# Exploring Semantic Constraints for Document Retrieval

**Hua Cheng, Yan Qu, Jesse Montgomery, David A. Evans**
Clairvoyance Corporation
5001 Baum Blvd., Suite 700, Pittsburgh, PA 15213, U.S.A.
{H.Cheng, Y.Qu, J.Montgomery, dae}@clairvoyancecorp.com

## Abstract

In this paper, we explore the use of structured content as semantic constraints for enhancing the performance of traditional term-based document retrieval in special domains. First, we describe a method for automatic extraction of semantic content in the form of attribute-value (AV) pairs from natural language texts based on domain models constructed from a semi-structured web resource. Then, we explore the effect of combining a state-of-the-art term-based IR system and a simple constraint-based search system that uses the extracted AV pairs. Our evaluation results have shown that such combination produces some improvement in IR performance over the term-based IR system on our test collection.

## 1 Introduction

The questions of where and how sophisticated natural language processing techniques can improve traditional term-based information retrieval have been explored for more than a decade. A considerable amount of work has been carried out that seeks to leverage semantic information for improving traditional IR. Early TREC systems such as INQUERY handled both natural language and semi-structured queries and tried to search for constraint expressions for country and time etc. in queries (Croft et al., 1994). Later work, as discussed in (Strzalkowski et al., 1996), has focused on exploiting semantic information at the word level, including various attempts at word-sense disambiguation, e.g., (Voorhees, 1998), or the use of special-purpose terms; other approaches have looked at phrase-level indexing or full-text query expansion. No approaches to date, however, have sought to employ semantic information beyond the word level, such as that expressed by attribute-value (AV) pairs, to improve term-based IR.

Attribute-value pairs offer an abstraction for instances of many application domains. For example, a person can be represented by a set of attributes such as name, date-of-birth, job title, and home address, and their associated values; a house has a different set of attributes such as address, size, age and material; many product specifications can be mapped directly to AV pairs. AV pairs represent domain specific semantic information for domain instances.

Using AV pairs as semantic constraints for retrieval is related to some recent developments in areas such as Semantic Web retrieval, XML document retrieval, and the integration of IR and databases. In these areas, structured information is generally assumed. However, there is abundant and rich information that exists in unstructured text only. The goal of this work includes first to explore a method for automatically extracting structured information in the form of AV pairs from text, and then to utilize the AV pairs as semantic constraints for enhancing traditional term-based IR systems.

The paper is organized as follows. Section 2 describes our method of adding AV annotations to text documents that utilizes a domain model automatically extracted from the Web. Section 3 presents two IR systems using a vector space model and semantic constraints respectively, as well as a system that combines the two. Section 4 describes the data set and topic set for evaluating the IR systems. In Section 5, we compare the performance of the three IR systems, and draw initial conclusions on how NLP techniques can improve traditional IR in specific domains.

## 2 Domain-Driven AV Extraction

This section describes a method that automatically discovers attribute-value structures from unstructured texts, the result of which is represented as texts annotated with semantic tags.



*Proceedings of the Workshop on How Can Computational Linguistics Improve Information Retrieval?*, pages 33–40,
Sydney, July 2006. 

We chose the digital camera domain to illustrate and evaluate the methodology described in this paper. We expect this method to be applicable to all domains whose main features can be represented as a set of specifications.

### 2.1 Construction of Domain Model

A domain model (DM) specifies a terminology of concepts, attributes and values for describing objects in a domain. The relationships between the concepts in such a model can be heterogeneous (e.g., the link between two concepts can mean inheritance or containment). In this work, a domain model is used for establishing a vocabulary as well as for establishing the attribute-value relationship between phrases.

For the digital camera domain, we automatically constructed a domain model from existing Web resources. Web sites such as epinions.com and dpreview.com generally present information about cameras in HTML tables generated from internal databases. By querying these databases and extracting table content from the dynamic web pages, we can automatically reconstruct the databases as domain models that could be used for NLP purposes. These models can optionally be organized hierarchically. Although domain models generated from different websites of the same domain are not exactly the same, they often share many common features.

