Multi-Hypothesis Abductive Reasoning for Link Discovery

Nicholas J. Pioch, Daniel Hunter, James V. White, Amy Kao, Daniel Bostwick, Eric K. Jones
ALPHATECH, Inc.
6 New England Executive Park
Burlington, MA 01803
781-273-3388

{npioch, dhunter, jwhite, akao, bostwick, ejones}@alphatech.com

ABSTRACT

Intelligence agencies are under increasing pressure to "connect the dots" between fragments of evidence from disparate sources to enable preemption of potential threats such as terrorist attacks. Most systems for threat detection in use today provide only data visualization tools for manual "link analysis," leading to methods that do not scale to massive data sets. The CADRE system (Continuous Analysis and Discovery from Relational Evidence) addresses this deficiency by automating the link analysis process. CADRE combines an expressive knowledge representation of threat patterns with efficient, constraint-based abductive reasoning algorithms to automatically infer links and construct coherent threat hypotheses from structured data. A compact, factored representation of multiple hypotheses avoids redundant storage and enables scaling to large data sets. CADRE efficiently manages the growth of the hypotheses using probabilistic evaluation models and a consistency checking algorithm to prune unlikely hypotheses.

Categories and Subject Descriptors

I.5.4 [**Pattern Recognition**]: Applications

I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods

General Terms

Algorithms, Performance, Design, Experimentation.

Keywords

Data mining, link analysis, abductive reasoning, Hidden Markov Models, pattern representation.

1. INTRODUCTION

With the increasing threat of global terrorism, and an ever-growing sea of computerized intelligence data, manual analysis techniques cannot provide enough coverage to reliably detect threatening activity. To date, most systems used by intelligence and law enforcement agencies have been limited to *link analysis* tools, such as Analyst's Notebook® [6], which enable rapid exploration of small- to medium-sized relational data sets. These tools depict data as a graph, in which nodes are usually people, places, or events, and links are binary relations holding between them. Analysts detect threats by manually "eyeballing" the data.

While manual link analysis methods can be usefully applied to small data sets, they break down when

- data sets become large or densely connected
- relevant clues are so widely scattered that they cannot be conveniently localized on a single visual display
- inference is required to understand the significance of disparate pieces of evidence in combination.

These challenges become all the more difficult when the task is to detect a terrorist threat before an attack occurs, as opposed to investigating after the fact. This suggests a need for new relational data mining methods to help automate link analysis. Such *link discovery* systems have been developed under DARPA's Evidence Extraction and Link Discovery program (EELD), including our system, CADRE: Continuous Analysis and Discovery from Relational Evidence.

CADRE implements a form of abductive inference to produce the best explanation of a set of observed facts. Previous research on abductive inference includes script-based methods [2] [12]; logic-based methods [4] [6] [8]; probabilistic methods [10], and hybrid systems [3] [11]. Our approach has its roots in Schank's and Charniak's script- and schema-based approaches to plan recognition for natural language understanding [2] [12]. Like this work, we represent threat patterns as hierarchically nested, templated event sequences. This approach has the advantage that it scales well since it enables top-down, pattern-directed refinement of hypotheses. However, CADRE's domain—the detection and prediction of threat events—presents several new challenges, including massive amounts of data, a vast hypothesis space, and low observability of attributes and links.

Unlike natural language understanding, threat detection involves correlating a relatively small number of clues scattered among very large datasets. These datasets contain mainly noise and clutter, where noise consists of random events that may happen to fit a subpattern of the threat pattern, and clutter consists of events conforming to a nonthreat pattern having some similarity to the threat pattern. Note that for this type of problem, we are not given a set of observations to explain, but rather must search for relevant observations, melding data mining with abductive inference. We regard the key challenge for this sort of problem to

Conference'04, Month 1–2, 2004, City, State, Country.

lie in the efficient generation of hypotheses, rather than in the evaluation of hypotheses already generated. The large size of the hypothesis space due to high noise and clutter renders De Kleer's candidate generation and conflict recognition algorithm [4] unfeasible here since one cannot construct all hypothesis nodes at a given level in the hypothesis lattice. Horn abduction faces a similar problem in that there is no way to focus the search for explanations so that in the presence of large amounts of noise and clutter, searching the hypothesis space is inefficient. A Bayesian network representation of the full hypothesis space [10] , even though it can compactly represent a very large state space through factorization, is not feasible since there may be millions of values for some individual variables in a flat Bayes net representation (e.g. the number of a priori possible values of a variable representing the leader of a threat event could be huge).

CADRE deals with the hypothesis generation problem for large hypothesis spaces by a carefully tailored combination of bottom-up and top-down processing, which focuses the search for hypotheses in such a way that the number of hypotheses generated is manageable. In addition, we have implemented a very compact encoding of large numbers of hypotheses that avoids the need to store full hypotheses explicitly. To deal with the incompleteness and sparseness of relevant data, we employ a hierarchical, constraint-based representation of patterns that allows incremental filling in of a hypothesis at different levels and that permits constraint-based inferences about missing data.

