A Non-Numerical Predictive Model for Asymmetric Analysis

by

Michael Valenzuela

A Thesis Submitted to the Faculty of the

Department of Electrical and Computer Engineering

In Partial Fulfillment of the Requirements
For the Degree of

Master of Science

In the Graduate College

The University of Arizona

2 0 1 0

Statement by Author

This thesis has been submitted in partial fulfillment of requirements for a Master of Science in Computer and Electrical Engineering at The University of Arizona and is deposited in the University Library to be made available to borrowers under rules of the Library.

Brief quotations from this thesis are allowable without special permission, provided that accurate acknowledgment of source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the head of the major department or the Dean of the Graduate College when in his or her judgment the proposed use of the material is in the interests of scholarship. In all other instances, however, permission must be obtained from the author.

SIGNED: _____

Acknowledgments

This thesis would not have been possible if it were not for the inspiration and peace I have found in God. Whenever the stress seemed too great, He has helped me overcome the hurdles, survive those 50 hour-weekends, and still remain relatively sain. Whether it be my classes, my research, or anything else, He has helped me through it.

This thesis was sponsored in part by the US Army Intelligence Battle Lab. Technical agents for part of this work were Fort Huachucha under Task Order Numbers 9T7ZDAIS705, 9T8ZDAIS803, and 9Q9SDAIS903.

I am indebted to my advisers and committee members: Rozenblit, Szidarovszky, and Sprinkle. Rozenblit has provided me with several opportunities and options as I began my advanced degree program. He was always very open and supportive, providing the guidance needed without restricting my research. Szidarovszky has helped guide my minor, verify the mathematical models presented in this thesis, and provided several enlightening classes. I would like to thank Sprinkle for agreeing to join the committee, even though my defense was scheduled two weeks later. He has also provided some insightful comments. I greatly appreciate their feedback, suggestions, and time they've dedicated to this project.

I would like to thank all of my colleagues. Somehow they made the time to help me proof reports, provide insights, and allowed me to bounce numerous ideas off of them.

Lastly, I would like to thank my family and close friends for their unflinching support throughout my academic career. I thank my parents for their whole- hearted support for a higher degree. I appreciate the many times my brothers, Danny and Tim, had dinner ready for me by the time I departed from the lab. My girlfriend, Heather, has been a pillar of support throughout the countless hours and several weekends she invested accompanying me while I worked. They have all been very accommodating, patient, and invaluable as I have worked on my thesis.

Table of Contents

Table of Contents—*Continued*

Table of Contents—*Continued*

List of Tables

List of Figures

Abstract

Predicting asymmetric threats (e.g., terrorist events) is becoming more important in the war against terrorism. Prior works have focused on tactical, statistical, and data-fusion systems. The thrust of our work instead has been the development of a novel system, with an emphasis on a non-numerical predictive model for automating and assisting intelligence analysts. The intelligence community uses a Template schema for assessing predictions. Our predictive model processes non-numerical data to arrive at automated assessment and certainty scores for these Templates. The predictive model is traceable, transparent, and utilizes Human-in-the-Loop data-fusion. In our future work, this predictive model will be further enhanced with behavioral filtering, which adjusts the assessment and certainty of the predictions by intelligently evaluating characteristic behavioral data. The non-numerical predictive model has been tested and verified inside the Asymmetric Threat Response and Analysis Program (ATRAP).

Chapter 1

Introduction

Intelligence analysts currently have a very difficult task. They must sift through massive amounts of data in order to build a higher level picture and answer questions about what is going on. When analysts are trying to answer questions about a situation, they traditionally make use of a Template schema. This schema involves composing a hypothesis and answering it by investigating several indicators, indicating a certain series of prerequisites. Since most of the data they use is non-numerical in nature (text, images, audio, multimedia), statistical software cannot help much. Without an aid to discover vital patterns in this information, critical information may be overlooked. Furthermore, without an automated method, rapidly changing factors may make historical data 'stale' before it can acted upon. Thus, this work discusses a tool to assist analysts in this process: a non-numerical predictive model for asymmetric analysis.

The non-numerical predictive model (NNPM) for asymmetric analysis to be discussed in this thesis was developed for the Asymmetric Threat Response and Analysis Program (ATRAP). The NNPM is essentially a flexible predictive model which generates a belief and a strength of that belief for predictions. The belief, hence forth referred to as an *assessment*, corresponds to an indication of truth in the prediction. The strength of that belief is called the *certainty*. The assessment and certainty are combined into one unified score (called *confidence*) for sorting purposes. The predictions are built using a Template schema similar to what intelligence analysts already use. The Templates/predictions are query-based, allowing for the search of

non-numerical information. The non-numerical information is derived from metadata and partially parsed textual data. This provides an easily searchable environment for the queries, which can then be matched either exactly, probabilistically, or fuzzily.

There are several applications of the non-numerical predictive model. It could be used in the analysis of the stability of asymmetric financial markets, law enforcement and criminal investigation, political inquiries, homeland security and modern warfare. One substantial application is in military intelligence analysis. ATRAP was designed for this purpose and hence includes a version of the non-numerical predictive model for asymmetric analysis.

Throughout this paper there are many technical terms and abbreviations are used. This can get confusing very quickly. Although all the terms are introduced before they are used, there are a large number of them and this paper is fairly lengthy. This makes it difficult to remember what all the terms and abbreviations mean. For a complete list of these terms refer to Appendix B.

1.1 Motivation

The generation of intelligence faces three major challenges. The first challenge is asymmetry, meaning opposing forces have very different capabilities. This is manifest in asymmetric financial markets, law enforcement, politics, homeland security, and modern warfare. Second, rapidly changing environments, such as technology and insurgencies [5, 18, 25], pose problems for systems that assume a static opponent. Lastly, due to the real-time confluence of information from automated sensors, Internet, unmanned aerial vehicles, and streaming reports, analysts have more data to sift through than previously. Information overload often obscures critical patterns in the data. To further compound this problem, the data analysts use is often in a textual

form, such as reports, which raises the need for non-numerical predictive models.

One major application is in asymmetric warfare, and modern warfare has taken a significant asymmetric shift. This includes peacekeeping operations and combating terrorist cells. Fighting terrorism is a significant current endeavor, which exhibits all three challenges simultaneously. First, the terrorist groups do not have the numbers nor equipment to fight a full-fledged (symmetric) war. To make combating terrorism more difficult, the terrorists remain flexible. According to [5] Al Qaeda has been successful in carrying out attacks partially due to its dynamic methods. Lastly, it can be difficult to track the actions of an entire group from individual reports on single persons or cells. All the data needs to be integrated to provide a full picture that an analyst can use to accurately assess the situation.

The predictive nature of the work of military analysts is not easily mathematically modeled. As such, evaluation of courses of actions (COAs) uses significant human resources. This work aims to provide a model which can automatically assist analysts by sifting through vast databases of information and leverage human intuition to help build expert predictions.

There are many key features spread out among the prior works, but none of them have all these features in a single package. Some of these features include: the ability for the user to tweak the model, the selection, evaluation, and generation of COAs, and support for tactical COAs as well as the COAs for the generation of information. Some offer better information management frameworks than others. Some prior works have better support for asymmetric analysis and human-in-the-loop capabilities. Most were lacking the ability to evaluate and generate new COAs. Our NNPM functioning inside of ATRAP offers all of these features in a single package.

1.2 Contributions

The NNPM is a contribution to the predictive modeling field. It was also developed as a data-mining application for ATRAP. This model is original compared to the predictive systems discussed in the next chapter (Chapter 2). There are many differences, which will be discussed in the next chapter, but one is the way the prediction works. Many of the prior works attempt prediction through tactical simulations, while the NNPM attempts prediction in a more general fashion via historical/doctrinal patterns. It is this prediction that makes the NNPM essential to the tasks of detecting, assessing and responding to COAs promptly and effectively. This project achieves this by automatically generating an assessment and certainty scores from textual data such as reports, HTML, emails, documents, and other similar archival artifacts. It also provides support for Human-In-The-Loop feedback via traceable, transparent, and overridable results. That is, a user can trace back to the source of the results, understand why the model provided the score it did, and optionally override the results to any subproblem. The non-numerical predictive model is more geared toward the generation of intelligence rather than the simulation of a scenario. Thus, given a vast database, the model can verify or deny an analyst's hypotheses.

With regards to data analysis, this project provides a mechanism for codifying COAs. The codified COAs can then be evaluated them from data including non-numerical sources. The codification process includes the construction of a tree-based Template representing COAs; the creation of "entities;" specification of optionally imprecise values, key terms, associations and relationships between "entities" for queries. Throughout the process, several parameters can be customized to provide a wide range of results. A selection of the parameters can be derived from several

example Templates with their recommended assessment and certainty scores.

The work reported here was supported by the US Army Intelligence Battle Lab. Technical agents for part of this work were Fort Huachucha under Task Order Numbers 9T7ZDAIS705, 9T8ZDAIS803, and 9Q9SDAIS903. Prior testing and results have been presented in [23].

1.3 Thesis Overview

In Chapter 2 we will introduce other predictive models and systems. Chapter 3 introduces the environment in which the non-numerical predictive model operates. Chapter 4 describes the methodology used when developing most of the algorithms used in this project. Chapter 5 describes how the model generates assessment, certainty, and confidence scores. Chapter 6 discusses experiments and results. Chapter 7 examines a potential enhancement to the non-numerical predictive model by making use of behavioral data. Chapter 8 summarizes the accomplishments of this thesis and suggests possible future improvements.

Chapter 2

Background

As a result of the third major challenge to the generation of intelligence—information overload—data-fusion systems have become critical. Data-fusion is broken down into five levels, where the lower levels deal with fusing sensor data and higher levels deal with identifying and predicting threats. The five levels (starting at level 0) according to the Joint Directors of Laboratories (JDL) are: Preprocessing, Object Refinement, Situation Refinement, Impact Assessment, and Process Refinement (see [13, 20] for details). There is also a proposed level 5 (sixth level) of data-fusion, User Refinement [2]. In Mahoney et al. [12], level 5 data-fusion is called Human-In-The-Loop Data-Fusion, in which higher level data-fusion tasks are performed manually by humans.

There have been numerous data-fusion systems and predictive systems developed to help deal with the problems of asymmetric conflicts, rapidly changing environments, and information-overload. Starlight^{TM}is a system that was developed as "visualization software" [7], with some data mining features. While Starlight slightly helps the information-overload problem, it did not address asymmetric predictions nor a rapidly changing environment. Neil Garra, a subject matter expert, saw a demonstration of the Starlight program in 1998. The version he saw had 30 parameters, none of which were changeable. He has also talked to other people who complained about the poor usability and large computing power [9].

Most recent research has gone into high level data-fusion (levels 2 and 3) via wargame simulators. Game theory and simulations are commonly used tools for determining threat assessments and evaluating enemy courses of action (ECOAs). An

attrition-type discrete time dynamic game model was designed to evaluate different approaches to a conflict [19]. In this simulator the focus was on military air operations and civilians who can retaliate. There are two forces, the friendly (blue) and enemy (red), and civilians can retaliate against any force that cause them harm (from collateral damage). The extensively developed game model evaluates COAs from a tactical standpoint, but lacks any mechanism to discover new COAs. In the closely related works [4, 24], the authors resolve this deficiency using adversarial Markov games, as part of a larger data-fusion system.

A game theoretic approach is also used in [4] with a multi-level data fusion system. Specifically, their system performs level-one through level-four data-fusion. The authors use a Hierarchical Task Network as an ontology representing various levels of detail for various ECOAs. They developed a Markov (stochastic) game which assumes there are three forces (friendly, hostile, and neutral). The hostile forces are also allowed to use deception to present false information to the friendly forces. The stochastic nature of noisy and uncertain environments is modeled well by Markov Decision Process, which is similar to a stochastic automaton. It is used in level-three data-fusion to help evaluate ECOAs as well as to discover their possible intents. The cumulative work of [4,19] are also included in [24], which improves the Markov game with added spatial dimensions. By using a modified spatial-temporal point-model, which was originally developed in [3], spatial forecasting can be used in conjunction with the terrain to generate a probability map of where certain events are likely to occur. There are three primary draw backs of this system. First is that the ECOAs are only tactical, rather than informational in nature. Additionally, there are few details on how scenarios are created and how information is entered into their system. Lastly, the system is too automated, lacking human-in-the-loop feedback.

The spatial prediction used in [24] was based on [3]. The authors of [3] offer

a significant improvement for spatial forecasting. The baseline method they used for comparison purposes was the fitting of a probability density function to past data. Their improvement came partially by increasing the dimensionality of the data by including additional information from geographic information systems. This allows them to intelligently increase the dimensionality of the data to find new correlations. Recent advances made by them in [15] include an investigation to select the appropriate algorithms based on the scenario and the available time for analysis. The authors' results showed a significant improvement over their baseline model of spatial forecasting. These new methods have little to do with predicting COAs, but are still a significant contribution in predicting the locations of reoccurring phenomenon.

