Intelligent Natural Language Processing for Media Data Query

Vincent Lum[1], Daniel A. Keim[2], Kyung-Chang Kim[3]

Department of Computer Science[4]
Systems Engineering, Chinese University
Shatin, Hong Kong

Abstract

This paper proposes an approach to manage multimedia data like sound and graphics that can now be stored and processed in today's computer systems. Present database management systems have not yet been developed to manage these data. To store these data in a computer is simple; to retrieve them in a manner similar to what we do with the traditional alphanumeric data is an extremely difficult task. The major problem is that this kind of data has very complex semantics that the computer cannot understand. This paper outlines an architecture of an enhanced database management system which accepts multimedia data with attached content descriptions in natural language form. These descriptions will be used for multimedia data retrieval based on their contents as specified by the queries. Methods to match the natural language descriptions to the queries will be dealt with.

1. Introduction

Although technological advances have made the storage and retrieval of multimedia (or simply media) data like graphics and images practical in real applications, progresses in database management systems (DBMS) have not kept pace with the advances and no conventional DBMS deals with this class of data effectively. Traditional DBMSs do not deal with media data because, until now, their large sizes are prohibitive to be stored in the normal databases.

Interests in developing a DBMS system with media data management capability existed for a number of years [1, 3, 13, 17, 18]. None of them, however, really dealt with the problem of processing media data on the basis of their contents. Unlike alphanumeric data, the semantics of which is very limited, media data is rich in semantics. Automatic deduction of semantics from media data is not possible today or in the near future.

In [12] we proposed to enhance the media data with natural language descriptions. Queries on their contents will also be specified in natural language. The system will match the specification in the queries with the stored descriptions. Unfortunately, as it has been long known to researchers in natural language processing, this method is intrinsically extremely complex. Because there are many ways to say the same thing and because a slight difference in syntactical structure or even the same syntactical structure with a slight change in very few words means an entirely different thing, it is very difficult to match one specification with another.

Unlike question-answering systems, content retrieval of media data only needs approximate answers as in text or information retrieval. But we cannot use text retrieval techniques where mainly keywords are used because this approach does not allow the users the ability to specify much semantics in their queries. The application of this method therefore will not be possible to retrieve information highly matching the users' needs in most cases, even when only approximate matching is desired. For example, suppose we want to retrieve the photos in the databases that show police brutality. We may specify in the query, 'police, hit, people', or something like that. But actually we want to say, 'police continuously hitting victim(s) even when victims have already been subdued or injured and cannot resist or can no longer command their bodily functions'. To be able to retrieve information with high pertinence, it is clear that the system must have a high degree of intelligence and approximate people's actions fairly well.

In the rest of the paper, we will first briefly discuss the architecture of our system and some key components and their functions. We will then discuss how we intend to

---

1. Current address: Systems Engineering Department, Chinese University, Shatin, Hong Kong. email: vlum@cucs93.cs.cuhk.hk, Fax: (852) 603 5505.
2. Current address: Institute of Computer Science, Leopoldstr. 11B, Ludwig-Maximilian-University, Munich, Germany.
email: daniel@dbs.informatik.uni-muenchen.de, Fax: (49) 89 2180 5246.
3. Current address: Computer Science Department, Hong-Ik University, Seoul, Korea. Fax: (822) 320 1122.
4. The work was supported in part by the Naval Postgraduate School, the Chinese University, NOSC, Direct Funding and the German Scholarship Foundation.

match the query specifications with the stored descriptions. Finally, we propose a method to approximate the way people go about to perform such a task by using object hierarchies that naturally exist in the real world. We will argue that such an approach is intrinsically the obvious way when faced with such a task.

2. System Architecture

The multimedia database management system (MDBMS) as shown in Figure 2-1 is mainly composed of a query processor, a conventional DBMS, a media object manager, a description manager, a matcher, and a parser. The media object manager is further decomposed into different media specific managers like image manager and sound manager. The conventional DBMS component, which in this system is a relational DBMS, handles the normal, traditional alphanumeric data.

Media data generally comes with registration data and raw data. As raw data is nothing more than pure bit strings, registration data, which gives information to the system to tell it how to handle the raw data, (for example, if it is image data, we need to know the color encoding, pixel depth, resolution, etc.), is needed to process these bit strings. As shown in Figure 2-2, we enhance the media data with a natural language description called the description data. Unlike registration and raw data, description data is generally redundant and will be required to be given by someone who knows the media data. Such a practice is not really out of place as it is generally done during data entry when users add information to the data at that time.

