Adaptive User Interfaces for Information Extraction

Peter Vanderheyden and **Robin Cohen**
Department of Computer Science, University of Waterloo
Waterloo, Ontario, Canada

Abstract

Information extraction systems distill specific items of information from large quantities of text. Current literature in the field of information extraction focuses largely on how different language models and machine learning approaches can improve system accuracy, with little mention of the role of interaction between the user and the system. In this paper, we explore many opportunities for interaction and adaptation during information extraction, and propose a system design to investigate these opportunities further.

Introduction

Much has been written about the flood of information currently available, and the challenge of locating relevant items amid that flood. Several areas of applied research — information extraction (IE), information retrieval (IR), and knowledge discovery in text (KDT) — seek to address this challenge. We are specifically interested in the field of information extraction. In this paper, we discuss opportunities for interaction between user and system, and for adaptation by the system to facilitate this interaction. We then develop objectives for designing a system that supports the user effectively during information extraction.

We motivate this discussion by observing that little is mentioned in current literature about the interaction between an information extraction system and a user. In contrast, for example, information retrieval literature discusses interactions between users and computer-based systems as well as between human librarians and library patrons (Twidale & Nichols 1998).

The following will be an ongoing example to which we refer throughout the paper: Person P is concerned that there has been a high number of terrorism cases reported recently[1], and is looking for more information on the matter. What P really wants to know (P's "global information need") is the following: "Is terrorism on the rise?" System S is able to accept many different kinds of queries, and then to search for information in a corpus of documents. How should P and S interact to satisfy P's information need?

[1]The reader may insert any other topic in place of "terrorism" (*e.g.*, management successions, cancer, rap music).

Information Extraction: What Is It?

Information extraction involves "extraction or pulling out of pertinent information from large volumes of texts" (MUC-7 1999). It is often defined by example, in terms of the results the system would return for a given query, and often contrasted with what an information retrieval system might return for a similar query.

The query-result perspective

Information retrieval generally interprets a query as a string of unrelated words or word phrases[2], matching it against the words in the corpus documents. Output from the IR system consists of either whole documents or segments, deemed relevant to the query words. Thus, for P's question "Is terrorism on the rise?", if the corpus contains a document that mentions a 'rise' in 'terrorism', and assuming P has confidence in the document, such a document will have answered P's question.

An IE system takes as its input a user's query expressed (or expressable) as a sentence template, containing unfilled slots, and relations between those slots. For example, a query to identify terrorism events contained in a document could be written as:

> "A terrorism event occurred in which `{GroupA}` `{ActB}` `{GroupB}`, and `{ActC}` `{ObjectC}`."

The system then processes each document in a corpus and, for each occurrence in the document of an event that adequately fills the query template, the system returns the filled template (*e.g.*, `{ActB}`="kidnapped", `{GroupB}`="10 civilians", *etc.*). Given P's question, converted into an appropriate template form, the system would ideally return all terrorism events mentioned in the corpus. Thus one possible scenario is that, if the IR system failed to answer P's question satisfactorily, the IE system could be used as a follow-up approach (Gaizauskas & Robertson 1997). The data returned by the IE system could be analyzed (the number of events counted) and plotted as a function of time, to

[2]Most current information retrieval systems consider only minor variations (*e.g.*, morphological) on the query words, though natural language approaches to IR exist as well (Lewis & Sparck Jones 1996).

see whether terrorism cases were indeed increasing in frequency.

Knowledge discovery in text (Feldman & Dagan 1995) begins with the kind of well-structured data often returned by an information extraction system. The system and user then cooperate, the system using statistical calculations and the user using intuition and knowledge of the world, to find correlative patterns among data categories, possibly defining new categories in the process. For example, a KDT system might analyze the terrorist event data returned by the IE query mentioned earlier, to identify patterns of occurrence over the a range of years. Thus S would compute the answer to P's question on its own, rather than relying on a document (that P trusts) in the corpus containing the answer. Finding information often does involve trying one approach, getting partial answers, integrating this new knowledge, and trying again with the same approach or with a different one.

