

Efficient Query Routing for Information Retrieval in Semantic Overlays

Hai Jin, Xiaomin Ning, Hanhua Chen, Zuoning Yin
Cluster and Grid Computing Lab

Huazhong University of Science and Technology, Wuhan, 430074, China
hjin@hust.edu.cn

ABSTRACT

A fundamental problem in peer-to-peer networks is how to locate appropriate peers efficiently to answer a specific query request. This paper proposes a model in which semantically similar peers form a semantic overlay network and a query can be routed or forwarded to appropriate peers instead of broadcasting or random selection. We apply *Latent Semantic Indexing* (LSI) in information retrieval to reveal semantic subspaces of feature spaces from documents stored on peers. After producing semantic vectors through LSI, we train a *support vector machine* (SVM) to classify the peers into different categories based on the extracted vectors. Peers with close categories are defined as semantic similarity and form a semantic overlay. Experimental results show the model is efficient and performs better than other non-semantic retrieval models with respect to accuracy. In addition, our approach improves the recall rate nearly 100% while reducing message traffic dramatically compared with Gnutella.

Categories and Subject Descriptors

H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval - Retrieval Models

General Terms

Algorithms, Experimentation, Performance

Keywords

Semantic Overlay, Information Retrieval, Query Routing

1. INTRODUCTION

Peer-to-peer networks are more scalable, fault tolerant, autonomic and cost effective compared with centralized systems. However, efficient query routing still remains a challenge. Query routing in current P2P systems is based on the following techniques: centralized indexing, query flooding, random walk or heuristic. The above systems have to search a large number of peers to ensure the recall/precise and cause much network traffic for a given query request. In addition, peers may not be willing to accept arbitrary messages from other peers. Structured overlays such as CAN [3] and Chord [4] based on *Distributed Hash Table* (DHT) provide good scalability and efficient query routing

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SAC'06, April, 23-27, 2006, Dijon, France.

Copyright 2006 ACM 1-59593-108-2/06/0004...\$5.00.

performance, but they just allow retrieval of (key, value) pairs and are not suitable for full text search.

Due to their inherent merits, peer-to-peer systems are becoming popular in building large-scale information retrieval systems. But current peer-to-peer systems require exact metadata such as filename or keyword to perform a successful search. In this paper we propose a semantic overlay model to address this problem. In this model, semantically similar peers are clustered together and a query is to be routed or forwarded to semantically related peers. Some well-studied classical information retrieval algorithms are combined to build the semantic overlay.

LSI [1, 2, 12] is a method for automatic indexing and retrieval, which is based on the *vector space model* (VSM) to compute the relevance scores of documents in a reduced, topic-related space. LSI overcomes the problems of synonymy, polysemy and noise in documents and reveals semantics of terms and documents. SVM [6, 11] is based on the Structure Risk Minimization principle from computational theory and is a solution to data overfitting in neural networks. To build a semantic overlay, we apply LSI and SVM to classify the peers into different categories and peers with close categories are defined as semantic similarity.

In brief, the model addresses the following problems:

- A semantic overlay based on LSI and SVM is feasible and performs well.
- Semantic query routing in addition to naive keyword match in semantic overlays is more efficient and accurate compared with other non-semantic models.

The paper is organized as follows. Section 2 proposes the system model, the semantic overlay algorithm and the query routing algorithm. Section 3 presents the simulation and experimental results. Related work is discussed in Section 4. Finally, we conclude in Section 5.

2. SYSTEM MODEL

2.1 Overview

We present the model for a document sharing P2P system, because it is appropriate to be extended to some other similar systems such as metadata retrieval of images. Let us consider the scenario in a computer science scholar P2P network where each peer has a collection of computer science articles. Researchers in the system want to share their metadata such as titles, authors, abstracts and references about the publications. They can issue queries based on simple keyword matching, but they may prefer performing more complex query capabilities for metadata retrieval that are semantically related to the query. Computer science articles can be classified according to *ACM Computing Classification System* (CSS). Summarization is a statistical representation for a peer's collection of articles based on ACM

CSS. We define the summarization as the peer's semantic representation. To reveal semantic of documents and establish the semantic overlay, *information retrieval* (IR) algorithms such as LSI and SVM are applied. We give a brief review of the basic components below.