From the epinions.com product specifications for 1157 cameras, we extracted a nearly comprehensive domain model for digital cameras, consisting of a set of attributes (or features) and their possible values. A portion of the model is represented as follows:

```
{Digital Camera}
    <Brand> <Price> <Lens>
{Brand}
    (57) Canon
    (33) Nikon
{Price} $
    (136) 100 – 200
    (100) >= 400
{Lens}
    <Optical Zoom> <Focus Range>
{Optical Zoom} x
    (17) 4
    (3) 2.5
{Focus Range} in., ”
    (2) 3.9 – infinity
    (1) 12 – infinity
```

In this example, attributes are shown in curly brackets and sub-attributes in angle brackets. Attributes are followed by possible units for their numerical values. Values come below the attributes, headed by their frequencies in all specifications. The frequency information (in parentheses) is used to calculate term weights of attributes and values.

Specifications in HTML tables generally do not specify explicitly the type restrictions on values (even though the types are typically defined in the underlying databases). As type restrictions contain important domain information that is useful for value extraction, we recover the type restrictions by identifying patterns in values. For example, attributes such as *price* or *dimension* usually have numerical values, which can be either a single number (“$300”), a range (“$100 - $200”), or a multi-dimensional value (“4 in. x 3 in. x 2 in.”), often accompanied by a unit, e.g., *$* or *inches*, whereas attributes such as *brand* and *accessory* usually have string values, e.g., “Canon” or “battery charger”.

We manually compile a list of units for identifying numerical values, which is partially domain general. We identify range and multi-dimensional values using such patterns as “A – B”, “A to B”, “less than A”, and “A x B”, etc. Numerical values are then normalized to a uniform format.

### 2.2 Identification of AV Pairs

Based on the constructed domain model, we can identify domain values in unstructured texts and assign attribute names and domains to them. We focus on extracting values of a domain attribute. Attribute names appearing by themselves are not of interest here because attribute names alone cannot establish attribute-value relations. However, identification of attribute names is necessary for disambiguation.

The AV extraction procedure contains the following steps:

1. Use MINIPAR (Lin, 1998) to generate dependency parses of texts.
2. For all noun phrase chunks in parses, iteratively match sub-phrases of each chunk with the domain model to find all possible matches of attribute names and values above a threshold:
    - A chunk contains all words up to the noun head (inclusive);
    - Post-head NP components (e.g., PP and clauses) are treated as separate chunks.
3. Disambiguate values with multiple attribute assignments using the sentence context, with a preference toward closer context based on dependency.

4. Mark up the documents with XML tags that represent AV pairs.

Steps 2 and 3 are the center of the AV extraction process, where different strategies are employed to handle values of different types and where ambiguous values are disambiguated. We describe these strategies in detail below.

**Numerical Value**

Numerical values are identified based on the unit list and the range and multi-dimensional number patterns described earlier in Section 2.1. The predefined mappings between units and attributes suggest attribute assignment. It is possible that one unit can be mapped to multiple attributes. For example, "x" can be mapped to either optical zoom or digital zoom, both of which are kept as possible candidates for future disambiguation. For range and multi-dimensional numbers, we find all attributes in the domain model that have at least one matched range or multi-dimensional value, and keep attributes identified by either a unit or a pattern as candidates. Numbers without a unit can only be matched exactly against an existing value in the domain model.

**String Value**

Human users often refer to a domain entity in different ways in text. For example, a camera called "Canon PowerShot G2 Black Digital Camera" in our domain model is seldom mentioned exactly this way in ads or reviews, but rather as "Canon PowerShot G2", "Canon G2", etc. However, a domain model generally only records full name forms rather than their all possible variations. This makes the identification of domain values difficult and invalidates the use of a trained classifier that needs training samples consisting of a large variety of name references.