The natural language perspective

Some degree of language understanding is required to recognize how the terms in the query template are related to one another. The documents in the corpus may contain either unstructured natural language (*e.g.*, news wire articles, in the MUCs; *e.g.*, (MUC-6 1995) and (MUC-7 1999)) or semi-structured text (*e.g.*, formulaic reviews of restaurants or movies, with consistent structure; discussed in many of the papers in (AAAI'98-AIII 1998)). When documents contain unstructured natural language, information extraction can be seen as a natural language understanding task restricted to the domain of knowledge relevant to the user's query. The system typically contains a knowledge base of words and phrases, semantic relations, and rules of syntax in the language. How these syntax rules are expressed is very much affected by the specific language model being used (*e.g.*, finite state automata (Appelt *et al.* 1995); statistical dependency model (Weischedel 1995)).

Information extraction systems acquire additional knowledge (about the lexicon, semantics, *etc.*) during the course of the task through machine learning techniques. Typically, this involves "batch" learning — learning in large increments — with no user involvement within each batch, and the specific approach taken to will be very much dependent on the language model.

Summary: there's still something missing

A good deal of the process involved in information extraction has yet to be discussed. For example, the design of an information extraction system should consider the kinds of users, queries, and corpora that will be involved. And information extraction may occur in isolation, but in the context of other activities.

The Information Task In Context

In this section, we briefly mention some of the parameters that will vary from one task to another, from one user to another, or within the span of a single task; this is by no means an exhaustive list. It is not our intention to suggest that any one system should explicitly make use of all of these parameters. However, the designers of a system should at least be aware of them, in order to take into account any that are relevant to the specific design and intended application of a given system.

Many parameters mentioned in this section come from (Allen 1996) and are not specific to information extraction, but are relevant to various information tasks.

Information-based variations

What kind of information is the user interested in? What kind of information is contained in the documents? Different kinds of information may not be conducive to the same techniques for representation or interaction. (For example, diagrammatic information is more successfully conveyed in a diagram than as text.)

Why does the user want the information that the system is trying to find? For example, is the user:

- searching for something in particular;
- generally browsing, or exploring, to become acquainted with information available on the topic;
- integrating new information into what the user already knows; or,
- trying to gain new perspectives on what the user already know?

The user's motivation for performing the IE task will affect how she examines each document and what kind of information she focuses on. As a result, it may affect how the user prefers to interact with the system.

How many solutions to the query is the user looking for? If the user is interested in finding a small number of solutions then a less accurate (and possibly faster) procedure may suffice. This is in contrast to a user interested in all solutions that occur in the corpus, or for the "best" solution (which may require finding all solutions, then using some criterion to select the best). Similarly, is the user satisfied with finding some of the elements in the query (*i.e.*, in information extraction, this means that partially filled templates are acceptable)? If so, how many and which elements (if they are not weighted equally in importance) are required?

With respect to documents in the corpus, does it matter how documents are related to one another? For example, in the legal domain it is important to relate a given case to possible precedents (Rose & Belew 1991).

User-based variations

How does the user process different kinds of information? In any given situation, different users will vary with respect to:

- what the user perceives, identifies, or focuses on[3];

[3] Allen (1996) discusses individuals' "bounded rationality", the fact that people do not always make use of all available information — *i.e.*, real people are not ideal users.

- how the user interprets the information;
- what courses of action the user will identify, and which one the user will pursue; and,
- what the user knows, and meta-knowledge about what the user doesn't know.

As well, users will vary in how quickly they learn over the course of a given task, and in what they require in order to be confident of the results.

In addition to their abilities, users will also vary with respect to preferences — how information should be presented, how much information should be presented at once, *etc.*.

Within-task variations

Has the user firmly established the query and domain of interest? An information task can be described in several phases:

- task initiation: recognizing the information need;
- topic selection: identifying the general topic;
- prefocus exploration: investigating information on the general topic;
- focus formulation: deciding on a narrower topic;
- information collection: investigating information on the narrower topic; and,
- search closure: completing the information search.

