Automatically Generating Term-frequency-induced Taxonomies

Karin Murthy **Tanveer A Faruquie** **L Venkata Subramaniam**
K Hima Prasad **Mukesh Mohania**
IBM Research - India
{karinmur|ftanveer|lvsubram|hkaranam|mkmukesh}@in.ibm.com

Abstract

We propose a novel method to automatically acquire a term-frequency-based taxonomy from a corpus using an unsupervised method. A term-frequency-based taxonomy is useful for application domains where the frequency with which terms occur on their own and in combination with other terms imposes a natural term hierarchy. We highlight an application for our approach and demonstrate its effectiveness and robustness in extracting knowledge from real-world data.

1 Introduction

Taxonomy deduction is an important task to understand and manage information. However, building taxonomies manually for specific domains or data sources is time consuming and expensive. Techniques to automatically deduce a taxonomy in an unsupervised manner are thus indispensable. Automatic deduction of taxonomies consist of two tasks: extracting relevant terms to represent concepts of the taxonomy and discovering relationships between concepts. For unstructured text, the extraction of relevant terms relies on information extraction methods (Etzioni et al., 2005).

The relationship extraction task can be classified into two categories. Approaches in the first category use lexical-syntactic formulation to define patterns, either manually (Kozareva et al., 2008) or automatically (Girju et al., 2006), and apply those patterns to mine instances of the patterns. Though producing accurate results, these approaches usually have low coverage for many domains and suffer from the problem of inconsistency between terms when connecting the instances as chains to form a taxonomy. The second category of approaches uses clustering to discover terms and the relationships between them (Roy and Subramaniam, 2006), even if those relationships do not explicitly appear in the text. Though these methods tackle inconsistency by addressing taxonomy deduction globally, the relationships extracted are often difficult to interpret by humans.

We show that for certain domains, the frequency with which terms appear in a corpus on their own and in conjunction with other terms induces a natural taxonomy. We formally define the concept of a term-frequency-based taxonomy and show its applicability for an example application. We present an unsupervised method to generate such a taxonomy from scratch and outline how domain-specific constraints can easily be integrated into the generation process. An advantage of the new method is that it can also be used to extend an existing taxonomy.

We evaluated our method on a large corpus of real-life addresses. For addresses from emerging geographies no standard postal address scheme exists and our objective was to produce a postal taxonomy that is useful in standardizing addresses (Kothari et al., 2010). Specifically, the experiments were designed to investigate the effectiveness of our approach on noisy terms with lots of variations. The results show that our method is able to induce a taxonomy without using any kind of lexical-semantic patterns.

2 Related Work

One approach for taxonomy deduction is to use explicit expressions (Iwaska et al., 2000) or lexical and semantic patterns such as *is a* (Snow et al., 2004), *similar usage* (Kozareva et al., 2008), *synonyms and antonyms* (Lin et al., 2003), *purpose* (Cimiano and Wenderoth, 2007), and *employed by* (Bunescu and Mooney, 2007) to extract and organize terms. The quality of extraction is often controlled using statistical measures (Pantel and Pennacchiotti, 2006) and external resources such as wordnet (Girju et al., 2006). However, there are

Proceedings of the ACL 2010 Conference Short Papers, pages 126–131,
Uppsala, Sweden, 11-16 July 2010.

domains (such as the one introduced in Section 3.2) where the text does not allow the derivation of linguistic relations.

Supervised methods for taxonomy induction provide training instances with global semantic information about concepts (Fleischman and Hovy, 2002) and use bootstrapping to induce new seeds to extract further patterns (Cimiano et al., 2005). Semi-supervised approaches start with known terms belonging to a category, construct context vectors of classified terms, and associate categories to previously unclassified terms depending on the similarity of their context (Tanev and Magnini, 2006). However, providing training data and hand-crafted patterns can be tedious. Moreover in some domains (such as the one presented in Section 3.2) it is not possible to construct a context vector or determine the replacement fit.

Unsupervised methods use clustering of word-context vectors (Lin, 1998), co-occurrence (Yang and Callan, 2008), and conjunction features (Caraballo, 1999) to discover implicit relationships. However, these approaches do not perform well for small corpora. Also, it is difficult to label the obtained clusters which poses challenges for evaluation. To avoid these problems, incremental clustering approaches have been proposed (Yang and Callan, 2009). Recently, lexical entailment has been used where the term is assigned to a category if its occurrence in the corpus can be replaced by the lexicalization of the category (Giuliano and Gliozzo, 2008). In our method, terms are incrementally added to the taxonomy based on their support and context.

