Document Title:	A Software System for Information Extraction in Criminal Justice Information Systems
Author(s):	Tianhao Wu ; Stephen V. Zanias ; William M. Pottenger
Document No.:	217681
Date Received:	March 2007
Award Number:	2003-IJ-CX-K003

A Software System for Information Extraction in Criminal Justice Information Systems

Law enforcement agencies across the country have enormous quantities of data that are simply not being well utilized. Police reports, affidavits, and various other forms of textual information remain an untapped resource because of technological and other barriers. In addition, officers, detectives, and investigators do not have time to analyze (or the ability to recall) this wealth of textual information. Furthermore, recognizing links between items of data presents an even greater problem. Fortunately, new technology is reaching the marketplace that enables multitudes of textual documents to be scanned and key items of information automatically extracted for use in solving crimes.

Using one of these technologies known as "information extraction," key items of data found within narrative textual documents can be processed and converted into useful, searchable information. Technically speaking, information extraction is concerned with the automatic discovery of patterns and relationships in textual data. This type of cutting-edge technology is exactly what the law enforcement community needs to process the wealth of law enforcement information and enhance investigative efforts.

Based on information extraction technology, Lehigh has developed a software system named the BPD_IE System (Bethlehem Police Department Information Extraction System) that automatically extracts key items of information from narrative textual data and links unsolved criminal cases to solved cases, providing investigators with valuable leads. The technology automatically obtains modus operandi and physical descriptions from these textual documents. This data is then stored into fielded, relational databases which can be easily searched.

While the effort covered in this summary, details the BPD_IE System and the work with the Bethlehem Police Department (BPD), the application of this system and its technology can be used in other departments to enhance efforts in law enforcement.

The BPD_FE sub-system, as mentioned earlier, is a conversion tool that extracts textual features from our data sources. These features are then saved in tables in the BPD_IE Database. BPD_IE System users can search the extracted features using the user interface. While the BPD_FE sub-system provides several capabilities, perhaps the most notable is batch loading and conversion. Investigators often keep their textual documents in a few (if not one) designated folder on their computer. The batch upload function allows the user to simply indicate a specific folder. The system automatically performs feature extraction and database operations on all the documents located within the folder.

A Software System for Information Extraction in Criminal Justice Information Systems

Tianhao Wu, Stephen V. Zanias, and William M. Pottenger

1. Introduction and Motivation

Law enforcement agencies across the country have enormous quantities of data that are simply not being well utilized. Police reports, affidavits, and various other forms of textual information remain an untapped resource because of technological and other barriers. In addition, officers, detectives, and investigators do not have time to analyze (or the ability to recall) this wealth of textual information. Furthermore, recognizing links between items of data presents an even greater problem. Fortunately, new technology is reaching the marketplace that enables multitudes of textual documents to be scanned and key items of information automatically extracted for use in solving crimes.

Using one of these technologies known as "information extraction," key items of data found within narrative textual documents can be processed and converted into useful, searchable information. Technically speaking, information extraction is concerned with the automatic discovery of patterns and relationships in textual data. This type of cutting-edge technology is exactly what the law enforcement community needs to process the wealth of law enforcement information and enhance investigative efforts.

Based on information extraction technology, we have developed a software system named the BPD_IE System (Bethlehem Police Department Information Extraction System) that automatically extracts key items of information from narrative textual data and links unsolved criminal cases to solved cases, providing investigators with valuable leads. Our technology automatically obtains modus operandi and physical descriptions from these textual documents. This data is then stored into fielded, relational databases which can be easily searched.

While the effort covered in this summary details the BPD_IE System and our work with the Bethlehem Police Department (BPD), the application of this system and its technology can be used in other departments to enhance efforts in law enforcement. It is our hope that this brief synopsis of our work will interest you in enhancing your efforts with information extraction technology.

2. Data Sources

Our current system combines and searches data from four different sources currently being used by the BPD. Detective reports created by the BPD and stored in Microsoft Word format (nearly 8,500 reports) provide the first source of data. The second source of information consists of crime report information that has been manually entered into the BPD Database. To date, we have entered nearly 250 crime reports (including over 300 associated narrative text supplements) into the system. The third component consists of narrative fields within the BPD Affidavit database that contain affidavits of probable cause; over 2,800 affidavit records have been converted to our BPD_IE database schema. Another database containing arrest records provides the fourth source of data and will soon be connected to our system.

In order to utilize these information sources, our first step was to determine which features were useful for matching cases' modus operandi. From our interaction with the BPD, we have identified several feature types necessary to recognize criminal modus operandi and physical description data, as shown in alphabetical order in Table 1.

Table 1: Modus Operandi and Physical Description Feature Types

• Age	• Hair Color	• Race
• Arrest Number	• Height	• Relationship
• Classification	• Item Stolen	• Report Date
• Clothing	• Location	• Report Officer
• Control Number	• Offense Date	• Residence
• Day of Week	• Offense Time	• SSN
• Driver License Number	• Offense Type	• Vehicle
• Drugs	• Person Name	• Weapon
• Eye Color	• Phone Number	• Weight

3. BPI_IE System Description

A high-level diagram of how our system works is depicted in Figure 1. The narrative text data (from detective reports, crime reports supplements, and/or narrative database field(s)) is entered using the BPD_FE, the Feature Extraction sub-system. This sub-system extracts items from the input data and loads them into the BPD_IE Database so that the information can more quickly and easily be searched and compared. A second subsystem, the BPD_MO (Modus Operandi) is then used to search the BPD_IE and arrest record databases to link unsolved and solved cases based on comparing information found within the cases. Using these tools helps detectives more easily identify suspects and more rapidly solve crimes.

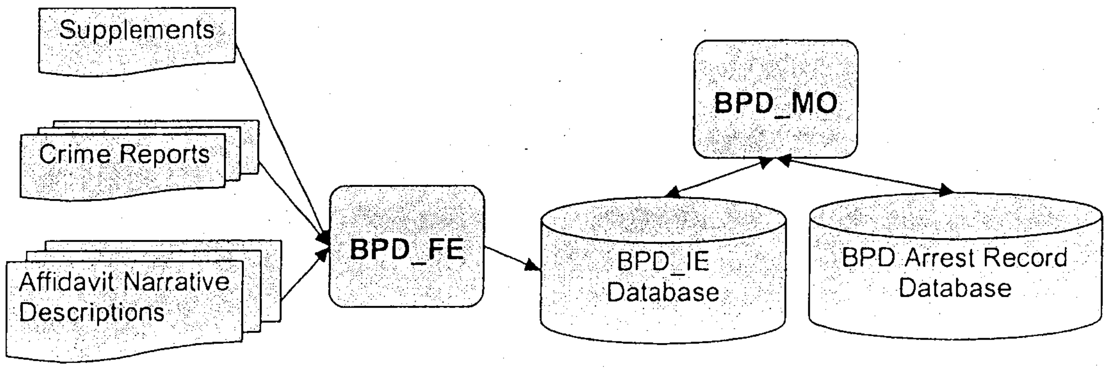


Figure 1: BPD_IE System Overview: Two sub-systems, BPD_FE and BPD_MO, work together

3.1. Purpose & Basic Functions of the BPD_FE Sub-system

The BPD_FE sub-system, as mentioned earlier, is a conversion tool that extracts textual features from our data sources. These features are then saved in tables in the BPD_IE Database. BPD_IE System users can search the extracted features using the user interface. While the BPD_FE sub-system provides several capabilities, perhaps the most notable is batch loading and conversion. Investigators often keep their textual documents in a few (if not one) designated folder on their computer. The batch upload function allows the user to simply indicate a specific folder. The system automatically performs feature extraction and database operations on all the documents located within the folder.

To provide for system growth, we designed the system to use XML files and XSD definitions. This will allow reports of different styles or formats to be recognized and their data to be more easily added to the system.

The BPD_FE sub-system also supports feature extraction rule file updating. By clicking the "Update" button on the user interface, the software automatically downloads the latest extraction rule file from a server located in the HDDI lab of the Computer Science and Engineering Department at Lehigh University. These updates allow the most up-to-date technology to be added into the system, leading to even more efficient and effective searches as our research efforts continue.

3.2. Purpose & Basic Functions of the BPD_MO Sub-system

The BPD_MO sub-system implements a search tool that helps users to search for relevant documents using features stored in the BPD_IE Database. Modus operandi and physical description searches can be carried out in one of three ways: search form, search-engine, and search by report. Our search form is similar to other forms currently in use in many law enforcement applications. The form has fields in which to enter the search criteria.

In our search-engine interface, the investigator inputs keywords in a search box in any order. Additionally, keywords may be preceded by a type designator that narrows the scope of the search. For example, "Age:15" would narrow the scope of search for the number 15 to the Age field. Multiple featureType:keyword pairs are permitted within the same query.

The third method is to drag and drop a narrative text investigative report onto the desktop BPD_MO icon. The system automatically fills out the search form described in the first method using the information contained in the report. The detective then selects the features of interest, and related cases are returned. The detective can also perform the same kind of searches employed in the two approaches described above.

Our system also provides "fuzzy search" capability. Due to variances in text from mistyping, misspelling, or other data entry errors, fuzzy searches allow inexact matches of values. For instance, Osama Bin Ladin's name has various different spellings, such as "Usama," "Osama," "Usamah", etc. Fuzzy searches allow for all of these names to be returned in a single search for "Osama." Such capabilities greatly enhance our system's usefulness.

3.3. Practical examples

Already, our system has provided benefits to the Bethlehem Police Department. Using our BPD_IE System, officers have been able to link cases and identify modus operandi with much success, as demonstrated in the following two examples.

Imposter Burglaries

One of the investigators at the BPD provided a summary of imposter burglaries that had occurred recently within the City of Bethlehem. The modus operandi for these burglaries involved diverting the owner's attention prior to breaking in and burglarizing the home. Using the Modus Operandi (MO) search tool, we identified three additional cases that fit the MO. By alerting investigative detectives to these cases, they are able to more quickly identify repeat offenders and possible suspects.

Auto Thefts

Another investigator at the BPD provided us with a set of solved crimes connected to a single serial auto thief. The thief's MO involved breaking in through a rear side or wing window of the vehicle. After the detective had spent a significant amount of time investigating the case, he discovered six previous cases for which the thief was responsible. Seeing this as an opportunity to test our system's effectiveness, we employed the third search method of the BPD_MO and dropped the most recent case onto the BPD_MO desktop icon. Our system recovered four of the six unsolved related cases among our top five search results. Capabilities such as this can greatly enhance an investigation's speed and effectiveness.

3.4. Technical Summary

Our system was developed using industry standard components and software. The system was designed with UML (IBM Rational Rose) and developed with Microsoft VC++ .NET and MySQL. Information extraction rules were learned using a Perl program, and the fuzzy search capabilities are MySQL functions. XML format files are used to contain extraction rules and manually entered crime reports. Some third party components, such as the Text Mining Infrastructure and a regular expression library for VC++ .NET, were also employed. Altogether, almost 350,600 lines of code have been written for this system.

4. Conclusion

We have described a software system that is able to not only provide a more efficient way of organizing textual law enforcement data, but also automatically discovers and extracts important features by which unsolved cases can be linked with solved cases. To learn more about using the BPD_IE System, a full version and documentation is available at http://hddi.cse.lehigh.edu.

Final Report on "A Software System for Information Extraction in Criminal Justice Information Systems"

William M. Pottenger, Tianhao Wu, and Stephen V. Zanias
Lehigh University Computer Science and Engineering Department
{billp, tiw2, svz2}@lehigh.edu

Executive Summary

Introduction and Motivation

Law enforcement agencies across the country have enormous quantities of data that are simply not being well utilized. Police reports, affidavits, and various other forms of textual information remain an untapped resource because of technological and other barriers. In addition, officers, detectives, and investigators do not have time to analyze (or the ability to recall) this wealth of textual information. Furthermore, recognizing links between items of data presents an even greater problem. Fortunately, new technology is reaching the marketplace that enables multitudes of textual documents to be scanned and key items of information automatically extracted for use in solving crimes.

Using one of these technologies known as "information extraction," key items of data found within narrative textual documents can be processed and converted into useful, searchable information. Technically speaking, information extraction is concerned with the automatic discovery of patterns and relationships in textual data. This type of cutting-edge technology is exactly what the law enforcement community needs to process the wealth of law enforcement information and enhance investigative efforts.

Based on information extraction technology, we have developed a software system named the BPD_IE System (Bethlehem Police Department Information Extraction System) that automatically extracts key items of information from narrative textual data and links unsolved criminal cases to solved cases, providing investigators with valuable leads. Our technology automatically obtains modus operandi and physical descriptions from these textual documents. This data is then stored into fielded, relational databases which can be easily searched.

While the effort covered in this summary details the BPD_IE System and our work with the Bethlehem Police Department (BPD), the application of this system and its technology can be used in other departments to enhance efforts in law enforcement. It is our hope that this brief synopsis of our work will interest you in enhancing your efforts with information extraction technology.

Data Sources

Our current system combines and searches data from four different sources currently being used by the BPD. Detective reports created by the BPD and stored in Microsoft Word format (nearly 8,500 reports) provide the first source of data. The second source of information consists of crime report information that has been manually entered into the BPD Database. To date, we have entered nearly 250 crime reports (including over 300 associated narrative text supplements) into the system. The third component consists of narrative fields within the BPD Affidavit database that contain affidavits of probable cause; over 2,800 affidavit records have been converted to our BPD_IE database schema. Another database containing arrest records provides the fourth source of data and will soon be connected to our system.

In order to utilize these information sources, our first step was to determine which features were useful for matching cases' modus operandi. From our interaction with the BPD, we have identified several feature types necessary to recognize criminal modus operandi and physical description data, as shown in alphabetical order in Table 1.

Table 1: Modus Operandi and Physical Description Feature Types

• Age	• Hair Color	• Race
• Arrest Number	• Height	• Relationship
• Classification	• Item Stolen	• Report Date
• Clothing	• Location	• Report Officer
• Control Number	• Offense Date	• Residence
• Day of Week	• Offense Time	• SSN
• Driver License Number	• Offense Type	• Vehicle
• Drugs	• Person Name	• Weapon
• Eye Color	• Phone Number	• Weight

BPI_IE System Description

A high-level diagram of how our system works is depicted in Figure 1. The narrative text data (from detective reports, crime reports supplements, and/or narrative database field(s)) is entered using the BPD_FE, the Feature Extraction sub-system. This sub-system extracts items from the input data and loads them into the BPD_IE Database so that the information can more quickly and easily be searched and compared. A second subsystem, the BPD_MO (Modus Operandi) is then used to search the BPD_IE and arrest record databases to link unsolved and solved cases based on comparing information found within the cases. Using these tools helps detectives more easily identify suspects and more rapidly solve crimes.

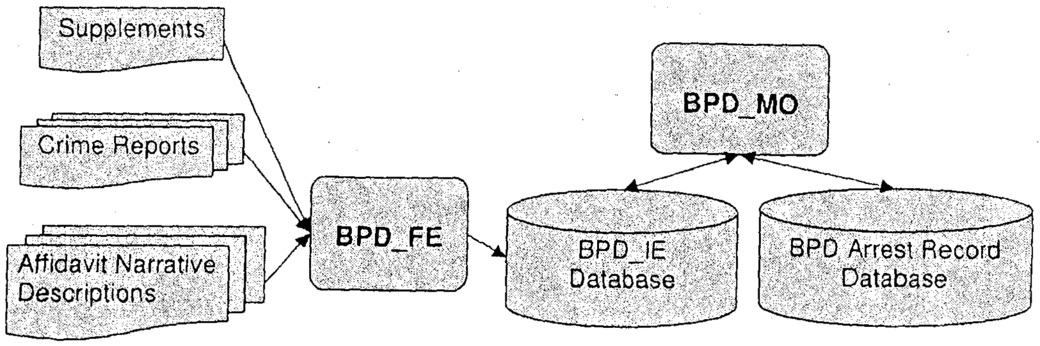


Figure 1: BPD_IE System Overview: Two sub-systems, BPD_FE and BPD_MO, work together

Purpose & Basic Functions of the BPD_FE Sub-system

The BPD_FE sub-system, as mentioned earlier, is a conversion tool that extracts textual features from our data sources. These features are then saved in tables in the BPD_IE Database. BPD_IE System users can search the extracted features using the user interface. While the BPD_FE sub-system provides several capabilities, perhaps the most notable is batch loading and conversion. Investigators often keep their textual documents in a few (if not one) designated folder on their computer. The batch upload function allows the user to simply indicate a specific folder. The system automatically performs feature extraction and database operations on all the documents located within the folder.

To provide for system growth, we designed the system to use XML files and XSD definitions. This will allow reports of different styles or formats to be recognized and their data to be more easily added to the system.

The BPD_FE sub-system also supports feature extraction rule file updating. By clicking the "Update" button on the user interface, the software automatically downloads the latest extraction rule file from a server located in the HDDI lab of the Computer Science and Engineering Department at Lehigh University. These updates allow the most up-to-date technology to be added into the system, leading to even more efficient and effective searches as our research efforts continue.

Purpose & Basic Functions of the BPD_MO Sub-system

The BPD_MO sub-system implements a search tool that helps users to search for relevant documents using features stored in the BPD_IE Database. Modus operandi and physical description searches can be carried out in one of three ways: search form, search-engine, and search by report. Our search form is similar to other forms currently in use in many law enforcement applications. The form has fields in which to enter the search criteria.

In our search-engine interface, the investigator inputs keywords in a search box in any order. Additionally, keywords may be preceded by a type designator that narrows the scope of the search. For example, "Age:15" would narrow the scope of search for the number 15 to the Age field. Multiple featureType:keyword pairs are permitted within the same query.

The third method is to drag and drop a narrative text investigative report onto the desktop BPD_MO icon. The system automatically fills out the search form described in the first method using the information contained in the report. The detective then selects the features of interest, and related cases are returned. The detective can also perform the same kind of searches employed in the two approaches described above.

Our system also provides "fuzzy search" capability. Due to variances in text from mistyping, misspelling, or other data entry errors, fuzzy searches allow inexact matches of values. For instance, Osama Bin Ladin's name has various different spellings, such as "Usama," "Osama," "Usamah", etc. Fuzzy searches allow for all of these names to be returned in a single search for "Osama." Such capabilities greatly enhance our system's usefulness.

Practical examples

Already, our system has provided benefits to the Bethlehem Police Department. Using our BPD_IE System, officers have been able to link cases and identify modus operandi with much success, as demonstrated in the following two examples.

Imposter Burglaries

One of the investigators at the BPD provided a summary of imposter burglaries that had occurred recently within the City of Bethlehem. The modus operandi for these burglaries involved diverting the owner's attention prior to breaking in and burglarizing the home. Using the Modus Operandi (MO) search tool, we identified three additional cases that fit the MO. By alerting investigative detectives to these cases, they are able to more quickly identify repeat offenders and possible suspects.

Auto Thefts

Another investigator at the BPD provided us with a set of solved crimes connected to a single serial auto thief. The thief's MO involved breaking in through a rear side or wing window of the vehicle. After the detective had spent a significant amount of time investigating the case.

he discovered six previous cases for which the thief was responsible. Seeing this as an opportunity to test our system's effectiveness, we employed the third search method of the BPD_MO and dropped the most recent case onto the BPD_MO desktop icon. Our system recovered four of the six unsolved related cases among our top five search results. Capabilities such as this can greatly enhance an investigation's speed and effectiveness.

Technical Summary

Our system was developed using industry standard components and software. The system was designed with UML (IBM Rational Rose) and developed with Microsoft VC++ .NET and MySQL. Information extraction rules were learned using a Perl program, and the fuzzy search capabilities are MySQL functions. XML format files are used to contain extraction rules and manually entered crime reports. Some third party components, such as the Text Mining Infrastructure and a regular expression library for VC++ .NET, were also employed. Altogether, almost 350,600 lines of code have been written for this system.

Conclusion

We have described a software system that is able to not only provide a more efficient way of organizing textual law enforcement data, but also automatically discovers and extracts important features by which unsolved cases can be linked with solved cases. To learn more about using the BPD_IE System, a full version and documentation is available at http://hddi.cse.lehigh.edu.

1 Introduction and Motivation

This report summarizes our research and development work on NIJ Grant Number 2003-IJ-CX-K003, "A Software System for Information Extraction in Criminal Justice Information Systems". The project involved many components including theoretical work, algorithm design, system design, system implementation, documentation, officer training, system deployment and pilot testing. Accomplishing these steps required the development of novel theory and multiple techniques that will be expounded upon in what follows.

The report has been divided into two parts. The first part (Sections 2 though 5) describes the theoretical foundation for the Information Extraction (IE) algorithms developed as part of this project. Section 2 reviews the background research conducted in the field of IE and provides an overview of the field. In Section 3, we discuss related work, highlighting popular IE systems and the learning algorithms associated with them. Given this background, we then describe (in Section 4) our own reduced regular expression (RRE) learning algorithm that has been designed for the system. Our evaluation results are presented in Section 5.

The second part of our report contains Sections 6 through 9. Section 6 details the development of the BDP_IE program, the application and deployment of our information extraction algorithm into a real world environment at the Bethlehem, Pennsylvania Police Department. In Section 7 we present our conclusions, and the future work related to this project is covered in Section 8. The ninth section contains our acknowledgements. A significant amount of additional information related to the project is provided in the appendices.

2 Overview of Information Extraction

In this section, we focus on providing an introduction to the field of IE by presenting background information on data sources, learning methods, and evaluation metrics. We will begin by briefly explaining what IE is. Then, we compare IE to both full text understanding and information retrieval, since these two other tasks are related to IE. Finally, we describe data sources, learning methods, and evaluation metrics at a high level. This background will aid the reader in understanding the IE applications described in later sections of this report.

2.1 *Information Extraction (IE)*

Research in IE has grown since the mid-1980s. The task of IE systems is to extract not only features, such as names or locations, but also the relationships among those features, from a natural language text. IE itself is based on the existence of implicit internal structure in natural languages. Researchers utilize such structure to build IE systems that convert unstructured information into structured information, such as specific features and values. Table 2.1 portrays a simplified example of extracted features and their values.

Table 2-1: IE Example - extracted features and their values

Feature	**Feature Value**
Name	ALEXANDER F. TOMLD
Height	SIX FEET TALL
Weight	170 TO 180 POUNDS
Eye Color	BLUE
Hair Color	BROWN

2.2 *Information Extraction vs. Full Text Understanding*

Full text understanding requires that a computer understand natural language text. This is a difficult task, as natural language text is often too complex to be fully understood by people – much less by a computer. In order to mitigate the complexity, the scope of full text understanding can be narrowed to IE. This is the viewpoint of [11], which argues that "[i]nformation Extraction is a more limited task than full text understanding".

While narrower in scope, IE maintains its usefulness in applications where users are only interested in identifying specific features in text rather than understanding the full set of text, such as name identification. For instance, if one would like to identify the people reported on in a given newspaper, it is not necessary for a computer to understand every article found in the newspaper. On the contrary, it is sufficient to scan the articles for a particular name; name identification is a typical task of IE. In general, IE simplifies the problem of full text understanding by ignoring much of the textual information.

