

EXPLORING KNOWLEDGE BASES FOR ENGINEERING A USER INTERESTS
HIERARCHY FOR SOCIAL NETWORK APPLICATIONS

by

MANDAR HARIDAS

B.E., University of Pune, India, 2004.

A THESIS

submitted in partial fulfillment of the requirements for the degree

MASTER OF SCIENCE

Department of Computing and Information Sciences
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas

2009

Approved by:

Co-Major Professor
Doina Caragea

Approved by:

Co-Major Professor
Gurdip Singh

Copyright

MANDAR HARIDAS

2009

Abstract

In the recent years, social networks have become an integral part of our lives. Their outgrowth has resulted in opportunities for interesting data mining problems, such as interest or friendship recommendations. A global ontology over the interests specified by the users of a social network is essential for accurate recommendations. The focus of this work is on engineering such an interest ontology. In particular, given that the resulting ontology is meant to be used for data mining applications to social network problems, we explore only hierarchical ontologies. We propose, evaluate and compare three approaches to engineer an interest hierarchy. The proposed approaches make use of two popular knowledge bases, Wikipedia and Directory Mozilla, to extract interest definitions and/or relationships between interests. More precisely, the first approach uses Wikipedia to find interest definitions, the latent semantic analysis technique to measure the similarity between interests based on their definitions, and an agglomerative clustering algorithm to group similar interests into higher level concepts. The second approach uses the Wikipedia Category Graph to extract relationships between interests. Similarly, the third approach uses Directory Mozilla to extract relationships between interests. Our results indicate that the third approach, although the simplest, is the most effective for building an ontology over user interests. We use the ontology produced by the third approach to construct interest based features. These features are further used to learn classifiers for the friendship prediction task. The results show the usefulness of the ontology with respect to the results obtained in absence of the ontology.

Table of Contents

List of Figures	vi
List of Tables	vii
Acknowledgements	viii
CHAPTER 1 - Introduction	1
Background	1
High level overview	2
CHAPTER 2 - Related work	4
CHAPTER 3 - Ontology engineering using Wikipedia and LSA	6
Approach	6
Discussion	10
CHAPTER 4 - Ontology engineering using Wikipedia Category Graph	12
Approach	12
Discussion	13
CHAPTER 5 - Ontology engineering using Directory Mozilla	15
Approach	15
Discussion	17
CHAPTER 6 - Ontology evaluation	18
Interest based nominal features without ontology	18
Interest based nominal features with ontology	18
Interest based numerical features	19
Graph based features	20
CHAPTER 7 - Experiments	22
Setup	22
Results	23
CHAPTER 8 - Conclusion	26
Summary	26
Wikipedia v/s DMoz	26
DMoz ontology limitations	29

CHAPTER 9 - Future work	30
Improvements to the Wikipedia/LSA approach	30
Searching interests in Wikipedia and DMoz dump	31
Improvement in constructing interest based features	31
Studying other prediction problems.....	31
References.....	32

List of Figures

Figure 3.1 Hierarchy over the user interests in Table 1 when WordNet/IMDB/AWS are used as knowledge bases.	7
Figure 3.2 Term-Document matrix snapshot for the sample data. Number of terms = 797 (only 14 shown), number of documents = 10.....	8
Figure 3.3 Matrix snapshot obtained after SVD and dimensionality reduction	8
Figure 3.4 Similarity matrix between documents in sample data.....	9
Figure 3.5 Ontology engineered for data in Table 1 using Wikipedia and LSA based approach	10
Figure 4.1 Fragment of the ontology engineered using the WCG-based approach.....	14
Figure 5.1 Snapshot of DMoz RDF dump (downloaded from http://rdf.dmoz.org/)	15
Figure 5.2 Snapshot of the ontology table for the DMoz based approach.....	16
Figure 5.3 Fragment of the ontology engineered using DMoz.....	16
Figure 6.1 Example to illustrate construction of interest based nominal features with ontology.	19
Figure 7.1 Graph of reported AUC values v/s level numbers in the ontology used, for Random Forest classifier	25
Figure 7.2 Graph of reported AUC values v/s level numbers in the ontology used, for Support Vector Machine classifier	25

List of Tables

Table 3.1 Interests data used to illustrate the ontology engineering approaches explored.....	6
Table 5.1 Ontology levels and number of nodes at each level	17
Table 7.1 AUC values for Random Forests (RF) and Support Vector Machines (SVM) classifiers. We assume that k% links are known in the test set, where k is 50, 25 and 10, respectively. The known links are used to construct graph features.....	23
Table 8.1 Comparison between Wikipedia and DMOz.....	27

Acknowledgements

Throughout the course of this work, I have been supported by professors, colleagues and family members alike, without the support of whom, this work would not have been possible.

Dr. Doina Caragea, my co-major professor, has made it possible to think out of the box solutions to problems, through her constant in flow of new ideas. Her support, suggestions and encouragement have helped meet crash timelines in the lifecycle of this work. With her vast knowledge and efficient solutioning, even the most complicated design issues could be overcome. Right from making sure the timely availability of resources for the project, to providing solutions to the most challenging problems, Dr. Caragea has well and truly been instrumental in helping get this work to completion.

Dr. Gurdip Singh, also my co-major professor, has greatly helped, getting the basics right. His teaching has been a source of a lot of learning as far as making my course work foundation stronger and increasing my desire to learn more about new technologies. I sincerely thank Dr. Singh to help clear even the slightest of doubts and the most basic of questions.

Dr. Torben Amtoft, member of my thesis committee, has educated me with important algorithmic concepts through his classes that have helped at every step of this work. I am thankful to Dr. Amtoft for always being around with constructive ideas and suggestions, which have formed a pivotal part of this work.

I cannot thank enough my parents and my wife who have been the pillars of strength, motivation and support for me. This work and degree are the result of their relentless faith, trust and continued support and assistance, on and for me.

I would also like to thank all my friends for chipping in with useful suggestions and for always being there to help me during the entire course of this work and degree. Thank you for your well wishes and blessings.

Finally, thank you Dr. Hsu and the KDD (Knowledge Discovery in Databases) group for sharing their data and valuable learnings on previous work.

CHAPTER 1 - Introduction

Background

Over the years, there has been a dramatic increase in the number of social network users. Collectively, the top ten social network sites have grown at a rate of almost fifty percent every year (Bausch and Han, 2006). In the coming years, it is expected that social networking will become more ingrained in mainstream websites. As a consequence, there is a greater need for data mining in social networks. For example, using data mining, users can be recommended new interests based on the interests of their current friends. Similarly, users can be recommended new friends based on their current interests and friends. To address such social network problems effectively, it is essential to organize user interests into an ontology, or more precisely a hierarchical ontology. An ontology is an explicit formal specification of the terms and relations among terms in a domain (Gruber, 1993). It can be achieved by a systematic grouping of domain concepts based on their definitions, in machine interpretable form. In the social networks domain, such systematic organization of user interests can provide good insights into common problems, such as interest and friend recommendations, to name a few.

Constructing an ontology over the interests specified by the users of a social network has attracted attention among some researchers previously. Bahirwani et al., (2008) have constructed an interest ontology by fetching definitions of interests from three online sources, namely WordNet-Online, Internet Movie Database (IMDB) and Amazon Associates Web Services (AWS). Each definition of an interest can be seen as an instance. Similarity between instances is computed as the dot product of the vectors representing the instances. Instances are grouped into a hierarchical ontology using a hierarchical agglomerative clustering approach.

Although the ontology constructed by Bahirwani et al., (2008) has proven helpful for improving the predictions of friendship relationships, the use of WordNet-Online, IMDB and AWS for a semantic understanding of user interests is cumbersome and may not always give complete and accurate definitions of interests. For interests belonging to topics, such as *Sports Persona*, that do not fit in the categories of *movies, books and words*, one would need to obtain definitions from a different knowledge base. For example, for the user interest *Pete Sampras*, definitions can be found neither on WordNet-Online nor on IMDB and AWS. Thus, the use of

such discrete, distributed knowledge bases is inconvenient and undesirable, and cannot represent interests from diverse topics. Furthermore, once the definitions corresponding to interests are obtained, the clustering approach in (Bahirwani et al., 2008) groups the definitions in a binary tree hierarchy, as opposed to an n-ary tree. The resulting clusters do not capture the concept information for related interests. If two interests, such as, *Laptops* and *Notebooks* are clustered, then the title of the new cluster becomes *Laptops & Notebooks*, when instead it would be desirable to derive the title *Portable Computers* to indicate the semantics of its children in a more meaningful way.

A different ontology engineering approach is required to address the above mentioned issues. A more effective ontology that captures better semantics for a wider variety of domains is needed. Such ontology could be used to construct interest features which can further be used in combination with graph based features to address prediction problems in social networks. Our work explores different ontology engineering approaches and more comprehensive knowledge bases (in particular, Wikipedia and Directory Mozilla) to address the limitations mentioned above.