An entirely different simulation environment is the Wargame Infrastructure and Simulation Environment (WISE) [17]. WISE was one of the approaches that the UK Ministry of Defence undertook to help with decision making in Command and Control (C2). It can be used both as a wargame, where military players make decisions, and as a simulator. Thus it is suitable for both experimentation of COAs and investigation of procurement options. It models many aspects including the difference between truth and perception, physical behavior, "plug and play" AI, and communicational modeling. WISE can log data for later examination. It uses two planning and decision making processes: both a strategic/operational level and a tactical level. The strategic/operational planner was enhanced in [11] by developing algorithms to identify collective behavior. The authors develop a data-fusion technique for the clustering of entities into groups. Their approach involves building a minimum spanning tree connecting all entities and breaking the tree up into forests if any connection exceeds a certain threshold. This produces a dynamic number of groups based on an attribute vector, opposed to a fixed number of groups as is the case

with k-means clustering. They also present a procedure for determining the spatial objective for a moving group and the confidence in that estimate. WISE with these enhancements is a truly remarkable environment, but it does not appear to cover everything. Most of the planning appears to be oriented around more conventional war rather than the generation of intelligence. Additionally, the system does not appear to be driven by a massive amount of data.

Sheherazade [22] is another wargame simulator, which uses genetic algorithms to evolve the COAs. Whereas other wargame simulators model Major Theater of War, Sheherazade was designed to model Stability and Support Operations (SASO). The wargame simulator alternates between playing out a series of friendly and hostile COAs, evaluating the results, and revising the COAs. Part of the evaluation includes examining factors such as animosity, civilian populations, and forces to determine when and where a riot is likely to erupt. This addresses the problem of the rapidly changing environment, reflecting how tactics change with respect to time. Sheherazade "co-evolves" the COAs for both the friendly and hostile forces [21], giving each side a chance to improve their tactics against the opponent's previous round's tactics. This co-evolving strategies show how an insurgency might change its strategies depending on the peace-keeping force's strategies. However, these systems take a narrow, tactical stance on COAs (limited to SASO), rather than a generic scope which intelligence analysts may need. Sheherazade admits that setting up a SASO environment, providing information about each unit, and information about the regions requires a non-trivial amount of effort.

A recent trend has been Human-In-The-Loop Data-Fusion, since humans offer many unique data-fusion skills. For example, a person can interject or act on knowledge that is missing from the system. The goal of [12] is to develop a high level data-fusion system that can easily be kept up to date and adapt to new Tactics, Techniques,

and Procedures (TTP). The authors do this with a prototype Human-In-The-Loop Data-Fusion system. Their system performs "COA impact assessments," which help assess the effectiveness of a COA under varying situations and environments. This is primarily an expert system with knowledge composed of several hundred hours of interviews and evaluations with many subject matter experts (SMEs). However, the authors quickly discovered that terminology changes quickly. So they expanded the system to include a Human-In-The-Loop to update the ontology and vocabulary to help keep terminology up to date. However, their system lacks some other features which may be desired for Human-In-The-Loop Data-Fusion. The prototype system does not state that the COAs are evolvable. Furthermore, most of the COAs mentioned are tactical in nature, rather than focusing on intelligence analysis. The system does not describe how information is feed to it nor how new data is to be entered for evaluation. Their system does not provide justification for its assessments, potentially leaving the user confused as to how it arrived at its answer.

Most of these previous works focused exclusively on one level of data-fusion, however a whole data-fusion system was introduced in [4,24] and multiple data-fusion levels were developed in [12]. These systems include feedback between different levels. The human-in-the-loop system allows the users to provide feedback to lower level data-fusion levels based on current deficiencies in the system.

Chapter 3

Environment

The Asymmetric Threat Response and Analysis Program (ATRAP) environment will be described to provide the needed understanding of the framework in which this NNPM operates. ATRAP is an ongoing project to assist analysts by functioning as a "Cognitive Amplifier." In other words, ATRAP's is a suite of many useful tools to assist analysts. ATRAP contains many individual pieces which build up the entire ATRAP environment. The predictions are initially human generated. A user codifies a prediction as a Template. These Templates are evaluated by the NNPM. More details about Templates can be found in Section 3.2.

There is also a component called an "entity," which is an object in the ATRAP database. Entities can be a person, organization, location, action, event, etc. An entity can have a place, time, relationships to other entities, and an entity type associated with it. The non-numerical predictive model finds and processes these entities to answer the questions posed by the Templates.

The non-numerical predictive model has been implemented as a part of ATRAP, written for Microsoft Windows XP and Vista. The programming language used was C#, using Microsoft .NET Framework 3.5. The database system was built upon .netTiers and Microsoft SQL Server 2005.

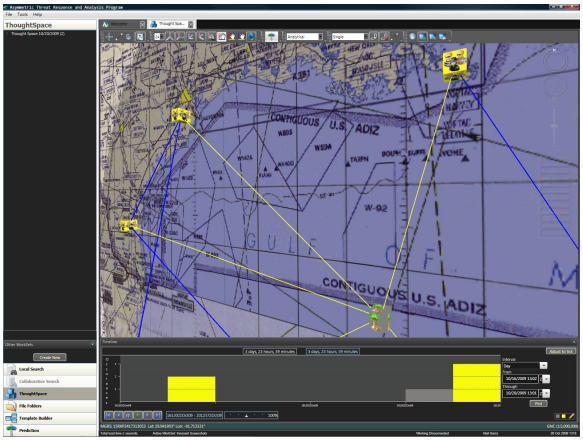


FIGURE 3.1. ThoughtSpace—ATRAP Visualization Tool

3.1 ATRAP co-development

A contracted developer, Ephibian, co-developed many aspects for ATRAP, including most of the graphical user interface (GUI). Ephibian worked closely with Neil Garra, a subject matter expert (SME), for various projects. They worked together on the creation of the database format, including the definitions for "entities." Another project they developed was a 3-dimensional visualization system for data (see Figure 3.1). This includes a visual editor for creating and editing entities. Ephibian created an automatic parser for documents to guess at proper nouns such as names of persons, places, events, etc. A text-highlighter was also developed to allow a user manual creation of entities from a report or other textual document. A visual search tool was created to help locate all "entities" in a user-defined region of the world.

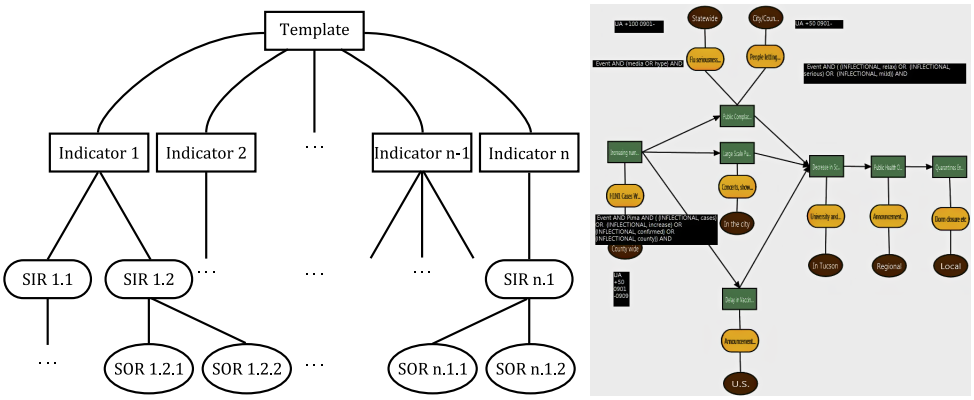
Research also went into optimizing the hardware on which ATRAP was to run. The major contributor to this field was Roman Lysecky. This involved looking into solutions for a parallelized ATRAP. Some areas of inquiry involved multi-core systems verse distributed systems, benchmarking, network bandwidth, memory and processing requirements. The solution that he and his team found should be sufficient

for many years as acceptance and use of ATRAP grows.

Jacob Gulotta looked into the generation of derivative enemy courses of action (DECOAs). To help make ATRAP more flexible and handle a rapidly changing environment, a genetic algorithm is used to generate new Templates from the initial user generated Templates. As of the writing of this document, the fitness function for the genetic algorithm looks at the confidence score of the Templates when recombining them to generate new "empirical" DECOAs. There is also ongoing research into making more "theoretical" Templates by looking at the requirements for subcomponents of the Templates rather than their performance based on available data.

Ephibian initially developed a text-highlighter tool to help extract entities, but it required a user to perform the tasks. To help automate this job, a couple people were recruited to help with computational linguistics. The computational linguists worked on a tool to identify all proper nouns in text. The task is more difficult than it sounds since the tool is designed to work for many regions around the globe. Future work involves automatically classifying proper nouns into groups such as individuals, locations, organizations, etc. Ideally, the tool could extract information associated with the proper nouns as well, such as time, location, and relationships to other entities.

Even with all these features, ATRAP lacked a basic predictive scheme. ATRAP was missing a scoring/ranking algorithm and the theoretical basis for scoring. This is where my work on the non-numerical predictive model comes into play. It affectively extends Templates as a formal mechanism for evaluating predictions based the data in ATRAP, which contains a significant amount of non-numerical data. Using generally accepted algorithmic components (e.g., Artificial Neural Networks), mathematical models (e.g., Cobb-Douglas Product) and feedback from expert Neil Garra, we built the non-numerical predictive model for asymmetric analysis that is the subject of this



(a) Template Structure (b) Example Template

FIGURE 3.2. Template

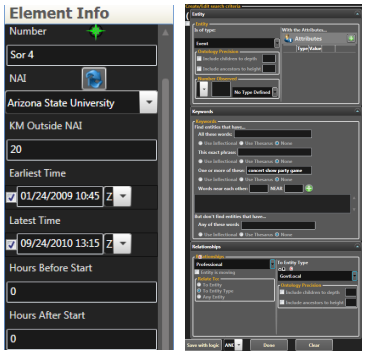
thesis.

3.2 Templates

This section will describe Templates. First, the connection between Templates and the non-numerical predictive model will be established. Then, the structure of the Template will be described in a bottom-up fashion. Lastly, this structure is extended to take on features from neural networks to increase the capabilities of a Template.

Templates can represent COAs, enemy COAs, and general predictions (see Figure 3.2). The NNPM makes use of this Template schema. Basically a Template is a codified hypothesis, including what information to look for that would support/deny it. The non-numerical predictive model extends and evaluates Templates. It gives Templates additional capabilities to make the querying process more powerful and reflect how a subject matter expert (SME) processes a Template. The resulting changes causes the Template to take on a Quasi-Neural-Network structure.

At the lowest level of a Template, there are the SORs, which have a link to its parent SIR and a spatial-temporal query. The spatial-temporal query represents a



(a) Spatial Temporal Query (b) Query

FIGURE 3.3. SOR and SIR Specification

5-tuple $\{A, T, t^-, t^+, r\}$, where A is a well defined area, T is a well defined time range, t^- is the allotted time before the start of T, t^+ is the allotted time after the end of T, and r is the allotted radius outside of A. A and T represent the perfect match constraints while t^-, t^+, and r represent extensions to A and T for imprecise matches.

In ATRAP, the spatial-temporal query resides inside of the "Template Builder" (see Section 5.1 for more on the Template Builder). When a user clicks on an SOR in a Template, a dialog along the right-hand side of the screen appears (see Figure 3.3(a)). The 5-tuple is reflected by the Named Area of Interest (NAI) (A), earliest-latest time (T), hours before start (t^-), hours after (sic) start (t^+), and KM Outside NAI (r). Any entities within the NAI between the earliest and latest time match the spatial-temporal query perfectly. The other parameters will extend the search, allowing for imprecise matches.

Above the SORs are SIRs. SIRs contain a search query for the ATRAP database (see Figure 3.3(b)), links to all its SOR children, and a link to its parent indicator. When constructing a query, there are options to automatically expand the query to

include inflections of the words and to make use of an ontology. When specifying the type of entity (person, place, event, etc.) the ontology can optionally include more specific types and more general types. For example, a query could specify an entity of type vehicle, and return a hit an entity of type truck.

Above the SIRs are Indicators, which contain a link to the parent Template and can either reference another Template or contains links to SIRs. A Template can take on a fractal nature by having Indicators referring to other Templates. Templates connected in such a way can be expressed in a form of a graph. This graph must remain acyclic, for Templates represent a hypothesis and a hypothesis in part based on itself can result in a circular argument. Alternatively, an Indicator may contain links to SIRs. Such an Indicator primarily provides structure and organization.

Templates consist of several links to their Indicator children and all their decedents. This represents a hypothesis. Intelligence analysts may build a Template for determining the capabilities of an organization or to evaluate COAs. There are primarily two types of Templates: Doctrinal and Situational. Doctrinal reflect the doctrine of some entity (organization, group, nation, individual, etc.). Typically Doctrinal Templates lack specifics such as time and location (SORs). A Situational Template is based off of a Doctrinal Template, but it contains all the specifics needed to evaluate it.

