# Enhance Genomic IR with Term Variation and Expansion: Experiences of the IASL Group at Genomic Track 2005

Tzong-Han Tsai, Chia-Wei Wu, Hsieh-Chuan Hung, Yu-Chun Wang,
Ding He, Yi-Feng Lin, Cheng-Wei Lee, Ting-Yi Sung, Wen-Lian Hsu

Institute of Information Science, Academia Sinica
115 Nankang, Taipei, Taiwan

{thtsai, cwwu, yabt, mainlander, derrick, lego, aska, tsung, hsu}@iis.sinica.edu.tw

## ABSTRACT
The rapid increase of biomedical literature available on the web has made it increasingly difficult to find precise information. To implement an accurate biomedical information retrieval (IR) system, we must deal with the variants of biomedical terms carefully. In this paper, we focus on the generation of aliases, synonyms, acronyms, and lexical variants of such terms. In addition, we also propose a hyphen handling technique for processing hyphenated terms. We use the original terms/phrases, and expanded terms/phrases to construct an Indri query, and evaluate the effectiveness of various methods by two indicators: MAP, and recall. Our experiment results show that tackling hyphenation improves information retrieval significantly. In addition, synonym expansion also enhances IR performance when the focus of a query is identified. For a natural language query, deep semantic analysis and more knowledge-oriented expansion should be applied.

## Keywords
Biomedical literature, information retrieval, lexical variation, query expansion

## 1. INTRODUCTION
Advances in biotechnology have given rise to a vast amount of biomedical literature, most of which is now available to the scientific community in an electronic format. However, the rapid growth in the literature has made it increasingly difficult to locate accurate information expeditiously. Clearly, natural language processing (NLP) applications (such as information retrieval and information extraction) are essential for navigating through and searching overwhelming biomedical texts.

Information retrieval (IR) identifies and extracts documents that are relevant to a user's query from a large database. Most approaches, such as the famous vector space model [11] score the degree of match between the terms in a query and the related terms in a document.

Unlike information retrieval in general domains, biomedical IR systems suffer from low recall, because biomedical terms usually have many aliases, abbreviations, acronyms, and synonyms. In addition, each biomedical named entity has many lexical variants. Therefore, proper management of terms in both user queries and documents is essential for achieving good retrieval quality in the biomedical domain.

We chose Indri as our biomedical IR search engine for its ability to express the complex relationship between original terms. Indri combines the language modeling [9] and inference network [12] approaches for information retrieval. Indri utilizes language modeling probabilities that increase robustness, whereas most approaches that use tf.idf-based term weights.

Genomic Track provides a good test bed for researchers in biomedical information retrieval. This year, we participated in the ad hoc retrieval task, the goal of which was to mimic conventional searching. The scenario was a user with a specific information need, i.e., searching the MEDLINE bibliographic database to find relevant articles for research. To provide systems with better defined queries for finding genomics information, the query topics are more structured than those in the 2004 track, mostly free-form topics.

In this paper, we develop several methods for term expansion and variation. We exploit databases and ontologies, such as AcroMed [10] and EuGenes [4] to find aliases, abbreviations, acronyms, and synonyms. We also apply the lexical variation rules described in [13] and particularly tackle the hyphen problem in biomedical terms in both query processing and document indexing. We use Indri query language to organize the original terms/phrases, expanded terms/phrases according to their relationships. We demonstrate the effectiveness of our methods for each query template.

## 2. SYSTEM OVERVIEW
An overview of our biomedical information retrieval system developed for the Genomics Track is given in Figure 1. The system comprises four main stages: document indexing, input query processing, Indri query construction, and IR searching. Most parts of the system can be adjusted by varying the respective parameters.

**Document Indexing**
This module stores all documents in an index file and transforms each one into a word list. The file connects the