High level overview

The data set used in this work consists of 1000 users of the *LiveJournal* online service. There are approximately 22,000 interests that these users have collectively specified. These interests belong to a wide variety of domains, including *Movies*, *Books*, *Sports*, *Social and Current Issues*, among others. Our ultimate goal is to organize user interests into a hierarchy, which does not present the limitations of the ontology produced by the approach in (Bahirwani et al., 2008). To achieve this goal, we first eliminate the dependency of the ontology engineering process on multiple data sources and instead explore the use of Wikipedia as a single data source. Wikipedia (Wiki) is a freely available on-line encyclopedia developed by a large community of users. Because of its popularity and generic nature, Wikipedia can serve as a centralized, yet comprehensive knowledge base for retrieving definitions (i.e., Wikipedia articles) of user interests across diverse domains. In our first approach, we obtain definitions of interests from Wikipedia and use the technique of latent semantic analysis (LSA) to measure the similarity between interests. LSA is a theory and method for extracting and representing the contextual-usage meaning of words by statistical computations applied to a large corpus of text

(Landauer and Dumais, 1997). The approach is to take advantage of implicit higher-order structure in the association of terms with documents ("semantic structure") in order to improve the detection of relevant documents (Deerwester, 1990). The particular technique used is singular-value decomposition, in which we take a large matrix of term-document association data and construct a "semantic" space wherein terms and documents that are closely associated are placed near one another. Singular-value decomposition allows the arrangement of the space to reflect the major associative patterns in the data, and ignore the smaller, less important influences (Deerwester, 1990). We believe that clustering wiki documents representing interests using LSA, works much better than determining similarity between the documents by just taking a dot product of their vector representations, because LSA helps mine "hidden" similarities within document structures.

While the Wikipedia/LSA approach produces a more sensible ontology than the one produced by the approach in (Bahirwani et al., 2008), this ontology is still a binary tree and consists of internal clusters labeled based on child information. To address these limitations, our second and third approaches explore the reuse of knowledge from existing hierarchies such as the Wikipedia Category Graph (WCG) and Directory Mozilla (DMoz), respectively, to group interests. In Wikipedia, every article belongs to some category. The articles form a network of semantically related terms, while categories are organized in a taxonomy-like structure, called WCG (Zesch and Gurevynch, 2007). Because WCG shares many properties with other lexical semantic networks such as WordNet-online, its analysis can help extract useful relationships in constructing the interest ontology. The resulting hierarchy is a n-ary tree and the internal nodes have meaningful names. However, it turns out that WCG does not have good coverage for the interests in our data set, producing a highly incomplete ontology. As an alternative to WGC, we also explore the use of DMoz, the largest, most comprehensive human-edited directory of the web (Varma, 2002). With more than four million categories arranged across fifteen levels, DMoz proves to be the best source of knowledge with respect to the problem of engineering a user interests ontology.

We will present the three approaches considered in our work in more details in Chapters 3, 4 and 5, respectively. For implementation of the three approaches, Wikipedia dump from October 8, 2008 was extracted and the XML documents were imported into MySQL database tables.

CHAPTER 2 - Related work

Exploiting the comprehensibility and coverage of Wikipedia has been the focus of widespread research. Amongst various other works, Gabrilovich and Markovitch (2006), Janik and Kochut (2007) use Wikipedia to assign category labels to documents. Syed et al. (2008) go a step further in attempting to predict categories common to a set of documents using Wikipedia whereas Strube and Ponzetto (2006), Gabrilovich and Markovitch (2007), Milne (2007) use Wikipedia to compute semantic relatedness between documents. Latent Semantic Analysis technique for text categorization has also been tried before; Lee et al. (2006) use Latent Semantic Indexing for multi-language text categorization. The method we use in Approach 1 is novel in a way that it tries to make best use of the advantages that Wikipedia, as a data source, and Latent Semantic Analysis, as a text categorization technique, have to offer. Though both Wikipedia and Latent Semantic Analysis have been used separately for text categorization and computing of document similarity, an approach that combines the benefits of both to engineer ontology, to the best of our knowledge, has not yet been explored.

The usefulness of DMoz for classification problems have been previously demonstrated as well. Grobelnik and Mladeni (2005) have shown that the use of large topic ontologies such as DMoz can help in classifying Web documents. Our WCG and DMoz based approaches rely on a principle similar to the one exploited in (Grobelnik and Mladeni, 2005): having an enriched input compensates for a simple approach to document classification. Grobelnik and Mladeni (2005) provide an enriched input by adding information in the form of web page content, obtained from the link structure of the web. In our case, a complete “category link information” for interests can be seen as enriched knowledge that can be reused in the process of ontology engineering. Based on our experiments, WCG proves not to have good coverage for user interests, and therefore it is inadequate to use as enriched input. However, DMoz can be successfully used for this task. Thus, we mine the enriched knowledge by searching for interest instances in the listing of external pages associated with DMoz dump categories.

The contribution of our work lies in the fact that we construct an accurate user interest n-ary hierarchy by efficiently and effectively reusing a single comprehensive knowledge base that covers a wide variety of interest topics. In the process, we derive useful observations about the

effectiveness of the LSA technique to compute the similarity between interests. Also, we compare the usefulness of Wikipedia versus DMoz as data sources in the ontology engineering process. While other authors such as Gabrilovich and Markovitch (2007) have found that Wikipedia is better than DMoz for certain tasks (e.g., computing semantic relatedness), we find that in the case of social network user interests, DMoz serves better than Wikipedia in the ontology engineering process.

CHAPTER 3 - Ontology engineering using Wikipedia and LSA

Approach

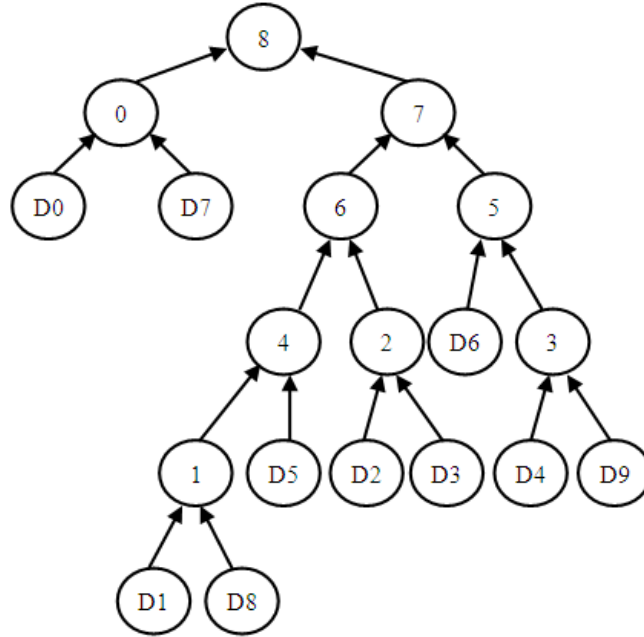
In this approach, we obtain definitions of interests from Wikipedia and compare the definitions using the standard LSA technique (Deerwester et al., 1990). For illustration purposes, we will use a small interests set consisting of 10 interests, picked up at random, from diverse fields such as *Movies*, *Sports* and *Current affairs*. The interests and the document index associated with each of them are shown in Table 1. The hierarchy constructed using the approach in (Bahirwani et al., 2008) is shown in Figure 3.1. As can be seen, this approach does not perform very well for the set of interests considered. We will show that Wikipedia/LSA approach produces a more sensible ontology compared to this baseline.

Table 3.1 Interests data used to illustrate the ontology engineering approaches explored

Interest Chosen	Document ID Assigned
<i>9-11</i>	<i>D0</i>
<i>Age of Empires</i>	<i>D1</i>
<i>Anti-terrorism</i>	<i>D2</i>
<i>Computer Gaming</i>	<i>D3</i>
<i>Forrest Gump</i>	<i>D4</i>
<i>Oscar Wilde</i>	<i>D5</i>
<i>Scuba Diving</i>	<i>D6</i>
<i>Tom Hanks</i>	<i>D7</i>
<i>Video Games</i>	<i>D8</i>
<i>Water Sports</i>	<i>D9</i>

To start with, for every user interest in our data, the relevant Wiki document is fetched (and regarded as the definition of the corresponding interest). The fetched documents are cleaned, as follows: stop words are removed, text in the documents is tokenized and tokens are

Figure 3.1 Hierarchy over the user interests in Table 1 when WordNet/IMDB/AWS are used as knowledge bases.



stemmed. If t is the number of distinct terms in all tokenized documents and d is the number of documents, then a $t \times d$ term-document matrix $A_{t \times d}$ is constructed.

A Snapshot of the constructed term document matrix is shown in Figure 3.2

The term-document matrix is then decomposed using the singular value decomposition (SVD) technique into $A = USV'$, where

- A is the original $t \times d$ term-document matrix whose rank is n ;
- U is a $t \times n$ left singular matrix (i.e., $UU' = I$);
- V is a $n \times d$ right singular matrix (i.e., $VV' = I$);
- S is an $n \times n$ diagonal matrix with singular values arranged in descending order;
- I is the identity matrix, V' is the transpose of V , and U' is the transpose of U .

The dimensionality of A is reduced by reducing U , V and S to only their first k columns, $k < n$, and taking their product. Thus $A_{k \times k} = U_{t \times k} \cdot S_{k \times k} \cdot V'_{k \times d}$ where $A_{k \times k}$, $U_{t \times k}$, $S_{k \times k}$ and $V_{d \times k}$ are all matrices with reduced dimensions (Deerwester et al., 1990). A snapshot of the SVD-reduced matrix for the ten sample interests is shown in Figure 3.3.