Association rule mining (Agrawal and Srikant, 1994) discovers interesting relations between terms, based on the frequency with which terms appear together. However, the amount of patterns generated is often huge and constructing a taxonomy from all the patterns can be challenging. In our approach, we employ similar concepts but make taxonomy construction part of the relationship discovery process.

3 Term-frequency-induced Taxonomies

For some application domains, a taxonomy is induced by the frequency in which terms appear in a corpus on their own and in combination with other terms. We first introduce the problem formally and then motivate it with an example application.

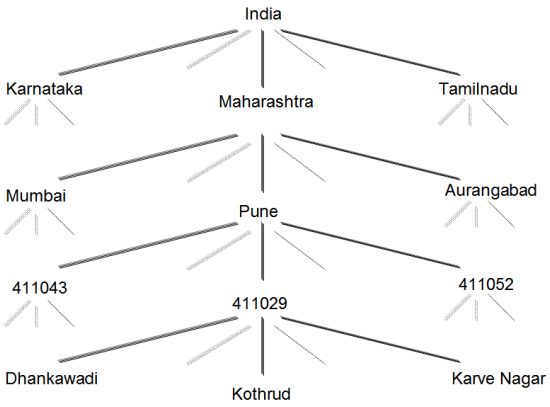

Figure 1: Part of an address taxonomy

3.1 Definition

Let C be a corpus of records r. Each record is represented as a set of terms t. Let $T = \{t \mid t \in r \ \wedge \ r \in C\}$ be the set of all terms of C. Let $f(t)$ denote the frequency of term t, that is the number of records in C that contain t. Let $F(t, T^+, T^-)$ denote the *frequency* of term t given a set of must-also-appear terms T^+ and a set of cannot-also-appear terms T^-. $F(t, T^+, T^-) = |\{r \in C \mid t \in r \wedge \forall t' \in T^+ : t' \in r \wedge \forall t' \in T^- : t' \notin r\}|$.

A term-frequency-induced taxonomy (TFIT), is an ordered tree over terms in T. For a node n in the tree, $n.t$ is the term at n, $A(n)$ the ancestors of n, and $P(n)$ the predecessors of n.

A TFIT has a root node with the special term $\perp$ and the conditional frequency ∞. The following condition is true for any other node n:
$\forall t \in T, F(n.t, A(n), P(n)) \geq F(t, A(n), P(n))$.
That is, each node's term has the highest conditional frequency in the context of the node's ancestors and predecessors. Only terms with a conditional frequency above zero are added to a TFIT.

We show in Section 4 how a TFIT taxonomy can be automatically induced from a given corpus. But before that, we show that TFITs are useful in practice and reflect a natural ordering of terms for application domains where the concept hierarchy is expressed through the frequency in which terms appear.

3.2 Example Domain: Address Data

An address taxonomy is a key enabler for address standardization. Figure 1 shows part of such an address taxonomy where the root contains the most generic term and leaf-level nodes contain the most specific terms. For emerging economies building a standardized address taxonomy is a huge chal-

Row	Term	Part of address	Category
1	D-15	house number	alphanumerical
2	Rawal	building name	proper noun
3	Complex	building name	proper noun
4	Behind	landmark	marker
5	Hotel	landmark	marker
6	Ruchira	landmark	proper noun
7	Katre	street	proper noun
8	Road	street	marker
9	Jeevan	area	proper noun
10	Nagar	area	marker
11	Andheri	city (taluk)	proper noun
12	East	city (taluk)	direction
13	Mumbai	district	proper noun
14	Maharashtra	state	proper noun
15	400069	ZIP code	6 digit string

Table 1: Example of a tokenized address

lenge. First, new areas and with it new addresses constantly emerge. Second, there are very limited conventions for specifying an address (Faruquie et al., 2010). However, while many developing countries do not have a postal taxonomy, there is often no lack of address data to learn a taxonomy from.

Column 2 of Table 1 shows an example of an Indian address. Although Indian addresses tend to follow the general principal that more specific information is mentioned earlier, there is no fixed order for different elements of an address. For example, the ZIP code of an address may be mentioned before or after the state information and, although ZIP code information is more specific than city information, it is generally mentioned later in the address. Also, while ZIP codes often exist, their use by people is very limited. Instead, people tend to mention copious amounts of landmark information (see for example rows 4-6 in Table 1).

Taking all this into account, there is often not enough structure available to automatically infer a taxonomy purely based on the structural or semantic aspects of an address. However, for address data, the general-to-specific concept hierarchy is reflected in the frequency with which terms appear on their own and together with other terms.