An added difficulty is that web texts often contain grammatical errors and incomplete sentences as well as large numbers of out-of-vocabulary words and, therefore, make the dependency parses very noisy. As a result, effectiveness of extraction algorithms based on certain dependency patterns can be adversely affected.

Our approach makes use of the more accurate parser functionalities of part-of-speech tagging and phrase boundary detection, while reducing the reliance on low level dependency structures. For noun phrase chunks extracted from parse trees, we iteratively match all sub-phrases of each chunk with the domain model to find matching attributes and values above a threshold. It is often possible to find multiple AV pairs in a single NP chunk.

Assigning domain attributes to an NP is essentially a classification problem. In our domain model, each attribute can be seen as a target class and its values as the training set. For a new phrase, the idea is to find the value in the domain model that is most similar and then assign the attribute of this nearest neighbor to the phrase. This motivates us to adopt K Nearest Neighbor (KNN) (Fix and Hodges, 1951) classification for handling NP values. The core of KNN is a similarity metric. In our case, we use word editing distance (Wagner and Fischer, 1974) that takes into account the cost of word insertions, deletions, and substitutions. We compute word editing distance using dynamic programming techniques.

Intuitively, words do not carry equal weights in a domain. In the earlier example, words such as "PowerShot" and "G2" are more important than "digital" and "camera", so editing costs for such words should be higher. This draws an analogy to the metric of Inverse Document Frequency (IDF) in the IR community, used to measure the discriminative capability of a term in a document collection. If we regard each value string as a document, we can use IDF to measure the weight of each term in a value string to emphasize important domain terms and de-emphasize more general ones. The normalized cost is computed as:

$$\log(TN / N) / \log(TN)$$

where TN is the total number of values for an attribute, and N is the number of values where a term occurs. This equation assigns higher cost to more discriminative terms and lower cost to more general terms. It is also used to compute costs of terms in attribute names. For words not appearing in a class the cost is 1, the maximum cost.

The distance between a new phrase and a DM phrase is then calculated using word editing cost based on the costs of substitution, insertion, and deletion, where

$Cost_{sub} = (Cost_{DM} + Cost_{new}) / 2$
$Cost_{ins} = Cost_{new}$
$Cost_{del} = Cost_{DM}$
$Cost_{edit} = \min(Cost_{sub}, Cost_{ins}, Cost_{del})$

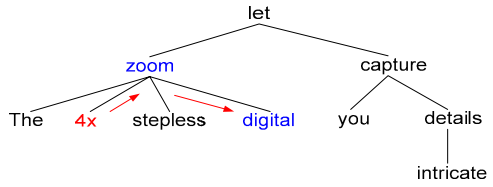
where $Cost_{DM}$ is the cost of a word in a domain value (i.e., its normalized IDF score), and $Cost_{new}$

is that of a word in the new phrase. The cost is also normalized by the larger of the weighted lengths of the two phrases. We use a threshold of 0.6 to cut off phrases with higher cost.

For a phrase that returns only a couple of matches, the similarity, i.e., the matching probability, is computed as 1 - $Cost_{edit}$; otherwise, the similarity is the maximum likelihood of an attribute based on the number of returned values belonging to this attribute.

**Disambiguation by Sentence Context**

The AV identification process often returns multiple attribute candidates for a phrase that needs to be further disambiguated. The words close to the phrase usually provide good indications of the correct attribute names. Motivated by this observation, we design the disambiguation procedure as follows. First we examine the sibling nodes of the target phrase node in the dependency structure for a mention of an attribute name that overlaps with a candidate. Next, we recursively traverse upwards along the dependency tree until we find an overlap or reach the top of the tree. If an overlap is found, that attribute becomes the final assignment; otherwise, the attribute with the highest probability is assigned. This method gives priority to the context closest (in terms of dependency) to the target phrase. For example, in the sentence "The 4x stepless digital zoom lets you capture intricate details" (parse tree shown below), "4x" can be mapped to both optical zoom and digital zoom, but the sentence context points to the second candidate.




