Simple Algorithms for Representing Tag Frequencies in the SCOT Exporter

Hak-Lae Kim
Digital Enterprise Research Institute
National University of Ireland, Galway
IDA Business Park, Lower Dangan
Galway, Ireland
haklae.kim@deri.org

Sung-Kwon Yang
Biomedical Knowledge Engineering Lab
Seoul National University
28-22 Yeonkun-Dong, Chongno-Ku
Seoul, Korea
sungkwon.yang@gmail.com

John G. Breslin
Digital Enterprise Research Institute
National University of Ireland, Galway
IDA Business Park, Lower Dangan
Galway, Ireland
john.breslin@deri.org

Hong-Gee Kim
Biomedical Knowledge Engineering Lab
Seoul National University
28-22 Yeonkun-Dong, Chongno-Ku
Seoul, Korea
hgkim@snu.ac.kr

Abstract

In this paper we describe the SCOT Exporter and its algorithms to create instance data based on the SCOT (Social Semantic Cloud of Tags) ontology for sharing and reusing tag data. The algorithms use tag frequencies and co-occurrence relations to represent statistical information via the SCOT ontology. We give an overview of the Exporter and the algorithms, and then discuss some experimental results.

1. Introduction

Social tagging systems encourage user participation through easy-to-use free tagging tools and produce aggregations of user metadata through bottom-up consensus. However, these systems on the Web are not sufficient to provide uniform way to define the semantics of the tag and to calculate its frequencies. In fact, folksonomies, statistically-weighted list of tags, have faced a number of important linguistic issues (i.e. polysemy, synonymy, homonymy, plurals, and spelling variants, etc.) and statistical issues (i.e. frequency, weighted value, etc.). In comparison with the linguistic issues, the statistical issues have not been focused on at the moment. However, when we intend to integrate or exchange tag data across various applications, different folksonomies, or different users, it is a critical problem because we do not know which type of frequency format (i.e. absolute and relative frequency etc) to use. To solve the limitations, we need a semantically and statistically enriched model to be able to describe tagging information.

The SCOT (Social Semantic Cloud Of Tags)[1] ontology is an ontology for sharing and reusing tag data and representing social relations among individuals. It provides the structure and semantics for describing resources, tags, users and extended tag information such as tag frequency, tag co-occurrence frequency, and tag equivalence.

In this paper, we describe the SCOT Exporter and the algorithms for constructing data based on the SCOT ontology. In particular, the contributions of this paper are:

- SCOT Exporters that can automatically construct SCOT instances from various data sources such as weblogs, relational databases, etc. The algorithms for the Exporters support an efficient way to generate the necessary information for SCOT.
- The experimental evaluation demonstrates that the SCOT Exporter is scalable and generates SCOT instances with good run time performance.

The rest of the paper is organized as follows: Section 2 describes the SCOT Exporters and its algorithms for

[1] `http://scot-project.org`

constructing tags and their co-occurrences. Section 3 describes the experimental setting and results. Section 4 discusses related work and Section 5 concludes the paper.

2. SCOT Exporter

We describe how data using the SCOT ontology can be automatically generated from both client-side weblog engines and databases. Two types of exporters for creating the SCOT ontology are as follows:

- ***Exporter for the WordPress***[2]. This allows the production of SCOT ontology from a blog. Its design is based on the assumption that categories in WordPress are used as tags. This plugin is activated in the plugin menu on the WordPress administration panel and it requires no user configuration in order to work. The ontology created by the plugin can be found at *http://yourhost/scot/scot.rdf*.
- ***Exporter for database***. This aims to create the SCOT ontology data from a large number of RSS feeds in a DBMS. This type of data set consists of multiple users from various blog systems, so it creates the ontology data per each individual user.

Both types of the exporters have almost the same functionalities that basically create user profiles such as name, blog name, blog url etc. and extended tag information such as name, frequency, co-occurrence, hierarchy between tags, equivalence of tags etc.

2.1. Tag Frequency Computation Algorithm

A tag has a weighted frequency that is associated with or assigned to certain items. The frequency of an individual tag would reflect how popular it is.

Frequencies can be expressed as absolute frequencies or relative frequencies [4]. The absolute frequencies are raw observations, that have not been normalized with respect to the base rates of the event in question. When we speak about the frequency of tags it usually means the absolute frequency format. At the same time, the 'relative frequency' means a frequency that is expressed in relation to a sample size or rate. This format is used to compare the occurrence of objects in two or more groups. Accordingly, this format is used in tag clouds where the size of each tag represents the proportion of the tags. Simply stated, an absolute frequency, the actual number of frequency, tells us how a frequently tag is used in a give domain while a relative frequency of each tag means its proportion in the total tag occurrence.

