tag community C , and to generate a characteristic extract from each of them, $ce(C, L)$ and $ce(C, G)$, which is inline with [3, 10]. TP, TR and TF can then be computed by averaging the tp , tr and tf of all the communities. tf is the harmonic mean of tp and tr . For each community, the evaluation metrics tp and tr can be computed as follows:

$$tp(C, L, G) = \frac{|ce(C, L) \cap ce(C, G)|}{|ce(C, L)|} \quad (7)$$

$$tr(C, L, G) = \frac{|ce(C, L) \cap ce(C, G)|}{|ce(C, G)|} \quad (8)$$

4.3 Baseline Methods

We compare the performance of each of the proposed tag relationship mining algorithms with state of the art methods. For parent-child relationship, we compared with co-occurrence based methods as used in [2, 6]. For sibling relationship, we followed a evaluation set up as in [3], where we use three similarity metrics, namely Jaccard similarity, Cosine similarity and KL-divergence, on the topics generated using unsupervised LDA as our baseline methods.

We compare the ontology generation algorithm with three popular Knowledge-bases (KBs) namely DBpedia [1], ConceptNet [9] and WebIsA-Graph [4], which has around 65 billion, 34 million and 3 million relationships,

⁵ <https://superuser.com/>.

respectively. DBpedia and ConceptNet contain both parent-child and sibling relationship, whereas WebIsAGraph contains only parent-child relationships between entities.

4.4 Tag Grouping Evaluation

Here, we present the the evaluation and results of our tag grouping algorithm. We run Infomap algorithm [7] on three tag graphs, G , G^{pop} and G^{cp} and evaluated them based on the modularity and clustering coefficient measures. G is constructed over all the tags in the dataset. For the communities to be comparable, they should be compared on the same nodes. So, after forming communities in G using all the tags, we removed all the infrequent tags before computing the two evaluation measures.

The modularity values are 0.28, 0.55 and 0.8 and the clustering coefficient values are 0.1, 0.26 and 0.6 for G , G^{pop} and G^{cp} , respectively. The communities from G are of very low quality because of infrequent tags and no edge directionality. Since G^{cp} considers both these factors, it gives very high scores.

4.5 Tag Relationship Mining Evaluation

Here, we present the results of our tag relationship mining algorithms by comparing them with the baselines and KBs listed in Sect. 4.3.

Table 1. Parent-child and Sibling

Classifier	Feature set	Performance (%)					
		Parent-Child			Sibling		
		Precision	Recall	F1-score	Precision	Recall	F1-score
LR	Our Method	97.05	93.8	95.4	78	68	70
	Baseline	77	76	76.5	67	62	64
NB	Our Method	95.66	90.8	93.2	79	72	74
	Baseline	71	65	67.9	55	56	56
RF	Our Method	98.9	93.8	96.28	76	60	62
	Baseline	82	83	82.5	58	53	55

To evaluate the performance of each relationship mining algorithm separately, we use three binary classifiers, namely Logistic Regression (LR), Naive Bayes (NB) and Random Forest (RF) as in [3].

For the parent-child case, we took entropy of both the tags participating in the parent-child relationship and edge weight $EWPA$ as the feature set to train the classifiers. In our labelled data, the ratio of positive to negative instances was 2:1. To overcome class imbalance, we reversed some random tag pairs to create the negative instances. For the baseline, we used the three co-occurrence

based features, namely support, confidence and mutual overlap, which was used in [3,6]. Table 1 shows the comparison of our method with the baseline for the three classifiers.

We can observe that all the classifiers achieved higher F1-scores for our method compared to the baseline. Baseline method gave an average F1-scores of 77%, whereas our gave average accuracy of 95%, which demonstrates that our entropy and probabilistic association based feature can capture the parent-child relationship very well.

For the sibling relationship case, we used the two scores, namely *sib* and *sibLLDA*, as features for our method. *sib* score is calculated using the similarities of the tag embeddings, and *sibLLA* score is obtained by calculating the similarities of the tag embeddings of the top-n words from the topics generated using the L-LDA algorithm. For the baseline, we use three similarity metrics given in Sect. 4.3. To maintain consistency with our results, we took the number of topics equal to the number of labels, as in our L-LDA method.

Table 1 shows the comparison of our method with the baseline for the three classifiers. We can observe that NB achieved the best performance on our method with an F1-score of 74%. Classification on the baseline method performed poorly with LR giving the highest F1-score of 64%. On an average, there was 15.63% improvement in prediction accuracy using our method compared to the baseline. One of the main reasons is that assigning the tag documents to the category links as pre-defined topic classes in case of L-LDA is more accurate rather than assigning them to less interpretable latent topics as in case of unsupervised LDA.