2.3 *Information Extraction vs. Information Retrieval*

It is important to begin by stating that information extraction is *not* information retrieval (IR). [12] defines IR as "the task of finding documents, usually text, which are relevant to a user's information need." Google™, a well-known IR system for the web, is a prototypical example of an IR system (see [13] for detail). Just like the results produced by a Google™ web search, the output of an IR system is a subset of documents that are relevant to a user's query. In contrast, the goal of an IE system is not to extract the documents themselves but, rather, to extract pre-specified *features* from the pertinent documents. In an IE system, these extracted features are usually entered into a database automatically. In short, IR is document retrieval while IE is feature retrieval. Figures 2.1 and 2.2 depict the difference between IE and IR.

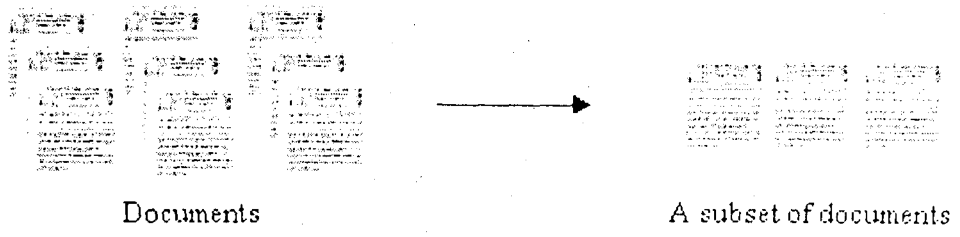


Figure 2-1: Information retrieval

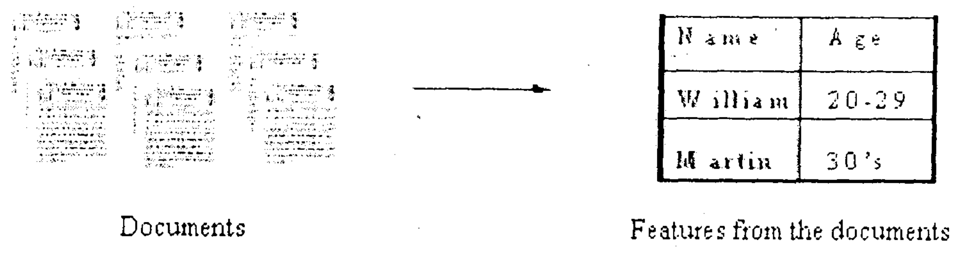


Figure 2-2: Information extraction

Although [14] points out that "[b]oth IR and IE are difficult because they must overcome the ambiguities inherent in language," the degree of their difficulty varies. IE poses unique difficulties because it requires more detailed knowledge about a document (such as its organization) than IR requires. Furthermore, IE systems are often required to establish relationships between features, which is not necessarily a requirement of an IR system.

Full text understanding, information extraction, and information retrieval can be viewed as three different types of textual information access, as all of them need to understand textual information at some level. However, as noted the degree to which each process understands the textual information varies. In general terms, this difference allows these three tasks to be ordered based on the difficulty of achieving textual information access. Of the three, IR has the least difficulty achieving textual information access, while full text understanding has the greatest difficulty. This progression of complexity is depicted in Figure 2.3.

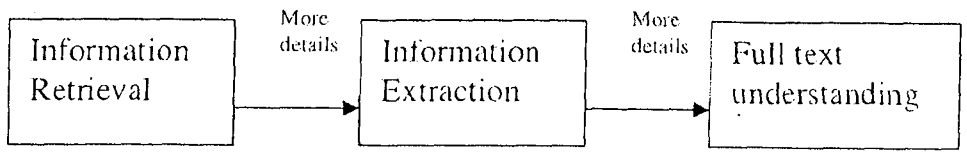


Figure 2-3: Relationship between IR, IE, and full text understanding

2.4 *Learning Methods*

There are many learning methods employed in IE. Generally, they can be classified into one of four categories: supervised learning, semi-supervised learning, active learning, and unsupervised learning. Among these categories are two more general categories: "supervised" and "unsupervised." If the input text for learning is annotated (i.e., labeled in a training set), the learning method employed belongs to the "supervised" category. If, on the other hand, the input text is unannotated, the learning method is termed "unsupervised." The remaining categories fall somewhere in between these general categories. To varying degrees, all of these methods require knowledge engineering; for example, annotating input text or crafting IE rules is often performed manually and requires extensive effort by human experts.

Additionally, learning methods used in practice may belong to more than one category. For example, transformation-based error-driven learning can be either supervised or unsupervised. We discuss an example of the use of transformation-based error-driven learning in IE in Section 3.2.1 below.

2.4.1 Manually Crafted Rule-based IE Systems

As noted all approaches to IE involve some degree of knowledge engineering. In some IE systems, human experts construct rules for the system manually using knowledge of the application domain. The computer system does not learn from the data but only implements what human experts have learned and programmed into the system. As such, the skill of the human expert plays a crucial role in the performance of these systems. While these systems are time-consuming to build and difficult to maintain, most of the best performing systems have manually crafted rule bases of this nature.

Systems based on manually crafted rules have advanced to the point that commercial tools are now available. AeroText™ [15] is one such tool that simplifies knowledge engineering. According to the Lockheed Martin M&DS, the company that developed AeroText™,

> "AeroText™ is a high-performance free-text natural language processing system, which includes a core knowledge base of compiled linguistic rules, and an integrated development environment combined with a graphical user interface for the creation of additional linguistic rules. M&DS developed AeroText as a powerful component technology to extract various entity types and relationships including the names of people, organizations, dates, locations, and telephone numbers."

One of the IE systems we developed in our preliminary work is based on this approach. The details of this system are discussed in Section 3.1.

2.4.2 Supervised Learning

Since manually crafting rules relies on extensive human expertise, other approaches have been developed to automate the process. As noted, one such method is supervised learning. The goal of supervised learning is to learn a model to classify instances automatically. Supervised learning is well known as a classification task. For example, if one wanted to build a system to help a person buy a used car, one could choose a car's make, color, mileage, and year as features. The system might have a list of sample instances (i.e., cars) with distinct values for each feature (e.g., color = "blue" or mileage = 76,510). Each instance is then manually assessed by a human expert and assigned a class that serves to classify the information. Continuing the used car example, the classes might be 'buy' or 'do not buy.' Together these instances and their class labels form a training set that can be used as input to a supervised learning scheme.

There are numerous supervised learning schemes that have been developed over the years such as decision trees, decision tables, and classification rules. As noted earlier, transformation-based error-driven learning can also be classified as a supervised learning method.

Supervised learning can be employed to learn patterns from training data (in the form of an annotated corpus) without the aid of a human expert. Depending on the difficulty of the (manual) annotation process, this can result in a significant reduction in knowledge engineering cost as compared to manually crafted rule-based systems. However, the success of supervised learning is dependent on having sufficient training data. In sum, although supervised learning saves human expert time that would otherwise be spent in, for example, rule development, the hidden cost is the labor-intensive annotation, or labeling, of training data.

2.4.3 Semi-Supervised Learning

To deal with the heavy reliance on human expertise that is required by supervised learning methods, semi-supervised learning is an alternative method that has been employed in IE systems. [16] describes semi-supervised learning as follows:

> "Using semi-supervised learning, a system learns from a mixture of labeled (annotated) and unlabeled data. In many applications, there is a small labeled data set together with a huge unlabeled data set. It is not good to use only the small labeled data set to train the system because it is well known that when the ratio of the number of training samples to the number of feature measurements is small, the training result is not accurate. Therefore, the system needs to combine labeled and unlabeled data during training to improve performance. The unlabeled data can be used for density estimation or preprocessing of the labeled data, such as detecting inherent structure in the domain. In other words, the system extracts patterns from the annotated data, and labels the unannotated data automatically using the patterns. As a result, all data are labeled for the training."

Therefore, semi-supervised learning saves human effort while maintaining the performance level of supervised learning techniques.

2.4.4 Active Learning

Active learning is another approach to solve the large knowledge engineering cost associated with supervised learning. As described in [17], active learning "identifies a subset of the data that needs to be labeled with the participation of human experts. After that, it uses this subset to generate classification models." One active learning task described in [18] is designed to "help users select suitable features in order to minimize the number of examples the user must annotate. This task is called selective sampling, which is significant in natural language processing since people usually like to annotate only the most important features in an abundance of text."

Like semi-supervised learning, active learning reduces human expert effort by annotating unlabeled text while still maintaining accuracy. The two approaches can be distinguished by the point at which the annotation is done: in supervised and semi-supervised learning, annotation is completed prior to the application of the learning algorithm; in active learning, annotation is part and parcel of the learning algorithm.

2.4.5 Unsupervised Learning

The focus of the various learning schemes discussed so far has been on reducing knowledge engineering cost. The ideal situation would be to discover IE rules from unannotated data, thereby significantly reducing knowledge engineering cost. While it would appear difficult to extract meaningful information from such data, it turns out that there are techniques for finding regularities in data and learning a model that captures these regularities. These techniques are classified as "unsupervised learning" because the algorithm works with little or no annotated data. [19] summarizes this learning method as follows:

> "The basic approach of unsupervised learning includes the following steps. First of all, an unsupervised learning system is seeded with a couple of labeled facts or patterns. Next, the system searches a large unannotated corpus for new candidate patterns based on the seeds. After the new patterns are found, the system can use them to uncover additional facts. The system then adds the facts to the seed set. After that, the system is retrained based on the new extended seed set. This process is repeated until no more patterns can be found."

2.5 *Commonly Used Evaluation Metrics*

Now that we have discussed various learning methods, it is important to establish a foundation on which to evaluate the techniques. For many IE applications, there are two sets of feature values used during learning. The first set is the "target set" in the input data, or the values the system should extract. These are feature values that, for example, in a supervised learning framework, would have been annotated or labeled by a human expert. This set is also referred to as the "ground truth." The second set, the "selected set," is set of values selected by the system. These are the feature values that the IE system actually extracts (as opposed to what the system should extract). Since the IE system can make mistakes, it is necessary to quantify the error. Thus, in IE it is common to define true positives (*TP*) as those feature values that the IE system extracts correctly – i.e., values that occur in both the target and selected sets. True negatives (*TN*) on the other hand are those feature values that are neither in the selected set nor in the target set. The wrongly chosen feature values in the selected set are called false positives (*FP*). The feature values in the target set that were incorrectly not selected are called false negatives (*FN*). Figure 2.4 from [1] depicts a diagram of the relationship between *TP*, *TN*, *FP*, and *FN*.

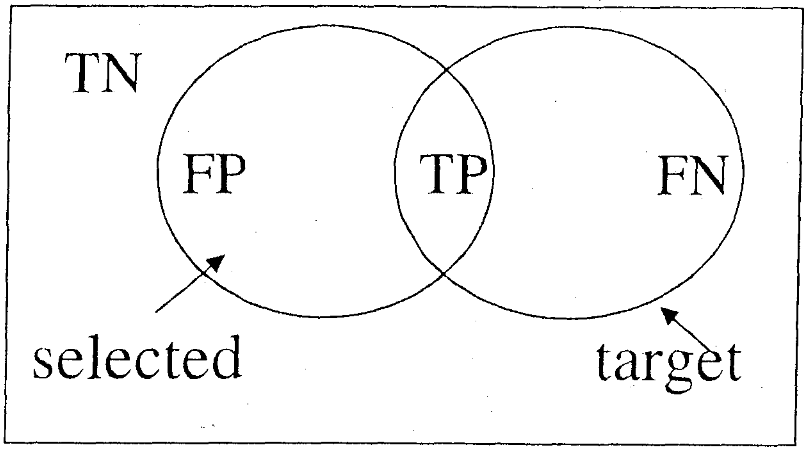


Figure 2-4: A diagram of TP, TN, FP, and FN

Frequently used evaluation metrics include precision and recall, which rely on the quantities defined above. Precision (P in Equation 2.1) is calculated as TP/(TP+FP), and recall (R in Equation 2.1) as TP/(TP+FN). Values for both precision and recall range between 0 and 1, inclusive. When evaluating a technique, the higher the precision and recall, the better the result.

One important observation is that precision and recall are frequently inversely related. One generally has to sacrifice precision to increase recall and vice versa. For example, if an information retrieval system returns an entire collection of documents in response to a query, the recall is 100%, but the precision may be very low. On the other hand, if the system returns only a few relevant documents, the precision would be high but recall likely low. Usually, a precision-recall curve is used to plot this tradeoff. Figure 2.5 shows a sample of such a curve.

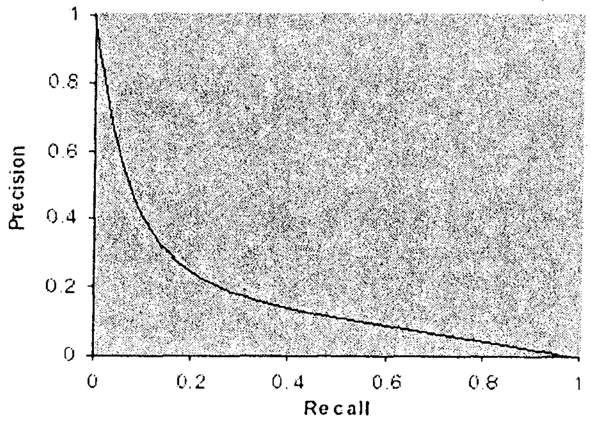


Figure 2-5: A sample precision-recall curve

Both precision and recall are important in assessing performance and can be combined into a single measure of overall performance. One such measure is F_β ("f-beta"), which combines precision and recall using Equation 2.1 from [2].

$$F_{\beta} = \frac{(\beta^{2} + 1)PR}{\beta^{2}P + R}$$

Equation 2-1: F_β

The parameter β determines the relative importance of precision. When β equals one, the weight given precision and recall is the same. If β is less than one, then precision is weighted more heavily than recall. On the other hand, if β is greater than one, then recall becomes more important than precision. Using F_β, it is straightforward to compare the performance of systems having different precision and recall. In summary, following [2] we conclude that

- $\beta=0$ implies that F_β = precision
- $\beta=\infty$ implies that F_β = recall
- $\beta=1$ implies that recall and precision are equally weighted
- $\beta=0.5$ implies that recall is half as important as precision
- $\beta=2$ implies that recall is twice as important as precision

3 Related Work

Much work has been done in study of information extraction (IE). Numerous algorithms and commercial systems have been developed to handle the task of understanding textual data. Yet, there still remains much work to be done. In this section, we begin by discussing several popular commercial IE systems that are currently available. As part of this analysis we identify limitations of these systems in order to motivate our own research. This is followed by a review of several prominent IE algorithms. The final subsection summarizes open problems in the IE field.

3.1 *Commercial IE Systems*

There are several commercial tools currently available that support information extraction. The more advanced of these systems are NetOwl® [21], ClearForest [22], AeroText™ [15], and IBM Intelligent Miner for Text [23]. At the time of this survey[1], the deficiency in NetOwl® and IBM Intelligent Miner for Text was that neither system supported user-defined feature extraction. This severely limits the types of features that can be extracted. For example, it is not possible to use the NCIC [24] codes as a basis for information extraction with such tools despite the fact that these codes form the basis for one of the nation's most advanced Criminal Justice Information Systems. The same survey results revealed that ClearForest and AeroText™, on the other hand, only supported the manual creation of user-defined feature extraction rules. As noted, this results in a very high knowledge engineering cost for users of such tools. In contrast, our proof-of-concept system supports the automatic generation of user-defined features based on our semi-supervised learning algorithm. This significantly reduces the knowledge engineering cost of using our system as compared to AeroText™, ClearForest, and other commercially available systems.

In addition, our segmentation algorithm[2] is more flexible than that provided, for example, in AeroText™. At the time of our survey, AeroText™ supported only the use of sentences and paragraphs as segments, but in our research we have determined that sub-sentence segmentation is required to precisely extract certain features [25]. As a result, our segmentation algorithm enables users to define their own segmentation methods. Another key capability that distinguishes our approach from other tools such as AeroText™ is the representation of patterns through the use of reduced regular expressions. AeroText™, for example, uses a manual pattern

[1] This survey was conducted in 2004 and conclusions are based on versions of the software available at that time.
2 Segmentation is the process of organizing reports into (portions of) sentences.

formation method based on assigning words in the input to bins in the pattern. Theoretically, the rules created using AeroText™ are not as powerful as the reduced regular expressions defined in [25], and, as a result, our algorithm is able to represent a wider range of patterns.

There are few if any published results relating to criminal justice that measure the performance of currently available commercial systems. Of the commercial systems we surveyed, results have been published for NetOwl®, ClearForest and AeroText™. Based on the widely employed F_β metric (Equation 2.1), NetOwl® (Isoquest. Inc, 1998) achieved a performance of 93.99% on the template-element extraction task of MUC-7 [27]. MUC-7. however, is a collection of newspaper articles. Results have been published for ClearForest as the winner of the 2002 KDD Challenge Cup competition in the biomedical domain [20]. In addition, AeroText™ results were presented in a technical report at the 2002 NIH Biomedical Computing Interest Group (BCIG) seminar [56]. Based on our search of the publicly available information, however, we found no published information extraction performance results in terms of precision, recall, or F_β related to the criminal justice domain for any of the commercial products we surveyed. This is not to say that these products are not being used effectively in the criminal justice domain – only that at the time our survey was conducted, we were unable to identify publicly available materials that report performance in terms of these metrics.

In contrast, we have demonstrated that our proof-of-concept system as described in [25] achieves excellent results on ten often-used features from police incident reports. We compare our approach and results with [29] as well as with other academic information extraction systems in the following section.

3.2 *IE Algorithms*

In this section we highlight a few different learning algorithms used in IE. Early named-entity extraction systems used hand-crafted rules which required domain experts to read through the training data, write rules using domain knowledge, test the rules, and refine the rules. Often, designing these systems required several iterations of this cycle to produce the desired results. This fully manual approach requires not only that the human expert have domain-specific knowledge, but also that they know how to write named-entity extraction rules.

Recognizing their efficiency, supervised machine learning approaches came to be widely used in named-entity extraction. In these approaches, annotated data is provided to a machine learning algorithm to automatically discover rules from the data. Once a learning algorithm has been developed, users do not need to know the details of the algorithm; the only job for users is to prepare good quality training data.

There are several popular named-entity extraction algorithms that are used to learn extraction rules. These include, but are not limited to, the following: Transformation-Based Learning (TBL), Hidden Markov Models (HMM), Maximum Entropy Models, and Support Vector Machines. In what follows we briefly survey these approaches.

3.2.1 Transformation-Based Learning (TBL)

Transformation-Based Learning (TBL) uses pre-defined templates to generate rules that determine whether or not a word belongs to a named entity. Since one of the early applications of TBL was part-of-speech tagging [10], it is helpful to think of each word in a named entity as having a special tag. TBL is adept at learning rules that assign these tags to words in previously unseen data.

TBL employs an iterative, greedy algorithm during learning. Although many different transformations may be discovered during learning, in a given iteration only the transformation that yields the best performance (e.g., the highest value of a scoring function) is chosen. The newly learned transformation is then added to a set of output rules for use in extraction.

Using TBL, William J. Black [30] extracted persons, locations, organizations, etc. from a collection of news wire articles written in Dutch and Spanish. His average $F_{\beta=1}$ scores were 82.35% for Dutch-language articles and 80.79% for Spanish-language articles.

3.2.2 Hidden Markov Models (HMM)

The use of Hidden Markov Models (HMM) is another learning approach used in IE. An HMM consists of a set of states Q, with initial state q_i and final state q_f. The model includes a set of transitions between states ($q \rightarrow q'$) and output alphabet $\sum = (\delta_1, \delta_2, \ldots, \delta_m)$.

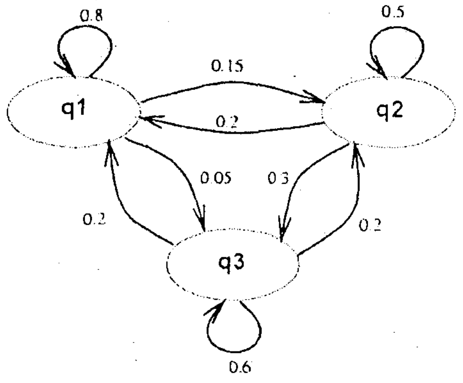


Figure 3-1: Hidden Markov Model

By applying the algorithm to the training data the states, transition probabilities $P(q \rightarrow q')$ and emission probabilities $P(q \uparrow \delta)$ are learned. During testing, the probability of a string being emitted by a HMM is computed as a sum over all possible paths.

In an application of HMMs, Kristie Seymore [31] extracted title, author, affiliation, address, email, date, phone, keywords, and several other features from headers (first paragraph or page) of computer science research papers. The average $F_{\beta=1}$ score obtained was 90.1%.

3.2.3 Maximum Entropy Model

Another popular information extraction algorithm is the Maximum Entropy Model. In this approach, each feature is divided into four sub-classes: feature-begin, feature-continue, feature-end, and feature-unique. Equation 3.1 depicts the most-often used equation utilized in the maximum entropy approach.

$$P(f \mid h) = \frac{\prod_i \alpha_i^{g_i\{h,f\}}}{Z_\alpha\{h\}} \qquad Z_\alpha\{h\} = \sum_f \prod_i \alpha_i^{g_i\{f,h\}}$$

Equation 3-1 Functions for the Maximum Entropy Model

These equations define the following values: P(f|h) is the conditional probability that the current word belongs to class f (e.g., a named entity) given a history of events h. $Z_\alpha\{h\}$ is used for normalization given α_i, which is a weight learned during training. $g_i\{f,h\}$ is a binary feature function that allows the learning algorithm to distinguish features. Consider, for example, the case of a capitalized word. If a word is capitalized, then we might assume that it is likely that the word is a personal name. If we used $g_i\{f,h\}$ to represent this scenario, we would then have the following binary function:

$$g_i\{h,f\} = \begin{cases} 1: \text{if } h \text{ is a capitalized word and } f = \text{named entity} \\ 0: \text{else} \end{cases}$$

In most of systems, the binary functions $g_i\{h,f\}$ are provided by the user. Since α_i is learned during training and P(f|h) can be computed, these values do not need to be provided by the user. However, a drawback of this approach is that the Maximum Entropy Model requires intensive feature-specific labeling to create the training data needed by the algorithm.

Hai Leong Chieu and Hwee Tou Ng applied the Maximum Entropy Model to extract features such as speaker, location, start time and end time from seminar announcements in their research [32]. The best average $F_{\beta=1}$ score they achieved was 86.9%.

3.2.4 Support Vector Machines (SVM)

The named-entity extraction problem can also be converted to a word classification problem and solved using support vector machines (SVM). The SVM algorithm attempts to discover a plane that gives the greatest separation between classes, termed the maximum margin hyperplane. Classes are distinguished by support vectors that are instances on the boundary of the maximum margin hyperplane. Therefore, the task of the SVM is to discover the support vectors and the corresponding maximum margin hyperplane.