If the query and domain are not already well defined at the beginning of the task, they will evolve over the course of the task as a result of:

- additional information learned from documents,
- constraints learned about information in documents,
- changes in user needs due to external influences.

How they evolve will depend in part on the user-based variations mentioned above, and may involve broadening, narrowing, replacing, or reorganizing individual elements within the current configuration, or the changing the scope of the entire problem definition.

Between-task variations

How is the current query related to previous queries? For example, the current query may be building on the results of earlier queries in the same task (*e.g.*, information extraction), or on the results of other tasks (*e.g.*, information retrieval, or knowledge discovery).

Similarly, a query may refer to a domain similar to that of an earlier query; some elements from the earlier task may be relevant, while others may not.

Information Extraction In Context

We now focus our discussion on information extraction specifically, and consider how this task can vary from one time to another. The model of information extraction that we have adopted is the annotation model: the results of analyzing a text are incorporated into that text as annotations (Bird & Liberman 1999); *e.g.*, semantic features are inserted into the document text as SGML markups[4].

Phases of an IE task

A typical information extraction task might be described as a series of five phases or steps. Many of these phases involve elements that are unconstrained in the specifications of the task, and these elements have been italicized; an italicized term X can be read as "some set of X", which may be the same or a different set than an X occurring elsewhere.

1. The user recognizes an information need, then defines and represents a specific problem:
   (a) the user identifies *Concepts* relevant to the query,
   (b) the user annotates *Concepts* as they occur in *Documents*, using *Annotations*.
2. The user and system explore possible strategies and generate possible intermediate solutions:
   (a) the system derives *Rules* for identifying *Concepts* in *Documents*,
   (b) the user reviews and possibly modifies directly how *Rules* are defined in the system,
   (c) the system applies *Rules*, annotates *Concepts* as they occur in *Documents*, using *Annotations*.
3. The user reviews and evaluates *Annotations* in *Documents*.
4. The user may choose to repeat the cycle if one of the following conditions holds, or to proceed to 5:
   - the system is missing a relevant concept — loop back to 1(a) and modify the representation, or loop back to 2(c) and overlook the omission;
   - the system rules are incorrect, but the relevant concept is defined correctly — loop back to 2(b) and modify the rules, or loop back to 2(c) and overlook the error;
   - the system solution is acceptable for the documents annotated so far, but additional unannotated documents remain — loop back to 2(c).
5. The system produces a report of final results.

Without any specification on how all information extraction systems should set these parameters, developers must make these decisions for each system. For example, the (italicized) elements may vary in:

- number or size — *e.g.*, how many documents does a user examine before proceeding to the next phase?
- selection — *e.g.*, which documents does the user examine? are they the same documents as in the previous phase, or novel documents?

[4]This is consistent with a number of the MUC systems (*e.g.*, FASTUS (Appelt *et al.* 1995) and Alembic Workbench (Aberdeen *et al.* 1995)), and we believe it is rich enough to also describe most other systems.

- order — *e.g.*, the system may process a next rule in some fixed sequence, or in order of how often it has been activated in the past; the order may be increasing, decreasing, or random.
- motives — *e.g.*, in phase 2(b), is the user's motive to identify new rules, to correct current rules, or to compare and contrast rules that overlap?
- control — *e.g.*, is it the system or the user who constrains each of these elements; if both, then which has priority during conflicts?
- timing — *e.g.*, when can a parameter be changed, and are these times predetermined or variable?

Proposed design. We argue that these parameters be variable through the course of an IE task, being set initially to some default value. Furthermore, while the user has priority in changing the parameter values, the system should be capable of suggesting values to the user on the basis of its evaluation of data patterns and, if the user does not object, resetting these values and thus adapting the interface to the task conditions.

Knowledge representation

An information extraction system must maintain an internal representation of the language model for processing document text. This will require such components as a lexicon of part-of-speech and semantic features for words and phrases, and a rule set for parsing syntactic and discourse structures. When documents are written in a semi-structured language, these components can be appropriately simplified.