It mostly holds that $f(s) > f(d) > f(c) > f(z)$ where s is a state name, d is a district name, c is a city name, and z is a ZIP code. However, sometimes the name of a large city may be more frequent than the name of a small state. For example, in a given corpus, the term 'Houston' (a populous US city) may appear more frequent than the term 'Vermont' (a small US state). To avoid that 'Houston' is picked as a node at the first level of the taxonomy (which should only contain states), the conditional-frequency constraint introduced in Section 3.1 is enforced for each node in a TFIT. 'Houston's state 'Texas' (which is more frequent) is picked before 'Houston'. After 'Texas' is picked it appears in the "cannot-also-appear"' list for all further siblings on the first level, thus giving 'Houston' has a conditional frequency of zero.

We show in Section 5 that an address taxonomy can be inferred by generating a TFIT taxonomy.

4 Automatically Generating TFITs

We describe a basic algorithm to generate a TFIT and then show extensions to adapt to different application domains.

4.1 Base Algorithm

Algorithm 1 Algorithm for generating a TFIT.

// For initialization T^+, T^- are empty
// For initialization l,w are zero
genTFIT(T^+, T^-, C, l, w)
// select most frequent term
t_{next} = t_j with $F(t_j, T^+, T^-)$ is maximal amongst all $t_j \in C$;
f_{next} = $F(t_{next}, T^+, T^-)$;
if $f_{next} \geq$ support **then**
  //Output node (t_j, l, w)
  ...
  // Generate child node
  genTFIT($T^+ \cup \{t_{next}\}, T^-, C, l+1, w$)
  // Generate sibling node
  genTFIT($T^+, T^- \cup \{t_{next}\}, C, l, w+1$)
end if

To generate a TFIT taxonomy as defined in Section 3.1 we recursively pick the most frequent term given previously chosen terms. The basic algorithm $genTFIT$ is sketched out in Algorithm 1. When $genTFIT$ is called the first time, T^+ and T^- are empty and both level l and width w are zero. With each call of $genTFIT$ a new node n in the taxonomy is created with (t, l, w) where t is the most frequent term given T^+ and T^- and l and w capture the position in the taxonomy. $genTFIT$ is recursively called to generate a child of n and a sibling for n.

The only input parameter required by our algorithm is $support$. Instead of adding all terms with a conditional frequency above zero, we only add terms with a conditional frequency equal to or higher than $support$. The $support$ parameter controls the precision of the resulting TFIT and also the runtime of the algorithm. Increasing $support$ increases the precision but also lowers the recall.

4.2 Integrating Constraints

Structural as well as semantic constraints can easily be integrated into the TFIT generation.

We distinguish between taxonomy-level and node-level structural constraints. For example, limiting the depth of the taxonomy by introducing a $maxLevel$ constraint and checking before each recursive call if $maxLevel$ is reached, is a taxonomy-level constraint. A node-level constraint applies to each node and affects the way the frequency of terms is determined.

For our example application, we introduce the following node-level constraint: at each node we only count terms that appear at specific positions in records with respect to the current level of the node. Specifically, we slide (or incrementally increase) a window over the address records starting from the end. For example, when picking the term 'Washington' as a state name, occurrences of 'Washington' as city or street name are ignored. Using a window instead of an exact position accounts for positional variability. Also, to accommodate varying amounts of landmark information we length-normalize the position of terms. That is, we divide all positions in an address by the average length of an address (which is 10 for our 40 Million addresses). Accordingly, we adjust the size of the window and use increments of 0.1 for sliding (or increasing) the window.

In addition to syntactical constraints, semantic constraints can be integrated by classifying terms for use when picking the next frequent term. In our example application, markers tend to appear much more often than any proper noun. For example, the term 'Road' appears in almost all addresses, and might be picked up as the most frequent term very early in the process. Thus, it is beneficial to ignore marker terms during taxonomy generation and adding them as a post-processing step.

4.3 Handling Noise

The approach we propose naturally handles noise by ignoring it, unless the noise level exceeds the support threshold. Misspelled terms are generally infrequent and will as such not become part of the taxonomy. The same applies to incorrect addresses. Incomplete addresses partially contribute to the taxonomy and only cause a problem if the same information is missing too often. For example, if more than $support$ addresses with the city 'Houston' are missing the state 'Texas', then 'Houston' may become a node at the first level and appear to be a state. Generally, such cases only appear at the far right of the taxonomy.

5 Evaluation

We present an evaluation of our approach for address data from an emerging economy. We implemented our algorithm in Java and store the records in a DB2 database. We rely on the DB2 optimizer to efficiently retrieve the next frequent term.