By extending the previously described model, Templates can take on additional capabilities. We call this extended model a Quasi-Neural-Network (QNN). The extension is achieved by allowing all the links in Templates to carry a weight. By adding a threshold value and activation function to each node in the Template, the Template behaves similarly to an artificial neural network (see Figure 3.4). The QNN are similar to other neural networks, except the input nodes are queries. Thus the queries are the location where transduction occurs. The significance of the QNN

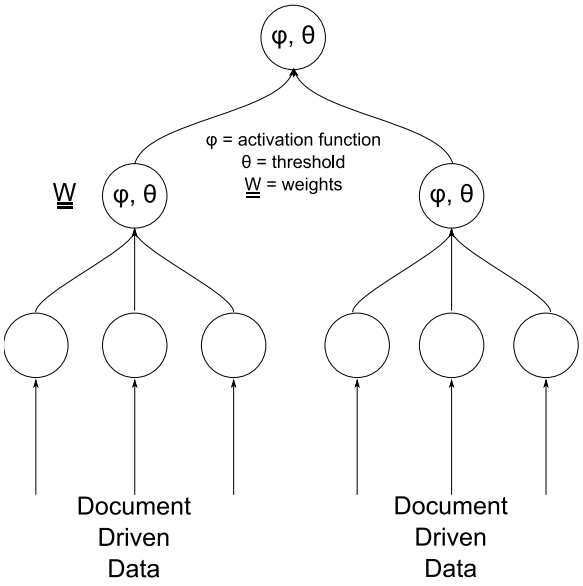


FIGURE 3.4. Quasi-Neural-Network Extension

extension will be explained further in Section 5.5.

Chapter 4

Methodology

This chapter describes the methodology used when generating the algorithms discussed in the next Chapter (5). Due to the unique nature of assigning a score to a Template, there has not been a significant amount of prior research into the evaluation of Templates. This chapter is outlined in three sections describing the steps used to generate several original algorithms. The first step discusses the process of defining requirements. Second, several algorithms are designed and compared in a trade-off table, iterating the first step as needed. Once no additional benefit as defined by the trade-off table can be obtained, a final review of the "best" algorithm attempts to determine whether further iteration is necessary or not.

4.1 Algorithm Building with Desired Properties

We worked closely with Neil Garra, an ex-military analyst, our primary SME with regards to evaluating Templates. Together we would brain storm test cases and the expected behavior. There were two types of test cases: extremal test cases and differential test cases. Extreme test cases defined how the algorithms ought to behave whenever any situation is at a potential extreme or boundary. For example, the behavior to occur at an SOR when the spatial constraint is perfectly matched and the temporal constraint is a perfect miss. Differential test cases helped define how the algorithms should behave as situations change slightly. An example of a differential test case is the expected behavior as more matches are found for a single SIR/SOR

query. Other examples include an entity moving or an entity type becoming more or less specific.

There were usually several desired behaviors from the algorithms besides the test cases. A few omnipresent desired properties were to keep the algorithms as simple as possible, as understandable as possible, traceable and transparent. Traceability refers to the ability to trace rationale and test cases back to the algorithm. Transparency encompasses the concept of the user being able to follow the algorithm—without black box components. The purpose for these behaviors lies in the fact that the non-numerical predictive model is supposed to be transparent and traceable to humans. As the algorithms grew in complexity, to fulfill the requirements and the desired properties, sub-steps were introduced for the purpose of understandability. Understandability not only applies to the algorithms, but also the intermediate results.

To fulfill several constraints, such as a diminishing returns, the algorithms typically would use classes of mathematical functions that already have such a behavior (e.g., sigmoid functions naturally exhibit diminishing returns and a maximum value). Individual components would then be combined to preserve the desired behavior of each component. This did not always lead to the simplest algorithms, but it did lead to highly understandable algorithms. Each component could be traced back to a test case or rationale. The algorithms are heavily mathematical, which is generally transparent. If users disagree with the specifics, they can easily tweak key parameters to change the behavior of individual components. The algorithm may not provide precisely the same solution as the user would, but the general trends would be correct. Furthermore, a mathematical SME could provide feedback on these types of algorithms. A couple of competing alternatives methods were considered when building up functionality include: logical decision trees, interpolation between saved data, and artificial neural networks.

Decision trees have their own sets of problem. First, they work much better for discrete problems than for continuous ones. Decision trees can fail to provide an answer if not all possible cases are covered. This is highly problematic since the decision space grows exponentially as the number of input variables increase. However, decision trees can be perfectly traceable and transparent. A user may disagree with some of the decision tree rules, which conceivably could lead to a disagreement with the system. Thus to account for this, the user might have to rewrite a large number of rules due to the exponential nature of the decision tree.

Interpolation, recording the SME's results for test cases and interpolating between them, is much like a continuous decision tree. It resolves the discrete/continuous issue present with decision trees, but shares many of its problems. The number of required interpolation points would generally grow exponentially as the dimensionality of the problem increases. Also the algorithm could fail if there are uncovered extremal cases. Interpolation would be difficult to understand what is happening as the algorithm interpolates. This means the algorithms would be traceable, but not transparent. Also the rationale and test cases for the stored data would be tied to the user who generates the data. Differences in opinion would result in a user either being responsible for providing his/her own test cases (potentially significantly increasing time required to configure the tool), or the data points would be non-transparent. The lack of transparency could potentially leading to disbelief of the computed answer.

Similarly to interpolation, artificial neural networks are trained to imitate the SME's responses. While this resolves the issue of missing test cases, the neural network is neither transparent nor traceable. Although it could perform arbitrarily well, there is no simple to way explain what the neural network doing nor a way to trace the test cases to the resulting neural network. The artificial neural network's complexity, run-time, and accuracy is governed by the decision surface and kernel

functions [10]. The dimensionality of the problem sets a lower bound on the initial number of neurons needed (which after training, may show some can be removed). Thus the dimensionality of the problem need not reflect the complexity of the neural network, which in turn means that this type of algorithm may run significantly faster or slower, but it is very difficult to determine without actually implementing the algorithm.

4.2 Trade-off Tables

To compare the different proposed algorithms for any particular part of the non-numerical predictive model, trade-off tables were made. A trade-off table is a two-dimensional grid with the proposed solutions along one axis. The other axis contains the desired properties along with their relative weights of importance. Usually more important properties have larger weights. The center of the grid shows an approximation of how well any solution satisfies any property. The more a solution satisfies a property, the larger the value. Then for each proposed algorithm, a score is generated from the sum of the products of the relative weights of importance of each property and how well that algorithm satisfies that property. The solution with the largest outcome is the winner. If there are two or more tied, then those solutions are considered equally good.

The majority of the algorithms used to score Templates underwent this process. Typically extremal cases had the highest importance, followed by transparency and traceability. Regardless whether or not there was a tie, the algorithms underwent another iteration of tweaking to try to improve their scores where they performed poorly, focusing on the heavier weighted attributes. Naturally, making an algorithm more complicated would worsen its simplicity score. Steps could be taken to ensure

the algorithm remained understandable though.

When no further improvements could easily be made, the algorithm was considered good enough at that point to implement. However, since the algorithms are not perfect, they are still subject to change. In each revision of ATRAP, usually a couple of bugs surface, showing a fault with the algorithm. Most of the faults are fairly insignificant.

4.3 Final Review of an Algorithm

Once an algorithm is a clear leader, appears to perform well in all the requirements and most of the desired properties, the algorithm is inspected for cases where it would fail to produce intelligent results. For instance, if the output from an algorithm exhibits a significant jump discontinuity, then there must exist a good explanation for it. An algorithm is coded and used if and only if it holds up to this level of scrutiny from two or more people.

Chapter 5

Score Flow

This chapter describes how the non-numerical predictive model provides a score to a Template. First, the process of building a prediction is reviewed, followed by the actual scoring algorithms. The scoring algorithms consist of five parts: information retrieval, evaluation of that information, resolving multiple hits, the aggregation of the score up the Template structure, and then the final scoring process used to assign a number to a Template to help sort it.

5.1 Building a Prediction

There are two prerequisites to running a prediction through the non-numerical predictive model. Entities must be extracted from the underlying non-numerical data and a prediction must be codified into a Template.

5.1.1 Entity Extraction

Entity extraction is a very important prerequisite. These entities are those that the Template's SIRs and SORs queries match against. If there are no entities, then the queries will fail to find anything. If the queries fail to find anything, then the non-numerical predictive model will provide the same assessment and certainty (neutral with no certainty) for all Templates.

Entity extraction can be done both automatically and manually. Automatic entity extraction has several limitations at this point. While it can identify many proper

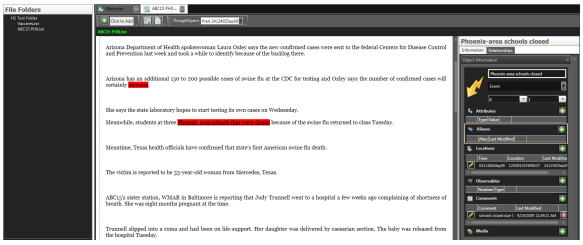


FIGURE 5.1. Text-Highlighter

nouns in text, it cannot currently classify the type of entity the proper noun represents. Furthermore, it currently cannot automatically attach relevant information from the document to the entity either. This means that much of the entity extraction must still be done by humans.

ATRAP has a text-highlighter tool to assist in manual entity extraction (see Figure 5.1). Once extracted, the entity can have additional information about it specified, including times and locations. Additionally, keywords and relationships to other entities may be included at this time. Once the entity has been extracted by any user and it resides in the ATRAP database and is visible to all users. This allows several users to build the ATRAP database.

5.1.2 Template Design

Templates are split into two categories: doctrinal and situational. A doctrinal Template is an outline or an abstraction of a specific behavior. It reflects the doctrine of an individual, organization, or nation. A situational Template is an applied doctrinal Template. It takes the abstraction of a behavior and fills in all the specifics (when, where, and other details). For example, a list of instructions for crossing rivers would be reflected in a doctrinal Template while the steps executed when crossing a specific river would be exhibited by a situational Template.

There is a tool called the "Template Builder" inside of ATRAP where these

(a) Template Builder blocks (b) Prediction Run

FIGURE 5.2. Template Builder and Prediction Run

Templates are built. Inside the Template Builder a user can load or copy existing Templates, create new Templates, or make a doctrinal Template a situational Template. When creating a Template, there are Indicators, SIRs, SORs, Decisions, and Options. The Decisions and Options are currently not used because they do not extend functionality. For more details on Templates please refer to Section 3.2. The SIR queries are shown specified as shown in Figure 3.3(b) and the SOR spatial-temporal constraints are entered as shown in Figure 3.3(a).

Once there are entities in the database and there are Templates, one or more Templates can be evaluated in a "Prediction Run." In ATRAP, a prediction run is initiated by selecting the "Prediction" tab from the bottom left corner of ATRAP, selecting the Templates which are to be compared against one another, and clicking "Run."

5.2 Information Retrieval

The first question to answer was how to take the query information from the SIRs and SORs and pull the appropriate information from the database. Since natural

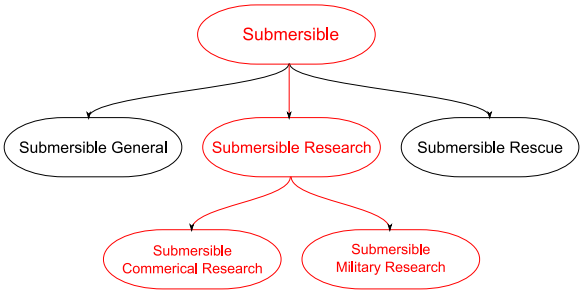


FIGURE 5.3. Example use of Entity Type Ontology

language processing is a very difficult problem, the system requires some degree of manual entity extraction.

There are three primary fields for SIR queries: entity type, keywords, and relationships. The entity type field specifies what is being searched for: a location, an attack, an IDE, etc. The entity type can optionally be expanded to include more specific and less specific entity types. For instance, if the user specifies an entity type of type "vehicle" and includes more specific types, the system may return a hit for an entity of type "truck." Figure 5.3 depicts the expansion of "Submersible Research" in red, including one level of generalization and specification.

The recognition of generalizations and specifications is achieved by using an entity type ontology, where the higher levels are more generic and the lower levels are more specific. This tool improves recall and trades off between recall and precision. Recall is a measure of the completeness of the search, and precision is a measure of true positives:

$$\text{recall} = \frac{|\{\text{relevant matches} \cap \text{matches found}\}|}{|\text{relevant matches}|} \tag{5.1}$$

$$\text{precision} = \frac{|\{\text{relevant matches} \cap \text{matches found}\}|}{|\text{matches found}|}. \tag{5.2}$$

By searching for more specific types in addition to the specified type, the recall is improved with a minimal impact to precision. Since a generalization provides less

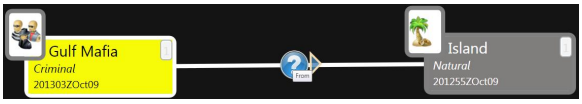


FIGURE 5.4. A Relationship Between Two Entities

information and may provide a false positive, a parameter has been introduced to provide a penalty for generalizations (see Section 5.3).

Keywords are useful for specifying additional details not captured by the entity type, location, and relationship to other entities. The search can optionally be expanded using inflections of keywords. This allows the search to look for singular/plural and different tenses of the provided keywords. While not currently implemented in the second revision of ATRAP, the optional use of a thesaurus has been discussed to improve the recall at the expense of precision of such queries.

The last part of the SIR query, entity relationships, is an incredibly powerful tool. This enables the user to specify a query where entities have relationships among each other. These relationships can be specified by either relationship type or to a specific entity. An example query may involve searching for an arms dealer with suspected relations with a criminal organization.