In addition, a representation of the information contained inside a document will need to be maintained; as relevant concepts are recognized in the document, they will be added to this representation and integrated into the representation by being related to other concepts already present. Such a semantic model may be maintained either explicitly (*e.g.*, as a semantic network (Morgan *et al.* 1995)) or implicitly in the system's language components. Relations between documents may also need to be represented.

We have proposed a "tripartite" semantic model, dividing the represented information into a query model, domain model, and corpus model (Vanderheyden & Cohen 1998). Briefly, the domain model is a network of concepts relevant to the domain — the general subject area relevant to the query. The query model is a subnetwork of the domain model, containing only the concepts asked for in the query and thus expected in the results returned by the system; the query model concepts, however, may have additional constraints (*e.g.*, values may be restricted to a subset of those allowed in the corresponding domain concept). Finally, the corpus model connects the domain model to information in documents that the system may have processed (in part or in full) but that is not contained in the domain.

As new concepts are identified by the system or user, they will need to be integrated into the system's various language components and related both to the text patterns in which they appear in a document as well as to the concepts already in the semantic model.

Proposed design. At present, we allow concepts to be related to one another vertically in a simple subsumption hierarchy (containing only AND and OR links), and horizontally using simple pre-defined relations (IS-A and HAS-A); all other relations are themselves defined as concepts (*e.g.*, relation:HAS-MORE-THAN-ONE-OF might be defined as relation:HAS-A → concept:MORE-THAN-ONE → relation:IS-A). Relations could therefore be involved in subsumption hierarchies as well.

Such a semantic representation provides a clear contrast to representing concepts implicitly using a rule syntax, and allows us to investigate issues related to both interfaces and interaction methods (see below).

Interface

The IE system stores its knowledge in a variety of repositories and forms, including:

- annotated document text,
- the semantic network,
- rule syntax,
- the lexicon, or lists of known words,
- a listing of query results found so far.

Any of these can be large, with the corpus possibly containing Gigabytes of text. It is challenging, therefore, to provide the user an intuitive access to all of this data. In knowledge discovery applications, which generally involve large amounts of numerical data, graphical interfaces are common; however, it may be difficult to visualize textual information graphically. A natural alternative would be to use a natural language interface, though this also presents a number of challenges.

One approach to information retrieval (Rose & Belew 1991) involves a graphical representation of a small amount of textual information; only the concepts deemed most relevant are shown. The user is then able to manipulate these concepts, as well as query them for further information.

Proposed design. We encourage making many views of the system knowledge available and allowing the user to select the preferred view according to personal, task, and information characteristics. And we are encouraged by the interface developed by Rose & Belew in which the user had access to several of these views in parallel. User-driven navigation enables both purposeful search behaviour as well as opportunistic browsing.

Navigation in one view, as well as any modifications made to the system knowledge, require the other views to be updated accordingly. There are some factors, however, that may not allow continuous real-time updates across all views. For example, too many changes occurring at once may be distracting to the user, while also taxing the system performance.

Interaction

The interface provides the user with a window into the system knowledge, allowing the user to learn about:

- document content,
- the current definition of query and domain models,
- the current mapping from query and domain models to document text.

With this information, the user may see that the system is missing a concept, or a mapping from the concept to the text. Information extraction can be performed as a completely user-driven task, with the system performing only those functions that the user specifies.

However, the user would need to be paying attention to a great deal of information in order to perform the task in an efficient manner. Are any of the annotations in the annotated documents incorrect? Are any of the rules of syntax or discourse, or semantic relations, more general or more specific than they should be? Are the documents being examined in the optimal order?

In any task involving this much information, the system should provide as much support to the user's actions and decisions as possible. There is some very interesting and relevant work in area of "mixed-initiative" planning and system design (we discuss some of this work in (Vanderheyden & Cohen 1999a)) in which the system and user cooperate in performing a task, each contributing the skills and knowledge for which they are best suited. While the user has the advantage of a high-level knowledge of the user's own information need and (at least to some extent) of the domain, the system can have the advantage of a low-level representation for patterns in the corpus as a result of its ability to process large amounts of text very quickly. Thus, the user and system can, in general, play complementary roles in guiding the task.