The descriptions are processed by a parser with the use of a dictionary to determine the domain specific semantics and turn the natural language descriptions into a set of Prolog predicates or literals [4, 5, 6, 12]. Similarly the natural language specification from the queries are processed by the parser as well. We shall further discuss about the parser later in the next section.

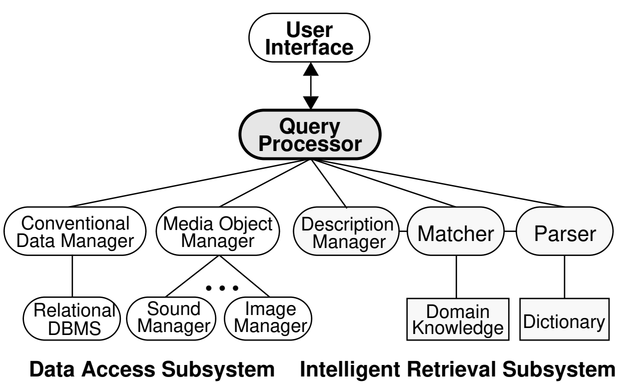


Figure 2-1: Architecture of MDBMS System

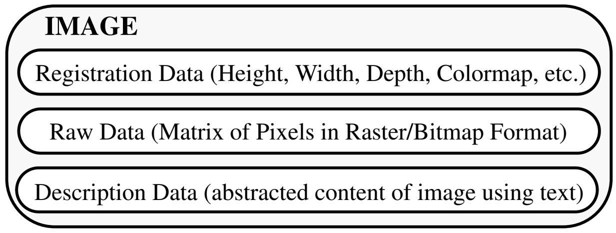


Figure 2-2: Example for the Media Data Format

The matcher will attempt to match the query specification with the stored descriptions. Unfortunately since there are numerous ways to say the same thing in natural language, exact matching is practically impossible. Approximate matching will be done. In fact, it is actually desirable to have the system retrieving the approximate matches as well as the exact matches. This is argued to be the case because in fact users rarely know or want precise matches in this kind of applications. We do, however, need to classify the degree of approximation. The subsequent sections will discuss more on this topic.

Both, parser and matcher naturally must use the lexicon and the dictionary information to do their tasks.

Media data generally is very large and cannot be stored with the normal data in the relational database. We use the abstract data type concept to allow the users to enhance the relations to include media data via keys acting as pointers. Just as standard data types have their set of operations like addition, subtraction, substring and concatenation of strings, a set of operations must be defined for the media data. Exactly as in the alphanumeric case, the set of legitimate operations depends on the type of media data. For example, the operation of 'reduce volume' has meaning to sound data but not image data. Naturally, since no database language is defined for the operation of media data, current database interfaces must be modified. SQL being the standard DB language and the most popular one was enhanced to accommodate media data management. All of these are discussed in [8, 12] and are beyond the scope of this paper.

Since the actual raw data of a media object is stored as a file and this is not tied to the normal alphanumeric data, we need to find some way to have these two categories of data internally connected, even though externally the users see them all as data. What we have decided to do is to embed the media data into the data of the traditional model [15]. The overall idea is that media data is basically accessed through the normal data and the data language reflects this structure. Thus, the general query format for all the data becomes

```
SELECT  <data & media data>
FROM    <relational data tables>
[WHERE  <conditions with/without media data>];
```