Latent Semantic Indexing. Vector Space Model is a major retrieval model in IR research. In VSM, documents and queries are represented as vectors of terms that occur within documents in a collection. The measure of similarity between document and query vectors is the cosine coefficient. LSI is an important extension of VSM and uses a statistical method called *Singular Value Decomposition* (SVD) to uncover the word associations between documents [2]. Suppose there are t distinct terms in a collection with d documents. The term-document matrix $A=(a_{ij}) \in R_{t \times d}$ represents the collection. Each column vector a_j corresponds to a document j . Weight a_{ij} represents the importance of term i in document j . The matrix $A(t \times d)$ is converted into three matrices U , Σ and V through SVD. U is a $t \times t$ orthogonal matrix having the left singular vectors of A as its columns. V is a $d \times d$ orthogonal matrix having the right singular vectors of A as its columns and Σ is a $t \times d$ diagonal matrix having the singular values $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_{\min(t,d)}$ of A in the order along its diagonal. By keeping the k largest singular values in Σ and the corresponding columns in U and V , A can be approximated by:

$$A_k = U_k \Sigma_k V_k' \quad (1)$$

The coordinates of a new $t \times 1$ document vector p' in the basis U_k are given by the elements of the vector $U_k' p'$.

Support Vector Machine. SVM is a pattern recognition method based on statistical computational learning theory [6] and is widely used for classifying large datasets. When used for classification, SVM creates a hyperplan that separates the training data with the maximum-margin into a feature space induced by a kernel function used as the inner product. The maximum margin training algorithm finds a decision function for the pattern vector x with dimension n belonging to either of two classes A and B . The input to the training algorithm is a set of p examples x_i with labels y_i :

$$(x_1, y_1), (x_2, y_2), \dots, (x_p, y_p), \text{ where } \begin{cases} y_k = 1, & \text{if } x_k \in \text{class } A \\ y_k = -1, & \text{if } x_k \in \text{class } B. \end{cases} \quad (2)$$

From the training set the algorithm finds the parameters of a decision function during a learning phase. After training, the classification of unknown patterns is predicted.

The semantic overlay model consists of two steps: (1) establishing semantic overlay, and (2) query preprocessing and query routing. We present the algorithms respectively.

2.2 Semantic Overlay Algorithm

The major objective of establishing semantic overlay is to classify the peers into different categories. Supervised classification using SVM involves a training phase and a prediction phase. During the training phase, a large set of documents with known category labels are used to train the classifier. Feature selection is important when applying SVM. To reduce dimensionality, we choose d columns of the matrix $\Sigma_k V_k'$ according to Equation 1 as training set instead of d documents directly. During the prediction phase, each document after preprocessing is represented by a $t \times 1$ document

vector p' and SVM classifies the document according to the vector $U_k' p'$ which is the coordinates of p' in the basis U_k .

All documents on the peer are classified into different categories and semantic of each peer is represented by its top categories and proportions of each category. When system initiates, each peer sends its semantic information to other peers with a small *Time To Live* value such as $TTL = 2$ and peers decide to accept or discard it according to whether the received semantic information is similar to their own. As semantic information of each peer is small and remains relatively stable, the message traffic caused by it is almost negligible compared with Ping, Pong, Query, Push and Query Hit in Gnutella. Algorithms 1-3 give the procedures of establishing the semantic overlay.

Algorithm 1: setupSemanticOverlay

Input: Initial peers $P=\{P_1, P_2, \dots, P_n\}$, Training documents with labels TD , Documents set on peers $D=\{D_1, D_2, \dots, D_n\}$.

Output: Semantic overlay $SP=\{SP_1, SP_2, \dots, SP_n\}$ with categories $C=\{C_1, C_2, \dots, C_n\}$ where C_i is the category set of D_i on P_i .

```

/*Assume the name of current peer is  $P_i$ .*/
1:   $SP := \Phi$ 
2:  model  $m := \text{trainSVM}(TD)$ 
3:  classifyPeer( $P_i, D_i, m, C_i$ )
4:  broadcastSemanticInformation( $P_i, C_i$ ) with  $TTL = 2$ 
5:  while received semantic info  $C_j$  from  $P_j$  do
6:    if  $\text{sim}(C_i, C_j) > \text{Threshold}_{\text{sim}}$  then
7:      accept  $C_j$  and set semantic shortcut  $SP_{ij}$  to  $P_j$ 
8:    end if
9:  end while

```

Algorithm 2: Train SVM: trainSVM

Input: Training documents with labels $TD=\{TD_1, TD_2, \dots, TD_m\}$, TD_j consists of documents with category label j .