In SCOT, this information is represented by the `scot:ownAFrequency` and `scot:ownRFrequency` properties which are subproperties of `scot:AFrequency` and `scot:RFrequency` respectively[3]. We use the Tag Frequency Computation (see Algorithm 1) to calculate the frequency of each tag.

Algorithm 1 Tag Frequency Computation

```
Require: tagList is a list of tags
  var tagList ← empty list
  var totalTagFrequency ← 0

  for all item ∈ getUserItemList(user) do
    for all tag ∈ getItemTagList(item) do
      if tagList.exist(tag) then
        tag.AFrequency ← tag.AFrequency + 1
      else
        tagList.add(tag)
      end if
      totalTagFrequency ← totalTagFrequency + 1
    end for
  end for
  for all tag ∈ tagList do
    tag.RFrequency ← tag.AFrequency / totalTagFrequency
  end for
```

Several functions are assumed: *getUserItemList(user)* returns the item lists for the user from the data source. *getItemTagList(item)* gets the tag lists for the given item. *tagList.exist(tag)* returns true if the tag exists in the tagList. *tagList.add(tag)* inserts the tag into the tagList.

2.2. Co-occurrence Frequency Computation Algorithm

People are inclined to generate one more tags when they tag in resources. The meaning of the tag becomes more specific when the tag is combined with a set of tags. To simply define the term co-occurrence: if an item contains both the tags ***semanticweb*** and ***blog***, these two tags are said to co-occur or have a first order co-occurrence. It can play an important role to reduce '*tag ambiguity*' in tagging systems. `scot:cooccurAFrequency`

[2] `http://scot-project.org/?page_id=12`

[3] The former is intended to describe the absolute format and the purpose of the latter is to represent the relative format

and `scot:cooccurRFrequency` properties describe co-occurrence as an absolute and relative value to the frequency amongst a set of tags. Co-occurrence frequencies will be computed by the Co-occurrence Frequency Computation, as shown in Algorithm 2. Note that *getUserItemList(user)* returns the item

Algorithm 2 Co-occurrence Frequency Computation

Require: cooccurList is a list of co-occurrences
  var cooccurList ← empty list
  var totalCooccurFrequency ← 0
  for all *item* ∈ getUserItemList(*user*) **do**
    itemTagList ← getItemTagList(*item*)
    if *itemTagList.count* ≥ 2 **then**
      cooccur ← cooccurList.exist(*itemTagList*)
      if *cooccur* ≠ *null* **then**
        cooccur.AFrequency ← *cooccur.AFrequency* + 1
      else
        cooccur ← newCooccur(*itemTagList*)
        cooccurList.add(*cooccur*)
      end if
      totalCooccurFrequency ← *totalCooccurFrequency* + 1
    end if
  end for
  for all *cooccur* ∈ *cooccurList* **do**
    cooccur.RFrequency ← *cooccur.AFrequency* / *totalCooccurFrequency*
  end for

list for the user from the data source. *getItemTagList(item)* gets the tag list for the given item. *cooccurList.exist(itemTagList)* returns the tag set if there exists a same cooccurring tag set in the *cooccurList*. *cooccurList.add(cooccur)* inserts the co-occurring tag set into the *cooccurList*.

3. Evaluation

3.1. Datasets

Planet Journals is a blog aggregator website that collects country-specific blogs for residents or citizens of a particular country. The Ireland site (planet.journals.ie) has a collection of around 1322 blogs, 119101 posts, and 13688 tags. The data set used here is taken from that website during the two years period between February 2005 and October 2006, and the software that powers the website (Planet PHP) also allows the aggregation of any tags that are used in the various blogs and that are passed through via RSS syndication feeds (e.g. using the dc:subject attribute for an RSS item).

3.2. Evaluation Results

All the tests were run on an Intel T2600 2.16 GHz machine with 2 GB of RAM. The mean number of posts per blog for the data set is 96.6 and the mean number of tags per blog is 29.4. The mean number of frequency of tag usage per blog is 97.4, therefore more than 30% of the tags are overlapping (i.e. 29.4/97.4). Table 1 shows the top five tags. We can see that several tags are associated with the country Ireland (i.e. *irishblogs* and *ireland* etc).

Tag Ranking	AF	RF
irishblogs	2557	3.7%
general	1828	2.7%
ireland	1821	2.7%
uncategorized	976	1.4%
music	714	1.0%
.	.	.
.	.	.
∑	68785	100%

Table 1. Top five tags

The co-occurring tags[4] in Table 2 are created by calculating tags which appeared together using Algorithm 2. We can see that the tags such as *irishblogs, ireland, blogs, and irish* in Table 1 are frequently used with others. This means that frequently used tags are likely to combine with other tags. In SCOT, only the first order co-occurrence that happened in real world usage without a conditional probability is described. It is not the main of the SCOT to describe a similarity among tags or other types of term relationships as this ontology is intended to reflect the fact that tags happened in a given domain.