5.3 Evaluation of Retrieved Data & Fuzzy Matching

We are now onto the vectors which provide the basis for the scoring. This section will discuss the evaluation of these vectors, the next section will discuss reducing the problem space, and the following section will describe how they are aggregated to form a vector for a Template.

Once the data and documents have been retrieved which support (or deny) the queries, that information needs to be evaluated and converted into numerical vectors. These vectors are three-dimensional in nature, consisting of: confidence in the infor-

mation, relevance of the match, and the assessment of the match. If the information came from a free-text report, the confidence in the information was entered by a human. If the information came from sensor data, the confidence is assumed to be very high. The relevance reflects how well the query matches the retrieved data. The assessment indicates whether the data supports or denies the query. By default these values are:

$$\begin{aligned} \text{relevance} &= R = 1 \\ \text{confidence} &= C = \text{Copied from information's confidence} \\ \text{assessment} &= A = \pi/4 \end{aligned}$$

A military intelligence SME provided guidelines for evaluating the data retrieved from the query. Some of these behaviors include the impact of fuzzily matching spatial-temporal constraints, fuzzily matching generalizations of the entity types, user adjustable parameters, and methods for tweaking how detrimental a fuzzy match is compared to an exact match. While the guidelines only allow our algorithms to capture the general trends the SME expected, the specifics that the analysts expect can be captured by tweaking or training the adjustable parameters.

The exact variables in the SOR constraint query, A and T, specify the perfect matches. Data retrieved from the database that fulfills these requirements perfectly do not suffer a penalty to their relevance score for the spatial-temporal constraints. However if there is a fuzzy match, their relevance is determined by the user-selectable spatial and temporal falloff functions, $\mathrm{F}_{\mathrm{spatial}}$ and $\mathrm{F}_{\mathrm{temporal}}$), which specify how the relevance transitions from the perfect match to a perfect failure:

$$R = \mathrm{F}_{\mathrm{spatial}}(\mathrm{Distance}) \cdot \mathrm{F}_{\mathrm{temporal}}(\Delta\mathrm{Time}) \tag{5.3}$$

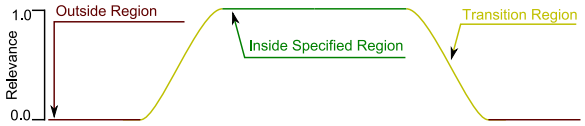


FIGURE 5.5. Example Falloff Curve

A perfect failure is defined as being outside the specified spatial-temporal region even when expanded by the parameters: t^-, t^+, and r. An example falloff function is shown in Figure 5.5.

The relevance score can also be impacted by a generalization of the entity type searched. If the user chooses to include all levels of generalizations of the specified entity type (e.g., truck) when building an SIR query, ambiguous entities may be found (e.g., vehicle). He/She can specify how many levels up the entity type ontology to travel when expanding the search and the penalty for each level. The user can also specify more specific entity types, but typically there is no penalty for more specific types (traveling down the ontology). For example, every level that must be traversed up the ontology (more general terms) to find a match, could multiply the relevance by 0.5. This is demonstrated in:

$$R' = R \cdot \sigma^{\max(0,L)}, \tag{5.4}$$

where R' is the relevance once adjusted for a generalization of the entity type, R is the relevance prior to accounting for the generalization, $\sigma \in (0, 1]$ is a user adjustable parameter which determines how detrimental generalization is, and L represents the number of levels of generalization. Formally, $L = D_s - D_f$, where D_s is the depth of the entity type specified and D_f is the depth of entity type of the query hit. Note that a positive value for L represents a generalization was found, a negative value represents a more specific type, and the value of zero (0) represents exactly the type

specified in the query. Thus, in equation (5.4) regardless the penalty parameter σ, only generalizations suffer from a penalty.

5.4 Problem Reduction

The next issue is reducing the problem space from potentially many three-dimensional vectors into a single twp-dimensional vector for the SORs, that can also be used for the rest of the Template scoring system. It was decided that the confidence in the information could then be combined with assessment and relevance to represent the data as a two-dimensional polar vector for the rest of the scoring procedure. This simultaneously simplifies the task and transforms the problem to something that matches the intelligence analyst SME's thought process better. The two-dimensional polar vector is that the assessment reflects what the underlying information indicates. A measure of certainty indicates the strength in that belief. We introduced Cobb-Douglas type products to accomplish this transform.

5.4.1 Cobb-Douglas-Products

Cobb-Douglas type products were selected primarily due to their behavior and wide acceptance in business and financial environments [6]. The Cobb-Douglas product is a weighted multiplication. Equation (5.5) shows a Cobb-Douglas product (CDP), where $\underline{x} = (x_1, x_2, \ldots, x_n)$ is a vector of components to be combined and $\underline{a} = (a_1, a_2, \ldots, a_n)$ is a vector of weights, both of length as n, and a constant A which defines the product when $(\forall i, x_i = 1)|(\text{CDP} = A)$. For our purposes we always let $A = 1$, since that corresponds to a perfect match:

$$\text{CDP}\,(\underline{x},\underline{a}) = A\prod_{i=1}^{n} x_i^{a_i}. \tag{5.5}$$

The Cobb-Douglas production model has traditionally been used in economics as a method of estimating productivity. In the economic case, it is common for there to be only two quantities multiplied: a measure of the employees and a measure of equipment. The main question that the literature has been trying to answer, since the Cobb-Douglas production model was introduced, is how to choose the weight values. The method for determining these weight values have progressed from simple regression techniques to Bayesian estimation techniques [14,26]. In our current version of the NNPM, there are a few methods in which a value can be assigned: by the user with a feeling for the meaning behind the $\underline{a}$ values, by regression methods, or by non-linear programming methods.

The $\underline{a}$ values have a real meaning. Via the manipulation shown in equation (5.6), a_i can be solved. This equation shows that each a_i is the ratio of the CDP relative growth rate to the relative growth rate of x_i, which is called the elasticity. Thus if an analyst has a feeling about these relative growth rate ratios, he can directly provide the $\underline{a}$ vector. Since:

$$f(\underline{x}) = A\prod_{i=1}^{n} x_i^{a_i}$$

simple differentiation shows that

$$\frac{\partial f(\underline{x})}{\partial x_i} = A\left(\prod_{j\neq i} {x_j}^{a_j} \;\cdot\; a_i {x_i}^{a_i-1}\right)$$

and so

$$\frac{\frac{\partial f(\underline{x})}{\partial x_i}}{f(\underline{x})} = \frac{a_i}{x_i}.$$

This relation can be rewritten as

$$a_i = \frac{\frac{\partial f(\underline{x})}{f(\underline{x})}}{\frac{\partial x_i}{x_i}} \tag{5.6}$$

Alternatively, one can see that the log of the CDP is linear in terms of a_i and the log of A:

$$\log\left(\text{CDP}\left(\underline{x}, \underline{a}\right)\right) = \log\left(A\right) + \sum_{i=1}^{n} a_i \log\left(x_i\right). \tag{5.7}$$

If A is not assumed to be 1.0, it can be treated as one of the values to be solved for in the least squares regression. This provides a linear least square solution for the logarithm of the CDP. An alternative method is the use of nonlinear regression, where the least square solution to the CDP requires minimizing equation (5.8). The primary difference between these two regression methods is the definition of the nonlinear error term:

$$\sum_{t=1}^{T} \left(f^{(t)} - A \prod_{i=1}^{n} x_i^{(t)\, a_i} \right)^2 \tag{5.8}$$

By further redefining the error as the L_1 norm or the L_∞ norm and using the equation (5.7), the $\underline{a}$ values can be obtained from a linear programming problem. Other definitions of error would most likely transform the problem into a non-linear programming problem.

5.4.2 Resolving multiple query matches

However the Cobb-Douglas Products is only half of the process, reducing the dimensionality to a domain more familiar to the hypothesis domain for analysts. Our intelligence analyst SME and team developed 11 key constraints and desired behaviors for resolving multiple hits. A few properties include: conflicting information should lower the confidence, order of elements should not matter, model dimensioning returns when there are many hits, various behaviors for different quality matches, and the simpler the better. When all aspects were combined, competing algorithms were then evaluated based on understandability and simplicity.

The current procedure proceeds as follows (see Pseudocode 5.6). First, the dimensionality is reduced to a two-dimensional polar vector using Cobb-Douglas Products. Next, the lengths (certainty) of all vectors are stretched by using the inverse of a user selectable sigmoid function (sigmoid functions model diminishing returns well). All vectors are then summed and the resulting vector is normalized using the user-selectable sigmoid function. Lastly if two or more pieces of information are in conflict, then the length of the normalized resulting vector is adjusted to reflect disagreeing data.

Since the last prototype of the model, we have talked with two SMEs for some additional constraints. This included methods for determining redundant data, measure of similarity, and a more statistical method for fusing the data based on the independence of the data. If a method for determining the dependence of two text-based pieces of information can be formulated, verified, and tested, then a statistical solution can be used for resolving multiple query hits. The intelligence analyst would then be able to select which algorithm to use for this part.

```
Vect3D scoredDataVectors;
UserParameter a, b, c, d;
UserFunction sigmoid;

//Reduce the dimensionality of each vectorized query match
List<Vect2D> polarVectors;
foreach( dV in scoredDataVectors ){
   certainty   = Cobb-Douglas( { dV.confidence, dV.relevance  }, {a, b} );
   assessment  = Cobb-Douglas( { dV.confidence, dV.assessment }, {c, d} );
   polarVectors.insert( new Vect2D( assessment, certainty ) );
}

//Stretching, adding, and normalizing the new 2D vectors
Vect2D vectSum(0,0);
foreach( pV in polarVectors ){
   certainty   = stretch( pV.certainty, sigmoid.inverse );
   assessment  = pV.assessment;
   vectSum     = vectorAdd( vectSum, new Vect2D( assessment, certainty ) );
}
vectSum.certainty   = stretch( vectSum.certainty, sigmoid );

//Determine general sway of the information & the data in conflict
double positiveSum  = negativeSum = runningSum = conflictValue = 0.0;
foreach( pV in polarVectors ){
   double yComp     = pV.certainty * sin( pV.assessment );
   runningSum      += yComp;
   if( yComp > 0)
      positiveSum  += yComp;
   else
      negativeSum  -= yComp;
}
if( runningSum > 0 )
   conflictValue    = negativeSum;
else
   conflictValue    = positiveSum;

//Reduce the certainty of the prediction by the amount of conflicting data
vectSum.certainty  /= ( 1 + conflictValue );
SOR.Vector          = vectSum;
```

FIGURE 5.6. Code outline

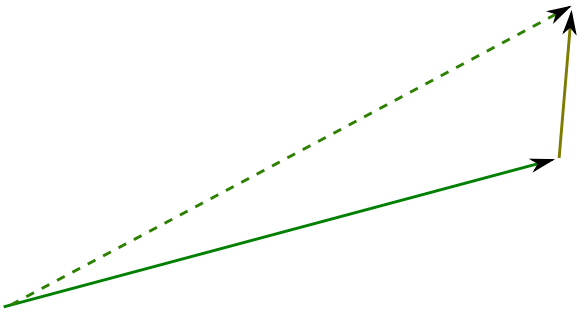


FIGURE 5.7. Interaction of certainty and assessment in score aggregation

5.5 Score Aggregation

Once a two-dimensional polar vector is obtained for each SOR, the system must propagate the scores to higher levels to answer the SIRs, Indicators, and eventually the Template. The aggregiation method is the activation function present at each node in the Template.

To make the activation function behave as a generally accepted artificial neuron response [10] without truncating information, a piece-wise linear activation function with thresholds was chosen. We extend this into two-dimensions by using weighted vector addition, where the certainty represents the length and the assessment represents the angle of a vector. Figure 5.7 demonstrates vector addition before normalization, where the dotted vector is the un-normalized result. This shows that vectors with greater certainty contribute more to the resulting score vector's assessment (direction). However, this is only the behavior within the linear region of the activation function, which is related to its slope. By default, the slope of the linear region of the function is determined by the inverse of the sum of the absolute values of the weights of all the child links. This has the affect of ensuring that by default the activation function only operates in the linear region. Thus, the resulting operation performs a weighted average if the threshold is either unspecified or equal

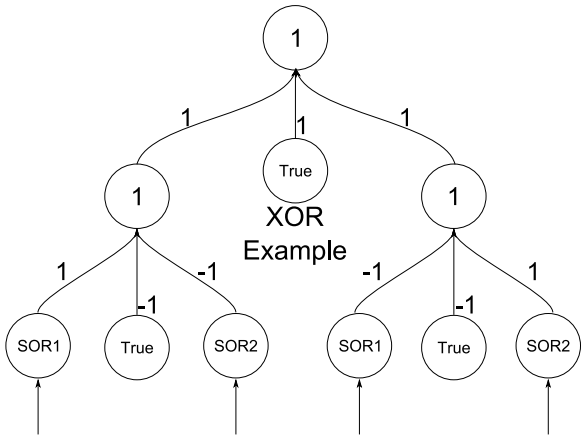


FIGURE 5.8. Weights and Thresholds for an XOR operation

to the sum of the absolute values of the weights. When the threshold is reduced to a value lower than its default value, the network behaves like a QNN.