The following sections describe CADRE's approach to handling the problem of hypothesis generation and evaluation for large, noisy datasets. Section 2 describes the relevant features of the problem domain. Section 3 gives a high-level view of the CADRE system, and the next two sections delve into the details of hypothesis generation and hypothesis evaluation.

2. PROBLEM DOMAIN

The EELD program (funded by DARPA) supported research on the problem of detecting threat activity in massive amounts of data. For testing, evidence and ground truth were generated by a simulator in which parameters such as the observability of events, the amount of noise and clutter, and the degree to which evidence suffers corruption, were systematically varied. The simulated data describes an artificial world in which there are threat and non-threats groups carrying out activities that on the surface may look very similar. A collection of individuals from the same group may form a team to carry out an "exploitation" directed at some "target." In the case of teams from non-threat groups, the exploitations are benign—they consist of the productive use of resources and capabilities applied to the target. Teams from threat groups, however, although they may sometimes engage in benign exploitations, also engage in threat exploitations, in which harm is done to the target.

Both threat and non-threat exploitations follow a common pattern: first an exploitation team is recruited through a series of communications, next needed resources for the exploitation are acquired, and finally the exploitation is consummated by applying the resources and capabilities of the team members to the target. Threat and non-threat exploitations differ only by probabilistic differences in the timing of events, by the particular combination of resources and capabilities employed (called threat vs. non-threat modes), and, of course, by the fact that only threat groups carry out threat exploitations.

In addition to the large dataset size, three particularly challenging aspects of the simulated data are noise, clutter, and partial observability. The simulated datasets used in the EELD 2003 evaluation featured up to 10,000 entities and 100,000 links, with clutter and noise events far outnumbering threat events. Only half of resource acquisitions and communications in the ground truth are published in the evidence data, and only half of the members of threat groups are declared. This means that link detection systems must provide effective mechanisms for partial matching. With the low signal-to-noise ratio, they must be able to efficiently represent and process large numbers of candidate hypothesis. To avoid being misled by the clutter, they must also have hypothesis evaluation algorithms to prune least likely hypotheses without breaking down if certain key elements of the hypothesis are omitted.

3. SYSTEM OVERVIEW

CADRE's hypothesis generation engine uses a declarative representation of a prototypical threat pattern to query evidence and automatically construct hypotheses. To facilitate rapid definition of such patterns we provide a compact language and knowledge base (see Figure 1) for specifying events, entities and relations using multiple inheritance and *contexts*, which are similar to the Cyc™ system's *microtheories* for partitioning information. This language also represents object and temporal constraints among pattern slots, where a slot is a binary relation between a pattern instance and another class instance or primitive type. To focus initial search, we allow analysts to annotate specific slots and constraints deemed useful for triggering candidate threats. As shown in Figure 1, during the *trigger* stage CADRE automatically compiles and executes queries for the annotated slots and constructs initial hypotheses from the results. It then *refines* these hypotheses by recursively querying for remaining unfilled slots based on constraints involving known slots. The most elaborate refined hypotheses are found at the leaves of a context tree in which each subcontext in the context inheritance hierarchy adds a single contributing piece of evidence. These leaf hypotheses are considered *local hypotheses,* because each one describes a single candidate threat event of interest.

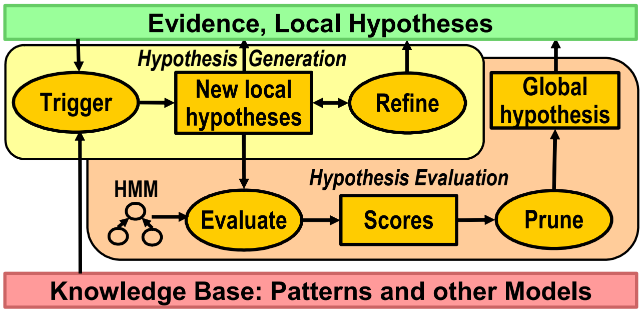


Figure 1. CADRE combines abductive reasoning and probabilistic models to support link discovery.

Hypothesis evaluation in CADRE takes as input all local hypotheses, and produces a single *global hypothesis* consisting of the most likely compatible set of local hypotheses. During the *evaluate* stage in Figure 1, a probability score for each local hypothesis is computed by mapping subevents and their intervening time intervals to observations in Hidden Markov Models (HMMs). We compute a global hypothesis by examining local hypotheses in order of descending score and *pruning* any hypotheses that conflict with surviving higher-scoring hypotheses. Conflicts are detected via a mutual exclusion algorithm that checks for differing slot values in a pair of hypotheses describing the same event.