Output: support vector model m to predict documents set D on peers.

```

1:  for each documents set  $TD_j \in TD$  do
2:    for each document  $TD_{jk} \in TD_j$  do
3:      /*Preprocess document  $TD_{jk}$ .*/
4:      apply stop words to remove common words
5:      stemming to eliminate morphological variants
6:    end for
7:  end for
8:  /*Setup vector space model before LSI.*/
9:  apply  $TF * IDF$  to setup VSM matrix  $A$ 
10: SVD convert  $A$  into three matrices  $U, \Sigma, V'$ 
11: keep  $k$  largest singular values, thus  $A_k \approx A$ 
12: model  $m := d$  columns of the matrix  $\Sigma_k V_k'$ 

```

Algorithm 3: classifyPeer

Input: Peer P_i , Documents set D_i on P_i , SVM model m .

Output: Category C_i of P_i .

```

1:  for each document  $D_{il} \in D_i$  do
2:    preprocess  $D_{il}$ 
3:    represent  $D_{il}$  as  $t \times 1$  document vector  $p'$ 
4:    SVM predict vector  $U_k' p'$  as category  $C_{il}$  using  $m$ 
5:  end for

```

- 6: choose top categories and proportions of each category as peer P_i 's semantic

2.3 Query Routing Algorithm

Intelligent query routing can be tackled based on the above semantic overlay. When a peer issues a query request, the peer selection and query routing model selects peers most semantically related to the query. If a related peer receives the query, it performs local query using keyword matching or calculating similarity based on LSI between the query and every document that has the same category as the query request. Results extracted from the related documents are returned to the peer that originally issued the query. Besides performing local query, the peer should also decide whether and where to forward the query.

In our scholar P2P network, a query request may comprise not only keyword matching such as author, title, abstract and year fields but also semantic search such as category. Even searching by title and abstract fields can be thought as a semantic search and results related to the title or the abstract should be returned. We treat the title and abstract fields as semantically related search fields for the rest of the paper.

Algorithm 4 describes the query routing procedure.

Algorithm 4: performQueryRouting

Input: Query request Q , Semantic topology SP_i of Peer P_i , Document set D_i with category labels.

Output: Result R similar to Q , Peers collection PF to be forwarded.

```

1:   $PF := \Phi$ 
2:  if  $Q$  is keyword exact-matching request then
3:    perform local keyword search and return  $R$ 
4:    random select known peers  $PF$  to forward  $Q$ 
5:  end if
6:  else if  $Q$  contains semantic search fields then
7:    predict  $Q$ 's category  $Q_c$ 
8:    perform local search and return  $R$ 
9:    for each peer  $p \in SP_i$  do
10:     calculate similarity  $\text{sim}(Q_c, C_p)$ 
11:     if  $\text{sim}(Q_c, C_p) > \text{Threshold}$  then
12:       put  $p$  into set  $PF$ 
13:     end if
14:   end for
15:   forward  $Q$  to peers collection  $PF$ 
16: end if

```

3. EXPERIMENTAL RESULTS

In this section, we evaluate the above algorithms. The experiment consists of two steps: (1) evaluating the accuracy of each peer's categorization, and (2) evaluating the recall rate and message traffic of the model compared with Gnutella.

3.1 Experimental Setup

In this experiment, we use a large set of abstracts from Computer Science Database¹ which is a bibliographic database covering literature in the field of Computer Science and Computer Technology with about 400,000 citations from 1972. These selected abstracts are divided into three top levels of ACM CSS

categories: C. Computer Systems Organization, G. Mathematics of Computing, and H. Information Systems. Table 1 summarizes the settings of accuracy evaluation.