Figure 3.2 Term-Document matrix snapshot for the sample data. Number of terms = 797 (only 14 shown), number of documents = 10

	D0	D1	D2	D3	D4	D5	D6	D7	D8	D9
9-11	2.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
abbreviated	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
ability	0.0000	0.0000	0.0000	1.0000	0.0000	0.0000	0.0000	2.0000	0.0000	0.0000
able	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000	0.0000
abraham	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000	0.0000
academy	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000	0.0000	2.0000	0.0000	0.0000
account	0.0000	0.0000	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
accounting	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000
accumulated	0.0000	0.0000	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
achieve	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000
achievement	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000	0.0000
across	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000
act	0.0000	0.0000	0.0000	0.0000	0.0000	1.0000	0.0000	0.0000	0.0000	0.0000
acting	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	2.0000	0.0000	0.0000

Figure 3.3 Matrix snapshot obtained after SVD and dimensionality reduction

	D0	D1	D2	D3	D4	D5	D6	D7	D8	D9
9-11	0.0001	0.0002	0.0001	0.0002	0.0000	0.0001	0.0000	0.0000	0.0002	0.0000
abbreviated	0.0007	0.0016	0.0005	0.0017	0.0000	0.0001	0.0000	0.0000	0.0020	0.0000
ability	0.0036	0.0017	0.0038	0.0016	0.0047	0.0067	0.0000	0.0067	0.0013	0.0002
able	0.0016	0.0003	0.0017	0.0002	0.0024	0.0033	0.0000	0.0033	0.0000	0.0001
abraham	0.0016	0.0003	0.0017	0.0002	0.0024	0.0033	0.0000	0.0033	0.0000	0.0001
academy	0.0032	0.0006	0.0035	0.0004	0.0047	0.0067	0.0000	0.0067	0.0001	0.0002
account	0.0004	0.0011	0.0003	0.0012	0.0000	0.0000	0.0000	0.0000	0.0014	0.0000
accounting	0.0000	0.0000	0.0000	0.0000	0.0000	0.0001	0.0000	0.0001	0.0000	0.0000
accumulated	0.0004	0.0011	0.0003	0.0012	0.0000	0.0000	0.0000	0.0000	0.0014	0.0000
achieve	0.0026	0.0067	0.0019	0.0070	0.0001	0.0002	0.0000	0.0000	0.0082	0.0001
achievement	0.0016	0.0003	0.0017	0.0002	0.0024	0.0033	0.0000	0.0033	0.0000	0.0001
across	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
act	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
acting	0.0032	0.0006	0.0034	0.0004	0.0047	0.0066	0.0000	0.0067	0.0001	0.0002

The advantage of such reduction is that interest documents that have many words in common get grouped closer to each other. Hidden relationships between interest documents are discovered, while weak undesirable relationships get eliminated. For the input data consisting of 22,000 interests, we experimented with various values of k ranging from 100 to 500. Intuitively, the best results, in terms of document similarity, were obtained for k = 150.

After applying the LSA technique, each document is represented as a vector of weights (Rosario, 2000). Similarity between a pair of documents is computed as the cosine of the angle between

the corresponding document vectors. For two documents A and B represented as vectors, the cosine similarity between them is computed as $\cos(\theta) = (A \cdot B / |A| |B|)$. Intuitively, interest documents that are very similar tend to have a very small angle between them and, therefore, a high similarity, as $\cos(0) = 1$. On the other hand, interest documents that are dissimilar have a larger angle between them indicating weaker similarity. Cosine similarity between each pair of documents is computed. The resulting matrix for the sample data, showing the similarity of each document with each other on a scale of 0-1 is shown in Figure 3.4.

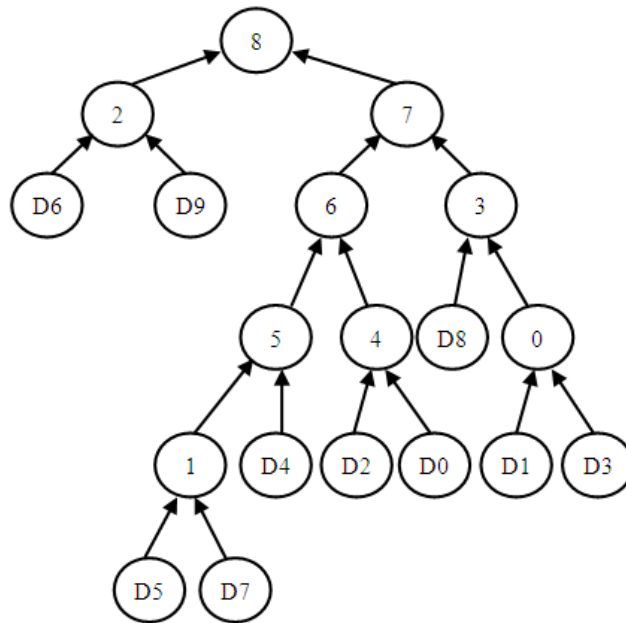
Figure 3.4 Similarity matrix between documents in sample data

	D0	D1	D2	D3	D4	D5	D6	D7	D8	D9
D0	1.0000	0.7741	0.9817	0.7583	0.7170	0.6973	0.2537	0.6672	0.7267	0.2930
D1	0.7741	1.0000	0.6428	0.9997	0.1146	0.1234	0.0766	0.0842	0.9970	0.0996
D2	0.9817	0.6428	1.0000	0.6238	0.8344	0.7928	0.3282	0.7673	0.5851	0.3683
D3	0.7583	0.9997	0.6238	1.0000	0.0901	0.1007	0.0669	0.0615	0.9985	0.0891
D4	0.7170	0.1146	0.8344	0.0901	1.0000	0.9460	0.3521	0.9414	0.0424	0.3879
D5	0.6973	0.1234	0.7928	0.1007	0.9460	1.0000	0.0310	0.9991	0.0640	0.0701
D6	0.2537	0.0766	0.3282	0.0669	0.3521	0.0310	1.0000	0.0160	0.0257	0.9990
D7	0.6672	0.0842	0.7673	0.0615	0.9414	0.9991	0.0160	1.0000	0.0253	0.0544
D8	0.7267	0.9970	0.5851	0.9985	0.0424	0.0640	0.0257	0.0253	1.0000	0.0466
D9	0.2930	0.0996	0.3683	0.0891	0.3879	0.0701	0.9990	0.0544	0.0466	1.0000

Once the similarity matrix is constructed, the instances (interest documents) are clustered using a hierarchical agglomerative clustering algorithm. During each iteration of the algorithm, we group the two documents having the highest similarity and recompute the similarity matrix. The similarity between two clusters C_i and C_j is defined as the average similarity between all pairs of documents D_i and D_j , where D_i belongs to C_i and D_j belongs to C_j .

Figure 3.5 shows the ontology that is constructed using this approach. In the figure, nodes 0 to 8 also indicate the order in which documents from D_0 to D_9 are clustered, with 0 indicating the first clustering and 8 indicating the last clustering. Thus, documents D_1 (corresponding to the interest *age of empires*) and D_3 (corresponding to the interest *computer gaming*) are clustered first (to form a new node 0), as they have the highest similarity. Next, documents D_5 (*Tom Hanks*) and D_7 (*Oscar Wilde*) are clustered (to form a new node 1), as they have the next highest similarity measure, and so on.

Figure 3.5 Ontology engineered for data in Table 1 using Wikipedia and LSA based approach



At the implementation level, the ontology may be maintained in a MySQL database table with columns *NodeId*, *Parent*, *Children* and *TreeId*. All nodes, corresponding to records in the table, can initially be singletons. In the table, they would have no parent and the *TreeId* set to their *NodeId*. When two nodes are clustered, we may actually cluster the nodes corresponding to their *TreeId* to form a new node. The new node would have the clustered nodes as *children*, no *parent*, and its *nodeID* as *TreeId*. Also the *TreeId* of all descendants of the newly constructed node would be updated to the *TreeId* of the newly constructed node.

Discussion

The Wikipedia/LSA approach produces the hierarchy shown in Figure 3.5, which is more accurate than the baseline shown in Figure 3.1. The use of Wikipedia to obtain interest definitions results in good coverage for users interests, without the need for multiple sources such as WordNet-Online, IMDB, etc., as explained in Chapter 1. Furthermore, the use of LSA helps in reducing much of noise in the data and unhiding latent relationships between documents. Such a technique to calculate the similarity between the documents is superior to more conventional ways of computing the similarity such as dot product or cosine similarity of the original tf-idf vectors representing the documents.