4. HYPOTHESIS GENERATION

CADRE supports automatic link analysis through two main stages: bottom-up inference via triggering rules, and top-down inference via abductive hypothesis refinement. Underlying both stages is an expressive framework for representing both patterns and hypotheses.

4.1 Pattern Representation

Threat and non-threat exploitations are represented in the language of the ALPHATECH Knowledge Server (AKS). AKS is a frame-based knowledge representation system built in Prolog. It represents concepts as a class hierarchy with multiple inheritance, slots with type and cardinality constraints, and default inheritance of slot values. In addition, it incorporates certain constructions from description logics, such as enumerations and class unions. Further details on AKS can be found in [5].

To facilitate automatic hypothesis generation, AKS enables us to place constraints on classes and their slots. A class is defined by a set of slots and constraints between slots. Each slot has a type, for example the slot twoWayComm is of type communication which means that any filler of twoWayComm must be an instance of the communication class. Constraints are expressed as relations between *slot chains*, which represent successive slots in the object hierarchy, each separated by a period. For example, the following class chainedCommunication represents a sequence of communications in which the respondent of the twoWayComm is constrained to be equal to the initiator of the twoWayComm of the subsequentComms slot.

```
Class chainedCommunication
   twoWayComm : communication
   subsequentComms : chainedCommunication
   twoWayComm.respondent =
                    subsequentComms.twoWayComm.initiator
   subsequentComms.twoWayComm.timeStamp =
                    twoWayComm.timeStamp + [0, 12] hours
End
```

AKS also represents temporal constraints among subevents. In the above example, we have a temporal constraint restricting the initial communication of the embedded chained communication to occur from 0 to 12 hours after the initial communication of the parent chained communication. Meiri's path consistency algorithm [9] is used to check the consistency of temporal constraints in hypothesized threat exploitation instances. Temporal and equality constraints are used to generate triggering rules for hypotheses and to generate queries for the refinement of those hypotheses. Furthermore, CADRE can operate on recursive patterns that contain a subevent of the same type as the pattern itself, as illustrated in the above example.

4.2 Hypothesis Representation

In the problem domain to which we have applied CADRE, there are typically many consistent ways of matching data to a pattern. Managing the large number of hypotheses that result was a central concern for the design of CADRE. To address this issue, CADRE employs a representation analogous to the track-tree representation in multi-hypothesis tracking, a widely used technique in data fusion applications [1]. CADRE applies two main concepts from multi-hypothesis tracking to relational link discovery: 1) an algorithm for assembling a global hypothesis on-the-fly from the best candidate local hypotheses (see Section 5), and 2) a packed tree representation of local hypotheses in which child nodes in the tree represent elaboration of additional links in the hypothesis. Each node is a separate *context* corresponding to a partially filled-out hypothesis. The root context corresponds to the raw evidence data. The tree is extended by making queries within each current leaf context, based on the assumptions of that context, and creating new child contexts corresponding to the different data matches obtained. The hypothesis tree structure compactly represents hypotheses via *context inheritance;* each context inherits the information present in the parent context and so only needs to store a single match to a query.

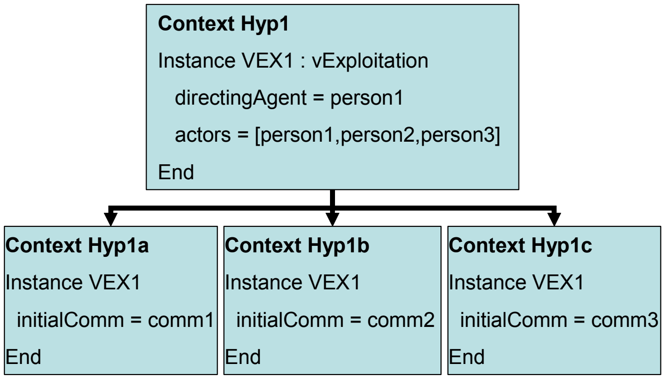


Figure 2. Hypotheses are elaborated in context trees.

In Figure 2, CADRE has generated a query for the slot initialCommunication, and found three matching data items, comm01, comm02, and comm03. Each context contains one hypothesis, an instance of the class vulnerabilityExploitation. In the top context, the hypothesis has values for the slots directingAgent and actors as indicated by the equality statements. In the next level of the context tree, only the values for the initialCommunication slot are stored.