## 3 Document Retrieval Systems

This section introduces three document retrieval systems: the first one retrieves unstructured texts based on vector space models, the second one takes advantage of semantic structures constructed by the methods in Section 2, and the last one combines the first two systems.

### 3.1 Term-Based Retrieval (S1)

Our system for term-based retrieval from unstructured text is based on the CLARIT system, implementing a vector space retrieval model (Evans and Lefferts, 1995; Qu et al., 2005). The CLARIT system identifies terms in documents and constructs its index based on NLP-determined linguistic constituents (NPs, subphrases and words). The index is built upon full documents or variable-length subdocuments. We used subdocuments in the range of 8 to 12 sentences as the basis for indexing and scoring documents in our experiments.

Various similarity measures are supported in the model. For the experiments described in the paper, we used the dot product function for computing similarities between a query and a document:

$$sim\ (Q, D) = \sum_{t \in Q \cap D} W_Q(t) \cdot W_D(t).$$

where $W_Q(t)$ is the weight associated with the query term $t$ and $W_D(t)$ is the weight associated with the term $t$ in the document $D$. The two weights were computed as follows:

$$W_D(t) = TF_D(t) \cdot IDF(t).$$
$$W_Q(t) = C(t) \cdot TF_Q(t) \cdot IDF(t)$$

where IDF and TF are standard inverse document frequency and term frequency statistics, respectively. $IDF(t)$ was computed with the target corpus for retrieval. The coefficient $C(t)$ is an "importance coefficient", which can be modified either manually by the user or automatically by the system (e.g., updated during feedback).

For term-based document retrieval, we have also experimented with pseudo relevance feedback (PRF) with various numbers of retrieved documents and various numbers of terms from such documents for query expansion. While PRF did result in improvement in performance, it was not significant. This is probably due to the fact that in this restricted domain, there is not much vocabulary variation and thus the advantage of using query expansion is not fully realized.

### 3.2 Constraint-Based Retrieval (S2)

The constraint-based retrieval approach searches through the AV-annotated document collection based on the constraints extracted from queries. Given a query $q$, our constraint-based system scores each document in the collection by comparing the extracted AV pairs with the constraints in $q$. Suppose $q$ has a constraint $c(a, v)$ that restricts the value of the attribute $a$ to $v$, where $v$ can be either a concrete value (e.g., 5 megapixels) or a range (e.g., less than $400). If $a$

is present in a document $d$ with a value $v'$ that satisfies $v$, that is, $v' = v$ if $v$ is a concrete value or $v'$ falls in the range defined by $v$, $d$ is given a positive score $w$. However, if $v'$ does not satisfy $v$, then $d$ is given a negative score $-w$. No mention of $a$ does not change the score of $d$, except that, when $c$ is a string constraint, we use a back-off model that awards $d$ a positive score $w$ if it contains $v$ as a substring. The final score of $d$ given $q$ is the sum of all scores for each constraint in $q$, normalized by the maximum score for $q$: $\sum_{i=1}^{n} c_i w_i$ , where $c_i$ is one of the $n$ constraints specified in $q$ and $w_i$ its score.

We rank the documents by their scores. This scoring schema facilitates a sensible cutoff point, so that a constraint-based retrieval system can return 0 or fewer than top N documents when a query has no or very few relevant documents.

### 3.3 Combined Retrieval (S3)

Lee (1997) analyzed multiple post-search data fusion methods using TREC3 ad hoc retrieval data and explained the combination of different search results on the grounds that different runs retrieve similar sets of relevant documents, but different sets of non-relevant documents. The combination methods therefore boost the ranks of the relevant documents. One method studied was the summation of individual similarities, which bears no significant difference from the best approach (i.e., further multiply the summation with the number of nonzero similarities).