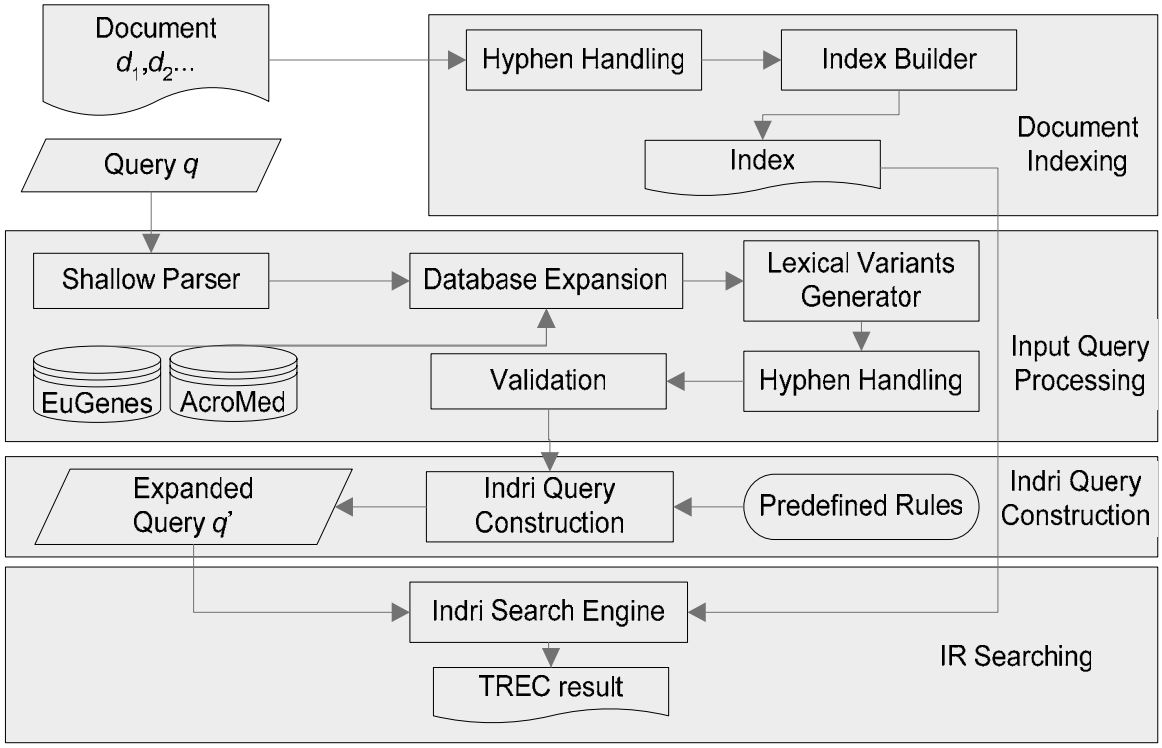


**Figure 1. Our biomedical information retrieval system**

query terms to the words in the documents. In addition, we remove all stop words, perform stemming on words, and convert words to lowercase. We also replace hyphens with a white space character and insert a white space character between an alphabetic character and a numeric character. This will be described in detail in Section 4.5.

**Input Query Processing**

The second stage is query processing in which we also remove stop words and perform stemming on words that do not trigger lexical variation rules discussed in Section 4.4. To ensure correct mapping between preprocessed query terms and words in the index file, all word processing methods in this stage should be consistent with those in the document indexing stage. Unlike stemming or lowercase conversion, which changes the original terms, lexical variation rules merely expand terms from the original terms. Thus, we do not have to apply lexical variation rules in both stages. In most cases, the rules are applied in the input query processing stage, rather than in document indexing, to avoid generating too many terms. After stop words have been removed, several synonym generation techniques are applied to the query. A detailed explanation of these techniques can be found in Section 4. The result of this stage is an expanded query containing possible synonyms and lexical variants of the terms.

**Indri Query Construction**

After processing the original input query and applying several expansion schemes, the query construction module uses these terms to construct an Indri query. The details are given in Section 5.

**IR Searching**

This module sends the Indri query generated by the query construction module to the Indri engine and records the results following TREC's result file format, which is then assessed by TREC's evaluation tools [1].

## 3. INDRI RETRIEVAL ENGINE

In this section, we briefly introduce the Indri retrieval engine used in our system. The following definition and description are taken from [7]. The retrieval model of Indri combines the language modeling [3, 9] and inference network [12] approaches to information retrieval. The resulting model allows structured queries similar to those used in INQUERY to be evaluated using language modeling estimates within the network, rather than tf.idf estimates. As in the original inference network framework, documents are ranked according to the probability $P(I \mid D, \alpha, \beta)$. More details about the inference network framework can be found in [5] and [12].

### 3.1 Document Representation

In Indri, documents are represented as multisets of binary feature vectors. The features can be any interesting binary observation of the underlying text. We shall discuss the features used to represent documents in our model in the following subsection. We assume that there is a single feature vector for each position within a document. The

approach moves away from modeling text towards modeling features of text. Hereafter, we refer to such models as language models.