The QNN structure of the Template is a very useful tool for performing fuzzy logical operations between different SORs and SIRs. Since all Templates have at least three layers (Indicators, SIRs, SORs), it is possible to perform complex logical operations such as an XOR. In Figure 5.8, the numbers inside each node represent the threshold values, the numbers along the connections show the weights, and the "true" nodes refers to a vector having maximum certainty and assessment (1, $\pi/2$).

The behavior for the threshold is subject to change, but one of the latest proposed methods is described here (see Appendix A). Basically it checks if the weighted sum of the vectors appears to be too ambiguous to apply any sort of thresholding effect. If the resulting vector is unambiguous, then the length of the vector is extended and the assessment is improved. The details are as follows.

The first step is to sum up all vectors multiplied by their weights. Determine the magnitude of the resulting vector and decompose the resulting vector into its Cartesian components (confirm-deny data and relevant but assessment-neutral data). If the confirm-deny component (y) divided by the threshold is greater than the

magnitude of the un-normalized resulting vector ($y/threshold > magnitude$), then proceed with the rest of this procedure. Otherwise, since the vector is considered too ambiguous, use the previously discussed activation function. Adjust the magnitude by the adaptive formula:

$$magnitude' = magnitude + \alpha \cdot (y/threshold - magnitude), \tag{5.9}$$

where $\alpha \in (0, 1]$ is a constant which defines how fast the threshold function thresholds, and $y/threshold - magnitude$ represents an amount of improvement. The improvement is always positive if the vector is considered unambiguous. This measure of improvement was chosen because of two properties. First, the algorithm converges to the previously discussed algorithm when the threshold is set to the sum of absolute values of the weights. Secondly, at $\alpha = 1.0$, the magnitude becomes the confirm-deny component divided by the threshold value (the usefulness of this will be shown later). Essentially, equation (5.9) sets the magnitude to the weighted average of its previous value and the amount of improvement. Then the confirm-deny component is adjusted:

$$y' = \frac{y}{\sum \left|\overrightarrow{weights}\right|} + \alpha \cdot (magnitude' - magnitude), \tag{5.10}$$

where $\overrightarrow{weights}$ corresponds to the weight vector and y' is the new confirm-deny component. Equation (5.10) performs a weighted average between the confirm-deny component before normalization and the amount by which the magnitude increased. If $\alpha = 1.0$, then y becomes the full length of the vector, which translate to a vector with maximum assessment. By replacing $magnitude'$ in equation (5.10) with its

definition in equation (5.9), we have

$$y' = \frac{y}{\sum \left|\overrightarrow{weights}\right|} + \alpha^2 \cdot (y/threshold - magnitude). \tag{5.11}$$

In the next step, we ensure the vector does not exceed the maximum length. This is accomplished done by clamping both the magnitude and the confirm-deny component to no higher than 1.0. Lastly the relevant but assessment-neutral component, (x') is calculated to preserve the magnitude:

$$x' = \sqrt{Magnitude'^2 - y^2} \tag{5.12}$$

The net result of this thresholding algorithm is as follows. Unambiguous vectors have their confidence increased in proportion to α and the measure of improvement. The assessment improves in proportion to α^2 and the measure of improvement. If $\alpha = 1.0$, the resulting vector is a vector composed entirely of just a confirm-deny of $y/threshold$. This is useful because it behaves like a hard-thresholding "or" function. This algorithm has been coded up in Matlab to test and verify its behavior (see Appendix A). Appendix A also has a simpler algorithm which appears to have similar properties.

5.6 Template Score & Ranking

The last step in evaluating Templates is providing scores and ranking all of the Templates selected for ranking. The first part of this procedure is identical to the aggregation of the score, where all of the Template's Indicator vectors are averaged to produce an assessment and certainty. This produces a final two dimensional vector which must be transformed into a one dimensional number for ranking purposes.

A single number is generated from taking the projection of the certainty and assessment onto an axis. There are three reasons why a projection is used instead of the aforementioned Cobb-Douglas product. First, projection is more intuitive due to the polar nature of the vectors. Secondly, both have similar behavior if at least one number becomes zero. Additionally, projection is simpler when working with negative numbers. The resulting number can then be used for ranking purposes.

There is one more step when it comes to Template scoring. It is dangerous to report a number which ranges between -1.0 and 1.0, since the number can be misinterpreted. Positive scores could accidentally be taken as likelihoods. Thus the last step is a monotonic distortion to prevent the user from interpreting the score as a percent likelihood:

$$\text{Confidence} = -100 \cdot \log\left(0.5 - 0.5 \cdot \|\overrightarrow{TempScore}\| \cdot \sin\left(\angle\overrightarrow{TempScore}\right) + \epsilon\right) \quad (5.13)$$

The monotonic property preserves the relative order of the Templates. In equation (5.13), the $\overrightarrow{TempScore}$ represents the polar two-dimensional vector representing the Template's assessment ($\angle\overrightarrow{TempScore}$) and certainty ($\|\overrightarrow{TempScore}\|$), and ϵ represents a small number to prevent taking the log of zero. Since the certainty ranges from 0.0 to 1.0 and the assessment ranges between $-\pi/2$ and $+\pi/2$, the minimum confidence is $-100\log(1+\epsilon) \approx 0$ and the maximum is $-100\log(\epsilon) \approx \infty$.

Chapter 6

Experimentation & Results

Several experiments were conducted to help verify the proper functioning of the non-numerical predictive model and to show validity of the model. Since the ATRAP project is still looking for a large database of non-classified data to run predictions against, most of the data in the experiments was fabricated for the purpose of verifying the proper functioning of the algorithms. There is one real on-going prediction being made with ATRAP as well. This "live" experiment is part of the on-going project to validate the model.

6.1 Experiment Setup

Several experiments were performed to verify the predictive model. Of the experiments that were performed, there are five categorizes of verification experiments: *white*, *black*, *gray*, *mixed*, and *live* test cases. The first three (*white*, *black*, and *gray*) were part of the first tests to verify if the algorithms were working. A *white* test case has a database containing information to confirm a Template. If the algorithms are functioning as they were designed, most to all of the indicators should have a confirming assessment with at least a moderately high certainty. Another type of test case is the *black* test case. It involves a database with irrelevant material for the Templates being scored. Few to none of the indicators should have any assessment other than neutral and all certainties should be extremely low or non-existent. There is a *gray* test case, which uses a much larger database, with both

relevant and irrelevant data. The purpose of the gray case was to test how well the system scales to having to search through more data. Since the gray case includes parts of both of the *white* and *black* test case data, the expected outcome should be between the two other cases.

In another batch of experiments the database remains constant, but several different Templates were evaluated at the same time in a *mixed* scenario. This involved holding the database constant, but sweeping what was being searched. In the *mixed* test case, our intelligence analyst SME created a gang-recruitment scenario. There were three different gangs, three different schools, and several events at each school. The three gangs are: the Right Hand of Freedom (Right), the Sinister Lifestyle Defenders (Left), and the Free Hand of Choice Fellowship (Ambidextrous). The three high schools from which the gangs might try to recruit from are: Leesville, Polk, and North. This generated a total of nine Templates. The data was mostly randomly distributed, but there was a heavy focus on the Sinister Lifestyle Defenders at the Polk High School. There were also smaller clusters of information. For example, North High School had much more activity around it than the other schools.

There was also a *live* test case where the non-numerical predictive model was run against a developing scenario (as of October 2009 and updated for January 2010): H1N1 outbreak on the University of Arizona (UA) and Arizona State University (ASU) campuses. The objective was to determine how serious an outbreak might be based on its location. The source documents include a transcript from Phoenix ABC, a transcript of the delay of H1N1 vaccines, and some of the weekly flu updates from Arizona Department of Health Services. Entity extraction and Template design were performed manually. The example Template in Figure 3.2(b) shows the Template for the UA.

There is the possibility of a few other types of experiment which were not yet

performed, but could be constructed. A *red* case would involve information that is relevant to the Templates, but the information denies the Template. This is different from the other cases because the assessment should be "less" than neutral with at least a moderate level of certainty. For this to work, the Template would have to make use of negative weights. Another type of experiment that could be constructed and tested is a mixture of the *red* and *white* case. This hybrid case would involve relevant data that both denies and confirms different parts of the Template. The expected outcome of this type of experiment should be a Template score near neutral assessment with a moderate level of certainty. However, there are three places in a Template where the data can be in conflict. Thus it is helpful to label the three additional cases as: *contradictory-SOR*, *contradictory-SIR*, and *contradictory-Indicator* cases. A *contradictory-SOR* case would involve an SIR with both positively and negatively weighted SORs, both returning matches. A *contradictory-SIR* case would involve an indicator having positively and negatively weighted SIRs, both of which return a positive match to the indicator. Likewise, a *contradictory-Indicator* would involve any two indicators having nearly opposite assessments.

6.2 Results

All results are summarized in Table 6.1. For the first batch of experiments, the *white*, *black*, and *gray* results are shown. As anticipated, the *white* case performed well, achieving an extremely high score. To confirm this score, we checked the indicators which all performed very well as well. Likewise all their children also scored highly. The average Indicator assessment was approximately +0.79 (appears to be happening) with very high certainty. The *black* case performed as hoped, all Indicators had a neutral assessment of 0.00 (non-negative and non-positive) with very low certainty.

There were no matches. The *gray* case did not take noticeably longer to run (no times were recorded), despite the database being an order-of-magnitude larger. Some of the Indicators found relatively good matches while other found little to no data.

The *mixed* case provided a spectrum of results. The top ranking Template from this batch is the Template that looks at Sinister Lifestyle Defenders's activity around the Polk High School. There were two Templates tied for a distant second place ranking. Most Templates performed better than the previously mentioned *black* test case, but were inferior to the *gray* test case. Almost all information found confirmed gangs were trying to recruit from the schools, but most of the evidence was limited, producing the low Certainty scores shown. These results reflect what the subject matter experts anticipated.

The *live* case showed that there was more information indicating ASU would suffer an H1N1 outbreak rather than the UA. Both universities showed similar assessments (confirming it will happen), but there was more evidence for ASU (reflected in the certainty score). The confidence measure shows that the non-numerical predictive model (using the provided data and the provided Templates) holds more confidence in the Template describing an H1N1 outbreak at ASU than the Template describing an H1N1 outbreak at UA. While there is no direct data on the campuses to validate the NNPM, there is data associated with the counties.

In an attempt to measure the severity of H1N1 outbreaks on the ASU and UA campuses, the suspected/confirmed cases for Pima and Maricopa were collected from [16]. Pima is where the University of Arizona resides and ASU is in Maricopa county. Since the counties differ in size, the populations are also examined for a fairer analysis. The official US government census [1] was used to provide population estimates. This data has been combined and summarized in Table 6.2. While Maricopa (closer to ASU) has many more cases than Pima (closer to UA) in total, the ratios of cases is

TABLE 6.1. Template Score Results

Test Case	Assessment	Certainty	Confidence
White	0.79	1.00	193.22
Black	0.00	0.00	69.31
Gray	0.78	0.35	97.57
Left - Polk	0.75	0.77	143.19
Left - Leesville	0.79	0.15	80.53
Left - North	0.79	0.19	93.16
Right - Polk	0.79	0.15	80.53
Right - Leesville	0.79	0.15	80.53
Right - North	0.79	0.30	93.16
Ambidextrous - Polk	0.00	0.00	69.31
Ambidextrous - Leesville	0.00	0.00	69.31
Ambidextrous - North	0.79	0.15	80.53
Live (ASU)	0.79	0.57	120.95
Live (UA)	0.78	0.35	97.33

TABLE 6.2. Arizona County Data

County	Total cases	60-day span cases	2000 Census	2008 estimate
Maricopa	7,819	41	3,072,149	3,954,598
Pima	2,150	2	843,746	1,012,018
Ratio	3.6367	20.5	3.6411	3.9076

nearly the same as the ratio of the populations. However, the Template for H1N1 anaylsis was updated in early January of 2010; examining all cases does not reflect the NNPM's prediction. However, in the a 60-day span (from December 27, 2009 to February 25, 2010), even though Maricopa has had only 41 cases, this is more than 20 times the suspected/confirmed cases in Pima. The population ratios cannot account for this discrepancy. Thus, it this result helps to validate the NNPM.

Chapter 7

Behavioral Filtering

Another on-going research aspect of ATRAP is enhancing the predictive capabilities (Template evaluation) by making use of behavioral data. Behavioral filtering would adjust the certainty and assessment scores of each prediction modeled by a Template. The next section describes the problem in more depth. This is followed up with a description of a prototype expert system.

7.1 The Problem

Behavioral data comes at a cost. Investigation into an individual's or agency's background takes time and could cost a lot of money. Sometimes it is impractical to gather the level of information needed for a good prediction of an entity's (person, organization, etc.) actions. Police cannot interview or perform a comprehensive psychological analysis on an uncaught criminal. Thus behavioral data must be able to be estimated from a person or organization's actions or estimated from demographics. Additionally, such estimations may contain a great deal of error.