Although better than the approach in (Bahirwani et al., 2008), the Wikipedia and LSA based approach has several shortcomings. First, the ontology engineered is still a binary tree. Second, new nodes that are constructed as a result of clustering of interest instances do not have explicit semantics associated with them. For example, if interests *Tom Hanks* and *Forrest Gump* are grouped together, then it would be better if the resulting node had the title *Hollywood* rather than just *Tom Hanks & Forrest Gump*. Also, in our sample ontology shown in Figure 3.5, the interest *Tom Hanks* is clustered with *Oscar Wilde* when in reality, *Tom Hanks* should perhaps be grouped much closer to *Forrest Gump*. The reason for such an anomaly is that LSA groups documents that have many words in common. The word *Oscar* may be occurring frequently in the documents for *Tom Hanks* and *Oscar Wilde*, hence the grouping. While the ontology produced using the Wikipedia/LSA approach presents some shortcomings it is still more accurate than the ontology produced using multiple knowledge bases and the traditional cosine similarity on the original tf-idf vectors, thus making the Wikipedia/LSA approach appealing. However, because the ontology is a binary tree and also cannot capture the concepts and associated semantics of the newly formed clusters, we explore two more approaches to ontology engineering.

CHAPTER 4 - Ontology engineering using Wikipedia Category Graph

Approach

The Wikipedia/LSA approach described in the previous chapter does not construct an n-ary hierarchy and does not associate concept information with clusters. Furthermore, the Wikipedia/LSA based approach is computationally expensive in terms of time and space required to perform the singular value decomposition procedure. Our working hypothesis is that a simpler approach might produce a better ontology, if all existing information in the input is better utilized and if the algorithm is provided with an enriched input. Motivated by this hypothesis, in the WCG based approach, we exploit the category information associated with every Wikipedia article. Just as in the Wikipedia/LSA approach, we obtain definitions of interests from Wikipedia. However, instead of comparing the documents corresponding to interests, we compare categories corresponding to interest documents. Now, interests belonging to the same categories are grouped together. This is done by mining the WCG for relationships between categories and grouping the categories themselves within each other. Again, just as in the Wikipedia/LSA based approach, in this approach too we maintain the ontology in a MySQL database table with columns *NodeId*, *Title*, *CategoryID*, *Parents* and *Children*. The table is initially empty.

The detailed approach, which follows a 4-step algorithm, is as follows:

Step1: For every interest from the input file, the relevant Wiki document is fetched and categories corresponding to this document are extracted. An entry corresponding to the interest title is created in the ontology table. Entries are also created for each of the categories that the interest belongs to. If an entry for a category already exists, we only append the current interest title to the list of children of this category. At the end of this step, there are a number of small “fragments” in the Ontology table that need to be combined. Thus, for example, the interest *Tom Hanks* belongs to categories *Film Actors*, *1956 Births*, *Best Actor Oscars* etc. A node is created

for *Tom Hanks* with Parents *Film Actors*, *1956 Births*, *Best Actor Oscars* etc. Similarly, nodes for the 3 categories are also created with *Tom Hanks* being the child node of each.

Step 2: At the end of step 1, all categories (entries with “non NULL” *children*) in the ontology table are independent of each other. In step 2, the Wikipedia dump is parsed and a category hierarchy is constructed. This category hierarchy provides specifics about parent-child relationships between the categories. Thus, the category *Best Actor Oscars* in fact is a sub category of *Film Actors*. Therefore, the former is appended in the *Children* of the later whereas the later is made the *parent* of the former. Such relationships between categories are extracted in this step.

Step 3: Now the interest *Tom Hanks* belongs to categories *Best Actor Oscars* and *Film Actors* whereas *Best Actor Oscars* is again a child of *Film Actors*. This scenario is undesirable because it introduces cycles. Ideally, if node *Tom Hanks* has *Best Actor Oscars* as parent which in turn is a child of *Film Actors* then *Tom Hanks* cannot be a child of *Best Actors*. Furthermore, the node *Tom Hanks* also has *1956 Births* node as parent. Thus, at this stage, the structure in the Ontology table has nodes that have multiple parents. Such nodes should be duplicated as in there should be a node *Tom Hanks 1* that is part of the *Film Actors* → *Best Actor Oscars* tree, whereas another node *Tom Hanks 2* should be a child of *1956 Births*. To address these problems, we traverse the graph in the ontology table in a breadth first manner and perform the required corrections and duplications.

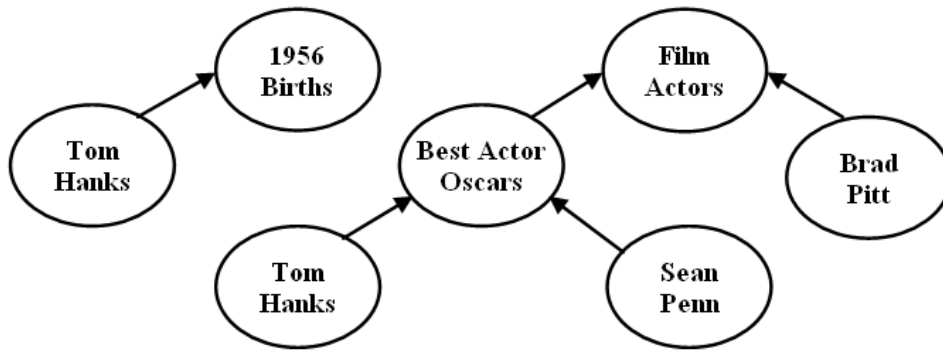
Step 4: All category nodes in the ontology table with no parent are clustered into a single node that becomes the root of the ontology. The number of nodes to be clustered in this step should be as small as possible

Figure 4.1 shows the fragments that are created as a result of Steps 1 to 3, for the example mentioned in the steps

Discussion

The WCG based approach serves the purpose of assigning semantics to newly formed clusters in the hierarchy. However, as can be seen with the interest *Tom Hanks* in the above example, in Wikipedia, each article can belong to an arbitrary number of categories. It is not possible to rank and filter the categories according to their importance, so that we can choose only the top k categories (e.g., k = 5). Thus, for the interest *Tom Hanks* the category *1956 Births*

Figure 4.1 Fragment of the ontology engineered using the WCG-based approach.



is as important as the category *Best Actor Oscars*. The large number of categories results in large scale duplication of interest instances in the resulting hierarchy. Furthermore, WCG contains cycles and disconnected categories (Zesch and Gurevynch, 2007). Breaking the cycles requires further duplication of nodes.

Another drawback of this approach is presented in what follows. The documents corresponding to the 10 interests shown in Table 1 have 45 distinct categories. After grouping related categories, we are left with 28 categories that have no parent. The problem of having a large number of nodes with no parent exists because it is not possible to extract the complete category link of a Wikipedia article. For example for an interest *Laptop*, with the current setting, we only get to know that it belongs to the category *Personal Computers*, whereas its complete category link is: *Technology and applied sciences* → *Computing* → *Classes of computers* → *Microcomputers* → *Personal computers* → *Laptops*. To the best of our knowledge, it is not possible to extract such “category link” information for a Wikipedia article using a simple approach.

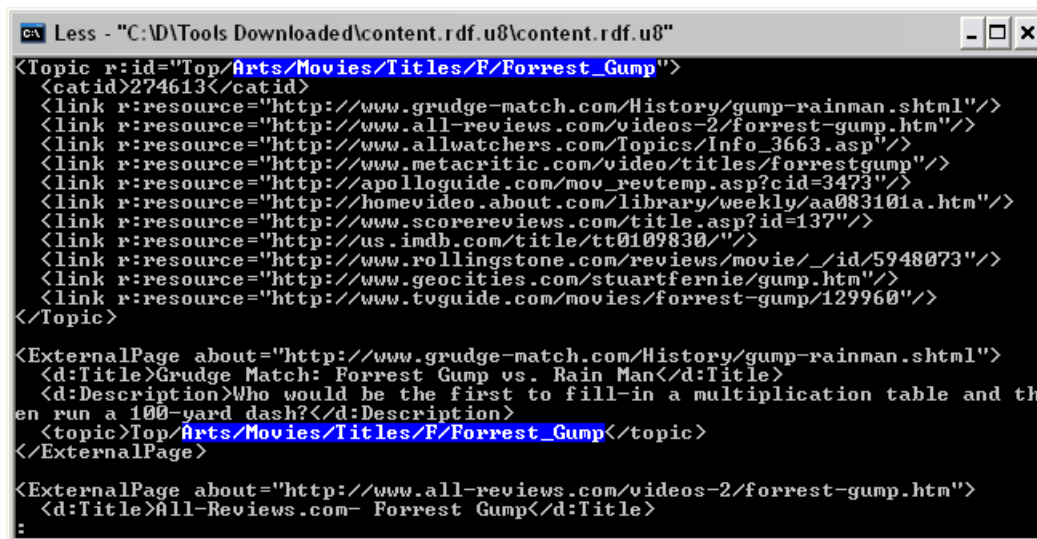
Due to the above mentioned shortcomings, this approach fails to provide the desired results. However, it motivates our next approach: if we can extract the “complete category link” to which interests belong and if we can rank the categories to which the interests belong based on their importance, the interests can be classified effectively and grouped into a hierarchy.

CHAPTER 5 - Ontology engineering using Directory Mozilla

Approach

The problem of retrieving the “complete category link” for an interest is resolved through the use of DMOZ category hierarchy. The approach is very simple: The interest is searched for in DMOZ. Every topic (category link) in the DMOZ has a brief listing and description of external pages associated with that topic. Categories, in which one or more of the external page descriptions contain the concerned interest word, can be selected and ranked in decreasing order of the matches. Thus for example one of the category results for interest *9-11*, as can be seen in Figure 5.1 is *Top* → *Society* → *Issues* → *Terrorism* → *Incidents*.