4.6 Ontology Construction Evaluation

Most of the previous work [3,10] have evaluated their ontology by comparing it with the ontologies of KBs. But since existing KBs have very low recall for our extracted relations, we compare our and KBs ontology with manually created ground truth ontologies.

Table 2. Comparison of Ontologies

Method	Data	Performance (%)		
		TP	TR	TF
Our Method	All Communities	49.6	74.4	57.65
	Big Communities	49.33	73.66	57.22
	Small Communities	48.8	74.26	56.17
Combined KBs	All Communities	39.35	4.17	7.55
	Big Communities	46.9	2.16	3.9
	Small Communities	35.41	3.37	6.01

Ground truth ontologies were constructed for five tag communities, in which there were three big and two small communities, each having around 200 and 60

relations, respectively. We compared the ontology of our method with ontologies of each KB. Due to space constraint, we omit those results. We observed that WebIsAGraph gives the best result. Since it only contains parent-child relations, we union it with ConceptNet that also had siblings. In Table 2, we present the comparison of our ontology with this combined KBs. The evaluation metrics were detailed in Sect. 4.2. We consider three cases: average of all five communities, average of the three big communities, and average of the two small communities.

For both the cases, our method gives an average TF score of 57%, whereas the combined KBs gave an average TF score of around 6%. As there were many missing relationships in the KBs, the TR values for all the three cases are very low. The precision for big communities for Combined KBs is higher than that of small communities. This is because, as the cluster size grows, the number of highly accurate relations increase. But we can observe that the recall of big communities for KBs is lesser than that of small communities. This is again because as the cluster size grows, the number of missed accurate relations also increase. Our method for all the three cases gives almost the same performance.

5 Conclusion

In this work, we propose algorithms for mining parent-child and sibling relationships between tag pairs, and for constructing an automatic tag ontology. We construct forest of tag ontologies where, each ontology only contains strong relationships between tags. Our evaluation shows that our algorithms can extract accurate tag relations compared to the existing works as well as the standard KBs.

References

1. Auer, S., Bizer, C., Kobilarov, G., Lehmann, J., Cyganiak, R., Ives, Z.: DBpedia: a nucleus for a web of open data. In: Aberer, K., et al. (eds.) ASWC/ISWC -2007. LNCS, vol. 4825, pp. 722–735. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-76298-0_52
2. Chen, K., Dong, X., Zhu, J., Shen, B.: Building a domain knowledge base from Wikipedia: a semi-supervised approach. In: SEKE, pp. 191–196 (2016)
3. Dong, H., Wang, W., Coenen, F., Huang, K.: Knowledge base enrichment by relation learning from social tagging data. *Inf. Sci.* **526**, 203–220 (2020)
4. Faralli, S., Finocchi, I., Ponzetto, S.P., Velardi, P.: WebIsAGraph: a very large hypernymy graph from a web corpus. In: CLiC-it (2019)
5. Liu, B., et al.: Giant: scalable creation of a web-scale ontology. In: Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data, pp. 393–409 (2020)
6. Rêgo, A.S.C., Marinho, L.B., Pires, C.E.S.: A supervised learning approach to detect subsumption relations between tags in folksonomies. In: SAC (2015)
7. Rosvall, M., Bergstrom, C.T.: Multilevel compression of random walks on networks reveals hierarchical organization in large integrated systems. *PloS ONE* **6**, e18209 (2011)

8. Saleh, I., El-Tazi, N.: Finding semantic relationships in folksonomies. In: WI, pp. 174–181 (2018)
9. Speer, R., Chin, J., Havasi, C.: ConceptNet 5.5: an open multilingual graph of general knowledge. arXiv preprint [arXiv:1612.03975](https://arxiv.org/abs/1612.03975) (2016)
10. Strohmaier, M., Helic, D., Benz, D., Körner, C., Kern, R.: Evaluation of folksonomy induction algorithms. TIST **3**(4), 1–22 (2012)
11. Yu, H., Li, H., Mao, D., Cai, Q.: A relationship extraction method for domain knowledge graph construction. World Wide Web **23**(2), 735–753 (2020). <https://doi.org/10.1007/s11280-019-00765-y>
12. Zhang, N., Deng, S., Sun, Z., Chen, X., Zhang, W., Chen, H.: Attention-based capsule networks with dynamic routing for relation extraction. In: EMNLP, pp. 986–992 (2018)