Determining which attributes to use for enhancing the predictive capability of ATRAP is problematic. The central problem to the behavioral aspect of ATRAP is optimizing the prediction accuracy per the difficulty, time, and cost of obtaining the behavioral information. Thus, the characteristic attributes must be measurable (even if the entity is unwilling to cooperate), cheaply obtainable, and provide a great

deal of predictive capabilities. Another problem involves designing robust prediction methods that work well even in the presence of flawed or incomplete behavioral data.

To help ensure a valid solution, the ATRAP team is currently seeking out SMEs in the behavioral field. The SMEs would assist in the selection of attributes as well as aid in the development of an expert system to use the attributes.

7.2 Expert System Framework

A very powerful and flexible prototype expert system framework has been developed to aid in the processing of the behavioral data. The expert system framework is effectively an inference engine builder (see Figure 7.1(a)). The inference engine builder has four primary inputs:

- syntax - Describes types and values for attributes
- attributes - Describes what the attributes entities have
- entities - Names of the input variables to the expert system
- rules - Basis of the expert system

These four modules allow the prototype to build an inference engine with several additional features. The inference engine (see Figure 7.1(b)) does not just execute rules but can also output an adjusted two-dimensional vector and a list of messages to show the user if certain rules fire and why. More formally, the inference engine generated has the following:

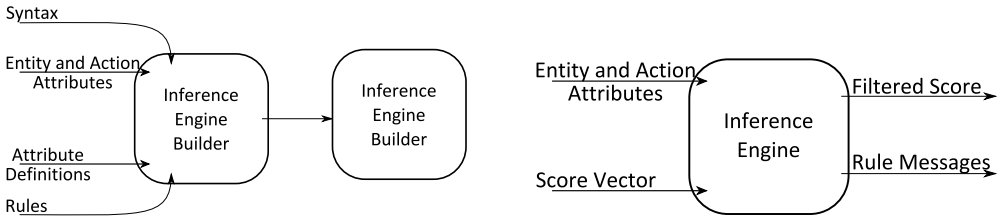
- entities - The input/output variables
- messages - An expert's messages that were triggered as the engine ran

- two-dimensional vector - An input/output ATRAP score vector

The syntax allows the rules to be written in an English-like language. Each non-numerical attribute has its syntax defined in the syntax component. Effectively, the syntax provides the programming equivalent of enumerated strings (enums), which the attributes can reference. Together this allows much of the rules to be written in an English-like language.

The attributes provide a generalized framework for any entity, event, or action to be passed into the inference engine. By taking the union of several abstracted objects, the inference engine can work with diverse sets of objects by treating them all as the same thing inside the engine. For example, an entity might have a desire for publicity and actions also have a certain amount of publicity associated with them. The resulting inference engine would be able to work with these two very different data types (entity and action). Additionally, all the attributes have two dimensions to them: their value and the confidence in that value. There is a trade-off for this additional degree of flexibility. This allows the expert to write rules that take uncertainty into account, at the cost of added complexity for the expert.

Three separate aspects make the rules very flexible and powerful. First, each set of rules starts with a list of aliases for the entities to be passed into the system, which can then be used throughout the rest of the rule set. This allows the rule developer



(a) Prediction Run (b) Template Builder blocks

FIGURE 7.1. Inference Engine Builder and Inference Engine

to use his/her own naming scheme while writing the rules. This feature also allows the inference engine to work on rules with an arbitrary number of entities.

Another aspect which makes the rule system powerful is the reading and writing to the entity's attributes. This allows for the modification of the entities during their processing. After the engine has exhausted all available rules, it is possible to read these modifications out of the inference engine entities and back into the rest of the ATRAP system. Due to the arbitrary number of entities that can be entered into the engine and the read-write capabilities, it is possible to introduce a dummy variable for keeping track of anything the author of the rules might want.

The third powerful aspect of the rules is the fact that they are modular. Modular rules allow for several classes of rules to be written for various scopes and offer the promise of hierarchical rule sets. One set of rules could be global while another set could be specific to a region or paradigm.

The two-dimensional vector that can optionally be passed in is a polar vector representing the assessment and certainty of a particular Template. These two components are treated as special variables inside the rule sets, which allow an expert to directly tweak the assessment and certainty as seen fit. If more than two variables are ever needed in the future, a dummy entity could be input into the system and used as an additional IO.

Chapter 8

Conclusions

We have developed and tested a non-numerical predictive model (NNPM) which makes use of non-numerical data to test predictions codified as Templates. Because the NNPM uses a Template schema with fuzzy matching, its applications are not limited to military intelligence. Virtually any domain which uses significant non-numerical data can benefit from this predictive model since hypotheses can be codified as Templates. These applications range from: criminal investigations, financial markets, homeland security to evaluating political outcomes. One specific application is that for ATRAP, a practical and flexible toolbox for intelligence analysts.

We worked with SMEs to help develop an NNPM that intelligence analysts could use. The model automatically generates three metrics for evaluating hypotheses (Templates). Our NNPM generates an assessment of the apparent truth of the prediction, a certainty or precision of that assessment, and a unified score called confidence. The algorithms have been tested and verified via the use of ATRAP. The experimental results have been shown to reflect an SME's expert evaluation of the same data. The *live* test reflected real world data, further validating the model.

Our NNPM offers several important features. It overcomes the three primary hurdles associated with modern intelligence analysis: asymmetry, rapidly changing tactics, and information overload. Compared to other works, it offers: extensive user customizability; COAs that go beyond tactics to include the generation of information, new COAs, and an evaluation; an information management framework; asymmetric and human-in-the-loop capabilities. While there are other programs which offer

several of these features, none of them incorporate all of these features into a single package. Our NNPM, functioning inside of ATRAP, offers all of these features in a single package.

The score flow consists of two prerequisites and five steps. The first prerequisite is that entities must be extracted for the non-numerical predictive model to match queries against. Additionally, predictions must be codified as Templates. The SIR queries can make use of an ontology, inflections of keywords, and relationships to other entities to help provide retrieve as much relevant information as possible. Then the retrieved information is evaluated to determine how well the data matches. The dimensionality is reduced and multiple hits are resolved into a single score for each SOR. The scores propagate up the Template structure, performing operations from neural networks. Lastly the assessment and certainty are combined and monotonically distorted to prevent misinterpretation of the output value.

The model is made malleable from its many options, including: an entity ontology to optionally expand search parameters, adjustable Cobb-Douglas products, various sigmoid and falloff functions, and weights and thresholds for Indicators, SIRs, and SORs. It provides means for automatically surveying codified hypotheses (Templates) by providing an assessment and confidence from non-numerical data. The predictive model has primarily five parts: information retrieval, evaluation of the retrieved data, resolving multiple hits, score propagation, and final score generation.

Future versions of the non-numerical predictive model will include further validation of Template scoring algorithms, computational scalability with orders-of-magnitude larger datasets, integration of the inference engine and other possible improvement of the Template model.

8.1 Future Research Directions

The NNPM could be extended in many different directions. Some are directions include: detecting "duplicate-like" information and accounting for it; introducing and processing of dependence among query matches; using machine learning to optimize all the parameters; generalizing the Template schema for improved flexibility and decreased time codifying hypothesis; integrating the behavioral inference engine; and improving data-fusion with stochastic linear motion models.

8.1.1 Duplicate-like Detection

One direction which has been investigated but not implemented, tested, or verified, is a "duplicate-like" detection tool. A duplicate-like detection tool would find sets of entities that are inexact duplicates, meaning that entities ought to really be one entry in the database instead of several. This tool would check various factors between entities such as:

- analyst who created the entity
- analyst level
- creation time
- name
- location
- time
- type
- source documents

This list is not exhaustive, and there are potentially many more factors to consider. By examining these factors we should be able to determine if entities are duplicate-like. For example, two different users (with very different ranks) create two very similar entities, sharing source documents, type, time, location, and similar names. It is reasonable to say these two entities might be duplicate-like.

The detection process could be on-the-fly (entities matching a query) or offline (when the system is idling). There are advantages to both approaches. First, on-the-fly detection could operate much faster by comparing smaller subsets of entities (if there N entities, each must be compared against $N - 1$ entities). On-the-fly detection could operate upon entities found matching an SIR/SOR query, which would automatically limit the number of entities. Also, if the entities are checked in real-time as they are added to the database, we could alert the user to a possible preexisting entity as soon as possible.

There are a few advantages to the offline detection process. The user's would never have to wait for the system to perform the checks, whether during Template evaluation or during entity extraction/entry. It would spare memory and cpu-time when the system is being heavily used. When the system is idle, the system can examine any entities which were introduced or changed to check for near duplicates.

8.1.2 Accounting for Dependence

The introduction and processing of dependence is another future research direction for the NNPM. When a single SOR query finds multiple matches, it is possible some of those events are not independent. For example, a search for a red truck in a small vicinity, over a large span of time might show multiple matches not because there are many red trucks, but because the same red truck has been spotted on several different occasions or even duplicate information entered into the database.

There is an analogous problem with random variables. This is the problem of linearly combining random variables to minimize the variance. With regards to query hits, the variance is a measure of uncertainty. The exact relationship between certainty and uncertainty is still currently an open question. Regardless, the basic idea is to combine query hits in a similar manner to the process used for combining unbiased random variables to obtain an unbiased estimate with minimum variance (or in our case minimum uncertainty):

$$Z = \sum_{i=1}^{n} \alpha_i X_i. \tag{8.1}$$

In equation (8.1), Z is the unbiased estimate (combined score), X_i are the random variables (query matches), and the α_i are the weights used to combine the X_i's. Solving for the α_i's that produce the minimum variance is trivial for a set of variables (query matches) that are mutually independent. The solution to the α_i's that produce the minimum variance estimate of Z (assuming independent random variables) is given by

$$\alpha_i = \frac{1}{\sigma_i^2 \sum_{j=1}^{n} \frac{1}{\sigma_j^2}}, \tag{8.2}$$

The process becomes complicated when some of the query hits are dependent. Our mathematics SME has provided a derivation for the closed form solution, which is presented below. In the following derivation (equations (8.3) – (8.12)), $X_1, X_2, \ldots, X_n$ are random variables such that $\mathrm{E}[X_i] = \mu$, $\mathrm{Cov}[X_i, X_j] = \sigma_{ij} = \sigma_{ji}$, $\mathrm{Var}[X_i] = \sigma_{ii} = \sigma_i^2$, $\underline{\alpha} = (\alpha_1, \alpha_2, \ldots, \alpha_n)^\intercal$, and λ is the Lagrange multiplier.

An unbiased estimate requires:

$$\mathrm{E}\left[\sum_{i=1}^{n}\alpha_i X_i\right] = \sum_{i=1}^{n}\alpha_i \mathrm{E}[X_i] = \mu\sum_{i=1}^{n}\alpha_i = \mu. \tag{8.3}$$

Thus, the constraint is

$$\sum_{i=1}^{n}\alpha_i = 1. \tag{8.4}$$

The variance of our estimate is:

$$\begin{aligned}
\mathrm{Var}\left[\sum_{i=1}^{n}\alpha_i X_i\right] &= \mathrm{E}\left[\left(\sum_{i=1}^{n}\alpha_i X_i - \sum_{i=1}^{n}\alpha_i\mu\right)^2\right] \\
&= \mathrm{E}\left[\left(\sum_{i=1}^{n}\alpha_i\left(X_i-\mu\right)\right)^2\right] \\
&= \sum_{i=1}^{n}\alpha_i^2\sigma_i^2 + \sum_{i=1}^{n}\sum_{j\neq i}\alpha_i\alpha_j\sigma_{ij} \\
&= \sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j\sigma_{ij} \\
&= \underline{\alpha}^{\intercal}\underline{\underline{C}}\,\underline{\alpha},
\end{aligned} \tag{8.5}$$

where $\underline{\underline{C}} = (\sigma_{ij})$ is a symmetric matrix.

Minimizing the variance produces the following constrained optimization problem:

$$\begin{aligned}
\text{minimize} \quad & \underline{\alpha}^{\intercal}\underline{\underline{C}}\,\underline{\alpha} \\
\text{subject to} \quad & \underline{1}^{\intercal}\underline{\alpha} - 1 = 0,
\end{aligned} \tag{8.6}$$

which by the Lagrange method is reduced to the following unconstrained optimization problem:

$$\text{minimize} \quad \underline{\alpha}^{\intercal}\underline{\underline{C}}\,\underline{\alpha} + 2\lambda\left(\underline{1}^{\intercal}\underline{\alpha} - 1\right). \tag{8.7}$$

Differentiating with respect to α_i (note $\underline{\underline{C}}$ is a symmetric matrix) and λ gives us:

$$2\underline{\underline{C}}\,\underline{\alpha} + 2\lambda\underline{1} = \underline{0} \tag{8.8}$$

$$\underline{1}^{\intercal}\underline{\alpha} - 1 = 0. \tag{8.9}$$

To solve for $\underline{\alpha}$ in closed form, take equation (8.8) and multiply by $\underline{\underline{C}}^{-1}$ from the left:

$$\underline{\alpha} + \lambda\underline{\underline{C}}^{-1}\,\underline{1} = \underline{0},$$

so

$$\underline{\alpha} = -\lambda\underline{\underline{C}}^{-1}\,\underline{1} \tag{8.10}$$

To solve for λ, multiply this equation by $\underline{1}^{\intercal}$ from the left to have

$$\underline{1}^{\intercal}\underline{\alpha} + \lambda\underline{1}^{\intercal}\underline{\underline{C}}^{-1}\,\underline{1} = 0,$$

or

$$\underline{1}^{\intercal}\underline{\underline{C}}^{-1}\,\underline{1} = -1$$

and so

$$\lambda = \frac{-1}{\underline{1}^{\intercal}\underline{\underline{C}}^{-1}\,\underline{1}} \tag{8.11}$$

Substituting this value for λ back into equation (8.10) produces show that

$$\underline{\alpha} = \frac{1}{\underline{1}^{\mathsf{T}} \underline{\underline{C}}^{-1} \underline{1}} \cdot \underline{\underline{C}}^{-1} \underline{1}. \tag{8.12}$$

Equation (8.12) defines the values for the α_i values to be used in equation (8.1). This solution requires a covariance matrix. The primary problem is generating the covariance matrix. Future work involves generating a method to estimate the covariance matrix and the implementation. Due to the relationship between correlation and covariance, we hope that an "independent-ness" metric for each variable (query hit) could be used to construct an estimate of the covariance matrix.