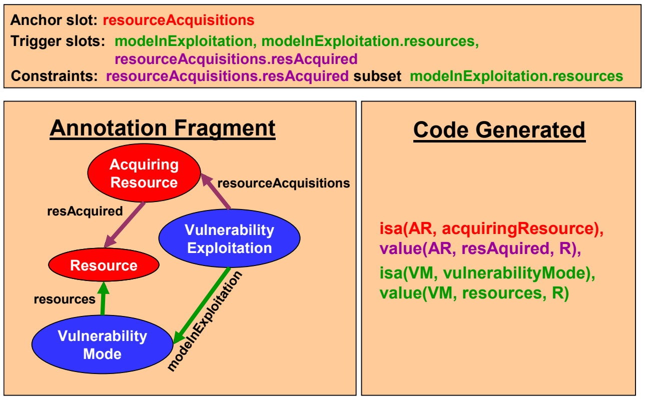
Since global hypotheses can be constructed by simply selecting one local hypothesis per context tree, it follows that a set of m context trees implicitly represents h^m distinct global hypotheses, if h is the number of local hypotheses in each context tree. Therefore, given a branching factor b, and depth d, the overall compression factor achieved by our factored hypothesis representation is given as follows:

$$\frac{h^m}{mb^d} = \frac{1}{mb^d}\left(1 + b\sum_{i=0}^{d-1}(2b)^i\right)^m \geq \frac{2^{(d-1)m}b^{d(m-1)}}{m}$$

Empirical studies based on characteristics of EELD's 2002 datasets show a space savings of ~10^{42} for b=1.5 and d=m=10.

4.3 Triggering Rules

Identifying instances of patterns in evidence begins with a bottom-up search for specific data that trigger the possibility of a hypothesis. We refer to these initial data queries as *triggering rules*. Triggering rules focus on relatively rare events with a high probability of appearing in the pattern. CADRE generates triggering rules based on annotated slots and constraints and automatically maps bound variables in the query to slot values in instantiated hypotheses. A single pattern may have multiple such annotation files to denote disjunctive triggers. Annotations include an *anchor slot* that identifies the first slot of the query, required trigger slots and constraints that must be filled by a match, and optional trigger constraints and slots whose filling will improve confidence in the match. Figure 3 illustrates how an anchor slot, several required trigger slots, and one required constraint are compiled into an executable query. The output of CADRE's triggering stage is the union of triggered hypotheses resulting from each annotation file.



Anchor slot: resourceAcquisitions
Trigger slots: modelnExploitation, modelnExploitation.resources, resourceAcquisitions.resAcquired
Constraints: resourceAcquisitions.resAcquired **subset** modelnExploitation.resources

```
isa(AR, acquiringResource),
value(AR, resAquired, R),
isa(VM, vulnerabilityMode),
value(VM, resources, R)
```

Figure 3. CADRE generates triggering queries based on annotated pattern slots and constraints.

Good triggering rules have the following features:

1. They involve events that are not common
2. They involve events that are strongly associated with the pattern of interest
3. They have connections to other parts of the pattern

Feature 1's importance has to do with computational efficiency and feasibility. For example, although it is part of the pattern for a threat exploitation that the team members communicate with one another, it is not computationally feasible to attempt to find threat exploitations by *first* sifting through *all* communications in search of clusters that satisfy the threat pattern. Looking at communication patterns may only be feasible *after* a set of suspects has been identified through other means. The second feature provides assurance that we are not likely to miss instances of the target pattern by focusing on features that, although perhaps sometimes found to occur in pattern instances, are frequently missing. Finally, a good starting place for hypothesis generation should have the property that it allows us to "unravel" the rest of the pattern in the data by following links from the subevents and individuals detected in triggering to other subevents or individuals that may be involved in the pattern instance.

4.4 Hypothesis Refinement

Following triggering, CADRE extends the triggered hypotheses in a top-down process we call *hypothesis refinement*. Hypothesis refinement takes as input the set of triggered hypotheses and emulates the manual link analysis process by incrementally expanding each partial match with new linked evidence that satisfies constraints involving known slots. The output of hypothesis refinement is a set of *local hypotheses* that have been maximally associated with available evidence.

We can view the evidence and pattern as graphs. Triggering thus produces a match of a subgraph of the pattern to a subgraph of the evidence. In this formulation, hypothesis refinement can be explained as the process of incrementally extending this match based on the pattern description. For a given hypothesis, CADRE generates queries for an unfilled slot by examining constraints that relate the unfilled slot to filled slots. For example, given the following hypothesis instance and class constraints, CADRE will query for fillers of the initialCommunication slot that meet the criteria that the initiator is person001 and the respondent is person001, person002, or person003:

```
Class vulnerabilityExploitationCase
   initialCommunication.initiator = directingAgent
   initialCommunication.respondent subset actors
End

Instance hyp001: vulnerabilityExploitationCase
   directingAgent = person001
   actors = [person001, person002, person003]
End
```

5. HYPOTHESIS EVALUATION

The result of hypothesis generation is a set of *local hypotheses*, each concerning a potential threat exploitation. The next stage, hypothesis evaluation, takes as input the most elaborate local hypotheses at the leaves of the context trees and produces a *global hypothesis*, which is the set of most likely mutually consistent hypotheses. To generate a global hypothesis from a set of local hypotheses, we again borrow from hypothesis management methods used in the multi-hypothesis tracking domain [1] to evaluate and prune local hypotheses. First, we generate a likelihood score for each hypothesis, using an HMM evaluator described below. Then we apply a greedy algorithm, starting with the most likely hypothesis, eliminating all hypotheses with which it is inconsistent, and then iterating with the next most likely local hypothesis.