## 3.2 Language Models

We estimate a multiple-Bernoulli model for each document, as in Model *B* of [6]. This resolves the theoretical issues encountered in [5].

We take a Bayesian approach and impose a multiple-Beta prior over the model ($\theta$). The Beta is chosen for simplicity, as it is the conjugate prior to the Bernoulli distribution. Thus, $P(D \mid \theta) \sim MultiBernoulli(\theta)$ and $P(\theta \mid \alpha, \beta) \sim MultiBeta(\alpha, \beta)$. Our belief at node $\theta$ is then

$$P(\theta_i \mid D, \alpha, \beta) = \frac{P(D \mid \theta_i) P(\theta_i \mid \alpha_i, \beta_i)}{\int_{\theta_i} P(D \mid \theta_i) P(\theta_i \mid \alpha_i, \beta_i)}$$
$$= Beta(\#(r_i, D) + \alpha_i, |D| - \#(r_i, D) + \beta_i)$$

for each *i*, where $\#(r_i, D)$ is the number of occurrences that feature $r_i$ is set to 1 in document *D*'s multiset of feature vectors.

We estimate the for the entire text of a document. Additionally, we estimate specific models for a number of XML fields. To do so, we treat all the text in a document that appears within a given field as a pseudo-document. For example, a model can be estimated for all the text that appears within the TITLE tags of a document.

## 3.3 Representation Nodes

The $r_i$ nodes correspond to document features that can be represented in an Indri structured query. Indri implements all of the terms and proximity operators available in INQUERY, including single terms, #*N* (ordered window *N*), and #uwN (unordered window *N*). See [5] for more details. The belief at a given representation node is computed as

$$P(r_i \mid D, \alpha, \beta) = \int_{\theta_i} P(r_i \mid \theta_i) P(\theta_i \mid D, \alpha_i, \beta_i) = \frac{\#(r_i, D) + \alpha_i}{|D| + \alpha_i + \beta_i}.$$

Furthermore, selecting $\alpha_i = \mu P(r_i \mid C)$ and $\beta_i = \mu(1 - P(r_i \mid C))$, we attain the multiple-Bernoulli model equivalent to the multinomial model's Dirichlet smoothing [14] estimate

$$P(r_i \mid D, \alpha, \beta) = \frac{\#(r_i, D) + \mu P(r_i \mid C)}{|D| + \mu},$$

where $\mu$ acts as a tunable smoothing parameter.

## 3.4 Query Nodes

The query node operators are soft probabilistic operators. All of the query operators available in INQUERY are also available in Indri, with the addition of a weighted version of the #and operator named #wand. The operators are #combine (same as #and), #weight (same as #wand), #or, #not, #sum, #wsum, and #max. See [5] for the details of how beliefs are computed at the query nodes.

# 4. INPUT QUERY PROCESSING

## 4.1 Input Query Types

In the 2005 ad hoc retrieval task, the query topics are more structured than the mostly free-form topics in the 2004 track. The purpose of this approach is to provide systems with better defined queries for finding genomics information. Therefore, systems can make better use of other resources, such as ontologies or databases.

As in 2004, the topics this year are collected from real biologists. Instead of soliciting free-form topics, biologists are provided with generic templates and asked to express information needs into the templates. The generic topic templates (GTTs) are derived from an analysis of the topics in the 2004 track and other known biologist information needs.

As in 2004, there are 50 topics in 2005. We reached closure on 5 GTTs, each of which have 10 instances, giving a total of 50 topics. The five GTTs are listed below. The semantic types in each GTT are underlined. For some semantic types, more than one instance is allowed. The five GTTs are:

1. Find articles describing standard methods or protocols for doing some sort of experiment or procedure.
2. Find articles describing the role of a gene involved in a given disease.
3. Find articles describing the role of a gene in a specific biological process.
4. Find articles describing interactions (e.g., promote, suppress, inhibit, etc.) between two or more genes in the function of an organ or in a disease.
5. Find articles describing one or more mutations of a given gene and its biological impact.

## 4.2 Shallow Parsing

Shallow parser, a basic tool for detecting phrase boundary, is widely used in natural language processing and information retrieval applications. Although words provide essential information for information retrieval, chunked phrases provide more accurate concepts. Therefore, we use phrasal expression for Indri query construction., and phrases are processed with ontological and database expansion to find alias/synonym phrases. In our system, we use the SPECIALIST Lexicon Text tools [8] as our shallow parser.

## 4.3 Removal of Stop Words

Stop words such as "in" and "at" increases ambiguity of an input query. It has little effect on finding relevant document. We therefore remove all common stop words.