The run time performance is a critical issue when creating and maintaining a SCOT data set, due to the complex analysis of word relations and the frequent updates. To generate the SCOT data, we must carry out analyzes such as tag frequencies and co-occurrence relations among tags using the Algorithms 1-2 in order. Figure 1 shows the runtime performance for the given data. The number of *tags+cooccurrences* on the Y axis is found from the sum of '*total frequency of tags*' and '*total frequency of tag sets*'. The time required depends on the amount of tags: the mean time for it is 1.1 seconds. Most cases in the data (under 200 tags)

[4] gaeilge is the Irish word for gaelic and podchraoladh is podcasting.

Cooccurrences	AF	RF
gaeilge irishblogs podchraoladh	71	0.8%
cork flickr ireland photography	36	0.4%
blogs irishblogs	35	0.4%
ireland real news	32	0.4%
irishphotos photography	31	0.4%
.	.	.
.	.	.
$\sum$	11514	100%

Table 2. Top five co-occurring tags

are generated in less than 1 second. Although there are several large tags in the data, it takes less than 6 seconds to generate the corresponding SCOTs. We believe that the algorithms and the tool produces an efficient performance in terms of time.

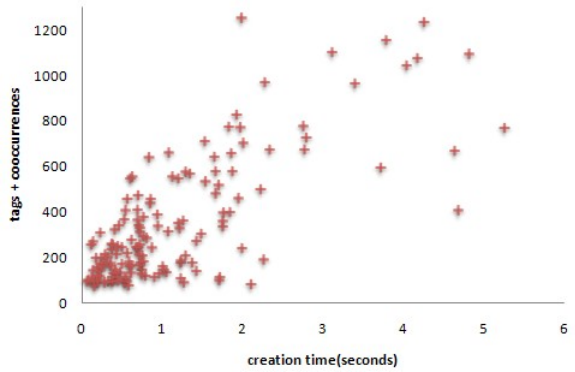

Figure 1. Runtime performance

4. Related Work

There are several efforts that try to represent the concept of tagging, the operation of tagging, and the tags themselves. Newman [1] describes the relationship between an agent, an arbitrary resource, and one or more tags. In his ontology, there are three core concepts such as Taggers, Tagging, and Tags to represent tagging activity. Gruber [2] describes the core idea of tagging that consists of object, tag, tagger, and source. Knerr [3] describes the concept of tagging in the *Tagging Ontology*. Since his approach is based on the ideas from [1, 2], the core element of the ontology is Tagging. The ontology consists of time, user, domain, visibility, tag, resource, and type. There are some other projects [5, 6] related to tag sharing and semantic tagging.

The approaches in the related work are focused on the tagging activities or events that people use to tag resources using keyword. Therefore the core concepts of the ontologies are *Tagging*, *Tagger* and *Resource* to represent users, events, and resources respectively. However, there is no way to describe frequency of tags in the ontologies. The SCOT ontology can be easily represent this information using properties of tag frequency.

5 Conclusion

The SCOT Exporter provides an efficient run time performance to construct SCOT instances. The algorithms for the exporter are simple and provide the complete set of information to describe metadata using the SCOT ontology. The exporters are implemented for each site or application since there are many possible kinds of social tagging sites for which exporters can be developed. Our approach is a starting point to represent the structure and the semantics of social tagging. We will provide further information through the project web site (http://scot-project.org).

Acknowledgments

This material is based upon works supported by the Science Foundation Ireland under Grant No. SFI/02/CE1/I131.

References

[1] R. Newman, Tag Ontology design, available at: http://www.holygoat.co.uk/projects/tags/ (viewed 16/08/2007), 2005.

[2] T. Gruber, Ontology of Folksonomy: A Mash-up of Apples and Oranges, First on-Line conference on Metadata and Semantics Research (MTSR'05). http://www.metadata-semantics.org/.

[3] Knerr, T. (2006),Tagging Ontology- Towards a Common Ontology for Folksonomies, available at: http://code.google.com/p/tagont/

[4] Clare Harries, N. H., Are absolute frequencies, relative frequencies, or both effective in reducing cognitive biases?, Journal of Behavioral Decision Making 13, Issue 4, 431-444, 2000.

[5] Tagcommons project website: http://tagcommons.org

[6] Tagora Project website: http://www.tagora-project.eu