To determine whether two local hypotheses are inconsistent, we use a *mutual exclusion* algorithm that compares values of corresponding *functional* and *uniquely identifying* properties. A functional property, such as the directing agent of the exploitation, cannot have more than one value. A *uniquely identifying* property is a property such that each possible value of the property can be held by at most one entity. If two hypotheses have different values for a functional property but the same value for a *uniquely identifying* property, then they are inconsistent.

5.1 HMM Evaluator

We have developed the algorithm HEVAL that evaluates a hypothesis H by computing three conditional probabilities: $P(V|H)$, $P(PT|H)$, and $P(PNT|H)$. These are the posterior probabilities that the hypothesis contains evidence from either a Vulnerability (V) exploitation case, a Productivity exploitation case by a Threat group (PT), or a Productivity case by a Non-Threat group (PNT). These three interpretations (case types) are mutually exclusive, and the probabilities sum to 1. The hypothesis H may be either a complete exploitation event or a collection of one or more substages within the exploitation.

A variety of optimal classifiers may be based on these probabilities. The *minimum-error classifier* assigns H to a case type that has the highest probability. If social costs are specified for classification errors, then the *optimal Bayes classifier* assigns H to a case type that has the smallest expected social cost.

The conditional probabilities are computed by using Bayes's rule, $P(x|E) = P(E|x)P(x)/P(E)$, where x is the case type and E is the evidence in the hypothesis. The likelihoods, $P(E|x)$ for x in {V,PT,PNT}, are computed using probabilistic models for the different types of evidence. Sequences of time-tagged evidence E(time) (about communications, acquisitions, target visits, and asset applications) are modeled as random processes using Hidden Markov Models (HMMs). Non-temporal pieces of evidence E(mode) (about exploitation modes), E(group) (about group memberships of actors), and E(assets) (about the number of assets that are confirmed by acquisitions or application events) are modeled separately. The complete set of models form a two-level Bayes net in which the exploitation case type is dependent on the outputs of temporal evidence, mode evidence, group evidence, and asset evidence.

The different types of evidence are modeled as conditionally independent, given the case type, so that $P(E|x)$ may be computed as the product:

$$P(E(\text{time})|x)P(E(\text{mode})|x)P(E(\text{group})|x)P(E(\text{assets})|x)$$

Partial observability, corruption, and noise in the evidence are easily modeled using HMMs, as depicted in Figure 4. Other types of errors may also occur in the hypothesis generation process. For example, if a resource application is unobserved, then the subsequent two-way communication, which always follows it, will also be missing from the hypothesis. Such temporally dependent errors are modeled by expanding the state space of the HMM as necessary.

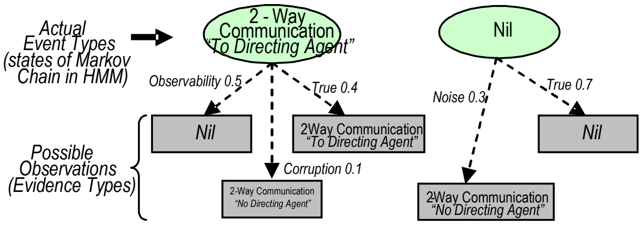
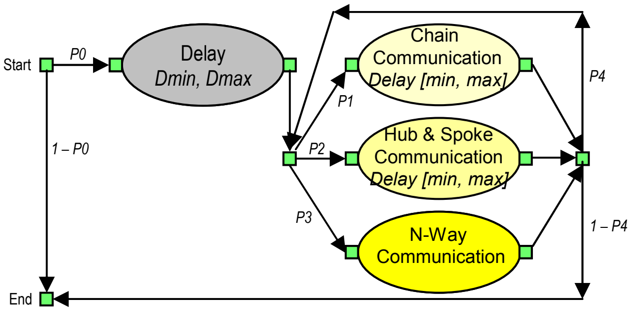

Actual Event Types (states of Markov Chain in HMM)
2 - Way Communication "To Directing Agent"
Nil
Observability 0.5
True 0.4
Corruption 0.1
Noise 0.3
True 0.7
Possible Observations (Evidence Types)
Nil
2Way Communication "To Directing Agent"
Nil
2-Way Communication "No Directing Agent"
2Way Communication "No Directing Agent"

Figure 4. Modeling evidence that is degraded by partial observability, corruption, and noise.