### 4.4 Lexical Variation

In biomedical information retrieval, hyphens, Greek letters, and numerical characters are the main types of lexical variants. Therefore, we develop several heuristic rules to generate a term's lexical variants. For example, given a term IL 2, we generate IL-2 and IL2 as its lexical variants. By lexical variation, documents containing these variants can be found. Detailed information of lexical variation can be found in [13].

### 4.5 Hyphen and Number Handling

Biologists usually add hyphen after a gene family name to stand for members in that family. For example, the IL-2 gene is a member of the IL gene family. Suppose a gene *G-i* is in the input query, and it belongs to the gene family *G*. We would prefer retrieving documents containing *G-i* but not *G*. However, when *G* is in the input query, all documents containing all *G*-family genes are to be retrieved. Lexical variation schemes cannot cover such variation.

For all document and query terms, we replace a hyphen character with a space character and insert a space between the alphabetic character and the numeric character. In addition, we treat the terms separated by the replacement and insertion of white space as a single phrase.

For example, after the replacement step, IL-2 will become "IL 2". Then we use the phrasal expression of Indri, "#1(IL 2)" to represent that "IL 2" is a phrase. By performing these hyphen handling steps, using "IL gene" can retrieve documents containing different IL-family genes, while using IL-2 can only retrieve documents containing IL-2.

### 4.6 Synonym and Acronym Expansion

A biomedical term and its synonyms/acronyms are usually used interchangeably in biomedical literature. We search for synonyms and acronyms by querying two databases: AcroMed and EuGenes. Phrases chunked by the shallow parser are also sent to retrieve their synonyms or acronyms.

In our experience, correct chucking (shallow parsing) is extremely important for expansion. To guarantee the quality of expanded terms/phrases, we use exact match to compare the original term/phrase and expanded terms/phrases.

### 4.7 Expansion Validation

Expansion validation is another way to ensure the quality of expanded query terms. Main types of incorrect expansion include that terms do not exist (i.e., terms do not correspond to any biomedical named entities) and the expanded terms have different meaning. Even though query expansion can improve the IR performance, incorrect expanded terms may dramatically degrade the IR performance. Therefore, expansion validation is necessary. Another advantage of applying validation is that the processing speed of IR engines can be accelerated by eliminating incorrect expanded terms..

We use very simple validation procedure to filter out inappropriate expanded terms described in [2]. First, we send the original query term to Indri for retrieving 100 relevant documents. Then we filter out the expanded terms that do not occur in those documents. Remaining expanded terms are used for constructing the Indri query in next section.

## 5. INDRI QUERY CONSTRUCTION

The Indri query of our system comprises three parts. We use Indri's #weight operator and assign a distinct weight to each part. We describe the meaning of each part in the following subsections.

### 5.1 Original Terms and Lexical Variants

This is the fundamental part that comprises all terms generated by input query processing. In addition, we consider that the lexical variants are also important for finding relevant documents. The expression of this part of the Indri query is

$$\#\text{combine}\ (q_1,\ q_2\ \ldots,\ q_i,\ \ldots,\ q_m,\ v_1,\ v_2\ \ldots,\ v_j,\ \ldots,\ v_n),$$

where $q_i$ stands for a original query term and $v_j$ stands for a lexical variant.

### 5.2 Sliding Window Limitation

One effective heuristic to determine if a document *D* is relevant to a query *Q* is to examine whether all terms in *Q* appear within a limited window. In our experience, window size of 45 yields the best results. Also, using an unordered window is better than an ordered window. The expression of this part of the Indri query is

$$\#\text{uw45}\ (q_1,\ q_2\ \ldots,\ q_i,\ \ldots,\ q_m,\ v_1,\ v_2\ \ldots,\ v_j,\ \ldots,\ v_n),$$

where $q_i$ stands for a original query term and $v_j$ stands for a lexical variant.

### 5.3 Synonyms

It is very important to exploit expansion and chunking information and encode it into an effective Indri query. We apply shallow parsing to find phrases, and use the methods described in Section 4 to find their expansions. Since in the Synonym and Acronym Expansion stage, our system derives synonyms and acronyms for each phrase $p_i$ in the input query, we can use <> operator to make Indri use the same probability for $p_i$ and its synonym phrase $s_{ij}$. The expression of this part in the Indri query is

$$\#\text{combine}\ (\langle p_1, s_{11}, s_{12}, \ldots\rangle,\ \langle p_2, s_{21}, s_{22}, \ldots\rangle,\ \ldots,\ \langle p_n, s_{n1}, s_{n2}, \ldots\rangle),$$

where $p_i$ stands for a phrase in the original query and $s_{ij}$ stands for an expansion phrase of $p_i$.