5.1 Dataset

The results are based on 40 Million Indian addresses. Each address record was given to us as a single string and was first tokenized into a sequence of terms as shown in Table 1. In a second step, we addressed spelling variations. There is no fixed way of transliterating Indian alphabets to English and most Indian proper nouns have various spellings in English. We used tools to detect synonyms with the same context to generate a list of rules to map terms to a standard form (Lin, 1998). For example, in Table 1 'Maharashtra' can also be spelled 'Maharastra'. We also used a list of keywords to classify some terms as markers such as 'Road' and 'Nagar' shown in Table 1.

Our evaluation consists of two parts. First, we show results for constructing a TFIT from scratch. To evaluate the precision and recall we also retrieved post office addresses from India Post[1], cleaned them, and organized them in a tree.

Second, we use our approach to enrich the existing hierarchy created from post office addresses with additional area terms. To validate the result, we also retrieved data about which area names appear within a ZIP code.[2] We also verified whether Google Maps shows an area on its map.[3]

5.2 Taxonomy Generation

We generated a taxonomy O using all 40 million addresses. We compare the terms assigned to category levels *district* and *taluk*[4] in O with the tree P constructed from post office addresses. Each district and taluk has at least one post office. Thus P covers all districts and taluks and allows us to test coverage and precision. We compute the precision and recall for each category level CL as

[1] http://www.indiapost.gov.in/Pin/pinsearch.aspx
[2] http://www.whereincity.com/india/pincode/search
[3] maps.google.com
[4] Administrative division in some South-Asian countries.

Support		Recall %	Precision %
100	District	93.9	57.4
	Taluk	50.9	60.5
200	District	87.9	64.4
	Taluk	49.6	66.1

Table 2: Precision and recall for categorizing terms belonging to the state Maharashtra

$$Recall_{CL} = \frac{\#\ correct\ paths\ from\ root\ to\ CL\ in\ O}{\#\ paths\ from\ root\ to\ CL\ in\ P}$$

$$Precision_{CL} = \frac{\#\ correct\ paths\ from\ root\ to\ CL\ in\ O}{\#\ paths\ from\ root\ to\ CL\ in\ O}$$

Table 2 shows precision and recall for district and taluk for the large state Maharashtra. Recall is good for district. For taluk it is lower because a major part of the data belongs to urban areas where taluk information is missing. The precision seems to be low but it has to be noted that in almost 75% of the addresses either district or taluk information is missing or noisy. Given that, we were able to recover a significant portion of the knowledge structure.

We also examined a branch for a smaller state (Kerala). Again, both districts and taluks appear at the next level of the taxonomy. For a support of 200 there are 19 entries in O of which all but two appear in P as district or taluk. One entry is a taluk that actually belongs to Maharashtra and one entry is a name variation of a taluk in P. There were not enough addresses to get a good coverage of all districts and taluks.

5.3 Taxonomy Augmentation

We used P and ran our algorithm for each branch in P to include area information. We focus our evaluation on the city Mumbai. The recall is low because many addresses do not mention a ZIP code or use an incorrect ZIP code. However, the precision is good implying that our approach works even in the presence of large amounts of noise.

Table 3 shows the results for ZIP code 400002 and 400004 for a support of 100. We get similar results for other ZIP codes. For each detected area we compared whether the area is also listed on whereincity.com, part of a post office name (PO), or shown on google maps. All but four areas found are confirmed by at least one of the three external sources. Out of the unconfirmed terms $Fanaswadi$ and $MarineDrive$ seem to be genuine area names but we could not confirm $DhakurdwarRoad$. The term th is due to our tokenization process. 16 correct terms out of 18 terms results in a precision of 89%.

Area	Whereincity	PO	Google
Bhuleshwar	yes	no	yes
Chira Bazar	yes	no	yes
Dhobi Talao	no	no	yes
Fanaswadi	no	no	no
Kalbadevi Road	yes	yes	yes
Marine Drive	no	no	no
Marine Lines	yes	yes	yes
Princess Street	no	no	yes
th	no	no	no
Thakurdwar Road	no	no	no
Zaveri Bazar	yes	no	yes
Charni Road	no	yes	no
Girgaon	yes	yes	yes
Khadilkar Road	yes	no	yes
Khetwadi Road	yes	no	no
Kumbharwada	no	no	yes
Opera House	no	yes	no
Prathna Samaj	yes	no	no

Table 3: Areas found for ZIP code 400002 (top) and 400004 (bottom)

We also ran experiments to measure the coverage of area detection for Mumbai without using ZIP codes. Initializing our algorithm with $Maharshtra$ and $Mumbai$ yielded over 100 areas with a support of 300 and more. However, again the precision is low because quite a few of those areas are actually taluk names.