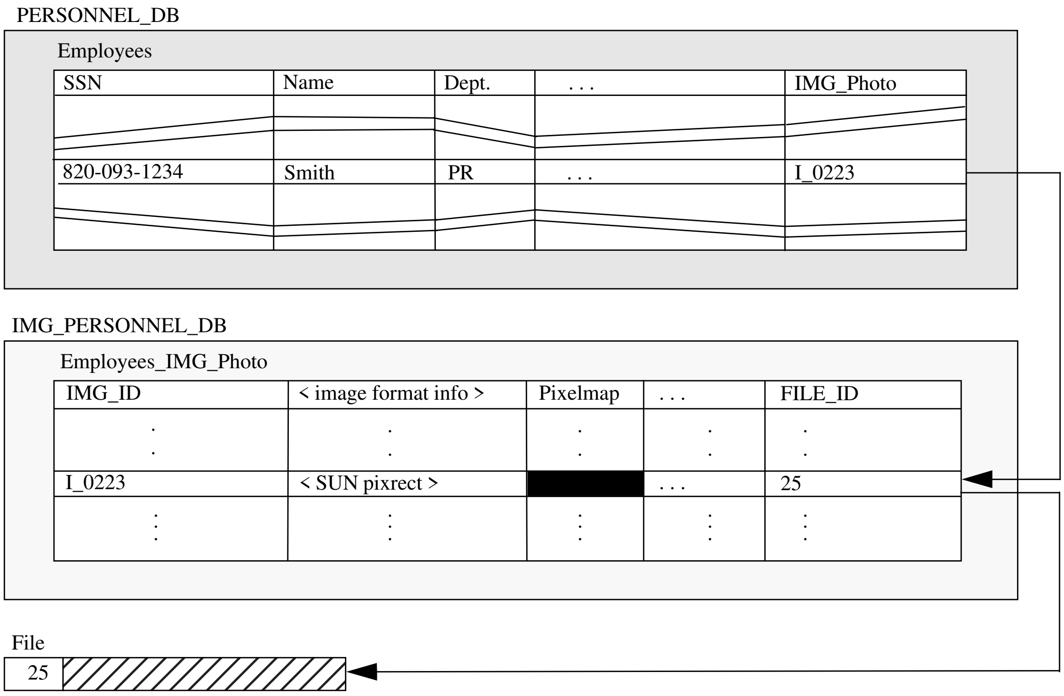


Figure 2-3: Embedding of Photographs into a Relational Database PERSONNEL_DB

This becomes clearer when we illustrate it by an example. Let us consider the example of having the media data of IMAGE type included in the Employees relation consisting attributes SSN, Name, Dept, . . . in the PERSONNEL database with normal alphanumeric data. The media attribute, Photo, is prefixed with IMG to become IMG_Photo, showing that it is of image type. The content in this column of IMG_Photo is actually the id's to another table where the information like colormap, pixel depth, etc. is stored. In turn there exists an attribute or column entry in this added table which actually contains the file id's of the image files. Figure 2-3 illustrates the case just presented.

When a query is received, the query processor determines if there is any reference to media data. If not, then the whole query is routed to the relational DBMS. If there is media data involved, the query is decomposed into multiple subqueries. The subqueries without media data are sent to the relational DBMS and the ones involving media data are sent to the appropriate object manager. Intermediate results are kept and are recomposed to obtain the final result.

Thus, the query

```
SELECT    . . . , IMG_Photo
FROM      Employees
WHERE     Dept='CS' and
          IMG_Photo contains 'has a beard';
```

will be decomposed into two subqueries, one to be processed with the Employees table and the other with the Employees_IMG_Photo table, and their results will be recomposed by a third SQL statement.

3. Processing Natural Language Descriptions

As stated before, media data descriptions are parsed by the parser to become Prolog predicates and literals. When a query requests for data based on the media data's contents, as in the above example of searching for employees with the description 'has a beard', the matcher will attempt to match the query specification with all the descriptions already stored in the database. Media data whose descriptions are matched will be retrieved. This implies exact match, in the sense of logical implication between those descriptions and the specification in the query. Generally it does not happen as there are many ways to say the same thing in natural language and their syntactic and semantic structures do not imply exact matches among them. For example, a person having a beard and a person growing a beard though having subtle differences can be said to match each other, but are likely to be missed. The more complex the descriptions and the query specification, the more likely the non-matches will occur.

Yet, because the specification of descriptions, whether in the query or in the original media data description, is at best an approximation of what one thinks, exact matching is rarely necessary. This situation occurs in real application environments as in information retrieval of the library science. Approximate matching is therefore not only desirable but becomes a must, if the system is to be practical.

In order for the parser to understand the natural language specifications, it is necessary to store the various routine and non-routine information like parts of speech, stems and affixes, and their meanings in the lexicon. Meanings or semantics can be domain independent and/or dependent. The organization of the lexicon must be able to contain both. Further, as the lexicon at any time cannot be complete, the structure of the lexicon must be such that it can grow and shrink dynamically. For any real application, the lexicon can be expected to be fairly big. If the system is to be run efficiently, the parser must be able to get to the words fast. Getting to the words quickly depends on the space, or the size, needed to store the lexicon. But of course both access time and space depend on the method to represent the lexicon information in the system. These issues of access time, space, and the representation of the lexicon are important but cannot be discussed in further depth here because of the overwhelming amount of details and variations. They are analyzed in [6].