Paul McNamee [33] applied a SVM to learn persons, locations, organizations, etc. from the same collection used by Black in [30]. His work produced $F_{\beta=1}$ scores of 59.52% for the Dutch-language articles and 60.97% for the Spanish-language articles.

3.3 *Open Problems in Information Extraction*

The growth of the IE field has brought with it several challenges that remain to be solved.

3.3.1 Knowledge Engineering Cost

One of these challenges is the high cost of knowledge engineering. Traditionally, as noted previously the usual approach to perform IE is to use hand-crafted rules. As was also discussed, manual construction of the rules requires not only linguistic and domain expertise, but a great deal of time. As mentioned previously, a common solution to this problem has been to adopt machine learning techniques well-known from corpus linguistics: to manually annotate a corpus of relevant texts and automatically learn rules from such corpora [34]. The use of machine learning is motivated by the fact that the cost of labeling documents is usually considerably less than the cost of writing the extraction rules by hand [35] [36]. However, the quality of inferred rules crucially depends on the size of the corpus, and manual annotation is also expensive. There are several methods that can be used to partially solve this problem, such as relevance statistics, active learning, and bootstrapping. However, the problem remains open at this time.

3.3.2 Scalability

Another open challenge in the IE field is to build scalable applications that cover new domains (e.g., medical abstracts, scientific papers, police reports, etc.) [37][38][39]. For most IE systems/algorithms, a domain-related lexicon or knowledge base is required to manipulate domain-specific concepts. It is very difficult to automatically generate these support files for different domains. A related scalability issue occurs when adapting IE systems to linguistic features specific to different domains since different grammars and lexicons must be provided to enable the system to cope with specific linguistic constructions typical of the application/domain. An example of this occurs in online news media, where web-based blogs can radically differ from online newspaper-like texts.

3.3.3 Accuracy

Data inaccuracy is also a source of problems. For instance, the data may be erroneous, incomplete, or inconsistent. Or, if the data is accurate, it may be poorly labeled (e.g., incomplete or inconsistent annotation). Inaccurate data impacts IE system performance [40] and is a problem that needs to be addressed.

4 Reduced Regular Expression Discovery Algorithm for Information Extraction

Having provided a background to the field of information extraction (IE) and outlined some of the open problem in the field, this section of our report presents a semi-supervised, active learning Reduced Regular Discovery IE algorithm that we developed as part of this project. The algorithm is a (greedy) covering algorithm that discovers reduced regular expressions that are used as patterns to perform information extraction. In this section we provide a definition of reduced regular expression, give an extensive description of our semi-supervised RRE Discovery algorithm, analyze the precise time complexity of the algorithm, survey the scoring functions evaluated for use in our algorithm, and present our approach to active learning within the semi-supervised framework of our RRE Discovery algorithm.

4.1 *The Definition of RRE*

Reduced regular expression (RRE): Given a finite alphabet Σ, a reduced regular expression is defined as a set:

- $\forall \alpha \in \Sigma$, α is an RRE and denotes the set $\{\alpha\}$.
- All words in our lexicon and all part-of-speech tags in the Penn tag set [1] belong to Σ.
- ^∈ Σ, $∈ Σ, where ^ is the start of a line, and $ is the end of a line.
- [0-9]∈ ∑, where [0-9] represents any single numeric digit.
- [A-Z]∈ ∑, where [A-Z] represents any single alphabetic character (upper or lower case).
- All punctuation characters belong to ∑.
- All white space characters belong to ∑.
- ε∈ ∑ is the null symbol (i.e., the empty string).
- (α|ε) is an RRE, where α ∈ ∑ is any single alphanumeric or punctuation character.
- If r and s are RREs denoting the languages R and S, respectively, then (rs) is an RRE that denotes the set RS.

The major difference between regular expressions and reduced regular expressions is that reduced regular expressions do not support Kleene closure (i.e., '*'). Examples of regular expressions that are not RREs include: α*, (α|β), and α+, where α and β are arbitrary word or part-of-speech members of Σ. We have not found it necessary to support such regular expressions to achieve high accuracies.

4.2 *The RRE Discovery Algorithm Framework*

In this section we present our semi-supervised approach to the discovery of RREs from a small set of labeled training segments. The process begins with pre-processing and is followed by the application of a greedy algorithm to discover RREs. We detail these steps in the following sections.

4.2.1 Pre-Processing

Our semi-supervised learning algorithm requires three pre-processing steps: segmentation, segment labeling, and part-of-speech tagging.

4.2.1.1 Segmentation

At this stage, each incident report is split into segments. Each segment becomes an instance in our system. The first step splits the input into sentences using the technique presented in [9] to detect sentence boundaries. The input data is further divided on commas, although there are a small number of cases where commas are not segment boundaries. We also assume that no features cross segment boundaries. This assumption is practical for a number of important attributes, including those discussed in section 5.2.1 on results.

4.2.1.2 Segment Labeling

Prior to the start of the labeling stage, domain experts must identify the attributes that will be extracted. For instance, if the high-level goal is to extract physical descriptions of suspects, the list should include Height, Weight, Eye Color, etc. During training set development, each segment is evaluated manually and assigned one or more labels. For example, if a segment includes Height and Weight information, the domain expert assigns both of these labels to the segment. After labeling, each feature has its own true set and false set.

4.2.1.3 Part-of-speech Tagging

Part-of-speech tags are also used in our RRE Discovery algorithm. Each word in the training set must be assigned its correct part-of-speech tag before the learning process begins. Currently, we are using Eric Brill's [10] part-of-speech tagger to tag our training sets. Brill's tagger uses the Penn tag set [1] (Table 4.1 contains some examples of Penn tags). We have enhanced our lexicon to include extra tags for feature extraction from police incident reports. For example, {CDS} is used for plural numbers such as "twenties".

Table 4-1: Example tags from Penn tag set

Tag	Category	Example
CD	Cardinal number	3, fifteen
IN	Preposition	in, for
PRP	Pronoun	they, he
PRP$	Determiner, possessive	their, your
DT	Determiner, article	the, both
CC	Conjunction	and, or
RB	Adverb	ago, very
JJ	Adjective	happy, bad
NN	Noun, singular	aircraft, data
NNP	Proper Noun	London, Reston
NNS	Plural Noun	books, years
VBG	Verb, present participle	taking, living
VBP	Verb, base present	are, take
VBD	Verb, auxiliary *be*, past	was, were
VBZ	Verb, auxiliary *be*, present	is

4.2.2 Learning Reduced Regular Expressions

The goal of our algorithm is to discover sequences of words and/or part-of-speech tags that have high frequency in the true set, while having low frequency outside the true set. The algorithm first discovers the most common element of an RRE, termed the root of the RRE. The algorithm then extends the RRE using an "AND" operator, after which it discovers the gaps between elements of the RRE. Finally, the start and the end of the RRE are learned. Figure 4.1 depicts the entire learning process. Figure 4.2 portrays the same process in algorithmic form. An example (in section 4.2.3) explains how the algorithm works at each stage of the learning process.

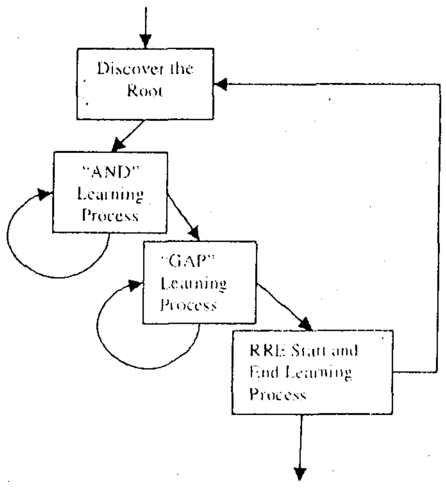


Figure 4-1: RRE Discovery

```
Do {
    Find the ROOT
    "AND" learning
    "GAP" learning
    "RRE Start/End" learning
    Prune true set
} While True Positives ≥ δ
```

Figure 4-2: RRE Discovery Algorithm

Our approach employs a covering algorithm. After an RRE is discovered, the algorithm removes from the true set all segments covered by the RRE. The remaining segments become a new true set and the steps in Figure 4.1 repeat. The learning process stops when the number of segments left in the true set is less than or equal to a threshold δ. We use this threshold because overfitting results if too few segments are used to discover an RRE. δ is a parameter in our system, and must be an integer. By default δ is set to two.

Our approach is a semi-supervised learning method. Instead of labeling the exact location of features in a training set, the training-set developer need only record whether a specific feature of interest occurs in a segment. Nonetheless, the RRE learned by the algorithm precisely matches the feature of interest – no more and no less. We depict the details of each step of the algorithm in what follows.

4.2.2.1 Discovering the Root of an RRE

In this step, each word and/or part-of-speech tag (specified as a simple RRE) in each true set segment is matched against all segments. The performance of each such RRE in terms of F_β (Equation 2.1) is considered. To reiterate, in Equation 2.1 P = precision = TP/(TP+FP) and R = recall = TP/(TP+FN), where TP are true positives, FP are false positives, and FN are false negatives. The parameter β is the ratio of recall to precision and enables one to place greater or lesser emphasis on recall versus precision depending on the needs of the application.

The element with maximum F_β is chosen as the root of the RRE. The algorithm discovers a word or part-of-speech tag that has a high frequency of occurrence in segments containing the desired feature. It must also have low frequency in segments that do not contain the desired feature. For example, the root discovered for the attribute Age in the example given in section 4.2.3 is the part-of-speech tag "{IN}". Because each element in each and every segment in the true set must be tested during root discovery, the time complexity of root discovery is equal to the number of elements in the true set.

Our approach places less emphasis on precision and more on recall during the root discovery process. We use the parameter β_{root} (by default set to six) to control this. Naturally, this results in a larger set of segments that match the root. These segments, however, are not necessarily all true positives. As a result, the "AND", "GAP", and "Start/End" learning phases all prune false positives from the set of segments that match the root RRE. As will become evident, this results in RREs that have both high precision and high recall.

4.2.2.2 "AND" Learning Process

After the root is discovered, the algorithm tests additional words and/or part-of-speech tags before and after the root. The algorithm places new candidate elements immediately before and after the root, thereby forming two new RREs. The RRE with the highest F_β replaces the previous RRE. For example, starting with the root "{IN}" for the attribute Age, the RRE "years <GAP> {IN}" may be discovered after the first pass of the "AND" learning process, where <GAP> is the gap discovered between these two elements[3]. Intuitively, element adjacency implies the use of an "AND" operator – thus the name "AND learning process."

The new RRE is then extended in the same way. As before, candidate elements are inserted before and after the current RRE. The algorithm measures the performance of each extended RRE, and the RRE with the maximum F_β is selected. In this sense, our algorithm is greedy. Continuing the previous example, the RRE after the second pass of the "AND" learning process may be "{CD} <GAP> years <GAP> {IN}", where "{CD}", "years", and "{IN}" are elements of the RRE and <GAP> represents the gaps that have been learned.

Candidate elements consist not only of words and part-of-speech tags, but also can be "numeric" tokens composed of the digits [0-9]. The lengths of such tokens provide clues useful in RRE discovery. For example, two digit tokens such as "23" are often a person's age, whereas four digit tokens such as "2003" are likely to be a year. Similarly, a person's height always has one digit for feet and one or two digits for inches, and a person's weight usually has three digits.

[3] The details of gap discovery will be discussed in section 4.4.4.

The overall complexity of the "AND" learning process depends on both the number of candidate elements and the maximum number of elements in an RRE (a system parameter). More formally, C_{AND} is the complexity of the "AND" process, where S_{ij} is the set of candidate elements for the jth position in the ith pass of the "AND" learning process in a single iteration of Figure 4.2. Although S_{ij} can contain many elements, there are only two positions to test in each pass of the "AND" learning process. One is the position before the current RRE and the other is the position after the current RRE. An example of how S_{ij} changes during the "AND" learning process is depicted in Figure 4.3. The actual size of S_{ij} depends on the number of candidate elements for each position j=1 and j=2 in a segment. The algorithm is data-driven in this regard.

Generalizing from this example, we can see that in the "AND" learning process there are at most N-1 passes required to test the N-1 positions between a maximum of N elements. Thus the "AND" learning process ends when at least one of the two following conditions is met: either the number of elements in the RRE reaches the maximum N, or all candidate elements in the segment have been tested in all allowable positions.

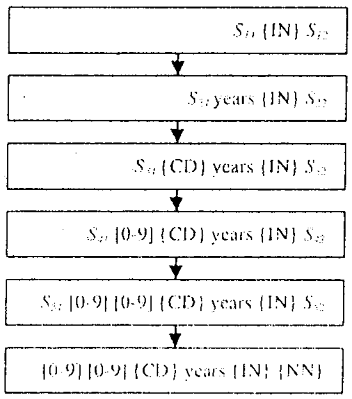

S_{11} {IN} S_{12}
S_{21} years {IN} S_{22}
S_{31} {CD} years {IN} S_{32}
S_{41} [0-9] {CD} years {IN} S_{42}
S_{51} [0-9] [0-9] {CD} years {IN} S_{52}
[0-9] [0-9] {CD} years {IN} {NN}

Figure 4-3: "AND" learning example for the *Age* attribute

Given that $0 \leq |S_{ij}| \leq |S|$, we have $0 \leq C_{AND} \leq 2(N-1)|S|$. Therefore, the complexity C_{AND} is bounded by $O(0) \leq C_{AND} \leq O(N|S|)$; it is represented in Equation 4.1.

$$0 \le C_{AND} \le \sum_{i=1}^{N-1} \sum_{j=1}^{2} | S_{ij} |$$

Equation 4-1: C_{AND}

The fact that the algorithm is data-driven improves its performance – this can be deduced by comparing the actual complexity of C_{AND} with the upper bound $2(N-1)|S|$. In other words, due to the fact that a given segment does not contain many words, in practice $|S_{ij}| << |S|$. In fact we have found a value of ten to work well for the max number of elements N allowed in an RRE.

4.2.2.3 "GAP" Learning Process

In the "GAP" learning process, our algorithm learns the length of the gap between elements of an RRE. For example, suppose an RRE learned for the Date attribute is "{CD} {MONTH}", where {MONTH} is the part-of-speech tag for January through December. Because our algorithm is designed to learn gaps between adjacent elements automatically, it can, for example, learn that a gap of zero to two elements between {CD} and the following {MONTH} is optimal. An example of this type of RRE is "{CD} {token}{0,3} {MONTH}", where {token} is a word or part-of-speech tag in $\sum$. This RRE allows at most three elements between a number and a following month, e.g., "the{DT} 5th{CD} of{IN} January{MONTH}". In this case, the word "of" followed by its part-of-speech tag followed by the word "January" form a gap that is three elements in size.

In this simple date attribute example we assumed that gaps are measured in terms of elements – in fact, the execution time performance of our algorithm is significantly improved if gaps are measured in terms of characters. As a result, we measure gaps in terms of characters and use an input parameter ψ to control the maximum gap allowed between any two adjacent elements in an RRE. Our system initializes ψ = 100 based on the observation that no adjacent elements are separated by more than 100 characters in our training data. For each gap, our algorithm first determines $\phi = \min(\psi, \omega)$, where ω is the longest gap in any segment in the true set between the two elements under consideration. For example, in the segment "the{DT} 5th{CD} of{IN} January{MONTH}", including spaces there are 15 characters between '{CD}' and '{MONTH}', and $\phi = \min(100,15)$, which is 15.

Once ϕ has been determined, the algorithm tests ".{0, ϕ}" as the gap. In this expression, the '.' represents any single character member of Σ. The syntax ".{0, ϕ}" means that any single character member of Σ can occur between 0 and ϕ times. The algorithm, in turn, tests gaps of ".{0, ϕ-1}", ".{0, ϕ-2}", ..., "{0,0}". The smallest gap that does not decrease the current RRE's performance is used as the final gap between two elements of the RRE. Gaps between different elements can differ in size.

The time complexity for the "GAP" learning process is C_{GAP}, where G_i is the number of comparisons to identify the optimal gap between RRE element i and element i+1. $0 \le G_i \le \psi$. Given N, the maximum number of elements in an RRE, the complexity of gap discovery C_{GAP} is thus $0 \le C_{GAP} \le (N-1)\psi$. Therefore, $O(0) \le C_{GAP} \le O(N\psi)$. C_{GAP} is depicted in Equation 4.2.

$$C_{GAP} = \sum_{i=1}^{N-1} G_i$$

Equation 4-2: C_{GAP}

4.2.2.4 Start and End Learning

The start symbol "^" and end symbol "$" of a segment are also useful in RRE discovery. As a result, our algorithm tests whether the current RRE should include the start symbol and/or the end symbol. We first insert the start symbol at the beginning of the RRE to form a new RRE. If it has equal or better performance compared to the previous RRE, then the new RRE with the start symbol replaces the previous one. We deal with the end symbol in a similar manner. In addition, if the initial element of an RRE is a part-of-speech tag, our algorithm ensures that the RRE also covers the word before the tag.

The time complexity of this learning process is O(1) since only two candidate RREs are tested.

4.2.3 Example

In this section we use the Age attribute to illustrate our semi-supervised learning algorithm in detail. Tables 4.2 and 4.3 contain the initial true and false sets respectively. The characters between "{" and "}" are part-of-speech tags. {MALE} is a special tag for words of male gender, and {FEMALE} is the tag for words of female gender.

There are two distinct Age patterns in the true set. One is illustrated by the feature value "in her twenties", the other by "25 years of age". Our semi-supervised algorithm discovers two distinct RREs, one based on each Age pattern. In other words, in this example our covering algorithm completes two iterations of Figure 4.2 during RRE discovery.

Table 4-2: True set for example

Segment Number	True segments
1	six{CD} feet{NNS} tall{JJ} and{CC} 25{CD} years{NNS} of{IN} age{NN}
2	both{DT} 30{CD} years{NNS} of{IN} age{NN} with{IN} cornrows{NNS}
3	in{IN} his{MALE} twenties{CDS} standing{VBG} six{CD} feet{NNS} tall{JJ}
4	they{PRP} are{VBP} in{IN} their{PRP$} thirties{CDS}
5	she{FEMALE} was{VBD} in{IN} her{FEMALE} twenties{CDS}
6	Tom{NNP} is{VBZ} in{IN} his{MALE} early{JJ} teens{CDS}

Table 4-3: False set for example

Segment Number	False segments
1	the{DT} first{JJ} man{MALE} is{VBZ} in{IN} his{MALE} car{NN}
2	in{IN} the{DT} roaring{VBG} twenties{CDS}
3	in{IN} the{DT} 111{CD} block{NN}
4	25{CD} years{NNS} ago{RB}
5	5{CD} feet{NNS} 8{CD} inches{NNS} tall{JJ} with{IN} a{DT} tall{JJ} thin{JJ} build{NN}
6	weighing{VBG} 180{CD} pounds{NNS}

In the first iteration of the algorithm, the root "{IN}" of the first RRE is discovered with $F_{\beta=6.0} = 0.98$. Next, the "AND" learning process extends this root to the five RREs portrayed in Figure 4.4 in sequence. In order to prune the false positives covered by the root, β is set to 0.5 in this example (by default it is set to two in our system). As noted previously, greater emphasis is placed on precision during "AND" learning in this way. Each of these five RREs has $F_{\beta=0.5} = 0.71$. Thus, in this example R_{and}, the RRE formed after the "AND" learning phase completes, is learned after five passes in the "AND" learning process.

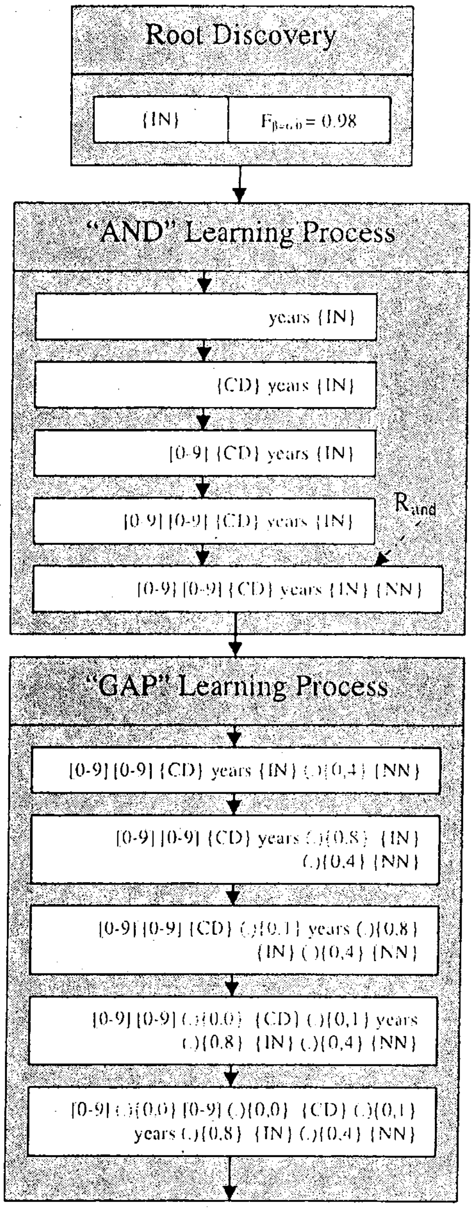

Root Discovery
{IN}
$F_{\beta=0.5} = 0.98$
"AND" Learning Process
years {IN}
{CD} years {IN}
[0-9] {CD} years {IN}
[0-9] [0-9] {CD} years {IN}
R_{and}
[0-9] [0-9] {CD} years {IN} {NN}
"GAP" Learning Process
[0-9] [0-9] {CD} years {IN} (.){0,4} {NN}
[0-9] [0-9] {CD} years (.){0,8} {IN} (.){0,4} {NN}
[0-9] [0-9] {CD} (.){0,1} years (.){0,8} {IN} (.){0,4} {NN}
[0-9] [0-9] (.){0,0} {CD} (.){0,1} years (.){0,8} {IN} (.){0,4} {NN}
[0-9] (.){0,0} [0-9] (.){0,0} {CD} (.){0,1} years (.){0,8} {IN} (.){0,4} {NN}

Figure 4-4: The first iteration of RRE discovery

After R_{and} is discovered, the "GAP" learning process takes place and tailors R_{and} to further improve precision. The process is depicted for our example in the "GAP" learning process in Figure 4.4. This process involves five steps, one for each of the five gaps between the six elements in R_{and}. Each intermediate RRE produced during "GAP" learning in this example has $F_{\beta=0.5} = 0.71$.

In this case, inclusion of either the start or the end symbol in the RRE decreases $F_{\beta=0.5}$, so the RRE discovered after "GAP" learning remains unchanged. This RRE is depicted at the bottom of Figure 4.4 and covers segments 1 and 2 in the true set. To prepare for the second iteration of the algorithm, segments 1 and 2 are thus removed from the true set. Therefore, the second iteration begins with segments 3, 4, 5, and 6 in the true set, and segments 1, 2, 3, 4, 5, and 6 in the false set. The steps to learn the second RRE are portrayed in Figure 4.5.