### 5.4 Combination of the Three Parts

According to [7], we use the #weight operator to combine the three parts of expressions. The weight of each part is automatically tuned according to the development set (10 topics) provided by TREC 2005. We choose the weights

**Table 1. Performance comparison of all configurations**

| | Shallow Parsing | Hyphen Handling | Query Expansion (GTT 1) | Query Expansion (GTT 2-5) | MAP | Recall |
|---|---|---|---|---|---|---|
| baseline | | | | | 0.200 | 2641/4584 |
| SP | √ | | | | 0.203 | 2668/4584 |
| SP+HH | √ | √ | | | 0.236 | 2930/4584 |
| IASLRun1 | √ | √ | | √ | **0.245** | 3419/4584 |
| IASLRun2 | √ | √ | √ | √ | 0.232 | **3431**/4584 |

**Table 2. Performance comparison with the average median score in each topic template**

| | IASL | Average Median Score | (+/-) |
|---|---|---|---|
| GTT 1 | 0.165 | 0.178 | -0.013 |
| GTT 2 | 0.280 | 0.265 | +0.015 |
| GTT 3 | 0.229 | 0.218 | +0.011 |
| GTT 4 | 0.326 | 0.224 | +0.102 |
| GTT 5 | 0.235 | 0.216 | +0.019 |
| Overall | 0.245 | 0.22 | +0.025 |

that achieve the highest MAP value for the development set. In our experience, the weights of the first, second, and third part are 1.5, 0.3, and 1.6, respectively.

# 6. EXPERIMENTS

## 6.1 Documents

The document collection for the ad hoc retrieval task was a 10-year subset of MEDLINE. The subset of MEDLINE used for the track consisted of 10 years of completed citations from the database inclusive from 1994 to 2003. Records were extracted using the Date Completed (DCOM) field for all references in the range of 19940101 - 20031231.

## 6.2 Topics

As described in Section 4.1, there are five topic templates in this year's task, each of which has 10 instances. We note that topics in template 1 are more like natural language queries and topics in other templates are more like structural queries, in which the main concepts such as gene or disease names are given.

## 6.3 Evaluation Metrics

The primary evaluation measure for the task was the mean average precision (MAP). Results were calculated using the trec_eval program, a standard scoring system for TREC. In addition to analyzing MAP, we also assessed recall, the portion of retrieved and relevant documents in the total relevant documents.

## 6.4 Results

In Table 1, we compare the performances of all configurations. The baseline configuration uses all terms in the input query to construct the Indri query. We can see that using shallow parsing only improves the MAP and recall slightly. This may be because, in this year's task, queries of Template 2, 3, 4, and 5 are structural. That is, the original query already contains the phrasal information. After adding hyphen handling, the MAP and recall significantly increases by 3.3% and 262 documents, respectively. This result demonstrates that our hyphen handling is effective. When we only apply query expansion in queries of GTT 2, 3, 4, and 5, our system achieves the best MAP. However, when we apply query expansion to GTT 1, the MAP decreases, but the recall increases slightly. We believe this is because queries in GTT 1 are in natural language form. Even if the stop words are removed, there are still some terms and phrases that are not very informative for finding relevant documents.

In Table 2, we compare our system's performance with that of the median system for all topic templates. We can see that, in GTT1, our system's performance is slightly worse than the average median score. In GTT 2, 3, and 5, our system outperforms the median system slightly. In GTT 4, our system outperforms the average median score by 10.2% of MAP. We think this is because in GTT4, two gene names and one disease name are provided. Our query construction rules use this information effectively and make the MAP of GTT 4 higher than other GTTs, while the median system only achieves a similar performance to other GTTs.

# 7. DISCUSSIONS

In this section, we study some cases in which our system achieves lower scores than the median scores to further understand the reasons for system errors.

**Topic 106:**

**"Chromatin IP (Immuno Precipitations) to isolate proteins that are bound to DNA in order to precipitate the proteins out of the DNA".**

In Topic 106, the abstracts retrieved by our system only contain single keywords such as "chromatin" and "precipitation". However, the relevant abstracts should contain the phrase "Chromatin IP" or "Chromatin Immuno Precipitation". This may be caused by incorrect chunking.