To perform approximate matching one must have a model to store the information. Knowledge representation has been studied for years but there is no general consensus or solution to the problem [2, 7, 10, 11, 14]. It is generally believed that a good knowledge representation model should possess the quality of generality to be able to represent very diversified real world situations, extensibility to accommodate the varying amount of information from one case to another, and decomposability to allow the representation to be subset into small components when needed. The semantic network model having most of the desirable elements can be thus used.

However, matching one semantic network to another is a very difficult problem. What we propose to do is to construct semantic network composed of major nodes of nouns and actions, and connect them to other minor or descriptor elements that are tied to these nouns and actions. Figure 3-1 shows the example of the description 'Stephen drives a (one) blue car'.

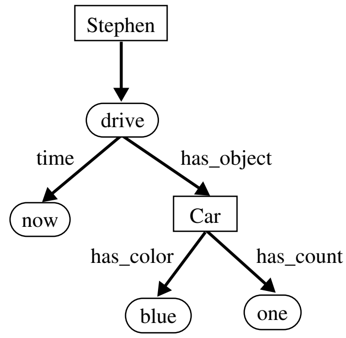


Figure 3-1: Semantic Net Representation of ‘Stephen drives a (one) blue car’

When a query is received and parsed, the subject noun will be first matched and its action will then be matched with the targeted net. If matching is found, the system proceed further to match the next node(s). If not, approximation will be used. Let us for the time being not go into this part any further, but proceed further with the assumption that matches at the nodes are found.

Having matched the noun and action nodes does not mean we have a match. We have to see if the minor or descriptor nodes like the ones connected to the 'car' node are right. Suppose the query wants 'Stephen drives a red car'. Then we have the major match but not the minor match (the color is wrong) and we call this an approximate match because it is believed that these two are highly relevant to each other and the user probably has strong interest to retrieve this target object in spite of the minor mismatch.

4. Enhancements for Better Matching

In addition, we need to handle cases where the noun or action nodes do not really match the query specification. In that case, we go to the knowledge base where we store the domain-dependent knowledge of the nouns and actions (verbs) in the form of object hierarchies. The nouns representing the objects are stored in one and the verbs in another hierarchy. It is well known that nouns representing objects can be arranged in hierarchical form. Shank and his colleagues have shown that verbs can also be reduced to primitive actions and therefore can be represented as hierarchies as well [16]. These hierarchies thus represent the concept of generalization and specialization. Figure 3-2 (a) and (b) show examples of such hierarchies.

To use these hierarchies to perform approximate matching, the system will first use the information in the lexicon to find the synonyms to locate the node specified by the query. It will then go down to all the descendants from that node. Since all the descendants represent the subcategories of the node, it satisfies the query specification even though it is not the term defined in the query. For example, a sedan is a car and is a descendant of car. If there is a description that says 'Stephen drives a blue sedan', then this of course is a match.

But suppose we do not find any such kind of descriptions after searching down the tree. In this case, the system will go to the parent node toward the root of an hierarchy. It then

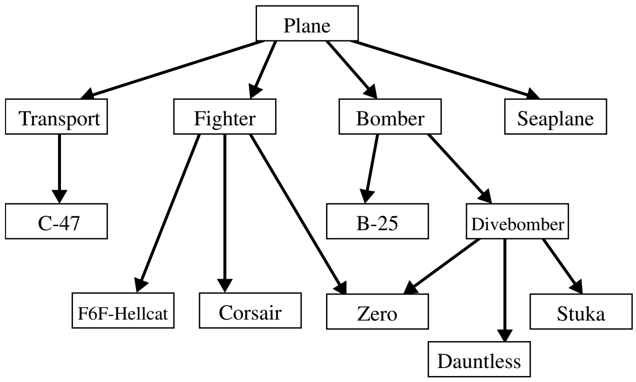
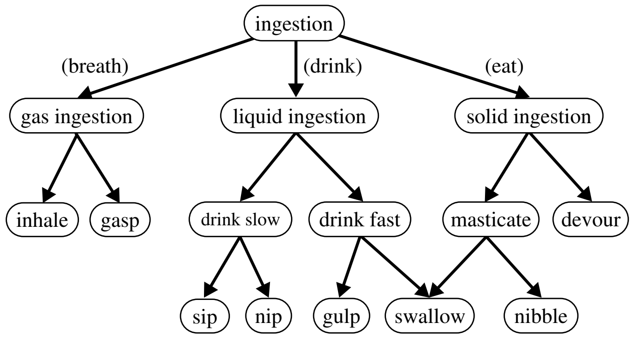