Our system therefore adopts the summation method for its simplicity. Because the scores from term-based and constraint-based retrieval are normalized, we simply add them together for each document retrieved by both approaches and re-rank the documents based on their new scores. More sophisticated combination methods can be explored here, such as deciding which score to emphasize based on the characterizations of the queries, e.g., whether a query has more numerical values or string values.

## 4 Experimental Study

In this section, we describe the experiments we performed to investigate combining terms and semantic constraints for document retrieval.

### 4.1 Data Sets

To construct a domain corpus, we used search results from craigslist.org. We chose the "for sale – electronics" section for the "San Francisco Bay Area". We then submitted the search term "digital camera" in order to retrieve advertisements. After manually removing duplicates and expired ads, our corpus consisted of 437 ads posted between 2005-10-28 and 2005-11-07. A typical ad is illustrated below, with a small set of XML tags specifying the fields of the title of the ad (*title*), date of posting (*date*), ad body (*text*), ad id (*docno*), and document (*doc*). The length of the documents varies considerably, from 5 or 6 sentences to over 70 (with specifications copied from other websites). The ads have an average length of 230 words.

```
<doc>
  <docno>docid519</docno>
  <title>brand new 12 mega pixel digital camera</title>
  <date>2005-11-07, 8:27AM PST</date>
  <text>
    BRAND NEW 12 mega pixel digital camera.....only $400,
      -12 Mega pixels (4000x3000) Max Resolution
      -2.0 Color LCD Display
      -8x Digital Zoom
      -16MB Built-In (internal) Memory
      -SD or MMC card (external) Memory
      -jpeg picture format
    ALSO COMES WITH SOFTWARE & CABLES
  </text>
</doc>
```

The test queries were constructed based on human written questions from the Digital Photography Review website (www.dpreview.com) Q&A forums, which contain discussions from real users about all aspects of digital photography. Often, users ask for suggestions on purchasing digital cameras and formulate their needs as a set of constraints. These queries form the base of our topic collection.

The following is an example of such a topic manually annotated with the semantic constraints of interest to the user:

```
<topic>
  <id>1</id>
  <query>
    I wanted to know what kind of Digital SLR camera I should buy. I plan to spend nothing higher than $1500. I was told to check out the Nikon D70.
  </query>
  <constraint>
    <hard: type = "SLR" />
    <hard: price le $1500 />
    <soft: product_name = "Nikon D70" />
  </constraint>
</topic>
```

In this example, the user query text is in the *query* field and the manually extracted AV constraints based on the domain model are in the *constraint* field. Two types of constraints are distinguished: hard and soft. The hard constraints must be satisfied while the soft constraints can be relaxed. Manual determination of hard vs. soft constraints is based on the linguistic features in the text. Automatic constraint extraction goes one step beyond AV extraction for the need to identify relations between attributes and values, for example, "nothing higher than" indicates a "<=" relationship. Such constraints can be extracted automatically from natural text using a pattern-based method. However, we have yet to produce a rich set of patterns addressing constraints. In addition, such query capability can be simulated with a form-based parametric search interface.

In order to make a fair comparison between systems, we use only phrases in the manually extracted constraints as queries to system S1. For the example topic, S1 extracted the NP terms "SLR", "1500" and "Nikon D70". During retrieval, a term is further decomposed into its subterms for similarity matching. For instance, the term "Nikon D70" is decomposed into subterms "Nikon" and "D70" and thus documents that mention the individual subterms can be retrieved.

For this topic, the system S2 produced annotations as those shown in the *constraint* field.