The mixed initiative must be monitored. The system and user need to make the other aware of what each is doing (referred to as "context registration"), in order that conflicts could be avoided.

When the system has information has relevant information, it may choose to "take the initiative" of interacting with the user. There will be constraints on when and how often interaction should occur; not so often as to annoy the user, for example.

Proposed design. When the user is setting task parameters, the system could provide information about relevant patterns in the document text. For example, by performing some lookahead to documents not yet seen by the user, the system could suggest which documents the user should examine next and in what order.

An opportunity for the system to initiate interaction with the user is when a partial answer to the user's query is identified in a document, or the document offers a critical opportunity for deciding whether a subset of the system's current rules is accurate or not.

Where possible, the system is processing information while the user is idle, but without causing undue delay when the user is ready to resume their activity.

Learning and adaptation

While it is important that the system support various opportunities for interaction with the user, it is nevertheless impractical for user and system to interact on every decision. The ability of the system to learn about the documents that it processes as well as about the user can help to keep this amount of interaction bounded. For example, information extraction systems use learning methods in order to integrate new words into their lexicon, acquire rules for parsing syntax and discourse, and recognize relations between domain concepts (Wermter, Riloff, & Scheler 1996). A system is thus able to adapt itself autonomously to changes in some aspects of the task.

The general rule of thumb has often been that learning in larger increments (*i.e.*, batch learning) improves accuracy. However, learning in smaller increments (*i.e.*, online learning) allows better adaptivity to changing elements in the task. Furthermore, when the system is able to control how data for learning is selected (*i.e.*, active learning), the requirement for interaction with the user is often further reduced.

Caution must be taken in how the system updates its knowledge bases as a result of patterns identified in the text. The user's information need may be evolving, for example, as the user learns more about the domain, and about what kind of information the system is identifying; this kind of "evolving query" (Vanderheyden & Cohen 1999b) is a natural consequence of an information gathering task. For example, a user may have a query about terrorist activities, asking for the names of perpetrators and the locations of targets; seeing an interim output from the system prompts the user to refine the query to define terrorist activities as only involving certain kinds of weapons.

The query is not the only element of the task that may change over time. Certainly the domain evolves as additional documents are processed. As well, when the corpus is very large or dynamic (*e.g.*, the Internet), the corpus itself may be seen as evolving if rules for mapping text patterns to query items that apply at one time or for one subcorpus no longer apply for another. The idea that task requirements evolve is consistent with much research within the information retrieval community, but in contrast to traditional approaches to information extraction — *e.g.*, as seen in recent MUC-6 (1995) and MUC-7 (1999) papers — which implicitly assume that task requirements are static for the duration of the task.

Proposed design. Using machine learning techniques, the system is able to suggest what values are appropriate for task parameters. For example, one of the parameters the system should monitor is how well the documents are fitting to its current rule set. In this way, critical documents can be identified and the system can decide between alternatives more efficiently

while preventing the user from annotating redundant documents that provide it with no new knowledge.

Conclusions

In our continuing work, we would like to show that the interactive approach that we have suggested for information extraction is both possible and practical from the perspectives of the user and of the system.

From the perspective of the user, research in the library sciences and in user-centered design both strongly suggest that an interactive approach is natural and necessary to identify and solve the user's information need, particularly when the user is not completely confident about the topic area, the kinds of information that are available, or the specifics of the system. People are not always effective at expressing their information need, perhaps not having completely formulated it in their own minds.

From the perspective of the system, there is evidence to suggest that interactive approaches to learning perform as well or better than approaches involving less interaction. However, the most convincing evidence will be a demonstration that this is the case.

In conclusion and for the purpose of raising discussion, we make the following claims:

Claim 1 The user must be involved in constructing and maintaining the system's knowledge base.