Three sets of HMMs were specified, one for each of the case types. Each HMM was built by interconnecting a set of submodels. The specification of these submodels and their interconnections were facilitated by a modeling technology [14] originally developed for modeling and analyzing amino-acid sequences of proteins. This technology is based on a gluing theorem that makes it practical for a human expert to specify any number of nested submodels, interconnect them, and thereby implement complex HMMs involving thousands of states. The submodels represent such processes as chain communications (the receiver of one communication acts as the sender of the next), full-team communications (a related set of communications involving all team members), target visits (an optional activity in which a threat team visits the target before the threat event), and resource acquisitions with associated confirming two-way communications (after acquiring a resource the buyer immediately contacts the directing agent).

An example of the model structure for a Full-team Communication Event is depicted in Figure 5. Each branch is labeled with the probability that the process will follow it. The probability distributions for the delays (uniform over a finite interval) and the branching probabilities depend on the exploitation case type.


Start
P0
Delay
Dmin, Dmax
1 – P0
P1
Chain Communication Delay [min, max]
P2
Hub & Spoke Communication Delay [min, max]
P3
N-Way Communication
P4
1 – P4
End

Exploitation Type	P0	P1	P2	P3	P4
V	0.2	0.64	0.27	0.09	0.2
PT	0.1	0.56	0.33	0.11	0.1
PNT	0.4	0.11	0.33	0.56	0.4

Exploitation Type	Dmin	Dmax
V	3	15
PT	3	7
PNT	2	10

Figure 5. Structure of Full-Team Communication Event.

To facilitate the independent evaluation of evidence in arbitrary substages of a hypothesis, only relative timing information within each stage was modeled. That is, the durations separating events were modeled but not individual event occurrence times. The probability durations separating events are different for the different case types and so relative timing provides a basis for distinguishing between the three different case types. The case types also differ in their modes and the group memberships of the actors.

The likelihoods of the HMMs were computed using the optimal filtering algorithm for categorical sequences in [14]. This algorithm computes the conditional probability distribution of the evidence that is about to occur at time $t+1$, given all past and present evidence at time t. The evidence in the hypothesis being evaluated is encoded as a set of categorical time series. For scalar communication processes that occur during the recruitment phase and during full-team communications, a single time series is sufficient. For resource acquisition and asset application during the consummation substage, multiple simultaneous time series are used to model independent processes occurring simultaneously. The probability distributions over the event-time differences for these simultaneous processes were precomputed and stored in multidimensional arrays. HEVAL uses these distributions and the HMMs: it applies optimal filtering and Bayes' rule to compute the conditional probabilities of the case types, given the evidence in the hypothesis being evaluated and a distribution of prior probabilities over the case types.

5.2 HMM and HEVAL Validation

The HMMs and HEVAL were validated by using simulated hypotheses constructed from ground truth before applying them to actual CADRE output. The test file contained 86 V exploitation cases, 8 PT cases, and 96 PNT cases. HEVAL was used to implement a minimum-error classifier with uniform prior probabilities over the three case types. The performance of the classifier for detecting V cases is summarized in Table 1. Here the *false-positive rate* is the conditional probability that the detector declares the evidence in the hypothesis to be a V case, given that the evidence is not from a V case. The *false-negative rate* is the conditional probability that the detector declares the evidence to be a PT or PNT case, given that the evidence is actually from a V case. *Recall* is the complement of the false-negative rate, and *precision* is the conditional probability that the evidence is from a V case, given that the detector assigned the evidence to a V case.

Table 1. Validation of HMMs and HEVAL.

Observability	Corruption	False pos. rate	False neg. rate	Recall	Precision
0.990	0.010	0.078	0.000	1.000	0.865
0.750	0.200	0.090	0.035	0.965	0.837

The *observability* (probability of observing events) is very high and the *corruption* (probability of mis-labeling or mis-assigning an event) is very low in these simulated hypotheses. Telling HEVAL that these parameters are 0.99 and 0.01, respectively, yields high recall (1.000) and reasonably high precision (0.865) as expected. When observability and corruption are mis-specified as 0.75 and 0.2 respectively, the recall and precision drop only slightly, which indicates that the evaluation of the simulated hypotheses is not overly sensitive to these parameters. These results validate that, given sufficient detail in refined hypotheses, HEVAL can accurately classify the hypothesis as a threat or non-threat exploitation.