Figure 5.1 Snapshot of DMOZ RDF dump (downloaded from <http://rdf.dmoz.org/>)



```
Less - "C:\D\Tools Downloaded\content.rdf.u8\content.rdf.u8"
<Topic r:id="Top/Arts/Movies/Titles/F/Forrest_Gump">
  <catid>274613</catid>
  <link r:resource="http://www.grudge-match.com/History/gump-rainman.shtml"/>
  <link r:resource="http://www.all-reviews.com/videos-2/forrest-gump.htm"/>
  <link r:resource="http://www.allwatchers.com/Topics/Info_3663.asp"/>
  <link r:resource="http://www.metacritic.com/video/titles/forrestgump"/>
  <link r:resource="http://apolloguide.com/mov_revtemp.asp?cid=3473"/>
  <link r:resource="http://homevideo.about.com/library/weekly/aa083101a.htm"/>
  <link r:resource="http://www.scorereviews.com/title.asp?id=137"/>
  <link r:resource="http://us.imdb.com/title/tt0109830"/>
  <link r:resource="http://www.rollingstone.com/reviews/movie/_/id/5948073"/>
  <link r:resource="http://www.geocities.com/stuartfernle/gump.htm"/>
  <link r:resource="http://www.tvguide.com/movies/forrest-gump/129960"/>
</Topic>
<ExternalPage about="http://www.grudge-match.com/History/gump-rainman.shtml">
  <d:Title>Grudge Match: Forrest Gump vs. Rain Man</d:Title>
  <d:Description>Who would be the first to fill-in a multiplication table and then run a 100-yard dash?</d:Description>
  <topic>Top/Arts/Movies/Titles/F/Forrest_Gump</topic>
</ExternalPage>
<ExternalPage about="http://www.all-reviews.com/videos-2/forrest-gump.htm">
  <d:Title>All-Reviews.com- Forrest Gump</d:Title>
:
```

Like in the first two approaches, we use a MySQL database table to maintain the ontology. The columns of the table are *Category*, *Parent*, *Children* and *TreeID*. The table is initially empty. Also, for demonstration purposes, we use the html dump of the interests data downloaded online. Thus, for every interest in our dataset, we download the html document corresponding to that interest.

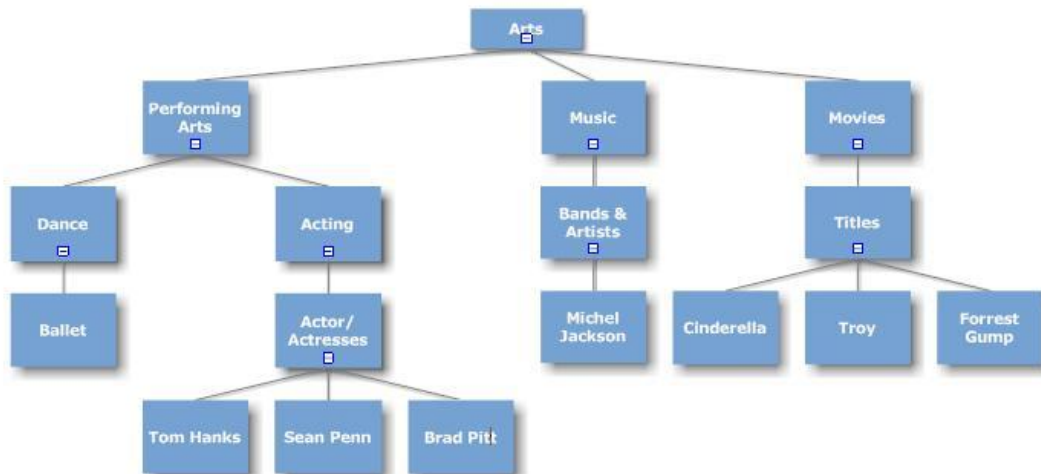
For every interest and its category links, entries are created in the table. Thus, for the interest *9-11*, entry is created for *9-11* with no children and parent *September 11, 2001*. Entry is created for *September 11, 2001* with children *9-11* and parent *Incidents* and so on. If entry for a category is already present in the table, we do not create a duplicate entry. Instead we just update its children if necessary. This procedure is repeated for every interest in the input data. Figure 5.2 shows a snapshot of the table.

Figure 5.2 Snapshot of the ontology table for the DMoz based approach

Title	CategoryID	Parents	Children
Anti-terrorism	197515	2588124 885355 701590	NULL
Category:Cleanup from September 2005	2588124	NULL	197515
Category:NPOV disputes	885355	NULL	197515
Category:Terrorism	701590	NULL	197515

It should be noted that we parse the complete category link and every term in the link becomes a node in the ontology. The interest is made a child of the lowest node in the hierarchy. Figure 5.3 shows a fragment of the ontology constructed for a set of interests (*Tom Hanks, Brad Pitt, Sean Penn, Ballet, Michael Jackson, Cinderella, Troy, Forrest Gump*) under the concept *Arts*.

Figure 5.3 Fragment of the ontology engineered using DMoz



As seen in the figure, all interests are accurately grouped under the respective categories to which they belong. The hierarchy constructed from all 22,000 interests in our data has 13 levels,

68083 leaf nodes (a consequence of multiple meanings for the interest words) and 52106 internal nodes. The maximum number of children for a node is 1912 (for the node *bands and artists*, sub-category of *music*). We should note that data mining techniques could be used to perform interest-word sense disambiguation, for example, by exploiting other interests of a user and the interests of his or her friends. The exact number of levels and the number of nodes at each level are detailed in the table 5.1.

Table 5.1 Ontology levels and number of nodes at each level

<i>Level Nr</i>	<i>Nr Of Nodes</i>	<i>Level Nr</i>	<i>Nr Of Nodes</i>
1	1	7	58261
2	16	8	64086
3	506	9	67694
4	6453	10	67970
5	27238	11	68058
6	45162	12	68081

Discussion

The DMoz based approach is a simple, efficient and effective approach that takes advantage of the existing information in the DMoz category hierarchy. Unlike WCG, the DMoz hierarchy has a tree structure. Thus, even if an interest can belong to multiple categories, no category can be a sub-category of more than one category. As a result, there is less duplication of nodes in the tree and the ontology is very “crisp”. Moreover, the DMoz hierarchy is very comprehensive. Topics covered range from *Arts, Sciences, and Computers to Movies, Business, and Health*, etc. Such coverage of topics is ideal for classification of a wide variety of user interests in social networks. Furthermore, the DMoz taxonomy is human edited and reviewed which makes it accurate and reliable. Thus, this approach constructs a very simple yet effective grouping of user interests and addresses all the issues discussed in chapter 1, presenting multiple advantages. In the next chapter we use this ontology to construct interest based features and use those features on classifiers to test the usefulness of this ontology in predicting friendship relations.

CHAPTER 6 - Ontology evaluation

The ultimate goal of this work is to study the effect of the interest ontology on the performance of learning algorithms at the tasks of predicting friendship links in the LiveJournal social network. The graph network used in the study consists of 1,000 nodes (users) and 7,500 links (declared friendships). Collectively, the users in the network specify about 22,000 interests which correspond to 68,083 concepts. These concepts are organized in an ontology using the DMOZ based approach as described in the previous chapter. Global cuts through these ontologies result in concepts specified at higher or lower levels of abstraction. The prediction task addressed can be stated as follows: given a pair of users $\langle A, B \rangle$, predict the existence (true or false) of a directed link from user A to user B. To do that, we need to represent pair of users $\langle A, B \rangle$ as a feature vector. Following are the different types of features that can be used to represent the feature vector.

Interest based nominal features without ontology

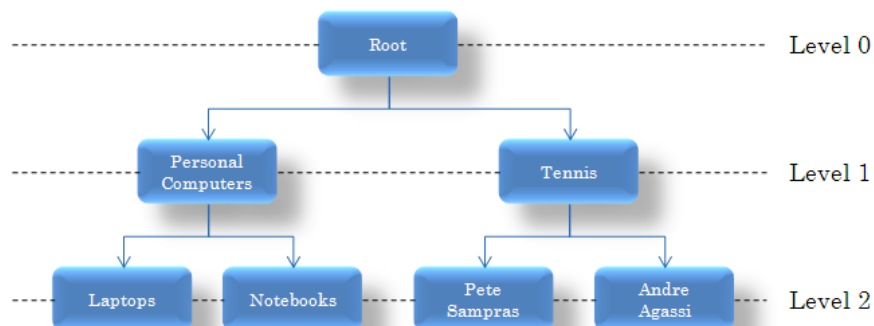
For a pair of users $\langle A, B \rangle$, a feature vector can be represented by a sequence of interests of user A followed by the interests of user B. These are interest based nominal features without ontology. Thus, for example, if user A has interests $\{Laptops, Sampras\}$ and user B has interests $\{Notebooks, Agassi\}$, then for the user pair $\langle A, B \rangle$, the interest based nominal features without ontology would be $\{Laptops, Sampras, Notebooks, Agassi\}$.