Table 1 gives a summary of the distribution statistics of terms and constraints for 30 topics selected from the Digital Photography Review website.

| | Average | Min | Max |
|---|---|---|---|
| No. of terms | 13.2 | 2 | 31 |
| No. of constraints | 3.2 | 1 | 7 |
| No. of hard constraints | 2.4 | 1 | 6 |
| No. of soft constraints | 0.8 | 0 | 3 |
| No. of string constraints | 1.4 | 0 | 5 |
| No. of numerical constraints | 1.8 | 0 | 4 |

**Table 1: Summary of the distribution statistics of terms and constraints in the test topics**

## 4.2 Relevance Judgments

Instead of using human subjects to give relevance judgments for each document and query combination, we use a human annotator to mark up all AV pairs in each document, using the GATE annotation tool (Cunningham *et al*, 2002). The attribute set contains the 40 most important attributes for digital cameras based on automatically computed term distributions in our data set. The inter-annotator agreement (without annotator training) as measured by Kappa is 0.72, which suggests satisfactory agreement.

Annotating AV pairs in all documents gives us the capability of making relevance judgments automatically, based on the number of matches between the AV pairs in a document and the constraints in a topic. This automatic approach is reasonable because unlike TREC queries which are short and ambiguous, the queries in our application represent very specific information needs and are therefore much longer. The lack of ambiguity makes our problem closer to boolean search with structured queries like SQL than traditional IR search. In this case, a human assessor should give the same relevance judgments as our automatic system if they follow the same instructions closely. An example instruction could be "a document is relevant if it describes a digital camera whose specifications satisfy at least one constraint in the query, otherwise it is not relevant" (similar to the narrative field of a TREC topic).

We specify two levels of relevance: strict and relaxed. *Strict* means that all hard constraints of a topic have to be satisfied for a document to be relevant to the topic, whereas *relaxed* means that at least half of the hard constraints have to be satisfied. Soft constraints play no role in a relevance judgment. The advantage of the automatic approach is that when the levels of relevance are modified for different application purposes, the relevance judgment can be recomputed easily, whereas in the manual approach, the human assessor has to examine all documents again.

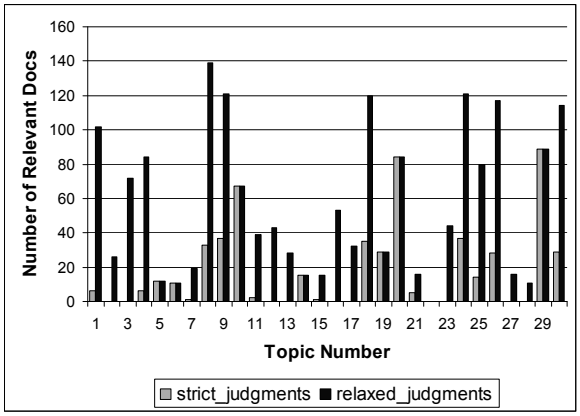



**Figure 1: Distribution of relevant documents across topics for relaxed and strict judgments**

Figure 1 shows the distributions of the relevant documents for the test topic set. With strict judgments, only 20 out of the 30 topics have relevant documents, and among them 6 topics

have fewer than 10 relevant documents. The topics with many constraints are likely to result in low numbers of relevant documents. The average numbers of relevant documents for the set are 57.3 for relaxed judgments, and 18 for strict judgments.

## 5 Results and Discussion

Our goal is to explore whether using semantic information would improve document retrieval, taking into account the errors introduced by semantic processing. We therefore evaluate two aspects of our system: the accuracy of AV extraction and the precision of document retrieval.

### 5.1 Evaluate AV Extraction

We tested the AV extraction system on a portion of the annotated documents, which contains 253 AV pairs. Of these pairs, 151 have string values, and the rest have numerical values.