**Topic 108:**

**"Methods for identifying in vivo protein-protein interactions in time and space in the living cell".**

For Topic 108, it is even harder to determine the important or proper phrases because the statement of this topic is quite general and no specific named entities as included.

**Topic 118:**

**Information describing the role(s) of a gene involved in a disease.**

**Gene: "Transforming growth factor-beta1 (TGF-beta1)"**

**Disease: " Cerebral Amyloid Angiopathy (CAA)"**

In Topic 118, most relevant abstracts contain "Transforming growth factor-beta1 (TGF-beta1)", but rarely contain "Cerebral Amyloid Angiopathy (CAA)". We observe that the relevant abstracts contain CAA-related terms (such as amyloid, Alzheimer's disease, amyloid beta-peptide…etc.) instead of "Cerebral Amyloid Angiopathy (CAA)". For this reason, more ontological resources and knowledge bases should be used in the query expansion method.

## 8. CONCLUSION

For the TREC 2005 Genomics Track, we develop several methods of term expansion and variation. We exploit databases and ontologies such as AcroMed and EuGenes to find aliases, abbreviations, acronyms, and synonyms. We also apply heuristic lexical variation rules described in [13].

Since the domain-specific knowledge in the genomics domain can be used to produce a very large number of possible synonyms for initial query terms, techniques to validate these expansions most found. We presented a simple way that can be used to verify if an expansion term appears in the context.

We particularly tackle the hyphen problem in biomedical terms in both query processing and document indexing that significantly improves the baseline result. We appropriately use Indri query language to organize the original terms, expanded terms, and their relationships. The experiment results shows that hyphen handling is more effective than other query processing schemes in this paper. When the focus terms/phrases are salient, such as topics in GTT 2, 3, 4, and 5, query expansion is more effective. In natural language queries (GTT 1), however, the expansion should be performed more carefully because there are some general terms that cannot help us distinguish relevant documents from irrelevant ones. In the future, we will use deep semantic analysis of the input query and knowledge oriented query expansion to improve the performance of biomedical information retrieval.

## 9. ACKNOWLEDGEMENT

We are grateful for the support of National Science Council under GRANT NSC94-2752-E-001-001.

## 10. REFERENCES

1. Buckley, C. trec eval IR evaluation package.
2. Büttcher, S., Clarke, C.L.A. and Cormack, G.V., Domain-Specific Synonym Expansion and Validation for Biomedical Information Retrieval. in *TREC-2004*, (2004).
3. Croft, W.B. and Lafferty, J. *Language Modeling for Information Retrieval*. Kluwer, 2003.
4. Gilbert, D.G. euGenes: A Eukaryote Genome Information System. *Nucleic Acids Research, 30* (1). 145-148.
5. Metzler, D. and Croft, W.B. Combining the language model and inference network approaches to retrieval. *Info. Proc. and Mgt., 40* (5). 735-750.
6. Metzler, D., Lavrenko, V. and Croft, W.B., Formal multiple bernoulli models for language modeling. in *SIGIR 2004*, (2004), 540–541.
7. Metzler, D., Strohman, T., Turtle, H. and Croft, W.B., Indri at TREC 2004: Terabyte Track. in *TREC 2004*, (2004).
8. NLM. The SPECIALIST Text Tools *http://specialist.nlm.nih.gov/TextTools.html*, 2003.
9. Ponte, J.M. and Croft, W.B., A language modeling approach to information retrieval. in *SIGIR 1998*, (1998), 275-281.
10. Pustejovsky, J., Castaño, J., Cochran, B., Kotecki, M., Morrell, M. and Rumshisky., A., Linguistic Knowledge Extraction from Medline: Automatic Construction of an Acronym Database. in *Medinfo-2001*, (2001).
11. Salton, G. *Introduction to Modern Information Retrieval*. McGraw-Hill, 1983.
12. Turtle, H. and Croft, W.B. Evaluation of an inference networkbased retrieval model. *TOIS, 9* (3). 187–222.
13. Wang, Y.-C., Hung, H.-C., Wu, C.-W., Tsai, T.-H., Liu, C.-L. and Hsu, W.-L., An Empirical Study of Lexical Variation Methods for Biomedical Information Retrieval. in *NCS-2005*, (Taiwan, 2005).
14. Zhai, C. and Lafferty, J. A study of smoothing methods for language models applied to information retrieval. *ACM Trans. Inf. Syst., 22* (2). 179-214.