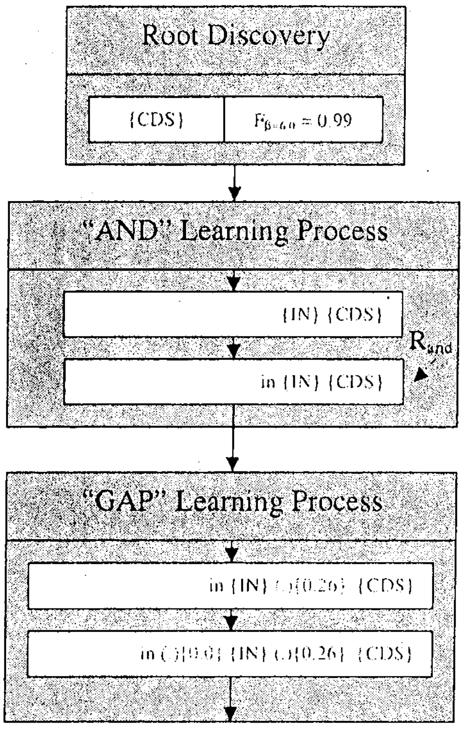


Figure 4-5: The second iteration of RRE discovery

As before, the addition of either the start or the end symbol decreases $F_{\beta=0.5}$. As a result, the final RRE for the second iteration is the RRE depicted at the bottom of Figure 4.5. The covering algorithm terminates after the second iteration because there are no true segments left in the true set – all have been covered by the two RREs discovered.

This example highlights a distinguishing characteristic of our algorithm – values of a given feature can be extracted precisely from a training set in which the features are imprecisely labeled. For instance, the RRE "in (.){0,0}{IN} (.){0,26} {CDS}" precisely matches only the Age feature "in{IN} his{MALE} twenties{CDS}" from the segment "in{IN} his{MALE} twenties{CDS} standing{VBG} six{CD} feet{NNS} tall{JJ}". Although this segment contains both Age and Height information and was labeled as such during training set development, our algorithm discovers an expression that precisely matches and extracts only Age features. Although we have not exemplified the discovery of an RRE for extracting Height, the same holds true: features for Height are also precisely matched and extracted despite the fact that the source segments are imprecisely labeled. To summarize, precise labeling of features is tedious and time-consuming, and this technique reduces training set development time without impacting performance. This is as noted what makes this a semi-supervised learning method.

4.3 *Scoring Functions*

The scoring function is used to evaluate the quality of the rule learned in each learning step. This function is critical for the RRE Discovery algorithm because the scoring function determines search heuristics, which impact the performance of the algorithm.

In this section, we introduce some well-known evaluation metrics, and the methods we used to select scoring functions for different applications.

4.3.1 Commonly Used Scoring Functions

There are several scoring functions that are commonly used in IE. The details of the metrics are dependent upon the definitions of true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN), as well as precision (P) and recall (R). These have been defined in detail in Section 2.5.

4.3.1.1 Precision

Precision is calculated as TP/(TP+FP). Generic covering algorithms such as PRISM use precision as the scoring function. Achieving high precision is critical for the inner loop of the RRE Discovery algorithm as well. We will discuss this measurement further in section 4.3.2.

4.3.1.2 Recall

Recall is calculated as TP/(TP+FN). Covering algorithms implicitly achieve high recall by combining rules learned in sequence. Naturally, recall is also significant for the RRE Discovery algorithm and will be discussed further in section 4.3.2 as well.

4.3.1.3 F_β

Since precision and recall are both important in assessing performance but are also inversely related (see Section 2.5), it is important to combine both values into a single measure of overall performance. One way to do this is F_β, which combines precision and recall using Equation 2.1 from [2]. The parameter β determines the relative importance of recall vs. precision. Using F_β, it is possible to compare the performance of systems with different precision and recall.

4.3.1.4 Accuracy

Accuracy is often used in machine learning algorithms. It is defined as the proportion of correct judgments over the entire dataset and is computer as shown in Equation 4.3.

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

Equation 4-3: Accuracy

4.3.1.5 Fallout

Fallout measures the proportion of returned irrelevant segments to all existing irrelevant segments. Since an IE system strives to minimize the number of irrelevant entities extracted, Fallout can be used as scoring function. Fallout is calculated using Equation 4.4.

$$Fallout = \frac{FP}{TN + FP}$$

Equation 4-4: Fallout

4.3.1.6 Overall

Overall is a scoring function designed to estimate the effort required for the user to change the selected result set to the target set. Accomplishing this change requires two operations: deletion and insertion. It is an assumption that it takes the same effort to delete an instance from the selected set as to insert an instance into the selected set. As can be deduced from the Venn diagram in Figure 2.6, it will take FP deletions and FN insertions to change the selected set to the target set. If the user manually labeled instances, it would take them |TP+FN| effort to construct the ground truth (assuming it was a best case scenario and no mistakes were made).

Therefore, the proportion of the adjustment effort required (after applying the algorithm) to the effort required by a full manual approach is represented by $\frac{FP + FN}{TP + FN}$. If we subtract this proportion from 1, it gives an estimate of the effort required to change the selected result set to the target set. In [3], overall is defined as represented in Equation 4.5.

$$Overall = 1 - \frac{FP + FN}{TP + FN} = recall * (2 - \frac{1}{precision})$$

Equation 4-5: Overall

If precision is less than 0.5, the value of overall will be negative. The highest value for overall (1.0) occurs when both precision and recall equal 1.0, as well. In all other cases, overall is less than both precision and recall.

In sum, Equation 4.5 satisfies the following properties:

- Precision=0.5 implies overall = 0
- Precision<0.5 implies overall < 0
- Precision=recall=1.0 implies overall reaches its highest value, 1.0

- Overall is smaller than both precision and recall if precision≠0 and recall≠1.0

4.3.1.7 Information_Gain

Information_Gain is another scoring function that calculates the amount of information gained in a given iteration of a covering algorithm. It "represents the total information gained regarding the current [true] positive examples, which is given by the number [of true positive examples] that satisfy the new test, multiplied by the information gained regarding each one of them" [4]. Information_Gain is calculated using Equation 4.6.

$$IG_{i+1} = TP_{i+1}(\log\frac{TP_{i+1}}{TP_{i+1}+FP_{i+1}} - \log\frac{TP_i}{TP_i+FP_i})$$

where i≥0 and by definition, $\frac{TP_0}{TP_0+FP_0}=1$

Equation 4-6: Information_Gain

4.3.2 Search Biases

The purpose of the semi-supervised RRE Discovery algorithm is to discover RREs that have both high precision and high recall. As a greedy covering algorithm with a top-down (general-to-specific) search strategy, the algorithm ideally obtains high precision for rules learned and improves recall when it combines the learned rules together.

First, each individual RRE discovered must have high precision because the overall precision of an expression that combines RREs is always greater than or equal to the lowest precision of its constituent RREs. In other words, the overall precision will be high if all the constituent RREs have high precision. Thus, by biasing the RRE discovery algorithm to obtain high precision for each RRE discovered, the algorithm ensures that the overall precision is high.

It is also necessary to improve recall as the RRE Discovery algorithm processes the data. Since our semi-supervised learning algorithm is a covering algorithm, the RREs discovered will be used in sequence to extract features from previously unseen data. If any single constituent RRE matches a feature value, overall recall is improved without sacrificing precision. This is due to the fact that, for a given feature, every RRE covers at least δ true positives, where $\delta>0$. As a result, when the RREs are used in sequence, true positives increase and false negatives decrease, coinciding with the goal to improve recall.

In general, the precision and recall biases in our algorithm enable the algorithm to discover expressions with both high precision and high recall as measured by the scoring metric employed. From a theoretical perspective, it is possible to prove for certain scoring functions that precision is guaranteed to increase monotonically during learning. Likewise, certain properties of recall can be demonstrated theoretically. It is thus useful to compare and contrast different metrics from a theoretical perspective. In what follows, we perform a theoretical analysis, and identify scoring metrics that guarantee a monotonic increase in precision during learning.

4.3.2.1 The Property of the Greedy Search Algorithm

The RRE discovery algorithm is clearly a *greedy search algorithm*. At the start of the learning process, the initial score is initialized to the minimum possible. For root discovery to occur, the algorithm must obtain a positive value for the scoring metric. Then, in each step of "AND" learning, the score of the extended RRE must be greater than or equal to the score of the previous RRE. This is demonstrated by Equation 4.7 (the scoring metric is abbreviated as F).

$$0 < F_i \leq F_{i+1}$$

Equation 4-7: Extending the RRE score

4.3.2.2 Theoretical Analysis of Recall

Regardless of the scoring function chosen, the recall monotonically decreases during learning. This is stated in Theorem 4.1.

Theorem 4-1: *The recall of an RRE monotonically decreases during learning.*

Proof:

- Suppose RRE_i is the RRE after the ith extension, and RRE_{i+1} is the RRE after the (i+1)th extension for a given attribute in the "AND" learning process, where i>0.
- We first show that all segments covered by RRE_{i+1} must also be covered by RRE_i.

In the "AND" learning process, $RRE_{i+1} = E_{i+1}RRE_i$ or $RRE_{i+1} = RRE_iE_{i+1}$ where E_{i+1} is the element (a simple RRE) discovered in the (i+1)th extension. Therefore, any string accepted by RRE_{i+1} will either take the form $S_{i+1}S'_{i+1}$ or $S'_{i+1}S_{i+1}$ where E_{i+1} accepts S_{i+1}, RRE_i accepts S'_{i+1}, and S_{i+1} and S'_{i+1} are substrings of the strings accepted by RRE_{i+1}. This implies that any segment containing a substring accepted by RRE_{i+1} must also include substrings accepted by RRE_i. Since the discovery of the start and end of an RRE is a special form of the "AND" learning process, the same conclusion holds.

In the "Gap Learning Process", the gap learned is the smallest one. Thus, strings accepted by the RRE with the smallest gap discovered must also be accepted by an RRE containing the same elements but having larger gaps between elements.

Based on the above, we conclude that all segments covered (i.e., containing strings accepted) by RRE_{i+1} must also be covered by RRE_i.

- Obviously, all true-set segments covered by RRE_{i+1} must thus be covered by RRE_i. This implies that $TP_i \geq TP_{i+1}$. Furthermore, we have $TP_i + FN_i = TP_{i+1} + FN_{i+1}$.
- Thus $$\frac{TP_i}{(TP_i + FN_i)} \geq \frac{TP_{i+1}}{(TP_{i+1} + FN_{i+1})}$$

Or $R_i \geq R_{i+1}$ **(Equation 4-8)**

4.3.2.3 Theoretical Analysis of Precision

As noted in the introduction to this section, the various scoring metrics have significantly different theoretical properties with respect to precision. In what follows we prove several theorems that treat the various scoring metrics overviewed above. In each case we refer to the scoring function as F in both the theorems and their proofs.

Theorem 4-2: *If the scoring function F is recall, the precision of an RRE monotonically increases during learning.*

Proof:

From Equations 4.7 and 4.8, we have

$F_i \leq F_{i+1}$ and $R_i \geq R_{i+1}$

$F_i \leq F_{i+1} \rightarrow R_i \leq R_{i+1}$ since $F_i = R_i$ and $F_{i+1} = R_{i+1}$

$\rightarrow R_i = R_{i+1}$

$$\rightarrow \frac{TP_i}{TP_i + FN_i} = \frac{TP_{i+1}}{TP_{i+1} + FN_{i+1}}$$

$TP_i + FN_i = TP_{i+1} + FN_{i+1}$

$\rightarrow TP_i = TP_{i+1}$

As we have proved in Theorem 4.1, any segment containing a substring accepted by RRE_{i+1} must also include substrings accepted by RRE_i. In other words, $TP_i + FP_i \geq TP_{i+1} + FP_{i+1}$. Therefore, we have $\frac{TP_i}{TP_i + FP_i} \leq \frac{TP_{i+1}}{TP_{i+1} + FP_{i+1}}$

$\rightarrow P_i \leq P_{i+1}$

Theorem 4-3: ***If the scoring function F is F_β, the precision of an RRE monotonically increases during learning.***

Proof:

Suppose P_i and P_{i+1} are the precisions of RRE_i and RRE_{i+1} in the i th and (i+1)th extensions respectively of the "AND" learning process. R_i and R_{i+1} are the recalls of RRE_i and RRE_{i+1} in the i th and (i+1)th extensions respectively of the "AND" learning process. F_i and F_{i+1} are the scores of RRE_i and RRE_{i+1} in the i th and (i+1)th extensions respectively of the "AND" learning process. TP_i and TP_{i+1} are the true positives of RRE_i and RRE_{i+1} in the i th and (i+1)th extensions respectively of the "AND" learning process.

1) Suppose $P_i > 0$ and $P_{i+1} > 0$

- We have $R_i > 0$, $R_{i+1} > 0$, $F_i > 0$ and $F_{i+1} > 0$
- $F_i = \dfrac{\beta^2 P_i R_i}{(\beta^2+1)P_i + R_i} \quad \rightarrow \quad P_i = \dfrac{F_i R_i}{(\beta^2+1)R_i - \beta^2 F_i}$
- $F_{i+1} = \dfrac{\beta^2 P_{i+1} R_{i+1}}{(\beta^2+1)P_{i+1} + R_{i+1}} \quad \rightarrow \quad P_{i+1} = \dfrac{F_{i+1} R_{i+1}}{(\beta^2+1)R_{i+1} - \beta^2 F_{i+1}}$
- We wish to prove $P_i \le P_{i+1}$

  Assuming $P_i > P_{i+1}$ we have the following:

$$\frac{F_i R_i}{(\beta^2+1)R_i - \beta^2 F_i} > \frac{F_{i+1} R_{i+1}}{(\beta^2+1)R_{i+1} - \beta^2 F_{i+1}}$$

$$\rightarrow F_i R_i((\beta^2+1)R_{i+1} - \beta^2 F_{i+1}) > F_{i+1}R_{i+1}((\beta^2+1)R_i - \beta^2 F_i)$$

$$\rightarrow F_i R_i(\beta^2+1)R_{i+1} - F_i R_i \beta^2 F_{i+1} > F_{i+1} R_{i+1}(\beta^2+1)R_i - F_{i+1} R_{i+1} \beta^2 F_i$$

$$\rightarrow (\beta^2+1)R_i R_{i+1}(F_i - F_{i+1}) > \beta^2 F_i F_{i+1}(R_i - R_{i+1})$$

From Equations 5-3 (on page +++) and 5-4 (on page +++), we have

$$F_i \le F_{i+1} \rightarrow 0 \ge (\beta^2+1)R_i R_{i+1}(F_i - F_{i+1})$$

$$R_i \ge R_{i+1} \rightarrow \beta^2 F_i F_{i+1}(R_i - R_{i+1}) \ge 0$$

$$\rightarrow 0 \ge (\beta^2+1)R_i R_{i+1}(F_i - F_{i+1}) > \beta^2 F_i F_{i+1}(R_i - R_{i+1}) \ge 0$$

$$\rightarrow 0 > 0$$

Since $0=0$, the assumption is incorrect.

- Thus $P_i \le P_{i+1}$.

2) Suppose $P_i = 0$ and $P_{i+1} > 0$

- It is obvious that $P_i \le P_{i+1}$.

3) Suppose $P_i > 0$ and $P_{i+1} = 0$

- $P_i > 0 \rightarrow TP_i > 0 \rightarrow R_i > 0 \rightarrow F_i > 0$
- $P_{i+1} = 0 \rightarrow F_{i+1} = 0$
- $\rightarrow F_i > F_{i+1}$
- Due to equation 1.1, this assumption can not exist.

4) Suppose $P_i = 0$ and $P_{i+1} = 0$

- It is obvious that $P_i \le P_{i+1}$.

5) Based on 1), 2), 3), and 4), we have $P_i \le P_{i+1}$.

Theorem 4-4: ***If the scoring function F is precision, the precision of an RRE monotonically increases during learning.***

Proof:

Suppose P_i and P_{i+1} are the precisions of RRE_i and RRE_{i+1} in the ith and (i+1)th extensions respectively of the "AND" learning process. F_i and F_{i+1} are the scores of RRE_i and RRE_{i+1} in the ith and (i+1)th extensions respectively of the "AND" learning process.

- $F_i = P_i$, and $F_{i+1} = P_{i+1}$
- From Equations 1.1 (on page +++) we have $F_i \le F_{i+1}$
- Thus $P_i \le P_{i+1}$.

Theorem 4-5: *If the scoring function F is overall, the precision of an RRE monotonically increases during learning.*

Proof:

Suppose P_i and P_{i+1} are the precisions of RRE_i and RRE_{i+1} in the ith and (i+1)th extensions respectively of the "AND" learning process. R_i and R_{i+1} are the recalls of RRE_i and RRE_{i+1} in the ith and (i+1)th extensions respectively of the "AND" learning process. F_i and F_{i+1} are the scores of RRE_i and RRE_{i+1} in the ith and (i+1)th extensions respectively of the "AND" learning process. TP_i, FP_i and FN_i are the true positives, false positives, and false negatives of RRE_i, respectively; TP_{i+1}, FP_{i+1} and FN_{i+1} are the true positives, false positives, and false negatives of RRE_{i+1}, respectively in the ith and (i+1)th extensions respectively of the "AND" learning process.

1) Suppose $P_i > 0$ and $P_{i+1} > 0$

- ❖ We have $R_i > 0$, $R_{i+1} > 0$
- ❖ $F_i = R_i(2 - \frac{1}{P_i}) \qquad \rightarrow \qquad P_i = \frac{R_i}{2R_i - F_i}$
- ❖ $F_{i+1} = R_{i+1}(2 - \frac{1}{P_{i+1}}) \qquad \rightarrow \qquad P_{i+1} = \frac{R_{i+1}}{2R_{i+1} - F_{i+1}}$
- ❖ We wish to prove $P_i \le P_{i+1}$

  Assuming $P_i > P_{i+1}$ we have the following:

$$P_i = \frac{R_i}{2R_i - F_i} > P_{i+1} = \frac{R_{i+1}}{2R_{i+1} - F_{i+1}}$$

$$\rightarrow 2R_iR_{i+1} - R_iF_{i+1} > 2R_{i+1}R_i - R_{i+1}F_i$$

$$\rightarrow R_{i+1}F_i > R_iF_{i+1}$$

$$\rightarrow F_i > \frac{R_i}{R_{i+1}}F_{i+1} \rightarrow 0 > \frac{R_i}{R_{i+1}}F_{i+1} - F_i$$

From Equations 5-3 (on page +++) and 5-4 (on page +++), we have

$$0 \le F_i \le F_{i+1} \rightarrow F_{i+1} - F_i \ge 0$$

$$R_i \ge R_{i+1} \rightarrow \frac{R_i}{R_{i+1}} \ge 1 \rightarrow \frac{R_i}{R_{i+1}}F_{i+1} - F_i \ge F_{i+1} - F_i$$

$$\rightarrow 0 > \frac{R_i}{R_{i+1}}F_{i+1} - F_i \ge F_{i+1} - F_i \ge 0$$

$\rightarrow$ 0>0

Since 0=0, the assumption is incorrect.

- ❖ Thus $P_i \le P_{i+1}$.

2) Suppose $P_i = 0$

- ❖ according to the definition of Overall, $F_i = 1 - \frac{FP_i + FN_i}{TP_i + FN_i} = \frac{TP_i - FP_i}{TP_i + FN_i} = -\frac{FP_i}{FN_i} \le 0$
- ❖ Due to equation 1.1, this assumption can not exist.

3) Suppose $P_{i+1} = 0$

- ❖ This assumption cannot exist. The proof is similar to 2).

4) Based on 1), 2), and 3), we have $P_i \le P_{i+1}$.

If fallout, accuracy, or Information_Gain is used as the scoring function of the RRE Discovery algorithm, the precision does not monotonically increase during learning. This conclusion can be demonstrated using the following example that employs the Height feature with Table 4.4 representing the training set.

Table 4-4: The training set for RRE Discovery

True Instances (Height)	In//IN his//PRP 20s//CD
	20//CD years//NNS old//JJ
	25//CD years//NNS of//DT age//NN
	In//IN his//PRP twenties//NNS
False Instances (Height)	Packer//NNP Avenue//NNP
	In//IN the//DT 20th//CD century//NN
	5//CD feet//NNS tall//JJ
	25//CD dollars//NNS
	2//CD days//NNS
	In//IN 5//CD hours//NNS

Suppose the root discovered is "//CD". From Table 4.5, we have the number of TP = 3, FP = 5, TN = 1, and FN = 1, while precision is $P_1=3/(3+5)=0.375$.

Table 4-5: The training set partition after "Root Discovery"

True Positive Instances	In//IN his//PRP 20s//CD
	20//CD years//NNS old//JJ
	25//CD years//NNS of//DT age//NN
False Negative Instance	In//IN his//PRP twenties//NNS
True Negative Instance	Packer//NNP Avenue//NNP
False Positive Instances	In//IN the//DT 20th//CD century//NN
	5//CD feet//NNS tall//JJ
	25//CD dollars//NNS
	2//CD days//NNS
	In//IN 5//CD hours//NNS

After the first iteration of the "AND Learning Process", the RRE becomes "in• {0.100}//CD". Table 4.6 depicts that TP = 1, FP = 2, TN = 4, and FN = 3, while precision is $P_2=1/(1+2)=0.33$. It is obvious that the precision has decreased, as $0.375=P_1 > P_2=0.33$.

Table 4-6: The training set partition after the first iteration of "AND Learning Process"

True Positive Instances	In//IN his//PRP 20s//CD
False Negative Instance	In//IN his//PRP twenties//NNS
	20//CD years//NNS old//JJ
	25//CD years//NNS of//DT age//NN
True Negative Instance	Packer//NNP Avenue//NNP
	5//CD feet//NNS tall//JJ
	25//CD dollars//NNS
	2//CD days//NNS
False Positive Instances	In//IN the//DT 20th//CD century//NN
	In//IN 5//CD hours//NNS

Based on this example, the values of fallout, accuracy and Information_Gain are:

Fallout:

The "Fallout" after root discovery is $F_1=1-5/(5+1)=0.167$,

The "Fallout" after the first "AND Learning Process" is $F_2=1-2/(2+4)=0.667$.

We have $F_1< F_2$ while $P_1 > P_2$

Accuracy:

The accuracy after root discovery is $A_1=(3+1)/(3+5+1+1)=0.4$,

The accuracy after the first "AND Learning Process" is $A_2=(1+4)/(1+2+4+3)=0.5$.

We have $A_1< A_2$ while $P_1 > P_2$

Information_Gain:

Suppose the initial precision in Information_Gain is 1, then, after root discovery we have

$$I_1 = TP_1(\log\frac{TP_1}{TP_1+FP_1} - \log 1) = 3\log\frac{3}{8} = -4.2451125$$

Then, the Information_Gain after the first "AND Learning Process" is

$$I_2 = TP_2(\log\frac{TP_2}{TP_2+FP_2} - \log\frac{TP_1}{TP_1+FP_1}) = 1*(\log\frac{1}{1+2} - \log\frac{3}{3+5}) = -0.169925001$$

We have $I_1 < I_2$ while $P_1 > P_2$.