The result shows a prediction accuracy of 50.6%, false negatives (missing AV pairs) 35.2%, false positives 11%, and wrong predications 3%. Some attributes such as *brand* and r*esolution* have higher extraction accuracy than other attributes such as *shooting mode* and *dimension*. An analysis of the missing pairs reveals three main sources of error: 1) an incomplete domain model, which misses such camera Condition phrases as "minor surface scratching"; 2) a noisy domain model, due to the automatic nature of its construction; 3) parsing errors caused by free-form human written texts. Considering that the predication accuracy is calculated over 40 attributes and that no human labor is involved in constructing the domain model, we consider our approach a satisfactory first step toward exploring the AV extraction problem.

### 5.2 Evaluate AV-based Document Retrieval

The three retrieval systems (S1, S2, and S3) each return top 200 documents for evaluation. Figure 2 summarizes the precision they achieved against both the relaxed and strict judgments, measured by the standard TREC metrics (*PN* – Precision at N, *MAP* – Mean Average Precision, *RP* – R-Precision)[1]. For both judgments, the combined system S3 achieved higher precision and recall than S1 and S2 by all metrics. In the case of recall, the absolute scores improve at least nine percent. Table 2 shows a pairwise comparison of the systems on three of the most meaningful TREC metrics, using paired T-Test; statistically significant results are highlighted. The table shows that the improvement of S3 over S1 and S2 is significant (or very nearly) by all metrics for the relaxed judgment. However, for the strict judgment, none of the improvements are significant. The reason might be that one third of the topics have no relevant documents in our data set. This reduces the actual number of topics for evaluation. In general, the performance of all three systems for the strict judgment is worse than that for the relaxed, likely due to the lower number of relevant documents for this category (averaged at 18 per topic), which makes it a harder IR task.

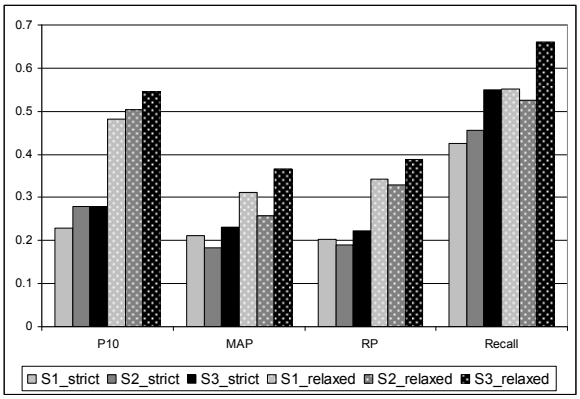



**Figure 2: System performance as measured by TREC metrics, averaged over all topics with non-zero relevant documents**

| *Paired T-Test (p)* | | *P10* | *AP* | *RP* |
|---|---|---|---|---|
| *strict* | (S1,S2) | .22 | .37 | .65 |
| | (S2,S3) | 1 | **.004** | .10 |
| | (S1,S3) | .17 | .48 | .45 |
| *relaxed* | (S1,S2) | .62 | .07 | .56 |
| | (S2,S3) | .056 | **<.0001** | **.0007** |
| | (S1,S3) | **.04** | **.02** | **.03** |

**Table 2: Paired T-Test (with two-tailed distribution) between systems over all topics**

The constraint-based system S2 produces higher initial precision than S1 as measured by P10. However, semantic constraints contribute less and less as more documents are retrieved. The performance of S2 is slightly worse than S1 as measured by AP and RP, which is likely due to errors from AV extraction. None of the metrics is statistically significant.

[1] Precision at N is the precision at N document cutoff point; Average Precision is the average of the precision value obtained after each relevant document is retrieved, and Mean Average Precision is the average of AP over all topics; R-Precision is the precision after R documents have been retrieved, where R is the number of relevant documents for the topic.

Topic-by-topic analysis gives us a more detailed view of the behavior of the three systems. Figure 3 shows the performance of the systems measured by P10, sorted by that of S3. In general, the performance of S1 and S2 deviates significantly for individual topics. However, the combined system, S3, seems to be able to boost the good results from both systems for most topics. We are currently exploring the factors that contribute to the performance boost.