Using a large number of addresses is necessary to achieve good recall and precision.

6 Conclusion

In this paper, we presented a novel approach to generate a taxonomy for data where terms exhibit an inherent frequency-based hierarchy. We showed that term frequency can be used to generate a meaningful taxonomy from address records. The presented approach can also be used to extend an existing taxonomy which is a big advantage for emerging countries where geographical areas evolve continuously.

While we have evaluated our approach on address data, it is applicable to all data sources where the inherent hierarchical structure is encoded in the frequency with which terms appear on their own and together with other terms. Preliminary experiments on real-time analyst's stock market tips [5] produced a taxonomy of (TV station, Analyst, Affiliation) with decent precision and recall.

[5] See Live Market voices at: http://money.rediff.com/money/jsp/markets_home.jsp

References

Rakesh Agrawal and Ramakrishnan Srikant. 1994. Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th International Conference on Very Large Data Bases*, pages 487–499.

Razvan C. Bunescu and Raymond J. Mooney. 2007. Learning to extract relations from the web using minimal supervision. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 576–583.

Sharon A. Caraballo. 1999. Automatic construction of a hypernym-labeled noun hierarchy from text. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 120–126.

Philipp Cimiano and Johanna Wenderoth. 2007. Automatic acquisition of ranked qualia structures from the web. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, pages 888–895.

Philipp Cimiano, Günter Ladwig, and Steffen Staab. 2005. Gimme' the context: context-driven automatic semantic annotation with c-pankow. In *Proceedings of the 14th International Conference on World Wide Web*, pages 332–341.

Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. 2005. Unsupervised named-entity extraction from the web: an experimental study. *Artificial Intelligence*, 165(1):91–134.

Tanveer A. Faruquie, K. Hima Prasad, L. Venkata Subramaniam, Mukesh K. Mohania, Girish Venkatachaliah, Shrinivas Kulkarni, and Pramit Basu. 2010. Data cleansing as a transient service. In *Proceedings of the 26th International Conference on Data Engineering*, pages 1025–1036.

Michael Fleischman and Eduard Hovy. 2002. Fine grained classification of named entities. In *Proceedings of the 19th International Conference on Computational Linguistics*, pages 1–7.

Roxana Girju, Adriana Badulescu, and Dan Moldovan. 2006. Automatic discovery of part-whole relations. *Computational Linguistics*, 32(1):83–135.

Claudio Giuliano and Alfio Gliozzo. 2008. Instance-based ontology population exploiting named-entity substitution. In *Proceedings of the 22nd International Conference on Computational Linguistics*, pages 265–272.

Lucja M. Iwaska, Naveen Mata, and Kellyn Kruger. 2000. Fully automatic acquisition of taxonomic knowledge from large corpora of texts. In Lucja M. Iwaska and Stuart C. Shapiro, editors, *Natural Language Processing and Knowledge Representation: Language for Knowledge and Knowledge for Language*, pages 335–345.

Govind Kothari, Tanveer A Faruquie, L V Subramaniam, K H Prasad, and Mukesh Mohania. 2010. Transfer of supervision for improved address standardization. In *Proceedings of the 20th International Conference on Pattern Recognition*.

Zornitsa Kozareva, Ellen Riloff, and Eduard Hovy. 2008. Semantic class learning from the web with hyponym pattern linkage graphs. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 1048–1056.

Dekang Lin, Shaojun Zhao, Lijuan Qin, and Ming Zhou. 2003. Identifying synonyms among distributionally similar words. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, pages 1492–1493.

Dekang Lin. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 17th International Conference on Computational Linguistics*, pages 768–774.

Patrick Pantel and Marco Pennacchiotti. 2006. Espresso: leveraging generic patterns for automatically harvesting semantic relations. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pages 113–120.

Shourya Roy and L Venkata Subramaniam. 2006. Automatic generation of domain models for call centers from noisy transcriptions. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, pages 737–744.

Rion Snow, Daniel Jurafsky, and Andrew Y. Ng. 2004. Learning syntactic patterns for automatic hypernym discovery. In *Advances in Neural Information Processing Systems*, pages 1297–1304.

Hristo Tanev and Bernardo Magnini. 2006. Weakly supervised approaches for ontology population. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*, pages 3–7.

Hui Yang and Jamie Callan. 2008. Learning the distance metric in a personal ontology. In *Proceeding of the 2nd International Workshop on Ontologies and Information Systems for the Semantic Web*, pages 17–24.

Hui Yang and Jamie Callan. 2009. A metric-based framework for automatic taxonomy induction. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 271–279.