If we assume all the variables, $X_1, X_2, \ldots, X_n$ are mutually independent, then:

$$\underline{\underline{C}} = \begin{bmatrix} \sigma_1^2 & & & \\ & \sigma_2^2 & & \\ & & \ddots & \\ & & & \sigma_n^2 \end{bmatrix},$$

$$\underline{\underline{C}}^{-1} = \begin{bmatrix} 1/\sigma_1^2 & & & \\ & 1/\sigma_2^2 & & \\ & & \ddots & \\ & & & 1/\sigma_n^2 \end{bmatrix},$$

$$\underline{\underline{C}}^{-1} \underline{1} = \begin{bmatrix} 1/\sigma_1^2 \\ 1/\sigma_2^2 \\ \vdots \\ 1/\sigma_n^2 \end{bmatrix},$$

and

$$\underline{1}^{\intercal}\underline{\underline{C}}^{-1}\,\underline{1} = \sum_{i=1}^{n} \frac{1}{\sigma_i^2},$$

so

$$\underline{\alpha} = \frac{1}{\sum_{i=1}^{n} \frac{1}{\sigma_i^2}} \cdot \begin{bmatrix} 1/\sigma_1^2 \\ 1/\sigma_2^2 \\ \vdots \\ 1/\sigma_n^2 \end{bmatrix}. \tag{8.13}$$

This is the same solution that was presented in equation (8.2).

8.1.3 Machine Learning

Future versions of the NNPM and ATRAP could use example Situational Templates, datasets, and experts' evaluations to optimize the many adjustable parameters. Due to the mostly linear nature of the weights and thresholds, a combination of least squares regression (8.14) and back-propagation (discussed extensively in [10]) should be able to help optimize the weights and thresholds for Doctrinal Templates.

Functions of a linear nature exhibit the following form:

$$f(x_i, \underline{\alpha}) = \sum_{j=1}^{m} \alpha_j \phi_j(x_i)$$

where the α_i coefficients are the weights of the linear components and ϕ_i is a function of x_i. Let for all i,

$$X_{ij} = \frac{\partial f(x_i, \underline{\alpha})}{\partial \alpha_j} = \phi_j(x_i),$$

Then the values for α_i, which minimize the sum of the squared errors, are obtained from relation:

$$\underline{\hat{\alpha}} = (X^\intercal X)^{-1} X^\intercal \underline{y}, \tag{8.14}$$

where $\underline{y}$ is a vector of the desired outcomes or training data.

Cobb-Douglas Production weights could also be optimized via regression mentioned earlier (equations (5.7) and (5.8). More advanced methods, such as genetic algorithms or simulated annealing, may be necessary for selecting the optimal functions (e.g., fall-off functions, sigmoid functions, etc.).

8.1.4 Generalized Templates

The Template structure could be revised to provide many benefits to the NNPM. There is a proposed "Generalized Template" which simplifies the Template schema and makes Templates more Simulink-esque. This work could vastly simplify the database for Templates. It can simplify the scoring code, provide forward compatibility, provide the users with more power and flexibility at greater ease of use, allow for more intelligently designed DECOAs, provide a framework for the Analysis of Competing Hypotheses, and speed up the Template design process. However, there are several considerations to account for before revising the Template structure.

A few features we were hoping for the Templates to include are: event-driven Templates, a more recursive structure, a simplified structure, variables, constraints, and operations.

We believe there is a need for an Event-Driven Template type. Basically Event-Driven Templates would have input-events which are monitored (e.g., something matching any query shows up or expires) and output-events. An output-event might send the user a warning or modify the database. An Event-Driven Template would not

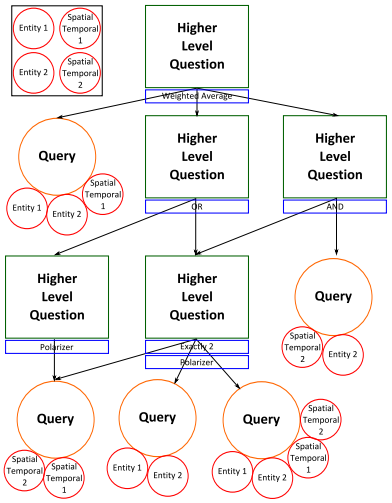


FIGURE 8.1. Generalized Template

wait for the analyst to run it, but an input event could trigger it. Once the Template is triggered, the Template can re-evaluate its score and fire an output-event if certain conditions hold. For example, upon data arriving that is associated with one or more of the queries, the system could update the branches of the Template that depended on that source, which in turn would update the Templates score. In turn, an updated Template score could trigger other Event-Driven Templates to re-evaluate their scores as well. Event-Driven Templates would have thresholds for automatically notifying the user if a score rises above or below a threshold. Additionally, there could be a "live" threat-level heat-map that is updated every time an Event-Driven Template fires.

In the proposed Generalized Template structure, a Template would be interchangeable with a higher level question (HLQ). This effectively merges an Indicator and a Template into a single structure, the HLQ. Basically an HLQ is simply a node from which further questions can be asked. An HLQ could potentially be collapsed (hiding all children) and saved as a Template to be drag-and-dropped in to another Template. The drag-and-drop could perform a copy or a reference, where the reference would

stay up to date if the Template changed, the copy would not. Visually, a referenced Template could be inserted into the HLQ block referencing it and expanded to view all queries. Additionally, this structure could allow for the easy creation of derivative enemy courses of action (DECOAs) by swapping around HLQs, constraints, variables, and operators.

Furthermore, the Template structure can be further simplified by merging the SIRs with the SORs into a general purpose query. The new query node would specify both the spatial-temporal constraints and the query (entity type, entity constraints, keywords, and relationships). While this does not follow the SIR and SOR structures directly, there is an easy mapping from the ATRAP 2.0 Template structure to this new proposed Generalized Template structure. The SIR will become an HLQ, with its spatial-temporal query converted into a variable that is applied to all its children. In addition to this merging, the query nodes are capable of taking variables.

Variables are a very important addition. They can do more than save time by allowing multiple queries to reference the same entity, location, keywords, relationships, or any other part of the query. They can be used to automate the process known as the Analysis of Competing Hypotheses. The variables can be allowed to rotate through a list of alternatives. For example, nine Templates representing three organizations operating at potentially three locations could be captured in a single Generalized Template with two variables, each rotating through three values. Furthermore, these variables allow for potentially highly effective DECOAs by rotating in untested values for these variables.

The new constraint nodes would operate on variables to produce complex, powerful operations. The most notable use is to be able to select all organizations, locations, etc. that fit certain requirements. The constraints can be either applied to a single variable or a group of variables. For example, if there are hundreds of terrorist groups,

then it would be easier to specify a single entity with the constraint that the entity is a terrorist group and in a specific region. A constraint could be applied to location variables to enforce that the location of query hits must be within a certain kilometer radius of each other.

The last major change is the way how the nodes are connected. Connections are made through operator blocks in a Simulink-esque fashion. Whereas the previous connection type was always a weighted average with optional thresholding, in the proposed Generalized Template model any number of operations can be chosen. A problem in ATRAP 2.0 is that the weighted average with thresholding was tied to the AND and OR operations, defining how thresholding operates. The new model offers various different operators: AND, OR, Weighted Average with thresholding, Exactly N, and Polarization (to name a few). Not only does this decouple the AND and OR from the Weighted Average, but it allows for more operations.

The operator blocks could further be used to allow the user to define other parts of the scoring algorithm (see Chapter 5). Currently a user selectable sigmoid function is used for combining multiple hits to a single query. However, various operator blocks could be used instead. It would allow the user to specify which sigmoid function to use or the option to use the statistical method that was discussed earlier. If all queries and HLQs have associated operator blocks, then this vastly simplifies the scoring algorithm. Inside the code, all nodes would have a score operator. HLQs would simply pass down the 'score' command to the operator node directly below it. Operators would perform work on their input after passing down the 'score' command to more operators, HLQs, or queries. The query would be able to generate several score vectors for each prediction match, which then combine them into a single score for the query via the use of the operator directly below it. This allows the builder of the Generalized Template to select how infobits are combined and whether to use

disambiguation, sigmoid thresholding, etc.

The combination of these two merges (Templates with Indicators and SIRs with SORs) and the use of operator nodes allow for vastly simplified code and database structure. First there would be fewer classes, which would benefit both the code and the database structure. This also models a much more design which makes use of inheritance far easier. The scoring of Templates would no longer make hard-coded scoring procedure calls. Rather these calls will be tied to the operator nodes, decoupling the Template scoring from the operations used.

8.1.5 Behavioral Filtering

An on-going research aspect is enhancing the predictive capabilities (Template evaluation) by integrating the behavioral filtering engine (see Chapter 7. The development of rules and selection of attributes for behavioral filtering is currently underway. This research is trying to determine which behavioral attributes to track as well as building rules around these attributes to maximize the accuracy of the NNPM per the time, cost, and difficulty of obtaining the behavioral attributes.

The behavioral filtering engine could benefit from an additional input: user-defined keywords. There are already a few built-in keywords such as "show" and "finished." A technical user could introduce new keywords along with their interpretation. Formally, a user-defined keyword is represented by the 5-tuple $\langle P, V, O, T, \phi \rangle$, where P is an English-Like pattern to match, V is the set of the variables in P, O is a boolean describing whether its an "output" or an"input" keyword, T the set of relations between variables and their types, and ϕ is the code translation of P. For example, to copy a variable, the user might define $P =$ "X1 copy X2," $V = \{X1, X2\}$, $O = true$, $T = \{X1 \equiv X2\}$ $\phi = \{X1.value = X2.value;\ X1.certainty = X2.certainty;\}$.

If the behavioral filter were to become slow due to a large number of rules, it could

potentially be sped up by implementing the Rete algorithm discussed in [8]. The Rete algorithm allows for the quick matching of many patterns by taking advantage of two facts. First, when a rule fires, it usually only affects a few other rules. Second, multiple rules can share patterns, so the pattern only needs to be evaluated once.

8.1.6 Stochastic Linear Motion Model

Another direction of research is that of stochastic linear motion data-fusion. There are two applications for motion data-fusion. First, when a single entity is moving, it will provide an estimate to where the entity might be after a given amount of time. Secondly, when a single entity may appear in the database multiple times if it is recorded by sensors multiple times, the system could determine if it appears plausible that the multiple sensor readings are all a single target (or at least reduce the number of suspected entities). The use of random variables in motion estimation equations could be used to provide a probability map of where an entity might be or to determine if sensor data suggests linear motion. Additionally, the random variables help account for measurement errors.

Stochastic linear motion can be described as:

$$\overrightarrow{X} - \overrightarrow{X_0} = \Delta T \cdot \overrightarrow{V}, \tag{8.15}$$

or alternatively,

$$\overrightarrow{X} = \overrightarrow{X_0} + \Delta T \cdot \overrightarrow{V},$$

where $\overrightarrow{X_0}$ is a random variable describing the initial position, $\overrightarrow{X}$ is a random variable describing a second position, ΔT is a random variable describing a change in time, and

$\overrightarrow{V}$ is a random variable describing velocity. Equation (8.15) is used to generate two distributions, $\overrightarrow{Z_1}$ from the differences in position and $\overrightarrow{Z_2}$ from the time and velocity.

