The Role of Mixed-Initiative Agent Interfaces in Intelligence Analysis

Extended Abstract

L. Karl Branting
BAE Systems, Inc.
Columbia, MD 21046, USA
karl.branting@baesystems.com

1. INTRODUCTION

The task of intelligence analysts is to derive information useful for law enforcement or security from diverse and heterogeneous data sources. Two assumptions support the position that mixed-initiative agent interfaces are desirable for automated systems to assist analysts. The first is that AI systems are most beneficial when they help analysts to do their jobs better, rather than to replacing analysts or supplanting their expert judgment. The second idea is that the relationship between analysts and AI systems should be governed by the metaphor of a collaborator who works with the analyst, rather than of an *idiot savant* who slavishly carries out calculations without any idea of whether they make any sense given the analyst's goals and the current problem-solving context. This distinction has been termed *interface-as-agent* as opposed to *interface-as-tool* [Chi98]. Collaborative interfaces for analysis have sometimes been referred to as *analyst's assistants*.[1]

The objective of this paper is to identify opportunities to apply mixed-initiative interface design techniques to improve intelligence analysis. The focus is not on technical solutions, but rather on potential applications in which successful mixed-initiative techniques might lead to significant benefits.

This paper sets forth several important factors that contribute to the difficulty of intelligence analysis and proposes a range of automation options addressing these factors.

2. IMPEDIMENTS TO INTELLIGENCE ANALYSIS

Analysts are trained to draw useful conclusions from data. However, at least six such factors impede the ability of analysts to perform their primary function:

1. *Repetition.* Much of analysts' time and attention is consumed by repetitive, routine activities rather than by the analytical tasks for which analysts are trained. For example, analysts are often required to perform highly repetitive sequences of queries to obtain a stream of relevant data.
2. *Superfluous data.* Analysts are often confronted by too much data. The large quantities of irrelevant data that typically surround relevant information can make analytical tasks, such as recognition and monitoring, slow and error-prone.
3. *Poor division of labor between computer and user.* Some analytical tasks are difficult for people to perform but easy for computers (e.g., tasks involving large numbers of computations), while others are easy for people but hard for computers (e.g., those involving common-sense). The optimal performance requires a division of labor in which the analyst performs the tasks at which humans are best and the system performs the tasks at which computers are best.
4. *Variable data format and quality.* Analysts' time is wasted by cleaning noisy entries from data sets and manipulating data into a consistent format.
5. *Extensive training requirements.* Training analysts is often a slow process, and the duration of service of many analysts is sometimes relatively short. As a result:
   - The period during which an analyst is highly proficient may be short relative to the length of an analyst's service.
   - Not all analysts are equally highly trained at any given time; inexperienced analysts are more prone to errors.
   - The required number of analysts is greater than if training were faster.
6. *Inconsistencies among analysts.* When different analysts draw different conclusions from the same data, important observations may be lost or false conclusions may be reached. Inconsistencies make quality control and training more difficult.

3. AUTOMATION OPTIONS

A range of automation options could potentially reduce the impediments described above. This section sets forth seven general approaches of complexity ranging from trivial to significant.

[1] See, e.g., analyst's assistants for data exploitation and interpretation http://www.wjminc.com/company/history.html and for report creation http://acl.ldc.upenn.edu/N/N03/N03-4004.pdf.

AAAI 2005 Fall Symposium on Mixed-Initiative Problem-Solving Assistants, November 4-6, 2005, Arlington, Virginia, USA.

3.1 Automation of Routine Non-analytical Functions.

The simplest automation option is to permit analysts to manually specify patterns of actions to be performed by the interface. This would reduce the first factor listed above, repetition.

Examples of automation of routine non-analytical functions include the following:

- Permitting the analyst to select sequences of actions as reusable macros, reducing repetitive actions.
- Monitoring, e.g., signaling the occurrence of significant events, or indicating the current level of some changing quantity.
- Performing periodic actions, such as executing queries at regular intervals.

The most straightforward implementation of an interface design permitting this form of manual customization would require only basic interface design and usability analysis [Nie94]. Possible mixed-initiative components could include dialogue management for specifying tasks and reporting the results of performing those tasks.

3.2 Auto-customization.

In auto-customization, the system would autonomously tailor the interface to the behavior and preferences of individual analysts. Examples include the following:

- Recognizing common sequences of actions, compiling the sequences into macros or templates, presenting them to the user, and permitting them to be viewed, copied, edited, ordered by frequency, and displayed.
- Detecting recurring actions and providing analysts the option of monitoring or scheduling periodic tasks based on these actions.
- Learning analysts' behavioral preferences (i.e., ordering relationships on actions), habits, schedules, and behavior.

Auto-customization can be performed without domain knowledge, requiring instead observations of user's behavior. For example, behaviorial regularities could be identified by running an induction algorithm at regular intervals over a log of analyst's actions. If regularities are detected, the system could unobtrusively ask the user whether he or she would like to (1) compile the pattern into a macro or template for reuse, (2) schedule the sequence for periodic performance, (3) monitor some variables, or (4) make any other modification of system behavior to conform to the analyst's preferences.