In this section, we have demonstrated theoretically that four metrics (F_β, Precision, Recall, and Overall) are suitable scoring functions for our greedy top-down covering algorithm (RRE Discovery). These four metrics guarantee that the precision of an RRE monotonically increases during the learning process. In other words, the overall precision will be relatively high. Metrics such as accuracy, fallout and Information_Gain are not suitable for use with our greedy algorithm since they cannot guarantee high precision in learning an RRE rule.

4.4 *Time Complexity Analysis*

In this section, we discuss the time complexity of the RRE discovery algorithm in detail. There are two aspects of the time complexity analysis. The first is the measurement of the complexity of generating reduced regular expressions. The second is the analysis of the time complexity of matching the reduced regular expressions to the input text. Actually, in the implementation we represent RREs using a RE matching library. Therefore, we first discuss how to match a regular expression to the input text.

4.4.1 The Time Complexity of Regular Expression Matching

The time complexity for regular expression matching is O ($|x||z|$), where $|x|$ is the length of a regular expression, and $|z|$ is the length of the input text measured in symbols of the alphabet.

The length of a regular expression 'x' is defined as $|x| = S_{\Sigma} + S_\varepsilon + S_| + S^* + S_{()}$ [5]

- S_Σ: the number of occurrences of alphabet symbols in the regular expression 'x'
- S_ε: the number of occurrences of ε in the regular expression 'x'
- $S_|$: the number of occurrences of alteration operations in the regular expression 'x'
- S^*: the number of occurrences of Kleene Closure operations in the regular expression 'x'
- $S_{()}$:the number of occurrences of parentheses in the regular expression 'x'

Definition 4-1: Length of a Regular Expression

The time complexity of regular expression matching includes two components. One is to convert a regular expression to a non-deterministic finite state automata with ε-transitions. The other component is the simulation of the automata. The simulation consists of a main matching procedure, a 'Closure' procedure, and a 'Transitions' procedure. The time complexities of these four parts are detailed in the following sections.

4.4.1.1 Convert a RE to ε-NFA (Thompson's approach)

This section is based on the theoretical work presented in [5] and [6].

There is exactly one initial state q_0 and one final state (double circled) in M, which is the Thompson's ε-NFA of a regular expression 'x'. Given the regular expression 'x'="a|ε", where 'a' is a symbol and 'ε' is the empty string, there are exactly two states in each of two NFAs: State(M_a) = 2, Edge(M_a) = 1, State(M_ε) = 2 and Edge(M_ε) = 1, where M_a is the Thompson's ε-NFA of the regular expression "a", and M_ε is the Thompson's ε-NFA for an empty string. (State(M) is a function that counts the number of states in M. Edge(M) is a function that counts the number of edges in M.)

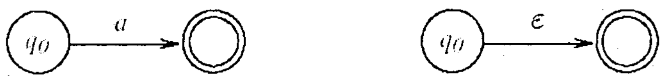


Figure 4-6: ε-NFA states

If a regular expression is "r=j|k", where r, j, and k are all regular expressions, then State(M_r) = State(M_j) + State(M_k) + 2 and Edge(M_r) = Edge(M_j) + Edge(M_k) + 4. That means that exactly two new states are added into the NFA for each alteration operation.

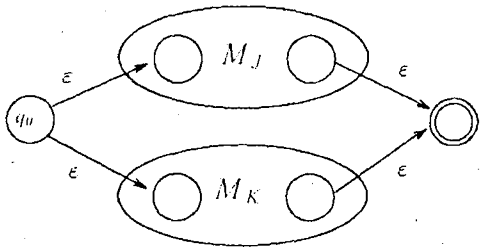


Figure 4-7: ε-NFA alteration operation

If a regular expression is "r=jk", then State(M_r) = State(M_j) + State(M_k) - 1 and Edge(M_r) = Edge(M_j) + Edge(M_k). That means there is exactly one state deleted from the NFA for each conjunction operation.

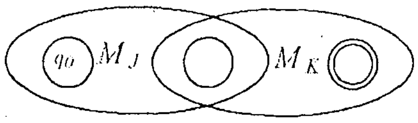


Figure 4-8: ε-NFA conjunction operation

If a regular expression is "r=j*", then State(M_r) = State(M_j) + 2 and Edge(M_r) = Edge(M_j) + 4. That means there are exactly two states added to the NFA for each Kleene Closure operation.

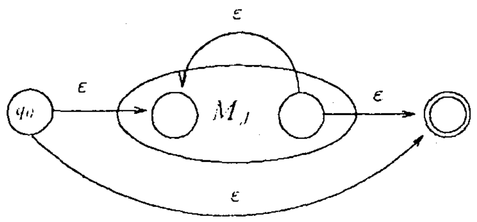


Figure 4-9: ε-NFA Kleene closure operation

These statements result in Theorem 4.6.

Theorem 4-6: "Let 'x' be a regular expression. There exists a ε-NFA recognizing lang(x) satisfying the following conditions from [6]:

1. The number of states is bounded by 2|x|
2. The number of edges labeled by symbols is bounded by |x|, and the number of edges labeled by ε is bounded by 4|x|
3. For each state the number of ingoing or outgoing edges is at most two, and is exactly two only when the edges are labeled by ε."

Overall, the time complexity to build the ε-NFA from a regular expression 'x' is O(|x|). The proof of Theorem 4.6 is found in [6].

4.4.1.2 The Algorithm for Computing Whether a Regular Expression 'x' Matches a Sub-string of the Input Text 'z'

An observation of regular expression matching is that "[s]earching a string 'z' for strings in language X is equivalent to searching for prefixes of 'z' that belong to the language $\sum^*X$, where $\sum^*$ is the power set of $\sum$. For example, suppose 'z' is 'uxv', $u, v \in \Sigma, x \in X$; to match 'x' to a sub-string of 'z' is equivalent to matching 'ux' to a prefix of 'z'" [6].

To test whether there is a prefix of 'z' that belongs to $\sum^*X$, where X is the language generated by the regular expression 'x', the following steps suffice:

```
1. Build an ε-NFA from the regular expression 'x', where the ε-NFA is defined as (Q, i, {t},
   E), where Q is the set of states, i is the initial state, t is the final state, E is the set of edges.
2. C ← Closure( E, {i} )
3. if t is in C, then 'x' matches a prefix of 'z'
4. for each symbol 'a' from the first to the last symbol of 'z'
5.     C ← Closure ( E, Transitions ( E, C, a ) + {i} )
6.     if t is in C, then x matches a prefix of 'z'
```

The time complexity of the test is O(Closure) + |z| * (O(Transition)+O(Closure)). The two functions Transition and Closure are presented below:

```
Transition (E, S, a)
  1. R←Ø
  2. for each p in S
  3.     for each q such that ( p, a, q ) is in E
  4.         R←R+{q}
  5. return R
```

Here S is the set of states that are reached by the previous Transitions and Closures, 'a' is the current symbol, and E is the transition function or the edges in the ε-NFA. The time complexity of the Transition function is based on how many states 'q' have been added into the set 'R'. This is $O(|4x|) = O(|x|)$ for the upper bound because $|S| \leq 2|x|$, and there are at most two edges labeled 'a' from any state 'p' (from Theorem 4.6).

```
Closure (E, S)
  1.  R ← S
  2.  Q ← EmptyQueue
  3.  for each state p in S
  4.     Enqueue ( Q, p )
  5.  while not QueueEmpty ( Q )
  6.     p ← Dequeue ( Q )
  7.     for each state q such that ( p, ε, q) is in E
  8.        if q is not in R
  9.           R ← R + { q }
  10.          Enqueue ( Q, q )
  11. return R
```

The purpose of the Closure function is to identity all states that can be reached from the current set of states by ε-transitions. The time complexity of Closure is also $O(|x|)$.

In general, the overall time complexity to match a regular expression 'x' to a sub-string of the input text 'z' is $O(|x||z|)$.

4.4.2 Root Discovery

In this section, we assume that one hash function evaluation is equivalent to one character comparison when computing time complexity.

There are two steps to find a "root" for a reduced regular expression (RRE). First of all, the algorithm needs to identify all unique words and part-of-speech tags in the true set of the training data. The time complexity of this step is defined as $C_{UniqueWordTag}$. We use a hash table to store each unique word or part-of-speech tag from the true set. Suppose the number of words and part-of-speech tags from the true set is 'k'. In the best case, the hash function assigns each unique word or part-of-speech tag a different hash code. Therefore, the time complexity is O(k), which is one calculation of the hash function for each word or part-of-speech tag in the true set. In the worst case, the basis of the time complexity is the calculations of the hash function and the character comparisons (when there is a collision in the hash table). If the hash table uses *chained hashing*, we have a single linked list in the worst case, in which all words or part-of-speech tags have the same hash code, and are linked together in one list. Therefore, there are k-1 collisions. Table 4.7 depicts the worst case scenario.

Table 4-7: Worst case scenario to identify unique words or part-of-speech tags

Unique words or part-of-speech tags	The number of hash function calculations	The number of string comparisons
1st	1	0
2nd	1	1
...	...	...
kth	1	k-1

Suppose $y_{longest}$ is the longest word or part-of-speech tag in the true set. In the worst case, every word or part-of-speech tag has length $y_{longest}$. Therefore, the comparison of any two of them will take at most $|y_{longest}|^2$ time[4].

The time complexity to put 'k' words or part-of-speech tags into a hash table is

$$O(k) \le C_{UniqueWordTag} \le O(k + \sum_{i=0}^{k-1} i\,|y_{longest}|^2)$$

$$\Rightarrow O(k) \le C_{UniqueWordTag} \le O(k + \frac{(k-1)k}{2}|y_{longest}|^2)$$

$$\Rightarrow O(k) \le C_{UniqueWordTag} \le O(k^2 |y_{longest}|^2).$$

[4] The time complexity of matching two strings (Str1 and Str2) is O(|Str1|) in the best case, where |Str1| ≤ |Str2|. The worst case time complexity is O(|Str1||Str2|). This comes from the Brute Force algorithm [8].

The second step is to test every unique word or part-of-speech tag in the true set against the entire training dataset, including both true and false sets. This step involves simple string matching. For a word or part-of-speech tag 'y', the string matching time complexity is $O(|y|) \leq C_{StringMatching} \leq O(|y||w|)$, where $|w| = \sum_{j=1}^{n} |z_j|$, z_j is the j^{th} segment, and 'n' is the number of total segments in the training set including both the true and the false sets. In the best case every word or part-of-speech tag in the true set matches the first word or the first part-of-speech tag of every segment in both the true set and the false set. Assuming 'k' unique words or part-of-speech tags, the best case root matching time complexity is $O(nk|y_{shortest}|)$, where $y_{shortest}$ is the shortest word or part-of-speech tag in the true set and n is the total number of segments in the training set including both the true set and the false set.

In the worst case, the algorithm needs to match each word or part-of-speech tag against each entire segment in both the true and the false sets. Therefore, the time complexity is $O(\sum_{i=1}^{k} \sum_{j=1}^{n} |y_i||z_j|)$, where y_i is the i^{th} unique word or part-of-speech tag in the true set. Given that $y_{shortest}$ is the shortest word or part-of-speech tag in the true set, and $y_{longest}$ is the longest word/tag in the true set, then $O(nk|y_{shortest}|) \leq C_{StringMatching} \leq O(|y_{longest}|k|w|)$. The final time complexity of Root Discovery for a single root is thus $O(k+nk|y_{shortest}|) \leq C_{rootOverall} \leq O(k^2|y_{longest}|^2+|y_{longest}|k|w|)$.

4.4.3 "AND" Learning Process

There are two steps to measure the "AND Learning Process" time complexity. We first measure the time complexity to extend an RRE. The second step is to measure the time complexity of the scoring function. In our algorithm, the scoring function is F_β (see section 2.5). As a result, the time complexity is proportional to the reduced regular expression matching time complexity on the entire training set. This is due to the fact that F_β computes both precision and recall based on the result of matching.

$$0 \le C_{ANDelement} \le \sum_{i=1}^{N-1} \sum_{j=1}^{2} | S_{ij} |$$

Equation 4-9: AND Learning Process time complexity

The time complexity of the first step is depicted in Equation 4.9, where $|S_{i1}|$ is the number of unique candidate words and part-of-speech tags occurring *before* the current RRE, and $|S_{i2}|$ is the number of unique candidate words and part-of-speech tags occurring *after* the current RRE. In the best case, the root cannot be extended at all. Therefore, the "AND Learning Process" best time complexity is 0. Figure 4.10 depicts an example of the "AND Learning Process" for the *Age* attribute, where six elements are in the RRE. All together there are 10 positions to be tested in Figure 4.10, with many possible words or part-of-speech tags potentially in each position. Given that N is an input parameter of the algorithm, we can see that in the worst case, 2*(N-1) positions must be tested.

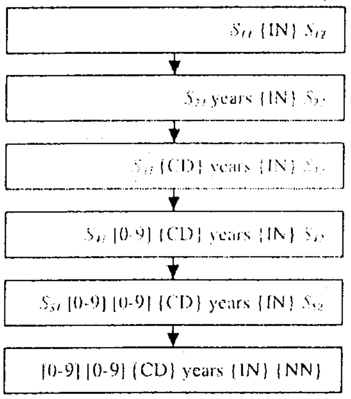

S_{11} {IN} S_{12}
S_{21} years {IN} S_{22}
S_{31} {CD} years {IN} S_{32}
S_{41} [0-9] {CD} years {IN} S_{42}
S_{51} [0-9] [0-9] {CD} years {IN} S_{52}
[0-9] [0-9] {CD} years {IN} {NN}

Figure 4-10: "AND" learning example for the *Age* attribute

To measure the time complexity of matching the extended RRE to the entire training set to compute F_β, we must first obtain the length of the current RRE. As described in section 4.4.1, the length of a regular expression 'x' is $|x| = S_\Sigma + S_\varepsilon + S_| + S^* + S_{()}$. After the i-th extension iteration of the current RRE, there are i+1 elements and 'i' gaps in the RRE. All elements contain only characters from $\sum$. Given that $y_{shortest}$ is the shortest element and $y_{longest}$ is the longest element in the true set, the length of all elements in the current RRE is between $(1+i)|y_{shortest}|$ and $(1+i)|y_{longest}|$. A gap between any two adjacent elements is expressed as "$(\alpha|\varepsilon)\{\psi\}$", which is "$(a|\varepsilon)\ (a|\varepsilon)\ \ldots\ (a|\varepsilon)$", where 'a' is any character in $\sum$, and ψ is the maximum gap (measured in characters) allowed between any two elements (ψ is an input parameter of the algorithm). This means we have up to ψ "OR" operations, ψ ε transitions, ψ parentheses, and ψ characters in a single gap, since an RRE does not include Kleene closure (i.e., '*'), $S^*=0$. Since there are 'i' gaps in the current RRE, we have $S_\varepsilon=\psi i$, $S_|=\psi i$, $S_{()}=\psi i$, and ψi characters in addition to the element characters in the elements that we considered earlier. As a result, the length of the current RRE after the i-th extension iteration is in the range

$$(1+i)\,|\,y_{\text{shortest}}\,| + 4\psi i \le RRE_{length} \le (1+i)\,|\,y_{longest}\,| + 4\psi i\,.$$

Per the overall time complexity to match a regular expression 'x' to input 'w', the time complexity to match the extended RRE to the entire training set 'w' is

$$((1+i)\,|\,y_{\text{shortest}}\,| + 4\psi i)\,|\,w\,| \le RRE_{matching} \le ((1+i)\,|\,y_{longest}\,| + 4\psi i)\,|\,w\,|$$

Equation 4-10: RRE Matching time complexity

The overall time complexity of the "AND Learning Process" is $0 \le C_{AND} \le \sum_{i=1}^{N-1}\sum_{j=1}^{2}|S_{ij}|((1+i)|y_{longest}|+4\psi i)|w|$, which is a combination of equations 4.9 and 4.10. Since $0 \le |S_{ij}| \le m$, where m is the number of words and part-of-speech tags in the true set,

$$0 \le C_{AND} \le 2m|w|\sum_{i=1}^{N-1}((1+i)|y_{longest}|+4\psi i).$$

4.4.4 Gap Learning Process

The measurement of the "Gap Learning Process" time complexity includes two steps. The first step is to measure the time complexity of the extension of an RRE. The second step is to measure the time complexity of RRE matching. Just like the "AND Learning Process", the matching of an RRE against the entire training set is used to calculate F_β, the scoring function used in the RRE discovery algorithm.

The time complexity for "Gap Learning Process is $C_{GAP} = \sum_{i=1}^{N-1} G_i$, where G_i is the time complexity to identify the optimal gap between RRE element i and element i+1. Similar to the "AND Learning Process," we must obtain the length of the current RRE in order to measure G_i. In the worst case, there are N elements in the current RRE during the "Gap Learning Process" (where N is an input to the algorithm). Given that $y_{longest}$ is the longest element in the true set, there are at most $N|y_{longest}|$ characters in these N elements. In the worst case, we suppose that all gaps (the number of these gaps is N-2) other than the current gap are expressed as $(\alpha|\varepsilon)\{\psi\}$, and the current gap is expressed in $(\alpha|\varepsilon)\{j\}$, where $0 \le j \le \psi$. According to the equation of the length of a regular expression "$|x| = S_\Sigma + S_\varepsilon + S_| + S^* + S_{()}$", the sum of the lengths of the RREs for the current gap learning is $\sum_{j=0}^{\psi}(N|y_{longest}|+4\psi(N-2)+4j)$. Therefore, we have

$G_i = |w| \sum_{j=0}^{\psi} (N | y_{longest} | + 4\psi(N-2) + 4j)$. In the best case, the time complexity is 0 because there is no gap in the current RRE. Finally, we have

$$0 \le C_{GAP} \le |w| \sum_{i=1}^{N-1} \sum_{j=0}^{\psi} (N | y_{longest} | + 4\psi(N-2) + 4j) \quad \Rightarrow$$

$$0 \le C_{GAP} \le |w| (N-1) \sum_{j=0}^{\psi} (N | y_{longest} | + 4\psi(N-2) + 4j) \quad \Rightarrow$$

$$0 \le C_{GAP} \le |w| (N-1)(\psi+1)(N | y_{longest} | + 4\psi(N-2) + 2\psi).$$

4.5 *Active Learning Approach*

Our active learning algorithm is seeded with an often-used description of a feature. For example, "six feet tall" may be a seed for the feature *Height*. Based on such choices for seeds, our semi-supervised learning algorithm discovers RREs representing the contexts surrounding the seeds. All segments with similar contexts become candidates for inclusion in the true set for a given feature.

The rule developer need only select those candidates that actually contain a description of a given feature. After this active learning phase, our semi-supervised learning algorithm is once again applied to discover RREs for each feature per the process described in section 4 of [7]. Figure 4.11 portrays a flowchart of the active learning process.

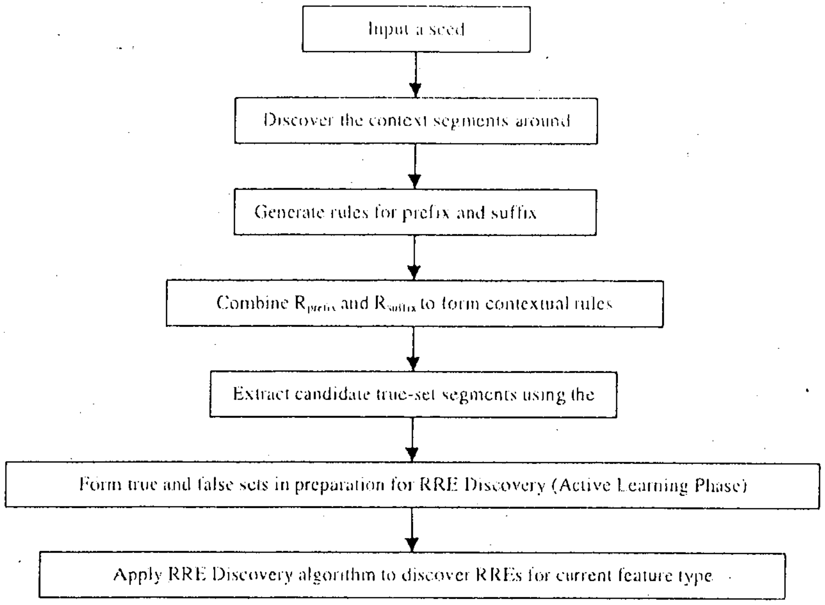

Input a seed
Discover the context segments around
Generate rules for prefix and suffix
Combine R_prefix and R_suffix to form contextual rules
Extract candidate true-set segments using the
Form true and false sets in preparation for RRE Discovery (Active Learning Phase)
Apply RRE Discovery algorithm to discover RREs for current feature type

Figure 4-11: Active learning flow chart

This approach is based on the observation that there is local context that can be leveraged to aid in the discovery process. For example, a person's *Age* may often follow *Race* and precede *Height* in a suspect's physical description. Therefore, if a segment occurs between *Race* and *Height*, it is likely an *Age* segment. In what follows we provide the details of our approach.

4.5.1 Context Pattern Discovery

There are several steps involved in the discovery of the context of the seed. First, all segments containing the seed in the training dataset are extracted. Let the set $S_{content}$ = {x | x is a segment containing the seed}. Second, let all segments immediately preceding each member of $S_{content}$ be defined as S_{prefix} = {y | y is a segment and the segment immediately following y contains the seed}. Third, let all segments immediately following each member of $S_{content}$ be defined as S_{suffix} = {z | z is a segment and the segment immediately preceding z contains the seed}.

After forming S_{prefix}, $S_{content}$, and S_{suffix}, we employ our semi-supervised learning algorithm to discover RREs for S_{prefix} and S_{suffix}. We then combine the prefix pattern with the suffix pattern to form a complete contextual pattern using the "AND" operator.

To discover the RREs for the prefix, we use S_{prefix} as the true set, and $S_{content} \cup S_{suffix}$ as the false set. The RREs discovered by our semi-supervised learning algorithm form the set $R_{prefix} = \{p \mid p \text{ is a prefix RRE}\}$.

To discover the suffix RREs, we use S_{suffix} as the true set, and $S_{content} \cup S_{prefix}$ as the false set. The RREs discovered by our semi-supervised learning algorithm form the set $R_{suffix} = \{s \mid s \text{ is a suffix RRE}\}$.

Combinations of members of R_{prefix} and R_{suffix} form contextual patterns that are used to discover candidates for the true set for the feature under consideration. Since not all possible combinations of such members occur in the training data, our algorithm employs a data-driven approach that ensures that only combinations that actually occur in the input text are used. As noted, the final expression for extracting candidate true-set segments consists of the logical "AND" of members of S_{prefix} and S_{suffix} with a one-segment gap between. We refer to this final expression as a *contextual expression* in what follows.

4.5.2 Active Selection of True Segments

In this step, our active learning algorithm identifies all candidates for a given feature using the contextual expression discovered as described above in section 4.5.1. Of course, not all such candidates are actually true positive segments, so at this point the training set developer selects the segments that form the final true set. We define the true set $T_a = \{t \mid t$ is a segment accepted by the contextual expression for feature type a in which a feature of type a actually occurs$\}$. The remaining segments that are not selected become the false set $F_a = \{f \mid f$ is a segment accepted by the contextual expression that does not contain a feature of type $a\}$.