Figure 3-2 (a): Generalization Hierarchy of a Plane



(b): Generalization Hierarchy of Ingestion

tries to match the descriptions using the new node as if that node was specified by the query.

However, when the searches go toward the root, information is less pertinent, from the user's point of view, because it is now a more general category than what the user states. Thus, a negative weight is assigned each time when we move to the parent. An algorithm is applied to assess the aggregated weights. If the final assessment of the description of a media data element satisfies the threshold stated by the user, it will be retrieved. Otherwise, it will be rejected.

It should be self apparent that the same strategy can be applied to both the noun or the verb hierarchy. Further, we can assign different weights to the different hierarchies so that a user can state whether the subject matter or the action is of higher or lower interest.

As stated before, there are many ways to say the same thing with natural language. Since our intention is to find pertinent but not precisely matching data, we perform the process of normalization, as done in some natural language processing. For example, a red car, a car of red color, and a car with a red body will all be stored in the same way. These are referred to as canonical or normal forms and the process is called normalization. Rather than attempting to understand all the implications in the natural language, a practical approach is adopted by storing the rules of normalization. The parser will then perform substitution to arrive at the result. Similarity grouping for adjectives and adverbs can also be thus achieved.

It can be argued that the variations of these forms of descriptions actually carry different semantics. Indeed this is generally true. However, if we were to keep the variations as distinct entities, what we will come up is to have no match rather than approximate match. We argue that in real applications user would prefer a close approximate match than exact match and the system should provide such help.

Unfortunately normalization is a very difficult process and little previous work has been found. Consider the example of a 'high frequency transistor oscillator'. It is not difficult to see that 'high' goes with 'frequency', although in certain cases this kind of association is not so obvious. But one can associate 'high frequency' with the transistor, or with the oscillator. Normalizing it one way or another makes much difference and can make the difference of locating or not locating the relevant objects in the database. The paper [19] discusses this topic and its application to approximate matching in more detail.

Another question comes to mind concerning what weights to use when we deviate from the user's specifications. This is indeed a difficult subject and there is not now, (and we believe there will never be), a concrete or absolute assignment possible for all situations. It is believed that we need the user's help to guide the system to locate the approximate matches by assigning the different weights to the different parts of the query to allow the system to come up with some answers. The paper [9] discusses this subject in depth and suggests how to assign weights to the searches in the hierarchies.

Thus, to help the system to find data better matching the user's need is to have the user specify not only the thresholds but also the importance of the different parts of the query specification. For example, the user can indicate the importance of subject and action by assigning weights to these terms. The system will calculate the combined weights of the descriptions and retrieve only those exceeding the requirements.

5. Conclusion

The method as proposed above allows users to retrieve media data like graphics and images on the basis of the data's contents. The approach not only can retrieve exact matches but more importantly the approximate ones as well. It separates the whole process into two major levels, one at the semantic net level without going to the object-oriented hierarchies and one using the noun and verb hierarchies. In the former case, matches on the minor items like the descriptors such as adjectives and adverbs may produce perturbations even when the major nodes like the subject nouns and verbs have already been matched exactly. If desired by the user, searches based on the hierarchies can proceed to locate grosser approximations. Such kind of approximations can vary little or greatly from the user's query specification. However, it is proposed that the user should interact with the system to let the system set the threshold in accordance with the user's input. Retrieved objects can then be ranked according to their threshold values calculated by the system.