Interest based nominal features with ontology

For a pair of users $\langle A, B \rangle$, a feature vector can be constructed by a sequence of interests of user A followed by the interests of user B, with interests of both users represented at some level of abstraction in the ontology. These are interest based features with ontology.

Consider Figure 6.1. If user A has interests $\{Laptops, Sampras\}$ and user B has interests $\{Notebooks, Agassi\}$, then interest based nominal features with ontology at level 1 would be $\{Personal Computers, Tennis\}$

Figure 6.1 Example to illustrate construction of interest based nominal features with ontology



Similarly, the interest based nominal features with ontology at level 0 would be $\{Root\}$.

Interest based numerical features

The intuition behind the interest-based numerical features we construct is that if two users have many interests in common, then it is possible that they are friends, regardless of exactly what those interests are. Furthermore, if the users don't have many common interests, but share a very rare interest, they might also be friends. We derive eight numerical features that capture this intuition by measuring the interestingness of a the set of common interests that two users A and B share (Aljandal et al. 2008). This is achieved by regarding each interest as an itemset whose elements are the users having that interest. From each interest itemset i , association rules of the form $A(i) \rightarrow B(i)$ (meaning, if A has interest i , then B has interest i) are derived (Caragea et al., 2009). The resulting association rules are used to define eight objective measures of rule interestingness, from which eight interest-based numerical features are constructed.

Suppose that $numIntA$ denotes the number of interests of user A, $numIntB$ denotes the number of interests of user B and $numIntAB$ denotes the number of mutual interests of users A and B. For a user pair $\langle A, B \rangle$, we define the following probabilities:

- *Probability that A has an interest: $numIntA / TotalNumberOfInterests$*
- *Probability that B has an interest: $numIntB / TotalNumberOfInterests$*
- *Probability that A and B have a common interest: $numIntAB / TotalNumberOfInterests$*

Using these probabilities and the Bayes' Theorem, the following eight objective measures of rule interestingness can be derived:

1. $Support(A \rightarrow B) = P(AB)$
2. $Confidence(A \rightarrow B) = P(B | A)$
3. $Confidence(B \rightarrow A) = P(A | B)$
4. $Lift(A \rightarrow B) = P(B | A) / P(B)$
5. $Conviction(A \rightarrow B) = P(A) * (1 - P(B)) / P(A \sim B)$
6. $Match(A \rightarrow B) = [P(AB) - P(A) * P(B)] / [P(A) * (1 - P(A))]$
7. $Accuracy(A \rightarrow B) = P(AB) + P(\sim A \sim B)$
8. $Leverage(A \rightarrow B) = P(B/A) - P(A)P(B)$

As pointed out in (Aljandal et al. 2008), these measures do not take into account the relative size of the itemset to which each candidate pair $A \rightarrow B$ belongs. For example, some interests that have low membership (i.e., small itemset size), such as *DNA replication* (an example of rare interest), often imply a more significant association between users listing them, than common interests, such as *music* or *games* that are shared by many users. To address this limitation, Aljandal et al. (2008) have derived a normalization factor that takes into account the popularity of particular interests that two users share, with the most popular interests (held by a significant proportion of users) being slightly less revealing than rarer interests. Experimental results (Aljandal et al. 2008) show that link prediction algorithms presented with normalized interestingness measures (as numerical features) give better results than those presented with the features obtained from non-normalized interestingness measures. Hence, we use the normalized versions of the eight association rule measures mentioned above to derive eight interest-based numerical features in our experiments.

Graph based features

Similar to (Hsu et al. 2006), for each user-pair $\langle A, B \rangle$ in the social network graph, we construct and use the following graph-based features in our study:

1. In-degree of A: The number of incoming edges to the node corresponding to user A (represents the popularity or importance of A).
2. In-degree of B: Similar to in-degree of A, this is the number of incoming edges to the node corresponding to B.
3. Out-degree of A: The number of friends of user A (except for user B). This number is computed by counting the number of outgoing edges (except for $A \rightarrow B$) from the node corresponding to user A in the social network graph.
4. Out-degree of B: Similar to out-degree of user A, this represents the number of friends of B except for A.
5. Mutual friends of A and B - 4 types considered:
 - a. Number of mutual friends C, s.t. $A \rightarrow C$ and $C \rightarrow B$.
 - b. Number of mutual friends C, s.t. $B \rightarrow C$ and $C \rightarrow A$.
 - c. Number of mutual friends C, s.t. $A \rightarrow C$ and $B \rightarrow C$.
 - d. Number of mutual friends C, s.t. $C \rightarrow A$ and $C \rightarrow B$.
6. Backward distance from B to A: The minimum distance from the node corresponding to user B to the node corresponding to user A in the graph.

Previous work by (Hsu et al 2006) and (Bahirwani et al 2008) have shown that use of Interest based nominal features with or without ontology does not improve accuracy of classifiers in predicting friendship links. Therefore, in our experiments, we make use of only graph based features and interest based numerical features.

CHAPTER 7 - Experiments

Setup

We evaluate the ability of the interest ontology to improve the performance of traditional learning algorithms such as Support Vector Machines (SVM) and Random Forests (RF) classifiers (Mitchell 1997) at the task of predicting friendship links. To do this, graph based features are used by themselves and in combination with interest-based numerical features with ontology. The expectation is that results are much more accurate, when graph based features are used in combination with interest based features computed in the presence of the ontology, than the results achieved with the use of only graph based features.

As mentioned before, our data set consists of 1,000 users and approximately 7,500 declared friendship links (out of 1000 by 1000 possible links in an undirected graph with 1,000 nodes). We make the assumption (obviously, violated in practice) that the graph network is complete, i.e. all declared friendships are positive examples and all non declared friendships are negative examples (Caragea et al., 2009). That means, our data set is highly skewed towards the negative class, the ratio between the positive and negative classes being less than 1 : 99. We randomly divide the original data set into three subsets (sample without replacement):

- A training set consisting of 50% positive links and 50% negative links (approximately 3,700 friendships and 3,700 non-friendship links). This data set is used to train the classifiers considered.
- A validation set consisting of data that has the original distribution (approximately, 1850 friendship links and 240,000 non-friendship links). This data set is used to find the best level of abstraction for an ontology.
- A test set also consisting of data having the original distribution (approximately, 1850 friendship links and 240,000 non-friendship links). We use this data set to evaluate the true performance of the classifiers, given the best level of abstraction for the ontology.

From each data set we remove the links that go across data sets to ensure that the three sets are independent. Interest based features are constructed using a particular level in the DMOz ontology. For the training set, graph-based features are constructed using all the available

friendship links. However, for the validation and test sets, we want to predict the friendship links. To be able to construct graph-based features for these sets, we follow the approach in (Taskar et al. 2003). More precisely, we assume that a certain percent of the links are known in the validation/test graphs (in particular, we explore scenarios where 10%, 25% or 50% links are known), construct graph-features based on the known links only and predict the "unknown" links. As our data set is highly imbalanced, we report the performance of a learning algorithm as the area under the ROC curve (called AUC). The ROC curve depicts the tradeoff between the true positive rate and the false positive rate.

The experiments we have designed are meant to address several questions, including:

- Does the ontology help improve the results obtained using only graph-based features?
- How does the performance of a classifier vary with the number of concepts used to construct the interest-based features (i.e., with the level of abstraction in ontology)?
- What is the best algorithm for the problem at hand?

Each experiment that we performed was repeated 5 times and the results were averaged over the 5 runs.

Results

Table 7.1 shows the results that were obtained for two widely used classifiers using the implementations provided by the Weka Data Mining Software (<http://www.cs.waikato.ac.nz/>).

Table 7.1 AUC values for Random Forests (RF) and Support Vector Machines (SVM) classifiers. We assume that k% links are known in the test set, where k is 50, 25 and 10, respectively. The known links are used to construct graph features.

Percent	Features	Random Forest	SVM
10%	Graph Only	0.7081	0.689
	Graph + Ontology	<u>0.7544</u>	<u>0.6818</u>
25%	Graph Only	0.7234	0.7101
	Graph + Ontology	<u>0.7564</u>	<u>0.747</u>
50%	Graph Only	0.8003	0.8214
	Graph + Ontology	<u>0.843</u>	<u>0.8446</u>

The results from table 7.1 help address the questions raised previously.

1. Does the ontology help improve the results obtained using only graph based features?

As can be seen, there is significant improvement in the AUC values when interest based numerical features in presence of the ontology are used in combination with graph based features in comparison with the AUC values achieved with use of only graph based features.

2. How does the performance of a classifier vary with the number of concepts used to construct the interest-based features (i.e., with the level of abstraction in ontology)?

As can be observed from Figure 7.2, For SVM classifier, better results were obtained as interest based features were constructed with cuts taken at lower levels in the ontology. The best results for SVM classifier were obtained at Level 11 in the ontology, for all 5 runs. For Random Forest classifier, higher AUC values were obtained at higher levels in the ontology. However, the level at which the best AUC values were obtained were less consistent as compared to SVM.