For a given feature, the ratio of true to false segments in these two sets is generally greater than the ratio of true to false segments in the training data overall. In other words, substantially less effort is required to develop these two sets using our semi-supervised active learning approach than would be required if these sets were manually generated by labeling every single segment in the training data. In fact, as will be seen in section 5.3, the reduction in effort is as great as 99%. As will also be seen in section 5.3, the performance of the RREs discovered with these sets is competitive with the performance of RRE discovery using all of the (manually labeled) training data.

The final true set for a given feature a is $T_a \cup S_{content}$ and the final false set is $S_{prefix} \cup S_{suffix} \cup F_a$. As noted, once these true and false sets have been generated with our active learning algorithm for a given feature, our semi-supervised learning algorithm is again applied, this time to discover an expression that extracts features for the feature of interest.

5 Experimental Results

In order to test the capabilities of our RRE Discovery algorithm, we conducted a series of experiments to verify our theoretical conclusions. Utilizing both police incident report and patent data, we carried out these tests in an iterative manner, beginning with a manual approach and moving on to a semi-supervised approach. We then combined the semi-supervised approach with an active learning algorithm to evaluate our semi-supervised active RRE Discovery algorithm.

5.1 *Manual Approach*

We first began our analysis by utilizing a manual approach for regular expression formation. Our source data was drawn from Fairfax County, Virginia police incident reports.

Based on this training data, we manually generated regular expressions for the seven attributes listed in Table 5.1. An independent test dataset was then used to assess the performance of the regular expressions. The performance of the regular expressions in terms of precision, recall, and F_β (Equation 2.1) was considered. The scoring functions are detailed in Section 2.5 and Section 4.3.1.

Table 5-1: Test performance of manual regular expressions

Attribute	Precision	Recall	F_β (β=1)
Time	100%	100%	100%
Race	94%	97.9%	95.91%
Age	100%	94.8%	97.33%
Height	100%	100%	100%
Hair Color	100%	90.6%	95.07%
Eye Color	100%	100%	100%
Weight	100%	100%	100%

Based on the results in Table 5.1, we concluded that regular expressions are suitable for the extraction of attributes in police incident reports. In the course of this work, however, we confirmed that it was both time consuming and tedious to create regular expressions for information extraction manually. In other words, knowledge engineering cost was significant.

5.2 *Semi-supervised RRE Discovery*

Due to the tedious and time-intensive nature of the manual approach, we embarked on a research effort to develop a data-driven algorithm to automatically discover regular expressions based on a small training set through means of a semi-supervised RRE Discovery algorithm. In this section, we will depict the results of the application of our semi-supervised learning algorithm on several data sources.

5.2.1 Results from Police Report Data

One of the data sources we used to test our semi-supervised learning algorithm was police incident reports. In order to analyze this data source, we used 10-fold cross validation based on 100 police incident reports that consisted of 1404 segments. For the attributes evaluated in the first experimental results reported below, the average values were 1264 training examples over 140 test examples. In a second experiment, we applied our combined semi-supervised active algorithm on a training dataset (404 items with 2983 segments) and an independent test dataset (149 items with 1462 segments).

Table 5.2 summarizes the results of our semi-supervised learning algorithm (without active learning). The ten different attributes evaluated are listed in the first column of Table 5.2. *Eye Color*, *Gender* and *Weekday* produced perfect results (F_β = 100%), in part because we have modified the lexicon as noted in section 4.2.1.3. The performance of *Age, Date, Hair Color, Height,* and *Race* were also excellent ($F_{\beta=1.0} \geq$ 90%). Even though the performance of *Time* and *Weight* is not as good as other attributes, they still achieved over 85% F_β. Given these results, we conclude that the RREs discovered for these ten features are of high quality overall.

Table 5-2. Results of semi-supervised non-active learning

Attribute	Average Precision	Average Recall	Average F_β (β=1)	Average true positives
Age	97%	89%	93%	12.6
Date	100%	95%	97%	8.9
Time	88%	83%	85%	7.7
Eye Color	100%	100%	100%	1.00
Gender	100%	100%	100%	33.6
Hair Color	90%	90%	90%	0.9
Height	100%	94%	96%	2.2
Race	97%	90%	91%	3
Week day	100%	100%	100%	9.8
Weight	90%	85%	87%	1.7

5.2.2 Results from Patent Data

Our semi-supervised algorithm was also applied to the extraction of the 'problem solved identifier' (PSI) in US patents [41]. The PSI in a patent identifies the particular solution that the patent addresses to an insufficiency in prior art.

For PSI extraction, our datasets included 55 patents; 15 of these, containing 7723 segments (sentences), were used in cross-validation. These patents were retrieved in the focused domain of text mining to enable us to label the training data more easily. Each segment in each patent was manually tagged by a human expert, thereby creating our target set. We split all true segments randomly into 10 folds for cross-validation. Each fold was given a roughly equal number of true segments (i.e., the folds were stratified). We then did the same thing with the false segments.

Using 10-fold cross-validation, we have achieved 56% average precision, 38% average recall, and 45% average $F_{\beta=1}$. These results are displayed in Table 5.3.

Table 5-3: 10-fold cross-validation test performance on patent data

Test sets	Precision	Recall	F-measure	# of true positives
1	85.71%	60.00%	70.59%	6
2	57.14%	40.00%	47.06%	4
3	62.50%	50.00%	55.56%	5
4	66.67%	40.00%	50.00%	4
5	37.50%	30.00%	33.33%	3
6	42.86%	30.00%	35.29%	3
7	40.00%	18.18%	25.00%	2
8	50.00%	27.27%	35.29%	3
9	71.43%	45.45%	55.56%	5
10	44.44%	36.36%	40.00%	4
Average	55.83%	37.73%	44.77%	3.9

An average precision of 55.83% indicates that over half of the sentences extracted using the RREs discovered by our algorithm contained information relevant to the problem solved by patents. Considering the complexity of natural language expressions used in patents, we consider this result promising. The average recall is 37.73%. This value is acceptable because, as noted previously, precision is more important than recall in this particular application. The average $F_{\beta=1}$ is 44.77%, indicating that we are currently successfully extracting PSI-related segments about half the time.

Another important metric is the distribution of correctly extracted PSIs across patents. In order to assess our algorithm's performance with regard to this metric, we measured the distribution of true positives from the 10 test folds across the 15 patents used to form the training set. Ideally, we would like to extract one or more PSIs from each patent. In this case, 80% of the original 15 patents were covered by at least one PSI. This result, too, is quite promising.

Our experimental results provide evidence that our approach to RRE discovery can be usefully applied to extract features from police incident reports and PSIs from patents. With the former application, we achieved excellent test set performance, and the latter application produced reasonable and stable test set performance. Our work with patent data is ongoing.

5.3 *Active Learning*

Having tested our semi-supervised RRE discovery algorithm, we then began evaluation of our semi-supervised, active learning RRE Discovery algorithm. Again, we utilized two data sources to perform this task: the police report data used to test the semi-supervised learning algorithm and a collection of narrative text university crime reports.

5.3.1 Results from Police Report Data

Table 5.4 depicts the reduction in training set development effort gained by the use of our semi-supervised active learning algorithm from the police report data. The second column of Table 5.4 is the ratio of true segments to false segments that are generated by the active learning algorithm described in Section 4.5. The third column presents the ratio of true segments to false segments overall in the training dataset. The fourth column is the reduction in labeling effort and is calculated as $C_4=1.0-C_3/C_2$, where C_4 is the value in the fourth column, C_3 is the value in the third column, and C_2 is the value in the second column.

Table 5-4: Labeling effort saved

Feature	Ratio of $\|T_a\|/\|F_a\|$	Ratio of true/false segments overall	Reduction in labeling effort	Training examples in active learning
Age	85 : 340	539 : 2444	11.78%	492
Date	9 : 1	222 : 2761	99.11%	22
Time	38 : 41	419 : 2564	82.37%	97
Eye Color	6 : 6	28 : 2955	99.05%	60
Gender	599:2168	1003:1980	-83.34%	2969
Hair Color	10 : 17	51 : 2932	97.04%	74
Height	27 : 36	93 : 2890	95.71%	117
Race	34 : 60	127 : 2856	92.15%	224
Week day	127 : 211	401 : 2582	74.20%	497
Weight	10 : 15	66 : 2917	96.61%	54

These results show that our active learning algorithm significantly reduces labeling effort for nine of the ten features. The significantly lower results for *Age* and *Gender* in Table 5.4 are due to the fact that these two features occur in many different contexts in the training data. In particular, *Gender* occurs in almost every possible context. In other words, our context-based active learning approach is not efficient when the feature is widely distributed among several contexts in the training data. Note that this does not necessarily imply that the performance of extraction will be degraded – as noted in Table 5.5 , *Gender* maintains F_β = 100%.

One disadvantage of this active learning approach is that the size of the true set is generally smaller than it would be if all of the training data were used. Using the *Time* feature as an example, the true set generated by the active learning process contains only 44 segments (38 in T_a, and 6 in $S_{content}$), while there are 419 true segments in the training set overall. In this particular case, however, our semi-supervised active learning algorithm exceeds the performance of the semi-supervised learning algorithm alone (98% vs. 85%). Nonetheless, in several other cases, the performance of our semi-supervised active learning algorithm is degraded compared to the semi-supervised learning algorithm alone. This result is due, in part, to the aforementioned bias of our active learning algorithm – the number of segments in the true set generated by the active learning process tend to be fewer than the actual number of true segments in the training data overall.

Table 5-5: Results of semi-supervised active learning

Feature	Precision	Recall	F_β (β=1)	Number of true positives
Age	76%	60%	67%	124
Date	100%	93%	96%	94
Time	98%	97%	98%	127
Eye Color	100%	100%	100%	5
Gender	100%	100%	100%	467
Hair Color	77%	83%	80%	10
Height	89%	89%	89%	16
Race	73%	84%	78%	16
Week day	100%	100%	100%	139
Weight	73%	73%	73%	11

Table 5.5 summarizes the test-set results for the ten features of interest using our semi-supervised active learning algorithm. As before, *Eye Color*, *Gender*, and *Weekday* had F_β = 100%. This implies that the RREs for these three features are stable. As already noted, the *Time* feature had a better test result than with the semi-supervised learning algorithm alone. The performance of *Date* in Table 5.5 was nearly identical to the performance achieved with our semi-supervised algorithm. Thus, for *Eye Color*, *Time*, *Date*, and *Weekday*, our semi-supervised active learning algorithm not only achieved a significant reduction of labeling effort (≥74%), but also maintained F_β performance equivalent to the performance of our semi-supervised algorithm alone.

The test performances of *Hair Color, Height, Race*, and *Weight* in Table 5.5 were not as good as the performance achieved when only using our semi-supervised algorithm. Nonetheless, the labeling effort was reduced dramatically for all four of these features (≥90%). Given that the reduction in F_β for these same features is less than 14%, we see that a tradeoff exists between training set development cost and extraction performance.

Our semi-supervised active learning approach does not work as well for the *Age* feature because there is an important sub-pattern of *Age* that was not discovered during the active learning phase. The seed that we used for this sub-pattern only covers a few segments. Therefore, the contextual patterns generated were not general enough to be used to find other similar patterns. As a result, the test performance of *Age* is degraded compared to our semi-supervised algorithm alone.

5.3.2 Results from Narrative Text University Crime Reports

We also evaluated our semi-supervised active learning algorithm on an online collection of narrative text university crime reports. The training set consisted of 74 reports with 160 segments, and the test set contained 48 reports with 117 segments. Four features of interest were considered in our experiments: *Date, Time, Item Value*, and *Gender*. Our results from using this data in our semi-supervised active learning algorithm are presented in Table 5.6. The fourth column of Table 5.6 portrays the reduction of labeling effort for these four features. The reduction of labeling effort was dramatic for the *Date, Item Value*, and *Gender* features (≥70%). The test performance of these four features was also quite good; these results are depicted in Table 5.7. All values obtained 100% precision. With respect to recall, *Date* and *Gender* were perfect (100%) while *Time* has nearly perfect at 99%. In general, our semi-supervised active learning algorithm performed well on these features extracted from online university crime reports.

Table 5-6: Labeling effort saved

Feature	Ratio of $\| T_a \| / \| F_a \|$	Ratio of true/false segments overall	Reduction in labeling effort
Date	46 :1	92 : 68	97.06%
Time	27 : 28	72 : 88	15.15%
Item Value	2 : 3	21 :139	77.34%
Gender	75 : 64	40 : 120	71.56%

Table 5-7: Results of semi-supervised active learning

Feature	Precision	Recall	F_β (β=1)	Number of true positives
Date	100%	100%	100%	67
Time	100%	98%	99%	47
Item Value	100%	82%	90%	9
Gender	100%	100%	100%	37

In this section, we have described how an active learning algorithm can be combined with our semi-supervised learning algorithm to extraction information from narrative text. These experimental results provide evidence that this approach significantly reduces training set development effort while, at the same time, preserving performance in terms of F_β.

5.4 *Comparison of RRE Discovery Algorithm with Popular IE Algorithms*

While our algorithm has produced significant results with test data, perhaps an even more important test is to compare our results with other popular IE algorithms. To conduct this comparison, we obtained the MUC7 dataset from LDC (*ISBN:* 1-58563-205-8), a package which also includes scoring software. Through this dataset, we can directly compare our IE algorithm with other systems that participated in MUC7 evaluation.

The MUC7 Named Entity task contains seven features that have been organized into three categories. These are listed in Table 5.8.

Table 5-8: MUC7 Named Entity categories and features

Named Entities	Temporal Expressions	Number Expressions
- ORGANIZATION	- DATE	- MONEY
- PERSON	- TIME	- PERCENT
- LOCATION		

The MUC7 IE tasks place no constraints on the methods employed. As a result, in order to improve the competitiveness of our system we prepared special purpose lexicons. These were constructed using a variety of sources. For example, some TIME-related words were downloaded from the following sources:

http://www.languageguide.org/english/grammar/full/part2/time.html
http://isu.indstate.edu/writing/handouts/setii/prints/SETII4AP.html
http://www.cist.org.uk/ay/gm/ga1601.htm
http://openforum.de/open.cgi/noframes/read/2395
http://www.enchantedlearning.com/languages/german/subjects/time.shtml.

For PERSON entries, we used three lists (last name, male first name, female first name) provided by the U.S. Census Bureau (http://www.census.gov/genealogy/names/names_files.html).

Since our algorithm can be provided with either relevant or irrelevant information as training data, we automatically converted the exact labeling of the MUC7 data into segment labeling data. All segments that contained exact feature labeling were treated as true segments. The segments without exact labeling were treated as false segments. While this conversion did lower the accuracy on some of the training data, the effects of this change were minimal.

To conduct our testing, we first used the training set (training.ne.eng.keys.980205) to train our algorithm. The dry-run set (dryrun.ne.eng.texts.970926) was used next to tune our IE system parameters. Then, after combining both training and dry-run datasets to our own training set, we trained the system. The final step was to test the system on formal run data. Using the same scoring software on formal run data (formaltst.ne.eng.texts.980301), the average MUC7 scores obtained and our system's scores are presented in Table 5.9. Of the seven features, only five have been evaluated to date; the two remaining features (LOCATION and ORGANIZATION) as well as PERSON are still undergoing tests.

Table 5-9: MUC7 results

Feature	MUC7 Average F1-measure	Our system F1-measure
DATE	90.24815%	87.44318%
TIME	87.12249%	88.87151%
MONEY	90.13344%	87.95455%
PERCENT	96.87039%	97.47692%
PERSON[5]	87.83406%	77.8882%

For two features, TIME and PERCENT, we had better performance than the MUC7 average; this is a significant achievement. PERSON is about 10% lower than the average, while DATE and MONEY are only slightly lower (< 3%) than the average. This performance is competitive for four of the five features in table 5.9 despite the fact that we employed our semi-supervised training algorithm, which significantly reduces knowledge engineering cost in terms of labeling[6].

6 An IE System Using the RRE Discovery Algorithm

Having established the accuracy of our semi-supervised active learning algorithm, we then went about constructing an actual system to utilize the RRE Discovery algorithm we had developed. The program we developed is the BPD_IE system, an acronym for the Bethlehem Police Department Information Extraction system.

[5] PERSON is still undergoing testing – these are not final results.
[6] Our MUC7 testing did involve additional knowledge engineering, however – for example, preparing special lexicons to enhance the performance of our system required additional knowledge engineering.

6.1 *System Analysis*

In this section, we provide an overview of the background of the BPD_IE system and also briefly introduce some of its components, techniques and functions. The details of the design work are expounded upon in Section 6.2.

6.1.1 A Description of the Relationship with the BPD

The BPD_IE program has been developed in conjunction and with the full support of the Bethlehem Police Department (BPD), located in Bethlehem, Pennsylvania, U.S.A. The arrangement has been mutually beneficial to both the officers and our research efforts. Working closely with Deputy Commissioner Randy Miller, Lieutenant Joe Kimock and Detective Matt Palmer, among others, we developed a system to aid the department in their criminal investigations. Using data provided by the BPD (in the form of narrative investigative report supplements, crime reports and affidavits), software we have developed for our research purposes converts this textual data into database records. Using a PC-based modus operandi search tool we developed for the project (described below), officers are able to instantaneously search thousands of records and reports – gathering information that would have previously taken uncounted hours to obtain.

Working with the BPD has provided not only an excellent test bed for our algorithm, but, perhaps more importantly, we have been able to provide a valuable service in disseminating the value of our tools to law enforcement officers. Generating IE rules learned from BPD investigator's reports using our IE learning algorithm ensures the accuracy of our system while also providing officers with crucial crime leads. Additionally, we have had the opportunity to receive feedback from the investigators that has helped us to design the user interface and further enhance the system.

6.1.2 Attribute Analysis

In order to conduct this project, it was necessary to determine which attributed, or features were useful for a search based on modus operandi. From our work with the BPD, we have identified several attributes relevant to criminal modus operandi. This involved the study of hundreds of BPD investigators' reports, and was further enhanced by our participation in numerous meetings with investigators. Table 6.1 lists some of the attributes identified in these meetings that are supported by our system.

Table 6-1: Attribute list

Feature	Type	Range
Classification	String	
Control Number	String	
Report Date	Integer	0000—1299 (mmyy)
Report Officer	String	
Location	String	
Arrest Number	String	
SSN	String	
Driver License Number	String	
Person Name	String	
Weight	Integer	0 – 1000 (pounds)
Eye Color	String	
Hair Color	String	
Height	Integer	0 – 120 (inches)
Relationship	String	
Race	String	
Phone Number	String	
Residence	String	
Age	Integer	0—200
Offense Type	String	
Day of Week	Integer	0—6 (Sunday =0, …, Saturday =6)
Item Stolen	String	
Clothing	String	
Drugs	String	
Time	Integer	0000-2400 (hours)
Vehicle	String	
Weapon	String	

6.1.3 Data Collection

There are three components that contributed to the data stored within the BPD_IE system. The first component is BPD detective reports in electronic format. Narrative text supplements of crime reports make up the second component; this information is entered manually to the current BPD database. The third and final component consists of narrative field(s) of the BPD Affidavit database that contains affidavits of probable cause.

There are many narrative text detective reports that are stored only in hardcopy (paper) form or in MSWord format. While police officers believe that there is much useful information contained in these reports, it is quite difficult for detectives to read through these reports manually. Our BPD_IE system uses a computer to 'read' through the documents and automatically extract useful information in the form of the features described in the following section. The output is then saved in a different BPD database named "BPD_IE", which is located on a server in the investigations department. This database can be utilized by BPD police officers for suspect identification.

All the BPD data is saved and used only in computer systems located within the BPD building. Detectives are able to use their personal computers in the BPD building to access the "BPD_IE" database to perform suspect identification. To maintain a high level of data security, these computers utilize a local network connection without access to the Internet.

6.1.4 RRE Rule File

RRE rules learned from the training set are saved in an XML file. Each RRE rule is divided into seven parts. The XML file starts with "<RRE>" and ends with "</RRE>"; a single XML file may contain multiple RRE rules in succession. Similarly, each component rule of the RRE starts with "<rule>" and ends with "</rule>"; there is no limit to the number of rules that may be contained in a particular XML file.

The first part of an RRE rule is *featureType*, which defines the type of feature the rule extracts. For example, if the rule is used to extract a person's height, then the value of the *featureType* tag is "*Height*". The second tag, *specialUser*, is used to identify the author of the rule if the rule was manually created (or automatically created and manually modified)

The next three tags, *prefix*, *pattern*, and *suffix* are directly related to the context of an RRE. *Prefix* contains an RRE that describes the features that precede the target feature (e.g., *Height*). *Pattern* is used to extract the target feature, and *suffix* denotes an RRE that describes the features following the target feature. These three RREs (*prefix, pattern*, and *suffix*) must all match in order for a feature value to be extracted – otherwise, the feature value is ignored.

Within each *prefix*, *suffix*, and *pattern* tag, there exists one or more *element* tags. Each *element* consists of *content, gapMin,* and *gapMax* tags, which are consistently arranged in this order. These tags may or may not contain values, but the *Pattern* tag must contain at least one *element* tag that has a value for *content*. *Content* is an RRE that matches the current element. *gapMin* and *gapMax* represent the minimum and maximum number of characters that are allowed to exist between the current element and the next element. In other words, the number of characters in a gap must be between *gapMin* and *gapMax*. By default, *gapMin* is set to zero. *Content* and *gapMax* are learned during training. (Note: for detailed definitions of *element* and *gap*, please refer to [25].)

The sixth tag, *probability*, is the training score; it is used to record the certainty or accuracy of the RRE rule and is derived during training. The seventh and final tag, *converter*, is the name of the function used in the RRE conversion routine to convert the textual value of an attribute to a numeric value as appropriate.

Figure 6.1 is a sample rule file that contains two sample RRE rules.

```
<RRE>
    <rule>
      <featureType>Height</featureType>
      <specialUser>Tianhao Wu</specialUser>
      <prefix>
            <element>
            <content></content>
            <gapMin>0</gapMin>
            <gapMax>0</gapMax>
            </element>
      </prefix>
      <pattern>
            <element>
                 <content>feet</content>
                 <gapMin>0</gapMin>
                 <gapMax>5</gapMax>
            </element>
      </pattern>
      <suffix>
            <element>
                 <content> </content>
                 <gapMin>0</gapMin>
                 <gapMax>0</gapMax>
            </element>
      </suffix>
      <probability>0.9</probability>
      <converter>height_func</converter>
    </rule>

    <rule>
      <featureType>Weight</featureType>
      <specialUser>Stephen Zanias</specialUser>
      <prefix>
            <element>
              <content ><content />
              <gapMin>0</gapMin>
              <gapMax>0</gapMax>
            </element>
      </prefix>
      <pattern>
            <element>
              <content>pounds</content>
              <gapMin>0</gapMin>
              <gapMax>10</gapMax>
            </element>
      </pattern>
      <probability>0.8</probability>
      <converter>NULL</converter>
    </rule>
</RRE>
```

Figure 6-1: RRE rule format

The XML format of an RRE has been designed to allow the user to easily read and modify the rules for a more customizable approach. A function accessible from both BPD_FE (the BPD_IE feature extraction subsystem) and BPD_MO (the BPD_IE modus operandi search subsystem) parses the XML file to recreate reduced regular expressions.