The requirements for auto-customization includes a representation language for the actions over which the induction is performed, and a action ontology to support generalization into templates.

The primary opportunities for mixed-initiative interaction in auto-customization arise in interacting with the user to determine whether the user would like to accept macros, templates, schedules, or monitoring regimens. This interaction must be designed carefully, because an interface that bombarded the user with large numbers of proposals would not be acceptable to most users, particularly if a significant proportion of the proposals were rejected.

3.3 Multimodal Interfaces.

Analysts may differ in the manner of interacting with a computer that they find most intuitive, and a single analyst's preferences may change over time. These individual preferences could be accommodated by a flexible user interface that permitted analysts to select among a variety of different interaction modalities, e.g., menu-based, text-based, voice-activated, etc., and interaction style, e.g., mixed-initiative, user-driven, system-driven.

There are such a wide variety of choices for multi-modal interfaces that it is impossible to identify a single essential technology, but appropriate technologies include multimedia interface techniques from the intelligence user interface community, natural-language understanding, speech-understanding, dialogue management and mixed-initiative interface design.

3.4 Automation of routine analytical tasks

A system with explicit domain knowledge can help automate portions of analysts' tasks, including:

- Recognizing relational and temporal structures, such as plans, social networks, and transactions. The required domain knowledge includes a model of plan structures, e.g., goals, states, operators, hierarchical task decomposition networks, etc. [JLM04]. The technology required includes link detection, plan recognition, graph indexing and pattern matching techniques.
- Finding patterns of information in text. The required techniques include standard text mining and information extraction, including hidden-Markov models and other statistical language models, POS tagging, shallow or deep parsing, and semantic analysis.
- Manipulating data into a standard format. This would require knowledge of the structure of target data, and could be performed using information extraction or other data mining techniques, depending on the condition of the target data.

There are numerous opportunities for mixed-initiative interaction in a system that uses domain knowledge to perform analytical tasks, including (1) dialogues in which the analyst specifies his or her goals, which could include requests for clarification or more information from the system, and (2) relevance feedback from the analyst.

3.5 Acquiring additional domain knowledge to improve performance in analytical tasks.

If a learning component is added to system with explicit domain knowledge, a number of opportunities for improving and customizing performance.

- Learning by observing analysts' actions and behavior, e.g., learning to recognize and anticipate analyst's goals. This is an extension of auto-customization, but with the addition that analysts' actions are interpreted in terms of underlying domain model, so the system can learn actions that are steps in similar plans even if the actions themselves are dissimilar.
- Active learning, i.e., selecting and asking questions that best discriminate among competing hypotheses.
- Learning by being told, e.g.,
  - Learning new lexical or taxonomic information
  - Learning to recognize new kinds of plans
- Exploration, e.g., of web.

Active learning is a particularly important opportunity for mixed-initiative acquisition of domain knowledge. It is very unlikely that any system can acquire error-free domain knowledge. As a result, the acquisition of domain knowledge would requiring interaction between system and user to (1) identify opportunities for learning, (2) correct or extend domain knowledge, (3) select between hypotheses, or (4) integrate new information into existing knowledge structures.

3.6 Training.

A system with an explicit domain or task model can help train analysts by:

- Answering procedural and substantive questions
- Providing suggestions
- Recognizing instructional opportunities
- Pedagogical planning, i.e., devising ways of teaching analysts important skills or declarative knowledge.

All mixed-initiation techniques developed for Intelligent Tutoring Systems would be applicable to the specific task of training analysts.

3.7 Continuity and cross-analyst consistency.

An important function of an analyst's assistant would be to maintain knowledge about analysts' practices to provide consistency over time and identify divergences between difference analysts and between individual analysts and the norm. A mixed-initiative interface would be important for this function because interaction with the system might take a variety of different forms depending on the context and the user. The system might function as an intelligent tutoring system for a novice and as a question-answering system for a more experienced analyst. The system might monitor analysts' activities, signaling when anomalous behavior is observed. In general, maintaining the optimal balance between actively engaging the analyst and passively responding to queries would require a detailed user model.

4. CONCLUSION

This paper has briefly sketched some opportunities for mixed-initiative agent interfaces to address the primary problems faced by intelligence analysts. Implementation of specific mixed-initiative components will require extensive analysis of individual analyst's tasks. However, the range and complexity of tasks arising in intelligence analysis makes it a challenging and potentially rewarding area for research in mixed-initiative techniques.

5. REFERENCES

D. Chin. *Readings in Intelligent User Intefaces*, chapter Intelligent Interfaces as Agents, pages 343–358. Morgan Kaufmann Publishers, Inc., 1998.

P. Jarvis, T. Lunt, and K. Myers. Identifying terrorist activity with aI plan recognition technology. In *Proceedings of the Sixteenth National Conference on Innovative Applications of Artificial Intelligence (IAAI 2004)*, pages 858–863, San Jose, California, July 25–29 2004. AAAI Press.

J. Nielsen. *Usability Engineering*. Morgan Kaufmann, 1994.