For Random Forest classifier,

- With 10 percent links known, best results were obtained for the five runs at levels (5, 6, 4, 4, 3)
- With 25 percent links known, best results were obtained for the five runs at levels (6, 6, 3, 6, 3)
- With 50 percent links known, best results were obtained for the five runs at levels (5, 7, 5, 6, 5)

The graph shown in figure 7.1 illustrates our point.

3. What is the best algorithm for the problem at hand?

Although higher AUC values could be observed for SVM classifier, greater number of nodes (cuts) were required to achieve the accuracy. On the other hand, the highest AUC value for RF classifier was slightly lesser than the corresponding AUC value for SVM. However, with RF classifier, smaller number of nodes are seen to be required to obtain a reasonably high accuracy. Also, because the difference in the AUC values between SVM

and RF classifiers is not very high overall the use of RF classifier would be preferred. The advantage of lesser concept representation for RF overrides the slightly higher accuracy provided by SVM.

Figure 7.1 Graph of reported AUC values v/s level numbers in the ontology used, for Random Forest classifier

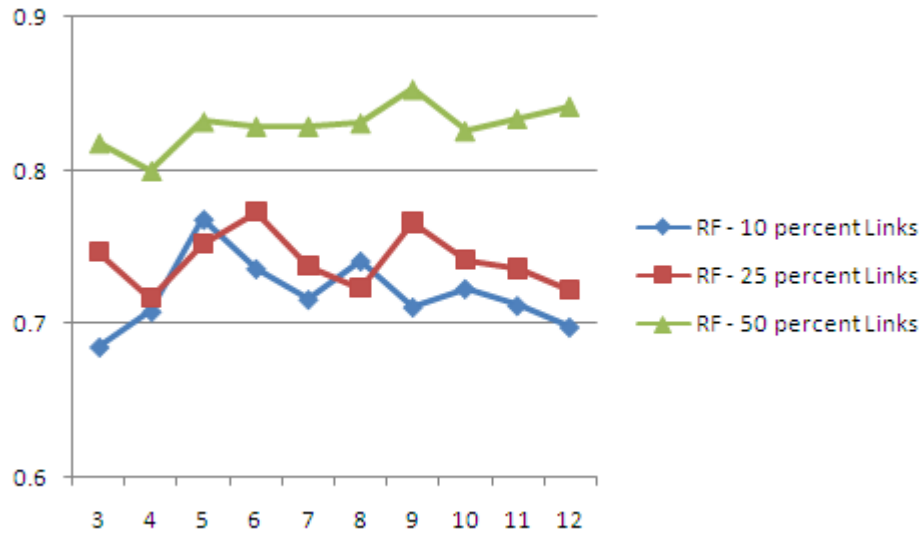
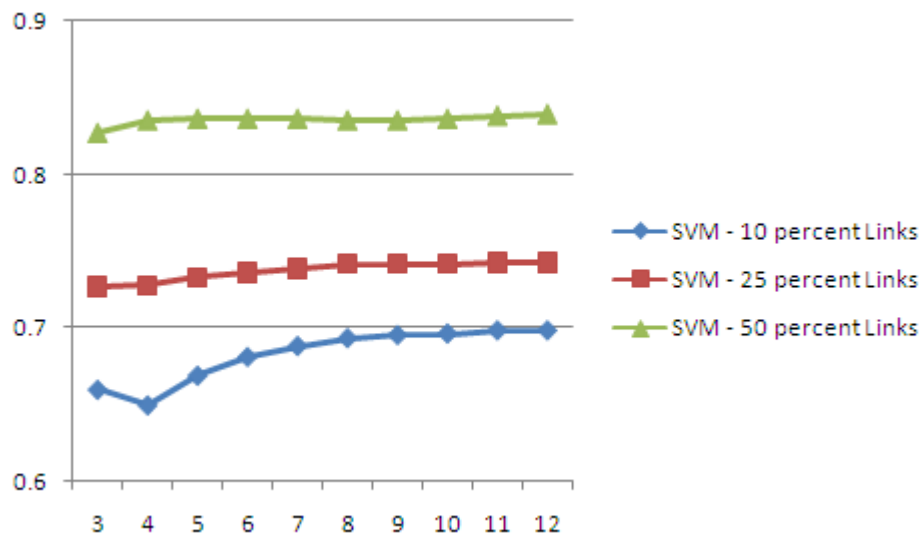


Figure 7.2 Graph of reported AUC values v/s level numbers in the ontology used, for Support Vector Machine classifier



CHAPTER 8 - Conclusion

Summary

In this work, we have explored three approaches to the problem of building an ontology over the interests specified by the users of a social network. The first approach uses Wikipedia as a comprehensive source of information and the LSA technique to find the similarity between documents corresponding to user interests. The second approach uses Wikipedia, and in particular the WCG to construct a hierarchy over user interests. Finally, the third approach exploits and extracts parts of the DMoz hierarchy to obtain the interest hierarchy. The first and third approaches produce usable hierarchies, although the Wikipedia/LSA hierarchy presents some limitations. While the second approach did not produce a useful ontology, it served as a bridge between the LSA-based and the DMoz based approaches. Moreover, it motivated the reuse of knowledge from existing hierarchies in the ontology engineering process.

Wikipedia v/s DMoz

When Wikipedia is used to address this problem, we basically “reduce” the problem of finding similarity between “interest terms” to the problem of finding similarity between “interest documents.” This is because using Wikipedia, it is not possible to compare terms, unless we obtain documents corresponding to the terms. Once documents corresponding to interest terms are obtained, there are various ways to semantically relate them as discussed in Chapter 2. The LSA technique works well for this task not only because document comparison is effective, but also because the documents (Wikipedia articles) are detailed.

However, one question to be asked is, when using Wikipedia, because we convert the “term comparison” problem to the “document comparison” problem, do we “reduce” the problem or do we convert a smaller problem to a bigger problem? That seems to be the case: although the LSA gives good results, it is computationally expensive (time and memory). That also exposes a problem with Wikipedia articles: they are detailed, but not always precise.

With DMoz, the opposite is true. Its category hierarchy is “crisp” even though it does not have as much coverage as Wikipedia. Searching for a term in the DMoz dump enables finding precise and accurate information as far as classifying the term is concerned, even if the approach

used is very simple. This is not surprising, however, because the simplicity of the approach is compensated by the rich categorization that DMoz provides. Additionally, as discussed in the previous section, the DMoz ontology, although not as detailed as Wikipedia, covers a wide variety of topics ranging from *Arts, Sciences, Computers* to *Movies, Business, Health* etc. This range of topics covers most of the domains, as far as user interests are concerned. Very importantly, the ontology engineered addresses all issues raised in chapter 1 and it appears to be useful for data mining problems, such as friendship prediction.

In summary, we believe that for engineering the interests ontology, DMoz clearly serves better than Wikipedia. In general, given a set of input “terms” it is more natural to categorize the terms into the DMoz hierarchy than to convert the problem to computing relatedness of documents using Wikipedia (or any other knowledge source). The above observations also help conclude that although Wikipedia is a comprehensive and reliable source of information, it may not always be suitable for engineering ontologies in certain domains and applications. Therefore, one cannot claim Wikipedia to be better than DMoz, in general. The vice-versa is also true. Although we believe that the DMoz hierarchy would help engineer better ontologies for most domains (as far as “interests” are concerned), this may not always be true. For example, for certain domains such as Bioinformatics, the DMoz hierarchy may not have adequate coverage. In such cases, the Wikipedia/LSA approach may be needed. Table 8.1 shows the summary of comparison between Wikipedia and DMoz on a point by point basis.

Table 8.1 Comparison between Wikipedia and DMoz

<i>Wikipedia</i>	<i>DMoz</i>
<i>A very detailed data source but not necessarily precise.</i>	<i>It is a web directory and therefore not very detailed. But provides a very accurate and precise categorization.</i>
<i>Category structure of Wikipedia is a graph (WCG). Thus, one category can be a sub category of multiple categories in the graph.</i>	<i>DMoz category hierarchy has a tree structure. No category in DMoz can be a sub category of multiple parent categories.</i>
<i>This makes WCG cumbersome to use, when</i>	<i>This makes DMoz hierarchy ideal to use for</i>

engineering a hierarchy.

Wikipedia is unsuitable when engineering an ontology for a very large number of input terms. The sheer size of Wikipedia articles makes an algorithm, using Wikipedia documents, time and space intensive.

Out of the 22,000 interest terms, no definitions were found on Wikipedia for about 8,000 terms. However, Wikipedia provides coverage for a wide range of domains.

*In summary, for applications requiring determining exact semantic relatedness between concepts, Wikipedia is an ideal data source to use as its documents are comprehensive and reasonably accurate. For example, for two instances **Tom Hanks** and **Forrest Gump**, using techniques that use Wikipedia yield a high similarity measure and hence indicate a high semantic relatedness between the two terms.*

classification of instances.

DMoz provides a very quick way to group very large number of terms into a hierarchy.

No information was found in DMoz for only 4,500 interests in input data, thus resulting in better coverage than Wikipedia. However, it may not have a good coverage for terms from rare domains.