6.1.5 BPD_FE

BPD_FE is the sub-system where the narrative text is converted and features are extracted and loaded into database table entries. A user interface has been created for this task. This section briefly mentions some of techniques used in this sub-system; for details of the design, please refer to Section 6.2.3.

6.1.5.1 Converting Features Extracted to Data Entries in Database

It is necessary to interpret all features extracted from the investigators' reports prior to their entry into the database. For example, there are different ways to express a person's height: "five feet tall", "five feet eight inches tall", and "5'8"" are just a few examples. These entries must be normalized into a standard expression so that modus operandi search can use a range search function (see section 6.1.6.2). In the example of a person's height, we use an integer to express a person's height in inches. Therefore, "five feet tall" is entered into the database as "60"; "five feet eight inches tall" and "5'8"" are both converted to "68." This normalization makes range search possible.

To convert features to database entries, we first modify RREs to indicate which elements need to be converted. Next, the program passes the elements to a programmer-specified function designed to convert the data. A simple example of this conversion process is depicted in Table 6.2, where \$1 and \$2 are two variables that contain the elements to be converted.

Table 6-2: Data entry conversion

Feature Extracted	five feet eight inches tall
RRE ("\s" symbolizes white space)	([a-z0-9]+)\sfeet([a-z0-9]+)\sinches tall
Elements to be converted	• five • eight
Conversion function	$1=5 (five) $2=8 (eight) result=$1*12+$2
Data Entry in database	68

To support even greater functionality, users can define their own conversion routines by modifying the "defaultRRErules.xml" XML file and the source code of the "RRE_converter." After a user creates a new function in the "RRE_converter", they can assign the function's name to a rule's *converter* tag in "defaultRRErules.xml". The function will be called automatically whenever the rule is fired.

6.1.5.2 Automatic Upload of Investigators' Reports

Support for batch loading and conversion is essential for the system. Investigators and detectives often keep their documents in relatively few (if not one) designated directories or sub-directories. It is advantageous to load all files from these directories into the system in a simple manner. When using the batch upload function, the user can simply indicate a specific file folder (i.e., the directory containing all the documents) and allow the system to automatically perform the feature extraction and database operations. This type of capability can be useful in allowing the user to perform this function to update the BPD_IE database on a regular basis. If the user begins the process every evening before leaving the office, the program can run during the night and provide the BPD_IE database with the most recent, up to date data available.

This capability is also beneficial when initializing the system. With the BPD, all investigators' reports are located on a BPD network file server rooted a specific directory. The batching loading and conversion capability allowed us to easily upload all investigators' reports to our BPD_IE database.

In order to handle any errors that may occur during the batch conversion, a error file is created and saved under the ./bin folder. This will allow any errors to be handled in an appropriate manner offline.

6.1.5.3 Upload XML Files to the BPD_IE Database

Although the BPD does not maintain electronic records of initial crime reports, these reports often contain useful supplements. As a result, as noted above, some of this data has been manually entered. To accomplish this, the crime reports are saved as XML files. To support this, we created a XSD file matching the BPD crime report (Appendix G), and use a free xml editor "XAmple XML Editor" (available at http://www.icewalkers.com/Linux/Software/520470/XAmple-XML-Editor.html) to enter the data.

After the data is entered as XML files, a function within the BPD_FE sub-system loads the files into the BPD_IE database. This function can also extract features from the narrative supplements of the crime reports.

6.1.5.4 Update RRE Rule File from HDDI Website

The BPD_IE System user can download the latest RRE rule file from http://hddi.cse.lehigh.edu/source/defaultRRErules.xml. To do so, clink the button pictured below in the BPD_FE tool to download and install the rule file automatically. You need an internet connection and must set up your proxy or personal firewall to allow the connection.

Once the download is finished, you must restart BPD_FE.

6.1.6 BPD_MO

BPD_MO is another component of the BPD_IE system; it is a search tool that helps users to search relevant documents using features stored in the BPD_IE database. We briefly mention some of techniques used in this sub-system in this section; for details of the design work of the BPD_MO, please refer to Section 6.2.4.

6.1.6.1 Scoring Function for Result Ranking

Since modus operandi search can return many reports for each query, it is necessary to develop a ranking method to sort these results. We defined a scoring function to assign a similarity score to each report retrieved from the database based on the report's similarity to the query attributes. The higher the score is, the better the report matches the query.

To determine the similarity between a report and a given query, modus operandi search calculates the similarity between attribute values of the given investigative report and the attribute values given in the query. The similarity is computed as depicted in Equation 6.1, where SimilarityScore is 0 if "$w_1+w_2+\ldots+w_n$" equals zero.

$$SimilarityScore = \frac{w_1}{w_1+w_2+\ldots+w_n}S_1 + \frac{w_2}{w_1+w_2+\ldots+w_n}S_2 + \ldots \frac{w_n}{w_1+w_2+\ldots+w_n}S_n$$

Equation 6-1: The Similarity function

The variables in Equation 6.1 are defined as follows:

- n is the number of attributes in the query, $n \geq 1$

- S_i is the similarity between the i^{th} attribute in the query and the report, $1 \le i \le n$, and
- w_i is the weighting factor of the i^{th} attribute in the query, $1 \le i \le n$, and $0 \le w_i \le 10$.

For numeric values, S_i is defined in Equation 6.2.

$$S_i = \begin{cases} \dfrac{|\,overlap(R_{di}, R_{qi})\,|}{Max(R_{di}, R_{qi}) - Min(R_{di}, R_{qi})}, & if \quad Max(R_{di}, R_{qi}) \neq Min(R_{di}, R_{qi}) \\ 1, & if \quad Max(R_{di}, R_{qi}) = Min(R_{di}, R_{qi}) \end{cases}$$

Equation 6-2: Attribute Similarity between query and report

The variables in Equation 6.2 are defined as follows:

- R_{di} is the range of the i^{th} attribute in the report
- R_{qi} is the range of the i^{th} attribute in the query
- overlap(R_{di}, R_{qi}) is the overlap of R_{di} and R_{qi},
- Max(R_{di}, R_{qi}) is the maximum value of R_{di} and R_{qi}, and
- Min(R_{di}, R_{qi}) is the minimum value of R_{di} and R_{qi}.

For example, if the query attribute range R_{qi}=(20, 30) and the report attribute range R_{di}=(25, 40), then overlap(R_{di}, R_{qi})=5, Max(R_{di}, R_{qi})=40, and Min(R_{di}, R_{qi})=20. Therefore, S_i=5/(40-20)=0.25.

For nominal values, S_i is 1 if the value of the i^{th} attribute in the query equals the value of an attribute in the report. Otherwise, S_i is 0.

Equation 6.2 has the following properties:

- S_i =1 if and only if R_{di} equals R_{qi}
- S_i =0 if and only if there is no overlap between R_{di} and R_{qi}
- $0 \le S_i \le 1$
- A higher value of S_i means R_{di} and R_{qi} are more similar to each other

Therefore, Equation 6.1 has the following properties:

- SimilarityScore =1 if all the attributes' values in the query are the same as the attributes' values in the investigative report

- SimilarityScore =0 if there is no overlap between any attribute value in the query and any attribute value in the investigative report
- $0 \leq$ SimilarityScore ≤ 1
- A higher value of SimilarityScore means the query and the document are more similar to each other

6.1.6.2 Fuzzy Search

After several discussions with BPD detectives and investigators, it became apparent that the information contained in investigators' reports is not always 100% accurate. An example of this occurs when a person's name is incorrectly entered. Suspects may also use variations of their phone number or social security number at times. The inaccurate information cannot be retrieved from the database using traditional SQL-based database search methods; such searches require exact matching to return results. Therefore, a *fuzzy search* must be supported by the modus operandi search subsystem. At this point, two types of fuzzy search have been implemented in modus operandi search: edit distance [42] search for string features, and range search for numeric features.

6.1.6.2.1 Fuzzy Search using Edit Distance

An edit distance function was implemented using Microsoft Visual C++ .NET. This function compares two input strings and returns the edit distance between them [42]. A user can define the minimum edit distance through the user interface. If the edit distance between a string value in the document and a string in the query is less than or equal to the minimum edit distance, then the system considers the two values as a match.

The edit distance function is used as a user defined function in the MySQL database and is compiled to a dynamic link library (editDistance.dll). The following command must be executed in the MySQL command console to create a user defined function named "editDistance" using the DLL file: "`CREATE FUNCTION editDistance RETURNS INTEGER SONAME "editDistance.dll";`"

Currently, an edit distance match is considered to be the same as an exact match. For example, suppose an investigator searches for an "offense type" and enters a query string of "robbery." If report_1 in the database contains "robbery", report_2 contains "Robbery", and the minimum edit distance is set to one, then report_1 and report_2 will have the same ranking score in the result set because only one 'edit' is needed to change an 'r' to and 'R'.

6.1.6.2.2 Search by Range

The investigators' reports often include features that have ranges. For instance, a person's height may be written as "from five feet six inches to six feet tall". Our feature extraction system can pick up this range information and convert it into start and end values; both start and end values can then be stored in the database.

If a user enters a single value in the query, then the modus operandi search system will return all documents that have the same value or have a range that covers the query value. For example, if the user enters the query "Age=20", the system will retrieve all reports that have "Age=20" or an age range covering 20, such as "Age=10 to 30". To differentiate between exact value matches and range matches, the ranking differs for the two kinds of reports returned.

If a user uses a range in the query, then the modus operandi search system will return all reports with a value or a range that overlaps with the query value. For example, if the user searches for "Age = 20 to 30", the system will retrieve all reports that have "Age=20", "Age=21", ..., "Age=30", or have an age range covering a number between 20 and 30, such as "Age = 15 to 25".

We use Equations 6.1 and 6.2 to sort the range search results. All reports with SimilarityScore > 0 are retrieved from the database. For the attributes that support range search, please refer to Table 6.1.

6.2 *System Design*

The Information Extraction System for the BPD_IE system contains three major subsystems. The first subsystem is the Model Builder which automatically generates information extraction rules from the training dataset. The second subsystem is the Feature Extractor; it makes use of the extraction rules generated by the Model Builder to extract features from the user input. The third subsystem is a Modus Operandi Search component through which users can search investigative reports containing specific modus operandi.

The following major functionalities of the Information Extraction System correspond to the three subsystems, respectively: "Generate RRE rules", "Drag and drop files to extract information", and "Modus Operandi Search". These subsystems and the system as a whole were designed using Rational Rose. A use case diagram of the BPD_IE System including all three functionalities is depicted in Figure 6.1.

There are five outside actors indicated in the use case: the Rule Developer actor, the TMI, the Detective, the GJXDM, and the BPD_IE database. A Rule Developer actor creates information extraction RRE rules that are implicitly used in the Feature Extractor. The TMI is the Text Mining Infrastructure that provides preprocessing functions such as sentence boundary detection, tokenization, part-of-speech tagging, and other similar capabilities. A Detective can perform both information extraction and conduct modus operandi search. The GJXDM and the BPD_IE databases are used to store all the original textual data, system files, and features extracted from the investigative reports. The BPD_IE database is the data source used by the Modus Operandi Search subsystem. We discuss the details of each of these functionalities (behaviors) in the following sections.

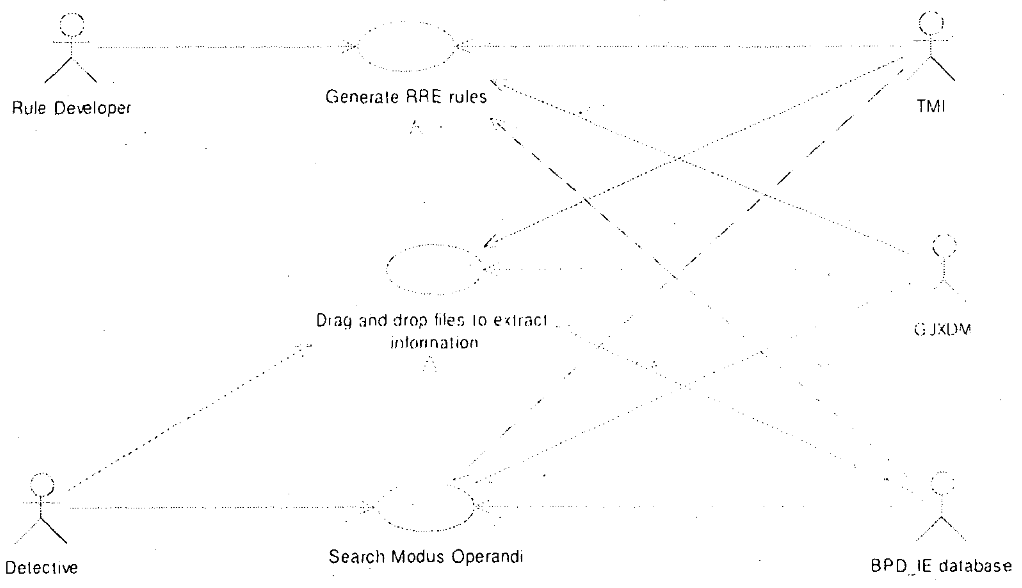


Figure 6-1: Use Case diagram for the BPD_IE System

6.2.1 Generate RRE Rules

Two components are used together to help the Rule Developer actor (a human user) to create information extraction rules. The first component is the TrainingSet Developer, which prepares a training set for the Rule Developer actor to label. The second component is the Model Builder in which a program automatically discovers information extraction rules expressed as Reduced Regular Expressions from the training set. The training set could come from the TrainingSet Developer or other sources. For example, the Rule Developer actor might directly label segments in a GJXDM XML file. We will explain these two components in detail in sections 6.2.1 and 6.2.2.

6.2.1.1 TrainingSet Developer

The TrainingSet Developer provides a user interface to help the Rule Developer actor to label the training set. To begin, the Rule Developer actor drags and drops training files or file folders onto the BPD_TD.exe icon, which is located on the desktop of the user's computer. This automatically invokes the Collection Builder to preprocess the training data. When the preprocessing is finished, the files in the training set have been segmented. Meanwhile, the TrainingSet Developer reads in all relevant feature types and associated labels from a data source indicated in the system configuration. After all segments, labels, and feature types are known, the Rule Developer actor assigns a <featureType, Label> pair to each segment according to their domain knowledge using the user interface provided by the TrainingSet Developer. Finally, all labeled segments are saved in the BPD_IE database or in a GJXDM XML file.

6.2.1.1.1 Preconditions

When a Rule Developer actor drags and drops documents onto the TrainingSet Developer, the documents must first be converted to plain text form. This is accomplished for MSWord documents by the invocation of an automatic file converter provided by Microsoft as part of their Office Suite. For older format files for which no automatic conversion utility exists, the user must convert the format of the input documents into plain text manually. This step can be completed by using a "Save As" function in any word processor.

6.2.1.1.2 Class Overview

As depicted in Figure 6.2, there are 16 classes in the TrainingSet Developer: BPD_IE, BPD_TD, Configuration, Document, DetectiveReport, PoliceIncidentReport, OldWordPerfectReport, Segment, Feature, CollectionBuilder, TrainingSet Developer, RREGenerator, Label, Storage, ODBC, and GJXDMConnecter.

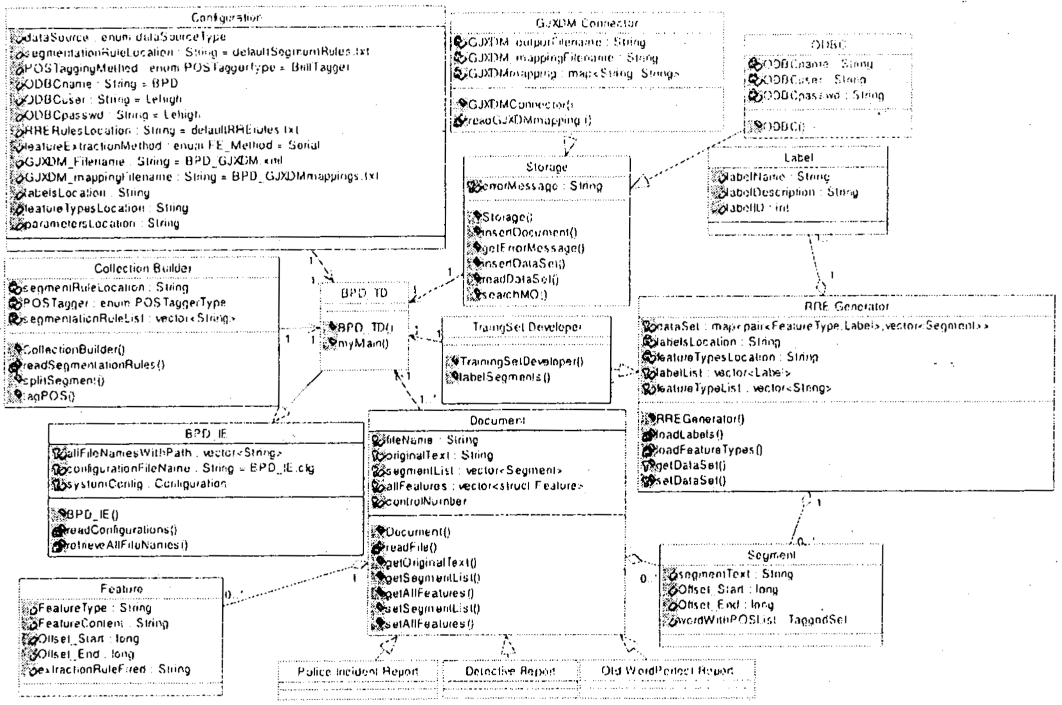

Configuration
dataSource : enum dataSourceType
segmentationRuleLocation : String = defaultSegmentRules.txt
POSTaggingMethod : enum POSTaggerType = BrillTagger
ODBCname : String = BPD
ODBCuser : String = Lehigh
ODBCpasswd : String = Lehigh
RRERulesLocation : String = defaultRRErules.txt
featureExtractionMethod : enum FE_Method = Serial
GJXDM_Filename : String = BPD_GJXDM.xml
GJXDM_mappingFilename : String = BPD_GJXDMmappings.txt
labelsLocation : String
featureTypesLocation : String
parametersLocation : String
GJXDM Connector
GJXDM_outputFilename : String
GJXDM_mappingFilename : String
GJXDMmapping : map<String, String>
GJXDMConnector()
readGJXDMmapping()
ODBC
ODBCname : String
ODBCuser : String
ODBCpasswd : String
ODBC()
Storage
errorMessage : String
Storage()
insertDocument()
getErrorMessage()
insertDataSet()
readDataSet()
searchMO()
Label
labelName : String
labelDescription : String
labelID : int
Collection Builder
segmentRuleLocation : String
POSTagger : enum POSTaggerType
segmentationRuleList : vector<String>
CollectionBuilder()
readSegmentationRules()
splitSegment()
tagPOS()
BPD_TD
BPD_TD()
myMain()
TraingSet Developer
TrainingSetDeveloper()
labelSegments()
RRE Generator
dataSet : map<pair<FeatureType, Labels>, vector<Segment>>
labelsLocation : String
featureTypesLocation : String
labelList : vector<Label>
featureTypeList : vector<String>
RREGenerator()
loadLabels()
loadFeatureTypes()
getDataSet()
setDataSet()
BPD_IE
allFileNamesWithPath : vector<String>
configurationFileName : String = BPD_IE.cfg
systemConfig : Configuration
BPD_IE()
readConfigurations()
retrieveAllFileNames()
Document
fileName : String
originalText : String
segmentList : vector<Segment>
allFeatures : vector<struct Feature>
controlNumber
Document()
readFile()
getOriginalText()
getSegmentList()
getAllFeatures()
setSegmentList()
setAllFeatures()
Segment
segmentText : String
Offset_Start : long
Offset_End : long
wordWithPOSList : TaggedSet
Feature
FeatureType : String
FeatureContent : String
Offset_Start : long
Offset_End : long
extractionRuleFired : String
Police Incident Report
Detective Report
Old WordPerfect Report

Figure 6-2: Class diagram of the TrainingSet Developer

BPD_IE is a super class of BPD_FE, BPD_MB, BPD_MO, and BPD_TD. BPD_IE loads all configuration information from a file and retrieves filenames from user input. All configuration information is stored in a Configuration object. The configuration information, filenames, etc. are accessed in all subclasses of BPD_IE.

The BPD_TD class instantiates the TrainingSet Developer because it contains the main function myMain().

RREGenerator is a super class of TrainingSet Developer and ModelBuilder. RREGenerator contains all feature types and associated labels. The entire training dataset is stored in the dataset attribute of TrainingSet Developer. RREGenerator uses both the Segment and Label classes. TrainingSet Developer interacts with the Rule Developer actor to complete the segment labeling task.

DetectiveReport, PoliceIncidentReport, and OldWordPerfectReport are subclasses of Document. Each subclass contains some attributes specific to certain document types used by BPD investigators such as detective reports, police incident reports, or old Word Perfect reports. The Document class contains most of the information related to an investigator's report. This information includes the original text of the report, the segments and part-of-speech tags of the original text, etc.

Feature and Segment do not contain methods (i.e., they are akin to C structs). The Feature class is used as a structure to keep all information related to features extracted from the original text. Likewise, the Segment class contains segment related information.

GJXDMConnector and ODBC are subclasses of Storage. GJXDMConnector is used to access a GJXDM XML file based on the mapping between feature type and the GJXDM library. ODBC is a bridge between the system and the BPD_IE database. The virtual methods insertDocument(), insertDataSet(), readDataSet(), and searchMO() in the Storage class must be implemented in the subclass. Only the insertDataSet() and getErrorMessage() methods are used in the TrainingSet Developer. The details for all classes are provided in Appendices D and E.

6.2.1.1.3 Interaction View

In this section, we explain how the 16 classes in Figure 6.2 interact with each other. We use both a sequence diagram and a collaboration diagram to illustrate the interaction. The sequence diagram depicting the development of a training set is shown in Figure 6.3, and Figure 6.4 is the collaboration diagram of the TrainingSet Developer.