Lets examine the first distribution:

$$\overrightarrow{Z_1} = \overrightarrow{X} - \overrightarrow{X_0},$$

which would have a probability distribution of

$$f(z_x, z_y, w_x, w_y) = \left.\frac{f_{XX_0}(x_x, x_y, x_{0x}, x_{0y})}{\left|\frac{\partial(z_x, z_y, w_x, w_y)}{\partial(x_x, x_y, x_{0x}, x_{0y})}\right|}\right|_{\substack{x_x = z_x + w_x \\ x_{0x} = w_x \\ x_y = z_y + w_y \\ x_{0y} = w_y}}.$$

These random variables are two-dimensional, but for now we shall use just one dimension (since x and y dimensions are assumed independent):

$$f(z, w) = \left.\frac{f_{XX_0}(x, x_0)}{\left|\frac{\partial(z,w)}{\partial(x,x_0)}\right|}\right|_{\substack{x = z + w \\ x_0 = w}}$$

$$f(z) = \int_{-\infty}^{\infty} \frac{f_{XX_0}(z + w, w)}{\left|\frac{\partial(z,w)}{\partial(x,x_0)}\right|}\,\mathrm{d}w$$

Letting $\overrightarrow{X}$ and $\overrightarrow{X_0}$ be independent yields:

$$f(z) = \int_{-\infty}^{\infty} \frac{f_X(z + w) f_{X_0}(w)}{\begin{vmatrix} 1 & -1 \\ 0 & 1 \end{vmatrix}}\,\mathrm{d}w$$

$$f(z) = \int_{-\infty}^{\infty} f_X(z + w) f_{X_0}(w)\,\mathrm{d}w. \tag{8.16}$$

If the second dimension is taken into account, the above equation is nearly identical.

The primary difference is the integral becomes a double integral to integrate out both w_x and w_y.

Now lets examine the second distribution:

$$\overrightarrow{Z_2} = \Delta T \cdot \overrightarrow{V}.$$

If the heading and speed of $\overrightarrow{V}$ are independent, we can operate on the magnitude of the $\overrightarrow{V}$ and the magnitude of $\overrightarrow{Z_2}$:

$$\|Z_2\| = \Delta T \cdot \|V\|.$$

The probability distribution of $\|Z_2\|$ is

$$f_{\|Z_2\|} = \int_{-\infty}^{\infty} \frac{f_{\Delta T}(t) f_{\|V\|}(v)}{\begin{vmatrix} t & v \\ 0 & 1 \end{vmatrix}} \,\mathrm{d}w \Bigg|_{\substack{v=z/w \\ t=w}}$$

$$f_{\|Z_2\|} = \int_{-\infty}^{\infty} \frac{f_{\Delta T}(w) f_{\|V\|}(z/w)}{|w|} \,\mathrm{d}w. \tag{8.17}$$

Since the heading and the speed are independent,

$$f_{Z_2}(\|z_2\|, \angle z_2) = f_{\|Z_2\|}(\|z_2\|) f_{\angle Z_2}(\angle z_2).$$

When converted to Cartesian coordinates, where $x = \|z_2\| \cos(\angle z_2), y = \|z_2\| \sin(\angle z_2)$ is

$$f_{Z_2}(x,y) = \frac{f_{\|Z_2\|}(\sqrt{x^2+y^2})f_{\angle Z_2}(\text{atan2}(y,x))}{\begin{vmatrix} \cos(\angle z_2) & -\|z_2\| \sin(\angle z_2) \\ \sin(\angle z_2) & \|z_2\| \cos(\angle z_2) \end{vmatrix}}$$

$$f_{Z_2}(x,y) = \frac{f_{\|Z_2\|}(\sqrt{x^2+y^2})f_{\angle Z_2}(\text{atan2}(y,x))}{\sqrt{x^2+y^2}} \tag{8.18}$$

Now the two distributions must be compared to determine how well they fit each other. At first, a two-dimensional Kolmogorov-Smirnov test was considered for the comparison of the two distributions. Upon further analysis of the problem, it was discovered that the distributions need not match. For example, suppose a velocity estimate is potentially flawed, then the random variable representing velocity would contain a large variance for speed and direction. If the locations are known with great accuracy, their variances would be much smaller. Even if the expected values were the same, the two-dimensional Kolmogorov-Smirnov test would show the distributions are dissimilar due to the differences in their variance (differences in their errors). Clearly we do not want the Kolmogorov-Smirnov test since it is likely to return a false negative if the sensors do not share the same degree of accuracy.

There are several methods which could be used to determine how well Z_1 and Z_2 fit each other. The Student's t-test would work except that the test is based on a number of samples. There are other statistical tests as well, but most assume sample sizes. Hence, this is still an open question.

Appendix A

```
function [ outs ] = Threshold_system( dataV, W, threshold )
%Threshold_system( dataV, W, threshold )
%   For investigating how this system behaves as data vector,
%   weight vector, and threshold changes

%transition speed: alpha = 1.0 ->
%   sudden jump to saturation (assessment->pi/2, magnitude=y-component)
%transition speed: alpha = 0.5 ->
%   average transition region(assessment & magnitude both increase)
alpha = 0.5;

V = dataV' * W';
x = V(1);
y = V(2);

%raw distance
dist = sqrt(x^2+y^2) / sum(abs(W));
distold = dist;
if(dist > 1)
    fprintf(1, 'WARNING VECTORS OF MAGNITUDE > 1\n');
end

%Test to see if threshold condition is met
if ( (y / threshold) > dist )
   y2 = y / sum(abs(W));     %y-average
   y = y/threshold;          %y-threshold

   %%soft-cap (if 0<alpha<1) the distance & y
   %weighted average(dist,dist+improvement)
   dist = dist + alpha * (y - dist);
   %weighted average(y, y+improvement)
   y = y2 + alpha * (dist - distold);

   %hard-cap magnitude to 1
   dist = min(1, dist);
   y = min(dist, y);

   %adjust related assessment-neutral information
   x = sqrt( dist^2 - y^2);
else %There isn't enough supporting information ==> default case
    y = y / sum(abs(W));
    x = x / sum(abs(W));
```

```
    if(dist > 1)
        theta = atan2(y,x);
        y = sin(theta);
        x = abs(cos(theta));
    end
end

outs = [x, y];
end



function [ outs ] = Experimental_threshold_system2( dataV, W, threshold )
%Experimental_threshold_system2 ( dataV, W, threshold )
%   For investigating how this system behaves as dataV changes

%Example new threshold system2 - for every 2 points gained in y past
%threshold -> 1 point lost from x
alpha = 0.5;

absW = sum(abs(W));
x = sum ( abs(W) .* dataV(:,1)' ) / absW;
y = sum ( W .* dataV(:,2)' );
y1 = y / absW;
if(threshold < absW)
    y2 = y / threshold;

    %raw distance
    D1 = sqrt(x^2+y1^2);          %Calculate dist without threshold
    D2 = sqrt(x^2+y2^2);          %Calculate dist with threshold

    D1P = D1 + alpha * (D2 - D1); %Allow D to grow some, shrink x some
    D1P = max( D1P, abs(y2) );    %Don't sacrifice y (primary signal)
    D1P = min( D1P, 1);           %Cap distance
    y2  = min( y2, +1);           %Cap y (from upper)
    y2  = max( y2, -1);           %Cap y (from lower)
    x   = sqrt(D1P^2 - y2^2);     %Recalculate x
else
    y2 = y1;                      %Pass through
end

outs = [x, y2];
end
```

Appendix B

ATRAP - Asymmetric Threat Response and Analysis Program

C2 - Command and Control

COA - Course of Action

DECOA - Derivative Enemy Course of Action

ECOA - Enemy Course of Action

HLQ - Higher Level Question

GUI - Graphical User Interface

JDL - Joint Directors of Laboratories

NNPM - Non-Numerical Predictive Model (for Asymmetric Analysis)

QNN - Quasi-Neural Network

SASO - Stability And Support Operations

SOR - Specific Orders and Request

SIR - Specific Information Request

SME - Subject Matter Expert

TTP - Tactics, Techniques, and Procedures

References

[1] U.S. Census: Population Estimates. website: http://www.census.gov/popest/counties/tables/CO-EST2008-01-04.csv, February 2010. accessed: February-25-2010.

[2] R. M. Akita. User based data fusion approaches. In *Information Fusion, 2002. Proceedings of the Fifth International Conference on*, volume 2, pages 1457–1462, 2002.

[3] D. Brown, J. Dalton, and H. Hoyle. *Intelligence and Security Informatics*, volume 3073 of *Lecture notes in computer science*, chapter Spatial Forecast Methods for Terrorist Events in Urban Environments, pages 426–435. Springer Berlin, 2004.

[4] G. Chen, D. Shen, C. Kwan, J. B. Cruz, and M. Kruger. Game theoretic approach to threat prediction and situation awareness. In *Information Fusion, 2006 9th International Conference on*, pages 1–8, July 2006. ID: 1.

[5] A. H. Cordesman. *The lessons of Afghanistan : war fighting, intelligence, and force transformation*. CSIS Press, Washington, D.C., 2002. ID: 50417230.

[6] P. H. Douglas. The cobb-douglas production function once again: Its history, its testing, and some new empirical values. *The Journal of Political Economy*, 84(5):903–915, 1976.

[7] J. A. Fisher and A. H. McMakin. Pnnl wins four technology transfer awards. *Breakthroughs. Science, Technology, Innovation*, 5, 2006.

[8] C. L. Forgy. *Rete: a fast algorithm for the many pattern/many object pattern match problem*, pages 324–341. IEEE Computer Society Press, Los Alamitos, CA, USA, 1990.

[9] N. Garra. Info about 'star-light' program. email, Nov 2009.

[10] S. Haykin. *Neural Networks and Learning Machines*. Pearson Education, New Jersey, Upper Saddle River, third edition, 2009.

[11] A. Hossain, N. Walmsley, and P. Pearce. A minimum spanning tree approach to identifying collective behaviour and inferring intent for combat models. In *International C2*, volume 2(2). DoD, September 26 2008.

[12] S. Mahoney, J. Pfautz, T. Fichtl, S. Guarino, E. Carlson, and M. Farry. Enabling robust c2 systems through evolvable human-in-the-loop data fusion. In *International Command and Control Research and Technology Symposium*, volume 14, page 147, Washington, DC, 2009. DoD.

[13] D. McDaniel. An information fusion framework for data integration. In *13th Annual Software Technology Conference*, Salt Lake City, UT, May 3, 2001 2001. Silver Bullet Solutions, Inc.

[14] W. Meeusen and J. V. D. Broeck. Efficiency estimation from cobb-douglas production functions with composed error. *International Economic Review*, 18(2):435–444, 1977.

[15] M. B. Mitchell, D. E. Brown, and J. H. Conklin. A crime forecasting tool for the web-based crime analysis toolkit. In *Systems and Information Engineering Design Symposium, 2007. SIEDS 2007. IEEE*, pages 1–5, April 2007. ID: 1.

[16] Dr. Henry Niman. Flu tracker. website: http://flutracker.rhizalabs.com/, February 2010], note=accessed: February-25-2010.

[17] P. V. Pearce, A. Robinson, and S. C. Wright. The wargame infrastructure and simulation environment (wise). In V. Palade, R. J. Howlett, and L. C. Jain, editors, *7th International Conference on Knowledge-Based Intelligent Information and Engineering Systems*, volume 2774, page 714722, Oxford, September 3-5 2003. Springer-Verlag, Berlin.

[18] E. Roth, R. Scott, S. Deutsch, S. Kuper, V. Schmidt, M. Stilson, and J. Wampler. Evolvable work-centred support systems for command and control: creating systems users can adapt to meet changing demands. *Ergonomics*, 49(7):688, 2006.

[19] D. Shen, G. Chen, J. B. Cruz, L. S. Haynes, M. Kruger, and E. Blasch. Game-theoretic modeling and control of military air operations with retaliatory civilians. In *Aerospace Conference, 2007 IEEE*, pages 1–10, March 2007. ID: 1.

[20] A. N. Steinberg, C. L. Bowman, and F. E. White. Revisions to the jdl data fusion model. In Belur V. Dasarathy, editor, *Sensor Fusion: Architectures, Algorithms, and Applications III*, volume 3719, pages 430–441. SPIE, 1999.

[21] L. Suantak, D. Hillis, J. Schlabach, J. W. Rozenblit, and M. Barnes. A coevolutionary approach to course of action generation and visualization in multi-sided conflicts. In *Systems, Man and Cybernetics, IEEE International Conference on*, volume 2, pages 1973–1978 vol.2, October 2003.

[22] L. Suantak, F. Momen, J. W. Rozenblit, D. Hillis, M. Barnes, and J. Schlabach. Modeling and simulation of stability and support operations (saso). In *Engineering of Computer-Based Systems, 2004. Proceedings. 11th IEEE International Conference and Workshop on the*, pages 21–28, Los Alamitos, CA, USA, 2004. IEEE Computer Society.

[23] M. L. Valenzuela, F. Szidarovszky, C. Feng, P. Reddy, F. Momen, J. W. Rozenblit, and B. Ten Eyck. A non-numerical predictive model for asymmetric analysis. Accepted to be presented in the *Engineering of Computer-Base Systems (ECBS'10), 17th IEEE International Conference on*, St. Anne's College, Oxford, UK, March 22-26 2010. IEEE Computer Society.

[24] M. Wei, G. Chen, J. B. Cruz, B. Jose, L. S. Haynes, and M. Kruger. Applying spatial-temporal model and game theory to asymmetric threat prediction. In *12th International*

Command and Control Research and Technology Symposium (ICCRTS), volume 12(3), page 63, Newport, RI, 2007.

[25] D. Woods and S. Dekker. Anticipating the effects of technological change: a new era of dynamics for human factors. *Theoretical Issues in Ergonomics Science*, 1(11):272–282, 1 July 2000.

[26] A. Zellner, J. Kmenta, and J. Dreze. Specification and estimation of cobb-douglas production function models. *Econometrica*, 34(4):784–795, 1966.