Table 1. Settings for accuracy evaluation

Parameter	Values
Data set	Abstracts of CompuScience
ACM CSS categories	3
# of documents for training	3009
# of documents for testing	300
# of distinct words	12582
Stop words list	SMART stop words
Stemming	Porter algorithm
Term weighting	TF*IDF
Decomposition	SDDPACK ²
SVM kernel function	Gaussian RBF kernel

To evaluate the recall rate and message traffic of the model compared with Gnutella, we simulate 8 topological graphs where the numbers of nodes range from 100 to 800. Each topology graph accords with a power-law with the exponent $\alpha=3.0$ and the average degree is 2.8~3.4. Table 2 summarizes the settings for evaluating the recall rate and message traffic.

Table 2. Settings for evaluating recalls and traffic

	Descriptions	Values
N	# of nodes in the network	100 – 800
α	Exponent α of power law	3.0
d	Average degree	2.7-3.4
T	# of ACM topics in the network	30
D	Max # of documents on each node	2000
Tsd	Threshold of similarity between peers	0.5
Q	Max # of queries by each peer	200
C	Max keywords in each document	25
TTL	Time to Live	2 - 5

3.2 Evaluation

In this experiment, we simulate various settings with different parameters and choose optimal parameters. Performance metric *Accuracy* describes the proportion between all documents to classify and the correctly classified documents. In the evaluation, we choose $k=90$ in SDD [10] decomposition and $\sigma=0.35$ for RBF-kernel. Table 3 shows that the average accuracy reaches 89.6% and the accuracy of category C is relatively higher than G and H. There are several reasons for the phenomena. (1) Before the evaluation we assume that each article belongs to a single ACM CSS topic, but that is not always appropriate. Computer science related articles usually can belong to several topics with a primary one, even we can not exactly confirm which topic an article should be classified into; and (2) Category C is distinct from G and H while Category G has some overlap with H, such as sub-topics G. Mathematics of Computing/*Mathematical Software* and H. Information Systems/*Information Systems applications*.

Table 3. Accuracy with optimal parameters

Category	Accuracy	Miss rate
C. Computer Systems Organization	96.5%	3.5%
G. Mathematics of Computing	85.7%	14.3%
H. Information Systems	86.6%	13.4%
Average	89.6%	10.4%

¹ <http://www.zblmath.fiz-karlsruhe.de/COMP/quick.html>.

² <http://www.cs.umd.edu/~oleary/SDDPACK/>.

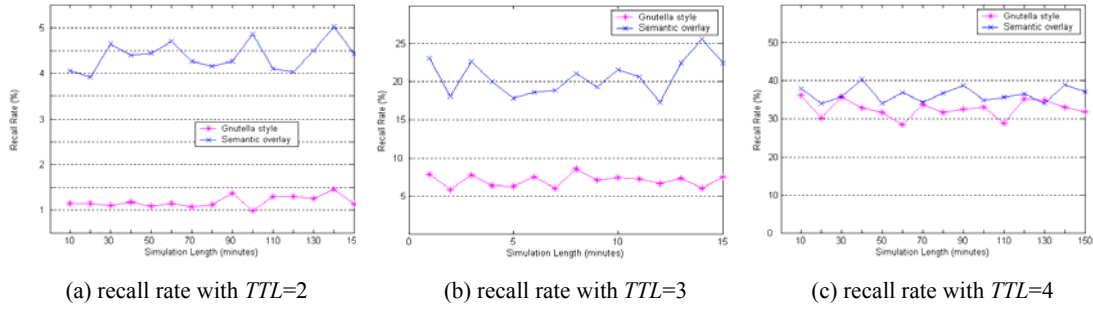


Figure 1. Comparing recall rates of Gnutella and Semantic overlay model when varying TTL values.

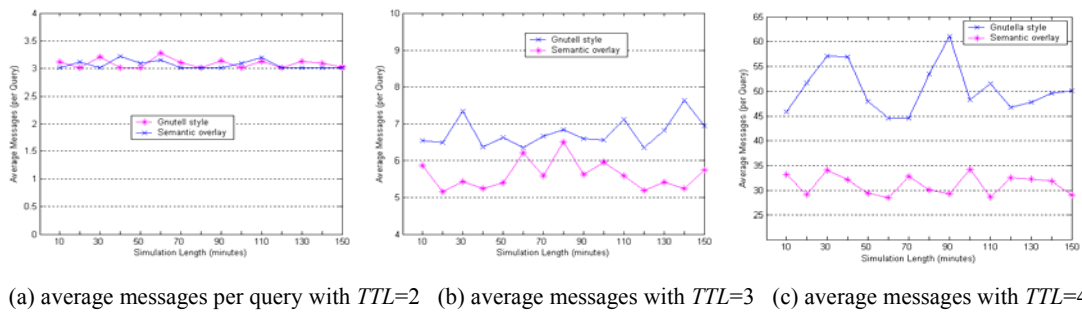


Figure 2. Comparing average messages per query request when varying TTL values.