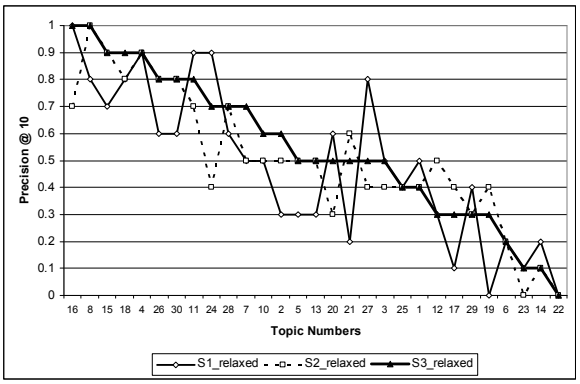



**Figure 3: Precision@10 for relaxed judgment**

A closer look at topics where S3 improves significantly over S1 and S2 at P10 reveals that the combined lists are biased toward the documents returned by S2, probably due to the higher scores assigned to documents by S2 than those by S1. This suggests the need for better score normalization methods that take into account the advantage of each system.

In conclusion, our results show that using semantic information can improve IR results for special domains where the information need can be specified as a set of semantic constraints. The constraint-based system itself is not robust enough to be a standalone IR system, and has to be combined with a term-based system to achieve satisfactory results. The IR results from the combined system seem to be able to tolerate significant errors in semantic annotation, considering that the accuracy of AV-extraction is about 50%. It remains to be seen whether similar improvement in retrieval can be achieved in general domains such as news articles.

## 6 Summary

This paper describes our exploratory study of applying semantic constraints derived from attribute-value pair annotations to traditional term-based document retrieval. It shows promising results in our test domain where users have specific information needs. In our ongoing work, we are expanding the test topic set for the strict judgment as well as the data set, improving AV extraction accuracy, analyzing how the combined system improves upon individual systems, and exploring alternative ways of combining semantic constraints and terms for better retrieval.

## References


Hamish Cunningham, Diana Maynard, Kalina Bontcheva and Valentin Tablan. 2002. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*. Philadelphia.

Bruce Croft, James Callan and John Broglio. 1994. TREC-2 Routing and Ad-Hoc Retrieval Evaluation Using the INQUERY System. In *Proceedings of the 2nd Text Retrieval Conference*, NIST Special Putlication 500-215.

David A. Evans and Robert Lefferts. 1995. CLARIT-TREC experiments. *Information Processing and Management*, 31(3), 385-395.

E. Fix and J. Hodges. 1951. *Discriminatory Analysis, Nonparametric Discrimination: Consistency Properties*. Technical Report, USAF School of Aviation Medicine, Texas.

Joon Ho Lee. 1997. Analyses of Multiple Evidence Combination. Proceedings of the 20th Annual International ACM-SIGIR Conference on Research and Development in Information Retrieval. Philadelphia, pp. 267-276.

Dekang Lin. 1998. Dependency-based Evaluation of MINIPAR. *Workshop on the Evaluation of Parsing Systems*, Spain.

Yan Qu, David A. Hull, Gregory Grefenstette, David A. Evans, et al. 2005. Towards Effective Strategies for Monolingual and Bilingual Information Retrieval: Lessons Learned from NTCIR-4. *ACM Transactions on Asian Language Information Processing*, 4(2): 78-110.

Robert Wagner and Michael Fischer. 1974. The String-to-string Correction Problem. *Journal of the Association for Computing Machinery*, 21(1):168-173.

Tomek Strzalkowski, Louise Guthrie, Jussi Karigren, Jim Leistensnider, et al. 1996. Natural language information retrieval, TREC-5 report. In *Proceedings of the 5th Text Retrieval Conference (TREC-5)*, pp. 291-314, Gaithersburg, Maryland.

Ellen Voorhees. 1998. Using WordNet for text retrieval. In *Wordnet, an Electronic Lexical Database*, pp 285-303. The MIT Press.