Claim 2 The knowledge represented in the system, the process for gathering information, and the way in which it is presented to the user must be able to change and evolve as the user's needs evolve.

Claim 3 The system and user should be responsible for those aspects of the task that they are each best qualified to perform, and these may change as the task progresses.

We look forward to an interesting symposium!

References

AAAI'98-AIII. 1998. *Proceedings of the AAAI'98 Workshop on AI and Information Integration*, Madison, Wisconsin, USA.

Aberdeen, J.; Burger, J.; Day, D.; Hirschman, L.; Robinson, P.; and Vilain, M. 1995. MITRE: Description of the Alembic system used for MUC-6. In MUC-6 (1995), 141–155.

Allen, B. L. 1996. *Information Tasks: Towards a User-Centered Approach to Information Systems.* Library and Information Science. Academic Press: San Diego, CA.

Appelt, D. E.; Hobbs, J. R.; Bear, J.; Israel, D.; Kameyama, M.; Kehler, A.; Martin, D.; Myers, K.; and Tyson, M. 1995. SRI International FASTUS system: MUC-6 test results and analysis. In MUC-6 (1995), 237–248.

Bird, S., and Liberman, M. 1999. Annotation graphs as a framework for multidimensional linguistic data analysis. In *Towards Standards and Tools for Discourse Tagging, Proceedings of the Workshop*, 1–10. Association for Computational Linguistics.

Feldman, R., and Dagan, I. 1995. Knowledge discovery in textual databases (KDT). In *Proceedings of the First International Conference on Knowledge Discovery (KDD-95).*

Gaizauskas, R., and Robertson, A. M. 1997. Coupling information retrieval and information extraction : A new text technology for gathering information from the web. In *Conference Proceedings of RIAO 97: Computer-Assisted Information Searching on the Internet*, volume 1, 356–370.

Lewis, D. D., and Sparck Jones, K. 1996. Natural language processing for information retrieval. *Communications of the ACM* 39(1):92–101.

Morgan, R.; Garigliano, R.; Callaghan, P.; Poria, S.; Smith, M.; Urbanowicz, A.; Collingham, R.; Costantino, M.; Cooper, C.; and the LOLITA Group. 1995. University of Durham: Description of the LOLITA system as used in MUC-6. In MUC-6 (1995), 71–85.

MUC-6. 1995. *Proceedings of the Sixth Message Understanding Conference (MUC-6)*, Columbia, Maryland, USA: Morgan Kaufmann: San Francisco, USA.

MUC-7. 1999. *Proceedings of the Seventh Message Understanding Conference (MUC-7).* Available online from the SAIC website at http://www.muc.saic.com/.

Rose, D. E., and Belew, R. K. 1991. Toward a direct-manipulation interface for conceptual information retrieval systems. Greenwood Press: Westport, Connecticut, USA. chapter 3, 39–53.

Twidale, M. B., and Nichols, D. M. 1998. Designing interfaces to support collaboration in information retrieval. *Interacting With Computers* 10(2):177–193.

Vanderheyden, P., and Cohen, R. 1998. Information extraction and the casual user. In AAAI'98-AIII (1998), 137–142.

Vanderheyden, P., and Cohen, R. 1999a. Designing a mixed-initiative information extraction system. In *Working Notes of the AAAI'99 Workshop on Mixed-Initiative Intelligence*, 142–146. Orlando, Florida, USA: AAAI'99-MII.

Vanderheyden, P., and Cohen, R. 1999b. A dynamic interface for a dynamic task: Information extraction and evolving queries. In *Working Notes of the AAAI'99 Spring Symposium on Agents With Adjustable Autonomy*, 120–123. Stanford, California, USA: AAAI'99-SSAWAA.

Weischedel, R. 1995. BBN: Description of the PLUM system as used for MUC-6. In MUC-6 (1995), 55–69.

Wermter, S.; Riloff, E.; and Scheler, G., eds. 1996. *Connectionist, Statistical, and Symbolic Approaches to Learning for Natural Language Processing.* Lecture Notes in Artificial Intelligence. Springer, Berlin.