The selection of dimension k is important for SDD decomposition and usually experiential. Figure 3 shows the effect of varying the values of matrix decomposition dimension k with three different Gaussian RBF kernel function parameters. For each parameter σ , there is a peak *Accuracy* value. When $\sigma=0.35$ and $k=90$, the *Accuracy* reaches the highest value 89.6% in our experiment. This figure suggests that selecting appropriate dimension k for SDD is important and very high or low dimension reducing drops the *Accuracy* of text categorization.

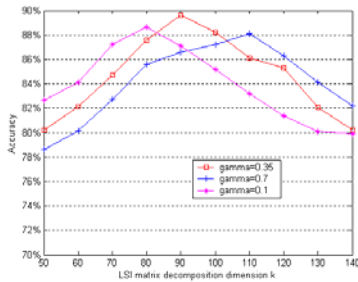


Figure 3. The effect of varying dimension k .

Figure 4 shows the effect of varying the values of parameters σ with three different dimensions k . The result is not so apparent compared with that of Figure 3. But we still can see that a peak *Accuracy* value for each dimension k exists. When σ varies from 0.05 to 0.15, the curves have steep jumps. The curves ascend or descend moderately when σ varies from 0.15 to 1.0. This is because very small radius σ can cause the overfitting problem.

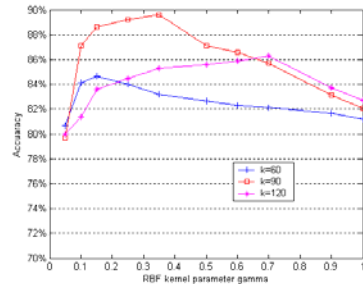


Figure 4. The effect of varying parameter σ .

Recall rate is important for full-text searching, but we should reduce message traffic at the same time. In this evaluation, we focus on average messages caused by each query request. We choose the topology with 800 nodes and increase TTL from 2 to 4. Figure 1 shows that both recall rates of our model and Gnutella increase as the TTL increases. When TTL is small (i.e., TTL=2 or 3) the recall rate of our model excels that of Gnutella apparently, but both recall rates are nearly close when TTL equals 4. This is because that the number of nodes in our experiment is relatively small and the average shortest distance is less than 5. When we set TTL=4, the flooding method of Gnutella can crawl almost all the network but causes very heavy traffic at the same time. Figure 2 shows that our model can reduce message traffic dramatically compared to Gnutella when TTL increases.

Figure 5 shows that when the number of nodes varies from 100 to 800, the recall rate of our model excels almost 100% over Gnutella. As the number of nodes increases, the proportion of recall rate between our mode and Gnutella increases apparently.

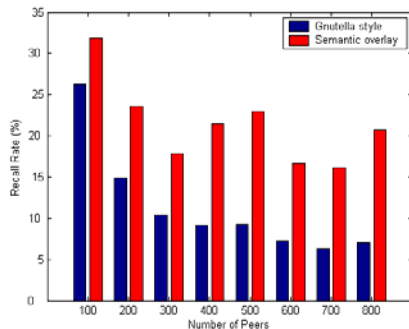


Figure 5. Comparing recall rates of Gnutella and Semantic overlay model.

From Figures 1, 2 and 5, we observe that the semantic overlay model can increase the recall rate and reduce message traffic dramatically. The model is efficient for document sharing P2P systems.

4. RELATED WORKS

Efficient query routing in peer-to-peer systems has been discussed for a long time. Flooding-based systems such as Gnutella consume amounts of network bandwidth and CPU cycles. Many approaches are proposed to address this problem. These approaches can be divided into random walk, heuristic or good peer model. Structured overlays such as CAN and Chord based on DHT provide good scalability and efficient query routing performance. But all these methods do not provide how a relevant resource can be found in the most appropriate place.