*Terms grouped under DMoz category may not necessarily be close semantically. However, if a loose grouping of instances is sufficient for the purpose of an application, it is significantly advantageous to use DMoz ahead of Wikipedia. For example, an approach that uses DMoz groups **Tom Hanks** under category **Arts\Performing Arts\ Actors and Actresses** while **Forrest Gump** is grouped under **Arts\Movies\Titles**. Because the 2 interests are under different categories, they are supposedly not closely related. However, for the purpose of engineering an interests ontology, it is enough, or in some cases, better to know that **Tom Hanks** is an actor and **Forrest Gump** a movie.*

DMoz ontology limitations

The DMoz based approach eliminated the drawbacks of the other two approaches by engineering an n-ary ontology. While the problem with previous approaches was that n was very small (n was 2), in the DMoz-engineered ontology, n is very large in certain cases. For example, the node *Bands and Artists*, a child of *Music* node, has 1912 children. As a result of such large values of n, the resulting ontology tree is a very dense and short “bush” with only 13 levels. The problem that a short ontology poses is that when cuts are taken at various levels in the ontology to construct interest based features, we gain or lose information very fast on moving from one level to the other. This in turn is bound to impact the performance of the prediction algorithm. To a certain extent, this is also pointed out by the results that are achieved for the Random Forest classifier. As was seen in Figure 7.1, the AUC curve reported with this classifier is not smooth and very much inconsistent as far as the levels at which best values are obtained are concerned. Therefore, in some cases, the results achieved with this ontology are not as satisfactory as those achieved in the previous work by (Bahirwani et al., 2008).

CHAPTER 9 - Future work

Improvements to the Wikipedia/LSA approach

The LSA based approach, as mentioned earlier, comes up with a very effective ontology of the interest documents. While this ontology is binary and the clusters do not represent the semantics of the interests, the approach un-hides latent relationships between documents because of a concept based approach as opposed to a word based approach. Such rich relationships enable expression of domain-specific knowledge, without the need to include domain-specific terms (Varma et al., 2008). This combined with the fact that documents fetched from Wikipedia are reliable and of generally good quality, helps us conclude that the ontology can be used very effectively for accurate prediction of user interests, in cases where DMoz hierarchy does not provide sufficient coverage for the interests.

The two drawbacks of the Wikipedia-LSA approach are that no semantics get associated with the clusters and the ontology engineered is binary. If we can work to get rid of these two drawbacks, the resulting ontology may potentially result in very accurate prediction of friendship relations.

Concepts may be tagged to clusters by combining the LSA approach with the WCG based approach. When documents for similar interests are clustered together in the LSA based approach, we may check for a Wiki category that is common to both the documents. If there exists such a category, then the cluster can be tagged with this category title. As another option, the work of (Syed et al. 2008) to predict concept information associated with a group of documents may also be used to label the clusters.

To engineer an n-ary hierarchy we may use a clustering approach which can work as follows: At each step, cluster two nodes. One of the nodes should be in the processing set (the set of instances which are still not part of the ontology) and the other should be in the processed set (the set of nodes in the ontology). Select that node A in the processing set which has the highest similarity measure with some node B in the processed set. If B is already part of some cluster such that C is its sibling and D as its parent, add the node A into the ontology such that A, B, C are all siblings with D as their parent. This way the ontology engineered would be n-ary.

Another problem with the LSA based approach is that it is very time and space intensive. The reason for this is that corresponding to the 22,000 interests, there are 68,000 document instances. Furthermore, each document (being a Wikipedia article) is fairly long having more than 100 terms. To construct a term document matrix for this data and performing the subsequent singular value decomposition therefore obviously has scalability issues. If, however, the number of terms in the documents can be reduced, the scalability can be much improved. In other words, the instance documents need to be condensed to shorter documents. One way of doing this is by selecting only the first paragraph of text in the Wikipedia document. The first paragraph supposedly may have the definition of the required interest which can provide sufficient number of terms to the LSA algorithm.

Searching interests in Wikipedia and DMoz dump

The search techniques that we use to obtain interest definitions from Wikipedia as well as DMoz are very primitive. As a result, a lot of interest words go unmatched. For example, for an interest like “avoiding sleep” no definition is found on Wikipedia. This means that there is no article on Wikipedia with document title “avoiding sleep”. However, this does not mean that the term cannot be found in any Wikipedia article. Therefore, to provide better coverage for the interest words, the text of every downloaded article may be indexed using a search engine library like Lucene.

Improvement in constructing interest based features

To construct interest based features in presence of the ontology, we compute the numeric values by using cuts made at a particular level in the tree. Better features may be obtained if the cuts in the ontology are made not just on the basis of level. Factors like number of child nodes, number of descendants, number of descendants that are leaf nodes etc may be used to come up with the variable depth cuts.

Studying other prediction problems

This work explores various ways to engineer ontologies of interests and studies the impact of the ontologies in predicting friendship relations. The constructed ontologies may also be used to study other prediction problems in Social Networks like predicting the interests associated with a user given the set of interests of all friends of that user.

References

- Bahirwani, V., Caragea, D., Aljandal, W., Hsu, W., 2008. Ontology Engineering and Feature Construction for predicting Friendship Links in the Live Journal Social Network. In *Proceedings of The 2nd SNA-KDD Workshop '08 (SNA-KDD' 08)*, Las Vegas, Nevada, USA.
- Bausch, S., Han, L., 2006. *Social Networking sites grow 47 percent, year over year, reaching 45 percent of Web users, according to Nielsen/Netratings* [online]. Available at < http://www.nielsen-online.com/pr/pr_060511.pdf >
- Caragea, D., Bahirwani, V., Aljandal, W., Hsu, W., 2009. Ontology based Link-prediction in the Live Journal Social Network. In *Proceedings of Association for the Advancement of Artificial Intelligence*.
- Cui, G., Lu, Q., Li, W., Chen, Y., 2008. Corpus exploitation from Wikipedia for ontology construction. In *European Language Resources Association, editor, Proceedings of the 6th International Conference on Language Resources and Evaluation*, Marrakech, Morocco.
- Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R., 1990. 'Indexing by Latent Semantic Analysis' *Journal of the American Society for Information Science*, 41, pp. 391-407.
- Gabrilovich, E., Markovitch, S., 2007. Computing Semantic Relatedness using Wikipedia-based Explicit Semantic Analysis. In *Proceedings of The 20th International Joint Conference on Artificial Intelligence*, Hyderabad, India.
- Gabrilovich, E., Markovitch, S., 2006. Overcoming the brittleness bottleneck using Wikipedia: Enhancing Text Categorization with Encyclopedic Knowledge. *Proceedings of the Twenty-First National Conference on Artificial Intelligence*, AAAI'06, Boston, MA.

- Grobelnik, M., Mladenić, D., 2005. Simple Classification into Large Topic Ontology of Web Documents. In *Proceedings: 27th International Conference on Information Technology Interfaces*, Cavtat, Croatia.
- Grobelnik, M., Brank, J., Mladenić, D., Novak, B., Fortuna, B., 2006. Using DMoz for constructing ontology from data stream. In *Proceedings of the 28th International Conference on Information Technology Interfaces*, Cavtat, Croatia.
- Gruber, T.R., 1993. *A translation approach to portable ontology specifications*. Technical Report 5(2):199-220, Knowledge Systems AI Laboratory, Stanford University.
- Janik, M., Kochut, K., 2007. *Wikipedia in action: Ontological Knowledge in Text Categorization*. Technical Report No. UGA-CS-TR-07-001, University of Georgia.
- Koller, D., Sahami, M., 1997. Hierarchically classifying documents using very few words, In *Proceedings of the 14th International Conference on Machine Learning ICML-97*, pp. 170-178, San Francisco, CA.
- Lee, C.H., Yang, H.C., Ma, S.M., 2006. A Novel Multi-Language Text Categorization System Using Latent Semantic Indexing. In *Proceedings of The First International Conference on Innovative Computing, Information and Control (ICICIC-06)*, Beijing, China.
- Milne, D., 2007. Computing Semantic Relatedness using Wikipedia Link Structure. In *Proceedings of the New Zealand Computer Science Research Student Conference, NZ CSRSC'07*, Hamilton, New Zealand.
- Nemrava, J., Svátek, V., 2005. Text mining tool for ontology engineering based on use of product taxonomy and web directory. In *Proceedings of the DATESO Annual International Workshop on DAtabases, TExts, Specifications and Objects*.
- Rosario, B., 2000. *Latent Semantic Indexing: An Overview*. Final Paper, INFOSYS 240 Spring 2000.

Strube, M., Ponzetto, S.P., 2006. WikiRelate! Computing Semantic Relatedness using Wikipedia. *American Association for Artificial Intelligence, 2006*, Boston, MA.

Syed, Z.S., Finin, T., Joshi, A., 2008. Wikipedia as an ontology for describing documents. In *Proceedings of the 2nd International Conference on Weblogs and Social Media, (ICWSM)*.

Varma, V., 2002. Building Large Scale Ontology Networks. *Language Engineering Conference, 2002. Proceedings*.

Zesch, T., Gurevynch, I., 2007. Analysis of the Wikipedia Category Graph for NLP Applications. In *Proceedings of the TextGraphs-2 Workshop (NAACL-HLT)*.