Calculating threshold values and weights when deviating from exact match is a most difficult problem. Although we have some intuitive assignments, such assignments can be far off from the user's expectation. We are continuing to look into this, although it is believed that there will never be a generally accepted algorithm, regardless what we can come up with. This is proven to be the case in information retrieval. Thus, we think that the best we can do is to have a system which works directly with the user to set these values. This is entirely different than the practice of the traditional database applications, but is the accepted practice in information retrieval. Perhaps, when media data is to be incorporated into the traditional data in the database systems, users will have no choice but to accept it if they want to treat the media data similarly to the traditional alphanumeric data.

Normalization as stated in the paper is another difficult problem to solve. Here it seems more hopeful because there is some commonly accepted style of specification where one can definitely say that specification in one manner is very closely the same as another. Nevertheless, normalization does lose some meaning and cannot be used in certain applications. In the application as specified, we think it can be acceptable.

While we do not have the whole system constructed as proposed, we have most pieces operational. The part that has not been finished is the smart parser. Work is continuing to define a smarter parser that can search on hierarchies, assign weights to the search strategies, and perform the normalization process. The portion to design and build an interface to interact with the user to set the values and thresholds has also not been done but it is not believed to be a difficult problem. The difficulty lies in the definition of an algorithm for such a purpose that would satisfy most, if not all, of the users.

References

[1] Agrawal R. et al: 'Ode View: The Graphical Interface to Ode', Proc. ACM SIGMOD Conf., Atlantic City, 1990.

[2] Allen J.: 'Natural Language Understanding', Benjamin/Cummings, 1987.

[3] Christoloukas S. et al: 'Multimedia Document Presentation, Information Extraction, and Documentation Formation in MINOS: A Model and a System', ACM TOOIS, vol. 4, no. 4, 1986.

[4] Dulle J.: 'The Scope of Descriptive Captions for Use in a Multimedia Database System', Master's thesis, Computer Science, Naval Postgraduate School, Monterey, CA, 1990.

[5] Guglielmo E.: 'Natural Language Processing of Captions for Retrieving Multimedia Data', Proc. Conf. on Applied NL Processing, Trento, Italy, 1992.

[6] Guglielmo E., Rowe N. and Lum V.: 'Natural Language Lexicon Design in Support of Multimedia Content Search', Technical Memorandum NAWCWPNS7202, 1992.

[7] Hahn U.: 'Topic Parsing: Account for Text Macro Structure in Full Text Analysis', Information Processing and Management, vol. 26, 1990.

[8] Holtkamp B. and Lum V.: 'Integration of Alphanumeric and Media Data', Technical Memorandum 48, Lehrstuhl für Software-Technologie, University of Dortmund, Germany.

[9] Kim K. C. and Lum V.: 'Towards Intelligent Data Retrieval in Multimedia Databases', Technical Report NPSCS-91-009, Naval Postgraduate School, Monterey, CA, 1991.

[10] Lewis D. D. et al: 'Language-Oriented Information Retrieval', Int. Journal of Intelligent Systems, vol. 4, 1989.

[11] Luger G. and Stubblefield W.: 'Artificial Intelligence and the Design of Expert Systems', Benjamin/Cummings, 1989.

[12] Lum V. and Meyer-Wegener K.: 'An Architecture for a Multimedia Database Management System Supporting Content Search', Springer Verlag's Lecture Notes on Computing Science, Advances in Computing and Information - ICCI '90, Eds: A.G. Akl, F. Fiala, W.W. Koczkodaj, 1991.

[13] Masunaga Y.: 'Multimedia Databases: A Formal Framework', Proc IEEE CS Office Automation Symp., Gaithersburg, MD, 1987.

[14] Mauldin M. L.: 'Retrieval Performance in Ferret - A Conceptual Information Retrieval System', Proc 14th Int. Conf. on Research and Development in Information Retrieval, ACM/SIGIR, 1991.

[15] Pongsuwan W.: 'Design and Implementation of a Multimedia DBMS: Retrieval Management', Master's thesis, Computer Science, Naval Postgraduate School, Monterey, CA, 1990.

[16] Schank R.: 'Conceptual Information Processing', North-Holland, 1984.

[17] Woelk D. and Kim W.: 'Multimedia Information Management in an Object-Oriented Database Management System', Proc. VLDB, Brighton, England, 1987.

[18] Woelk D. et al: 'An Object-Oriented Approach to Multimedia Databases', Proc. ACM SIGMOD, Washington, DC, 1986.

[19] Wong K. P. and Lum V.: 'Approximate Matching of Natural Language Text to Queries', in preparation.