6. EXPERIMENTAL RESULTS

The 2003 EELD end-year evaluation involved thirteen simulated datasets with varying levels of observability, clutter, and corruption. Variants with high connectivity or high rates of threat cycle completion were also included as individual specialized datasets. The evaluation contractor, IET Inc., provided only evidence filtered according to the above parameters, while retaining full ground truth for scoring purposes. IET developed automated scoring software taking as input a set of LD-output hypotheses and a corresponding "answer key" containing ground truth threat events only. Full details of the scoring algorithm are provided in [13]. Essentially, the software first searches for best matches of LD hypotheses to answer key threats using a greedy algorithm that compares a limited set of attributes. Then, a graph edit distance algorithm examining all attributes is used to determine a match score for each case pair. This score is weighted to reflect varying importance of attributes, then normalized. Variations of information retrieval's traditional recall and precision metrics were defined as follows:

- Weighted recall = sum of normalized match quality of paired cases divided by actual # of LD-output cases.
- Weighted precision = sum of normalized match quality of paired cases divided by actual # of ground truth cases.

6.1 Overall Results

Figure 6 shows CADRE's weighted precision vs. weighted recall for the first eleven datasets, as compared to scores resulting from another EELD link detection system and scores resulting from raw, unprocessed evidence. CADRE's results reflect tuning and debugging performed after the original experiment, due to problems with CADRE's database interface at the time of the evaluation. Results from the other link detection system are from the original evaluation. In all cases, CADRE's weighted recall scores were significantly higher than the evidence basis. For low observability datasets 1, 3, 5, and 7, CADRE's weighted recall and its relative improvement over raw evidence were significantly lower than for the other datasets. For these datasets, only half of the communications and resource acquisition events were reported in the raw evidence, leading to missed detections during triggering.

Note that weighted precision of the raw evidence basis is typically high because this metric only scores the evidence assertions corresponding to ground truth threats, that is, no false alarm threat events are included. Thus, any lack of precision in the evidence basis scores is due to corruption.

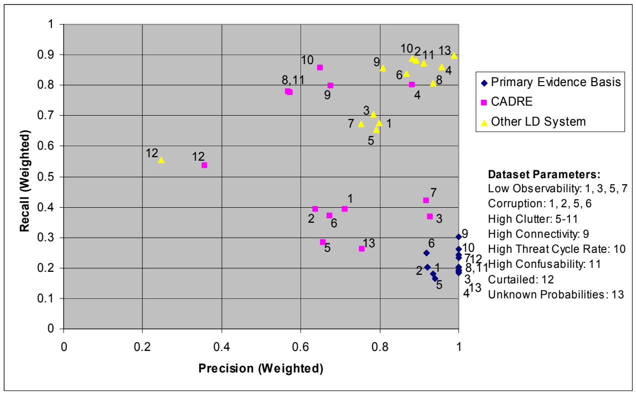


Figure 6. CADRE's weighted recall and precision as compared to raw evidence and another link detection system.

A particularly interesting result is CADRE's high precision on the uncorrupted low observability cases 3 and 7. By relaxing the triggering conditions, CADRE's low recall for these cases would have likely increased at the expense of precision. Also note that CADRE was able to find over 50% of the data for "curtailed" threat events in 12, meaning that CADRE was able to predict a threat event would occur using only evidence available prior to the actual attack. Because of the relatively scant evidence for these events, precision suffered, at 35%, but given the high social cost of real threat events, this may be an acceptable false alarm rate for law enforcement.

6.2 Impact of Hypothesis Evaluation

During follow-on experiments, we assessed the impact of HEVAL's probabilistic hypothesis evaluation as compared to an alternate ad-hoc evaluation method, which computes the percentage of slots which were assigned values for each local hypothesis and uses that percentage as the hypothesis' score. The alternate method emphasizes weighted recall over precision, since hypotheses with more information will always score higher than those with less, even if they are less accurate. Thus, we would expect that HEVAL would yield better weighted precision scores than the alternate method.

Dataset	Slot Prec.	Slot Rec.	Heval Prec.	HMM Rec.
6 Corrupt	0.64	0.69	0.7	0.62
8 Default	0.88	0.72	0.95	0.69
19 Fair connectivity	0.35	0.71	0.39	0.71
22 Easy target duty	0.91	0.71	0.94	0.63

Table 2. HEVAL yields slight improvements in weighted precision.

Using training datasets for which ground truth was provided, we manually derived the minimum acceptable threshold posterior probability for HEVAL leading to peak precision performance. Table 2 shows the weighted precision and recall results from selected datasets comparing HEVAL to the recall-based method. As expected, HEVAL led to small precision improvements. However, these improvements were balanced by slight losses in recall. One possible explanation for the loss of recall this is the use of the cutoff probability, since any hypotheses containing elements of ground truth threats falling below the cutoff would not be considered. The slot percentage evaluation method did not use a cutoff, rather, it accepted all hypotheses that were not in conflict with a higher-scoring hypothesis.