In pSearch [5, 9], a semantic overlay network is introduced. pSearch distributes document indices through the P2P network based on document semantics generated by LSI and organized as a CAN. The search cost for a given query is reduced, since the indices of semantically related documents are likely to be co-located in the network. Although the pSearch approach works well to find documents close to a query, its performance under the dynamic conditions common for peer-to-peer systems is unknown.

Bibster [7, 8] is another model based on semantic overlay. In bibster, peers use a shared ontology to advertise their expertise in peer-to-peer networks. The knowledge about the expertise of other peers forms a semantic topology, independent of the underlying network topology. Although this approach can forward queries to those peers that have a good chance to answer it, the semantic topology is not reliable. In bibster, the expertise of peers is extracted from documents just using lexical analysis and leads to many false categorization results (only approximately 10% accuracy according to our evaluation on bibster), so the semantic topology based on the expertise in bibster is not dependable.

5. CONCLUSIONS

In this paper we present a semantic overlay model that is based on LSI and SVM to cluster semantically related peers together. A query can be routed to related peers and thus increases the recall rate while reducing hops and messages. Experiments have shown the following results: (1) establishing a semantic overlay model based on LSI and SVM is feasible and performs well, and (2) query routing in the semantic overlay is efficient.

In our current work, we have some assumptions, such as the static SVM model and setting parameters manually. For future work, we should tackle with these problems. We summarize our future directions as follows: (1) how to establish a semantic overlay with hierarchical categorization, and (2) how to integrate all parameters and automatically change parameters setting with the application situation changing to yield optimal results, and (3) how to manage dynamic document collections.

6. ACKNOWLEDGMENTS

This paper is supported by the National 973 Key Basic Research Program under grant No.2003CB317003.

7. REFERENCES

- [1] M. Berry, Z. Drmac, and E. Jessup, "Matrices, Vector Spaces, and Information Retrieval", *SIAM Review*, 41(2):335-362, 1999.
- [2] M. S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman, "Indexing by Latent Semantic Analysis", *Journal of American Society for Information Science*, 41(6):391-407, 1990.
- [3] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker, "A Scalable Content-Addressable Network", *Proceedings of the ACM SIGCOMM'01*, 2001.
- [4] I. Stoica, R. Morris, D. Karger, M. Frans Kaashoek, and H. Balakrishnan, "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications", *Proceedings of the ACM SIGCOMM '01*, 2001.
- [5] C. Tang, Z. Xu, and S. Dwarkadas, "Peer-to-Peer Information Retrieval Using Self-Organizing Semantic Overlay Networks", *Proceedings of the ACM SIGCOMM'03*, 2003.
- [6] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A Training Algorithm for Optimal Margin Classifiers", *Proceedings of the 5th Annual ACM Workshop on Computational Learning Theory*, ACM Press, July 1992, Pittsburgh, PA, pp.144-152.
- [7] J. Broekstra, M. Ehrig, P. Haase, F. Harmelen, M. Menken, P. Mika, B. Schnizler, and R. Siebes, "Bibster - a Semantics-based Bibliographic Peer-to-Peer System", *Proceedings of the WWW'04 Workshop on Semantics in Peer-to-Peer and Grid Computing*, 2004.
- [8] C. Tempich, S. Staab, and A. Wranik, "REMINDIN': Semantic Query Routing in Peer-to-Peer Networks based on Social Metaphors", *Proceedings of the 13th Int. World Wide Web Conference*, 2004.
- [9] C. Tang, S. Dwarkadas, and Z. Xu, "On Scaling Latent Semantic Indexing for Large Peer-to-Peer Systems", *Proceedings of the ACM SIGIR'04*, 2004.
- [10] T. Kolda, *Limited-Memory Matrix Methods with Applications*, PhD thesis, 1997, University of Maryland at College Park.
- [11] J. C. Burges, "A Tutorial on Support Vector Machines for Pattern Recognition", *Data Mining and Knowledge Discovery*, 2(2):121-167, 1998.
- [12] D. Hull, "Improving Text Retrieval for the Routing Problem Using Latent Semantic Indexing", *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, 1994, pp.282-291.