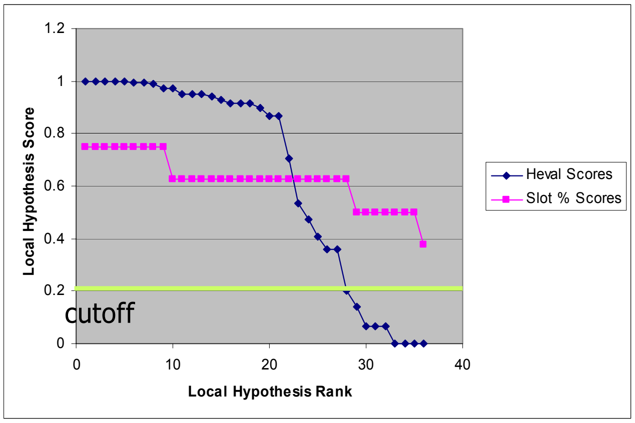


Figure 7. HEVAL produces a smoother performance curve than ad hoc metrics.

Despite the relatively small improvement in precision, HEVAL has a second advantage over the slot-percentage metric, namely that it exhibits a much smoother performance curve that is more amenable to choosing an optimal cutoff for low probability hypotheses. Figure 7 shows the performance plot for a single dataset, in which the local hypotheses have been sorted by descending score for both HEVAL and slot-percentage metrics. Because of the limited number of slots included in the computation, the slot-percentage metric exhibits only a few "quantum states," making it difficult to pick an optimal cutoff. In fact, attempts to find an optimal cutoff by hand always led to rejected hypotheses that corresponded to ground truth threats. With HEVAL, we found experimentally that a threshold of 0.2 maximized the precision benefit while minimizing the recall loss.

7. CONCLUSIONS

These results suggest that CADRE has significant advantages over other systems for abductive inference when the hypothesis space is so large that exhaustive enumeration of hypotheses is out of the question. In contrast to systems such as those described in [3] and [11], CADRE constructs hypotheses incrementally, starting from data known to be strongly relevant and using constraint-based reasoning to guide the search for additional pattern matches. This approach allows CADRE to be usefully applied to realistic problems involving massive amounts of data in which instances of the target pattern are sparse and incomplete.

8. ACKNOWLEDGMENTS

This research is sponsored by the Defense Advanced Research Projects Agency and managed by Rome Laboratory under contract F30602-01-C-0195. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied of the Defense Advanced Research Projects Agency, AFRL/IF, or the United States Government.

9. REFERENCES

[1] Blackman, S. and Popoli, R. *Design and Analysis of Modern Tracking Systems*, Artech House, Norwood, MA, 1999.

[2] Charniak, E. Motivation, Analysis, Abductive Unification, and Nonmonotonic Equality. *Artificial Intelligence, 34*, 3 (April 1988), 275-295.

[3] Charniak, E., and Goldman, R.P. A Bayesian model of plan recognition. *Artificial Intelligence, 64*, 1 (Nov. 1993), 53-79.

[4] de Kleer, J. and Williams, B.C. Diagnosing Multiple Faults. *Artificial Intelligence*, 32, 1 (April, 1987), 97-130.

[5] Everett, J., Bostwick D, and Jones, E. Rapid Knowledge Base Design via Extension of Mid-level Knowledge Components. In *International Conference on Integration of Knowledge-Intensive Multi-Agent Systems (KIMAS '03) (Cambridge, MA, Sep. 30-Oct. 4, 2003)*. IEEE Press, Cambridge, MA, 2003, 535-541.

[6] Hobbs, J.R., Stickel, M., Appelt, D., and Martin, P. Interpretation as abduction. Technical Note 499, SRI International, Menlo Park, CA, 1990.

[7] I2 Inc. 2004. http://www.i2inc.com/Products/Analysts_Notebook/default.asp.

[8] Josephson, J.R. and Josephson, S.G. eds. *Abductive Inference: Computation, Philosophy, Technology*. Cambridge University Press, Cambridge, UK, 1994.

[9] Meiri, I. Combining Qualitative and Quantitative Constraints in Temporal Reasoning. *Artificial Intelligence*, *87*, 1-2 (Nov. 1996), *295-342*.

[10] Pearl, J. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, CA, 1988.

[11] Poole, D. Probabilistic Horn Abduction and Bayesian Networks. *Artificial Intelligence*, *64*, 1 (Nov. 1993), 81-129.

[12] Schank, R.C. and Abelson, R.P. *Scripts, Plans, Goals, and Understanding*. Lawrence Eribaum Associates, Potomac, Maryland, 1977.

[13] Schrag, R. *EELD Y2 LD-PL Performance Evaluation*. Unpublished EELD Technical Report, 2003.

[14] White, J., Stultz, C., and Smith, T. 1994. Protein Classification by Stochastic Modeling and Optimal Filtering of Amino-Acid Sequences. *Mathematical Biosciences,* 119, 1 (Jan. 1994), 35-75.