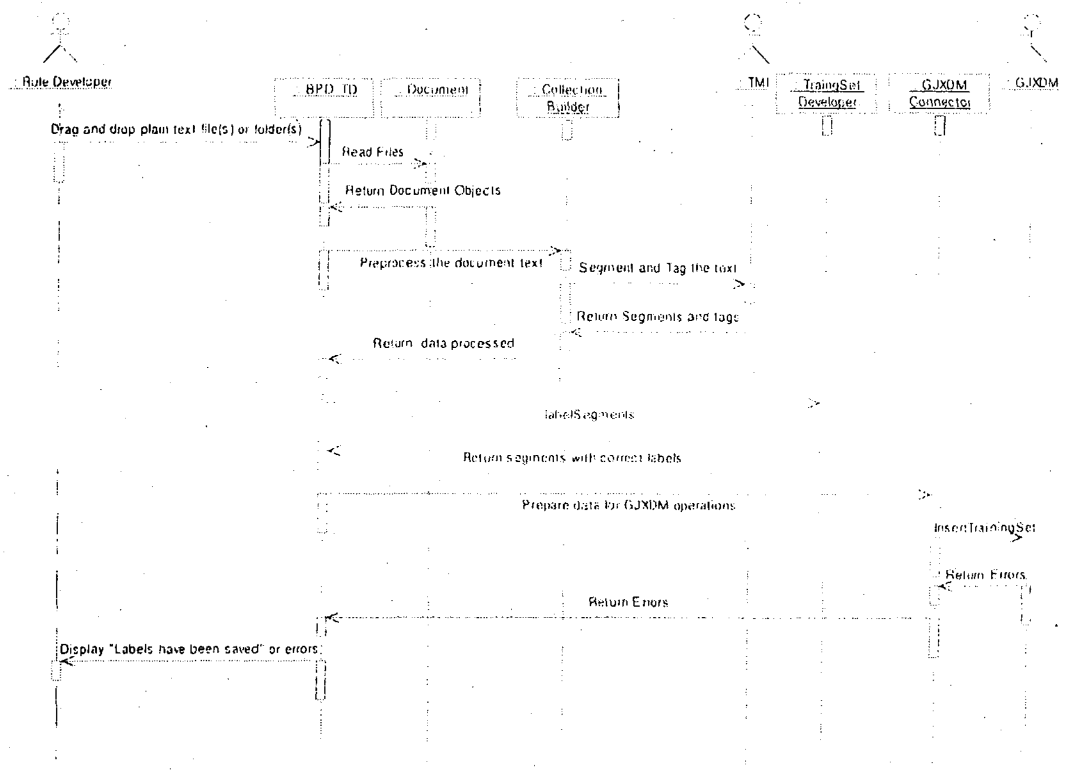


Figure 6-3: Sequence diagram of the TrainingSet Developer

As noted previously, to initiate the process the Rule Developer actor must drag and drop all training documents onto the BPD_TD.exe icon. Note that all training documents can be located in a single folder, and the Rule Developer actor can simply drag and drop the folder to the icon due to the batch processing supported by the system. Upon system startup, a BPD_TD object is instantiated. The BPD_TD object reads input documents into Document objects. Second, documents are segmented by the Collection Builder that calls TMI library functions to perform sentence boundary detection. After documents are segmented, segments are passed to the TrainingSet Developer. Meanwhile, all possible feature types and the associated labels are read into the program. The Rule Developer actor must then assign one <FeatureType, Label> pair to each segment; multiple labels per segment are not supported in the current system. Next, segments labeled are saved into the BPD_IE database or a GJXDM XML file through a Storage object (BPD_IE storage via ODBC is not pictured in the diagrams since it is similar to GJXDM storage). Finally, an acknowledgement message pops up to inform the Rule Developer actor of any errors or to indicate success in saving the labeled training data.

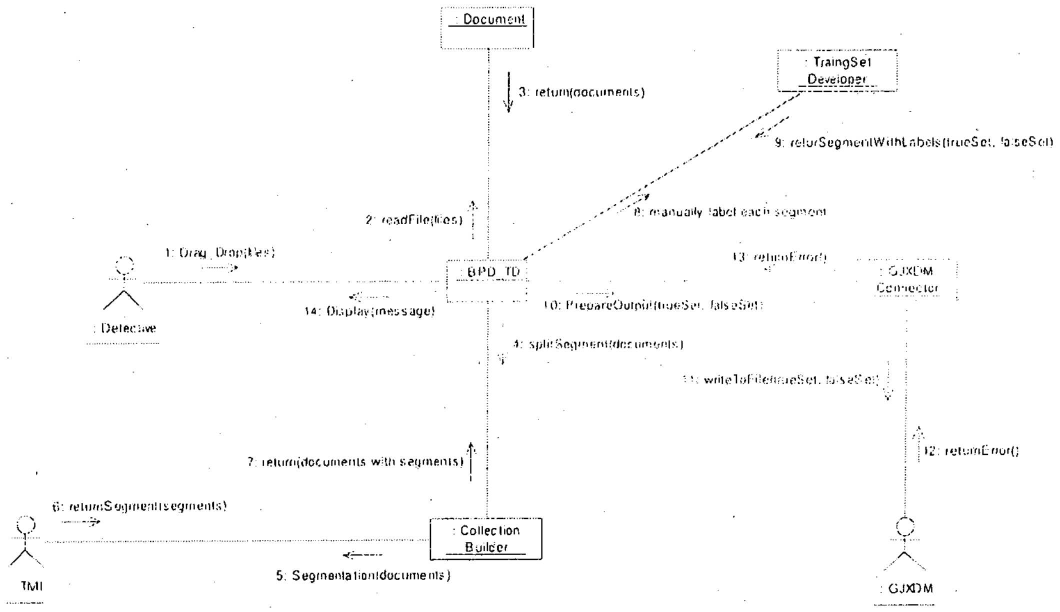

: Document
3: return(documents)
: TraingSet Developer
9: returSegmentWithLabels(trueSet, falseSet)
2: readFile(files)
8: manually label each segment
1: Drag_Drop(files)
: BPD_TD
13: returnError()
: GJXDM Connecter
14: Display(message)
10: PrepareOutput(trueSet, falseSet)
: Detective
4: splitSegment(documents)
11: writeToFile(trueSet, falseSet)
7: return(documents with segments)
12: returnError()
6: returnSegment(segments)
: Collection Builder
TMI
5: Segmentation(documents)
: GJXDM

Figure 6-4: Collaboration Diagram of the TrainingSet Developer

6.2.1.2 Model Builder

The semi-supervised RRE Discovery algorithm is contained in the Model Builder subsystem. This algorithm discovers and generates rules for feature extraction based on the training set. The Rule Developer actor interactively selects training sets and feature types. The Model Builder generates rules for these feature types automatically, and then stores these rules in a database or GJXDM XML file for future use by the Feature Extractor.

6.2.1.2.1 Preconditions

The Model Builder requires a set of training data that is segmented and labeled. The training set could be the output of the TrainingSet Developer. Alternatively, other segmentation and labeling methods could be utilized. For instance, users could write a small program to perform their own segmentation and then manually label the training set using a word processor.

6.2.1.2.2 Class Overview

There are thirteen classes in the Model Builder: BPD_IE, BPD_MB, Configuration, ModelBuilder, Parameter, RREGenerator, Segment, Label, Rule, Element, Storage, ODBC, and GJXDMConnecter. A class diagram of the Model Builder is in Figure 6.5.

BPD_MB is a subclass of BPD_IE. BPD_MB contains the only main function in the Model Builder. This main function, myMain(), controls the flow of execution. It uses the ModelBuilder class to generate information extraction rules (Reduced Regular Expressions) and stores the rules using the Storage class.

ModelBuilder is a subclass of RREGenerator into which the labeled training set and feature types are preloaded. In ModelBuilder, all rules are generated based on the training set. The final rule list is saved as a rule file, within GJXDM, or in the BPD_IE database.

Rule is a class that contains the details of a rule. The Rule class consists of Element objects, and Rule is embedded in the ModelBuilder class. The Parameter class includes all parameters used in the Model Builder. The parameters are stored in the trainingParameters attribute by the Model Builder.

GJXDMConnector and ODBC were discussed previously in Section 6.2.1.2. Only the readDataSet() and getErrorMessage() behaviors (methods) are used in the Model Builder.

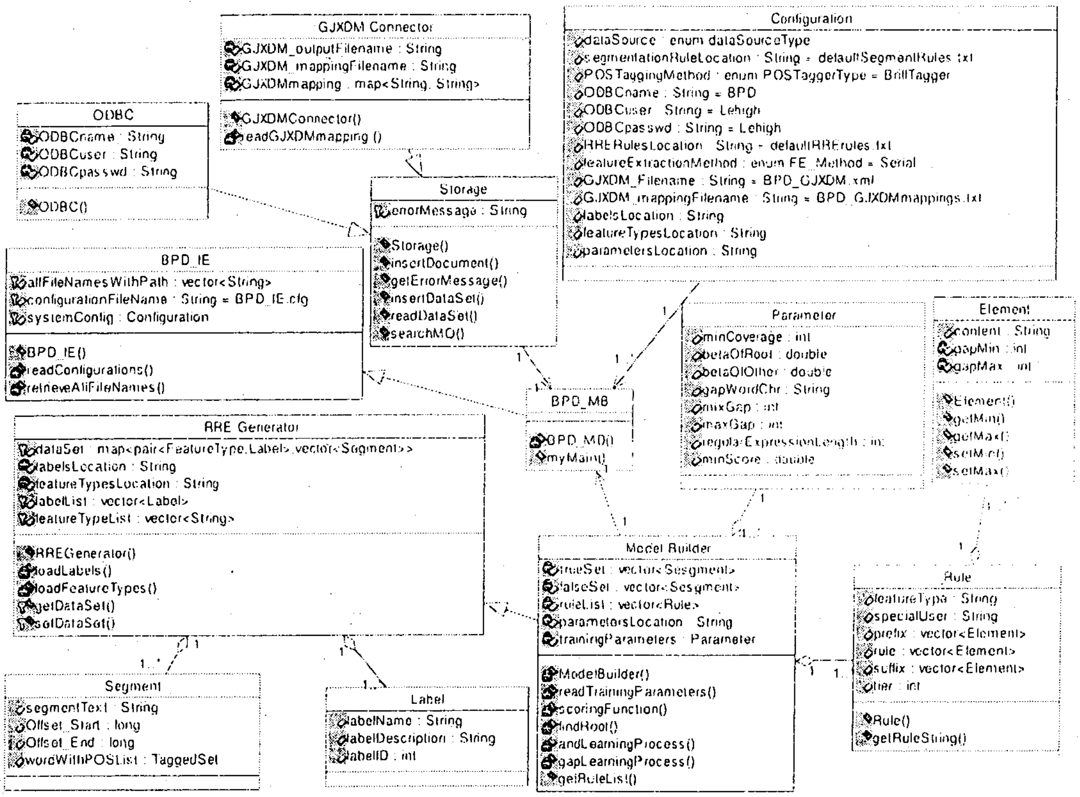


Figure 6-5: Class diagram of the Model Builder

6.2.1.2.3 *Interaction View*

There are two ways to invoke the Model Builder. The first method is to drag and drop GJXDM XML file(s) onto the BPD_MB.exe icon. The Model Builder will then ignore the GJXDM settings in the configuration file, and use the input XML file as the training set. The second method is to provide a GJXDM XML file name or ODBC settings in the configuration file and double-click on the BPD_MP.exe icon.

Figure 6.6 is the sequence diagram for the first method (the second method is nearly identical and is not shown). First, the BPD_IE reads system configurations and prepares input file names for the BPD_MB, in which a ModelBuilder object is instantiated. The ModelBuilder object reads in the feature types and the labeled training set upon the invocation of the RREGenerator's constructor. If the Rule Generator actor uses the first method outlined above to invoke the program, RREGenerator reads training data from the input XML file(s) using a GJXDMConnector. Otherwise, RREGenerator accesses a Storage object to read this training data either from an XML file or via an ODBC connection. Training parameters are loaded in the constructor for ModelBuilder. Following this, the Model Builder generates information extraction rules using several private methods. Finally, the rules generated are saved using either the GJXDM Connector or ODBC class.

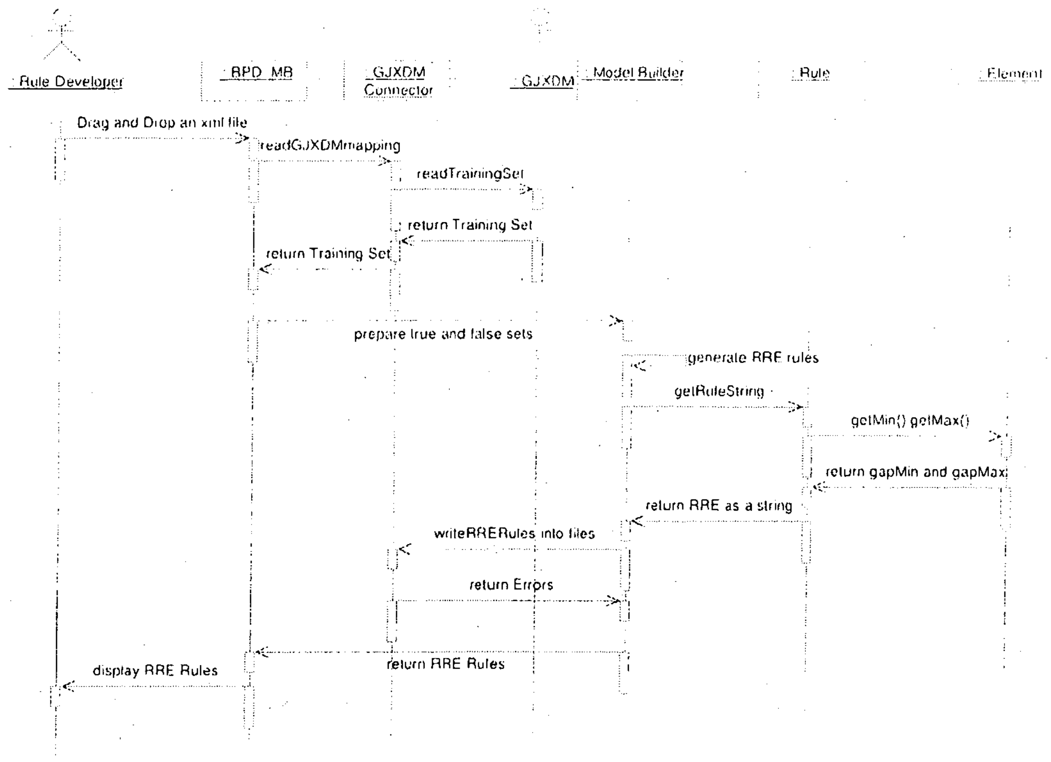


Figure 6-6: Sequence diagram of the Model Builder

6.2.2 Drag and Drop Files to Extract Information

This is one of the most important behaviors in which a Detective is involved. A brief summary of the behavior is as follows: first, a Detective drags and drops file(s) or file folder(s) onto the BPD_FE.exe icon. The BPD_FE will call the TMI library to preprocess the input files. Following this, a Feature Extractor is used to automatically extract features such as a suspect's physical description and modus operandi from the files. Finally, all features extracted and the original textual data are stored in a GJXDM XML file and in the BPD_IE database for use in modus operandi search.

6.2.2.1 Preconditions

All input files must be either in plain text or a known file format. Otherwise, the methods detailed in Section 6.2.1.1 describe how to manually convert input into plain text files.

6.2.2.2 Class Overview

There are eighteen classes related to this behavior: BPD_IE, BPD_FE, Document, DetectiveReport, PoliceIncidentReport, OldWordPerfectReport, Feature, Segment, CollectionBuilder, FeatureDiscover, SerialFeatureExtractor, ParallelFeatureExtractor, Configuration, Rule, Element, Storage, GJXDMConnector, and ODBC. A class diagram is depicted in Figure 6.7. For the details of each class, please see Appendices D and E.

In Figure 6.7, the classes SerialFeatureExtractor and ParallelFeatureExtractor are subclasses of FeatureDiscover. These two subclasses have different methods to handle multiple documents. SerialFeatureExtractor uses a sequential loop to handle documents one by one. ParallelFeatureExtractor handles documents using a multi-threaded process. The virtual method extractFeatures() in class FeatureDiscover must be implemented in the subclass.

To understand the class diagram properly, it is necessary to distinguish between a class and an instantiation of a class, termed an object. A single class can be instantiated multiple times, producing multiple objects. In what follows we discuss several such objects.

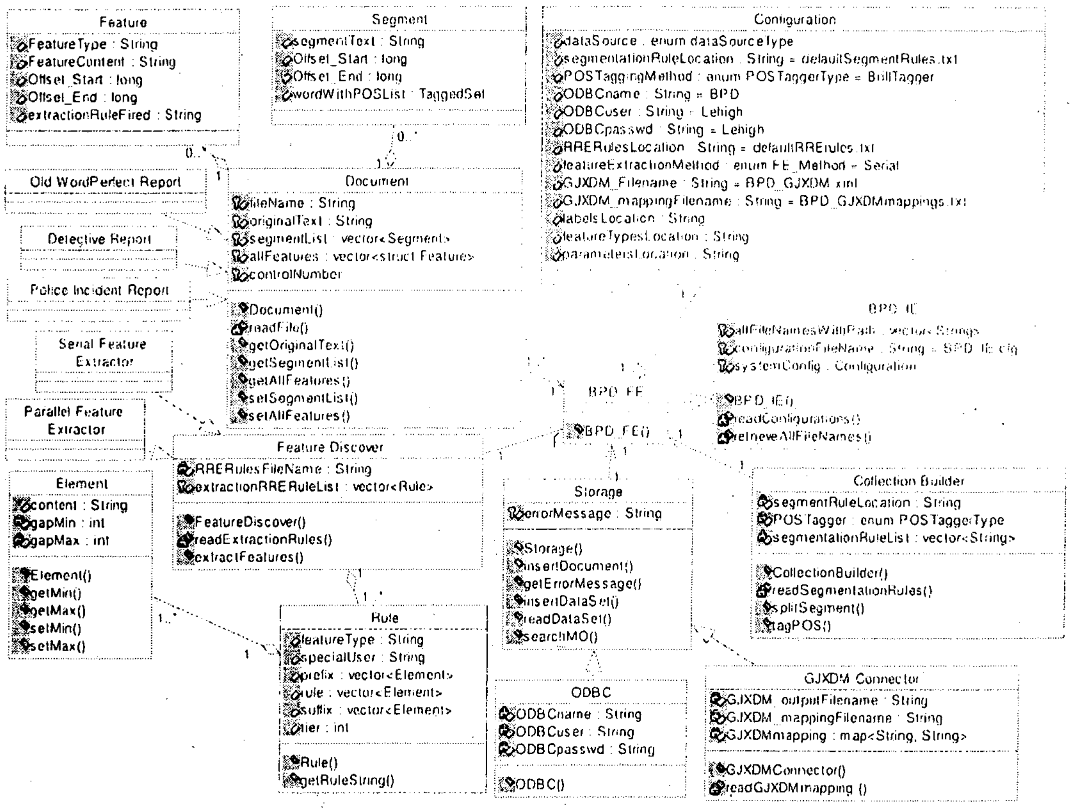


Figure 6-7: Class diagram of "Drag and Drop Files to Extract Information"

There is only a single Configuration object for a given BPD_FE object. On the other hand, a single Configuration can be used to initialize one or more BPD_FE objects. Each BPD_FE object can process one or more Document objects (i.e., reports). Each Document object, however, must be processed by only a single BPD_FE object in order to prevent processing the same report twice. Each Document object contains zero or more segments and features, and each segment or feature maps to the given Document object. Each BPD_FE object corresponds to a single CollectionBuilder object and a single FeatureExtractor object. Likewise, all features are extracted using only a single FeatureExtractor object. However, all reports (Document objects) provided in the input are preprocessed by a single CollectionBuilder object. Each Storage object is associated with a single BPD_FE object. In addition, all features extracted for all Document objects are stored either in a single GJXDM XML file or in the BPD_IE database through a single GJXDMConnector or ODBC object. A dynamic interaction view of this behavior is portrayed in the following section.

6.2.2.3 Interaction View

In this section, we describe the "Drag and Drop Files to Extract Information" behavior over time. A sequence diagram of this behavior is depicted in Figure 6.8. A collaboration diagram is portrayed in Figure 6.9.

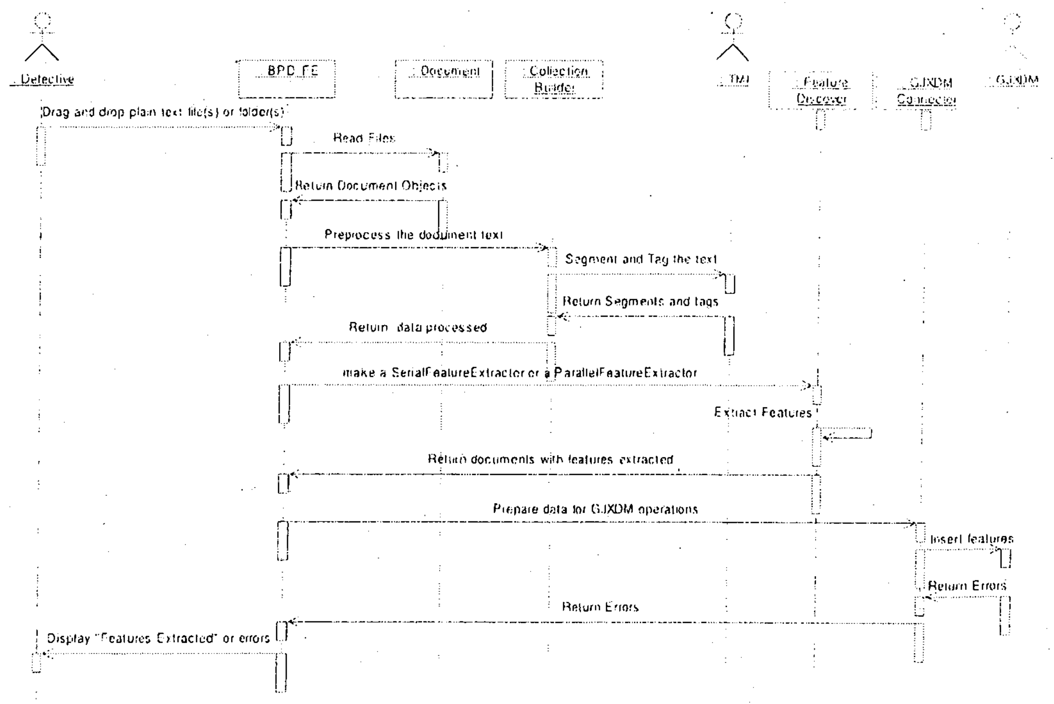


Figure 6-8: Sequence diagram of "Drag and Drop Files to Extract Information"

The user can drag and drop input reports(s) onto the BPD_FE.exe icon to activate the BPD_FE's main procedure to invoke the "Drag and Drop Files to Extract Information" behavior.

It should be noted that multiple files can be dragged and dropped at the same time. Similarly, one or more file folders can be supported, but files contained in subfolders of the input folder will be ignored.

Initially, Document objects are instantiated from the input report(s). The text of the report is stored in the originalText attribute of each Document object. Next, a CollectionBuilder object is instantiated. The Document objects are passed to the splitSegment() and the tagPOS() methods of the CollectionBuilder object for preprocessing. These methods call functions in the TMI library to perform sentence boundary detection, tokenization, and part-of-speech tagging. After the preprocessing is complete, all Document objects are returned to the BPD_FE object's main procedure (from link 1 to link 11 in Figure 6.9).

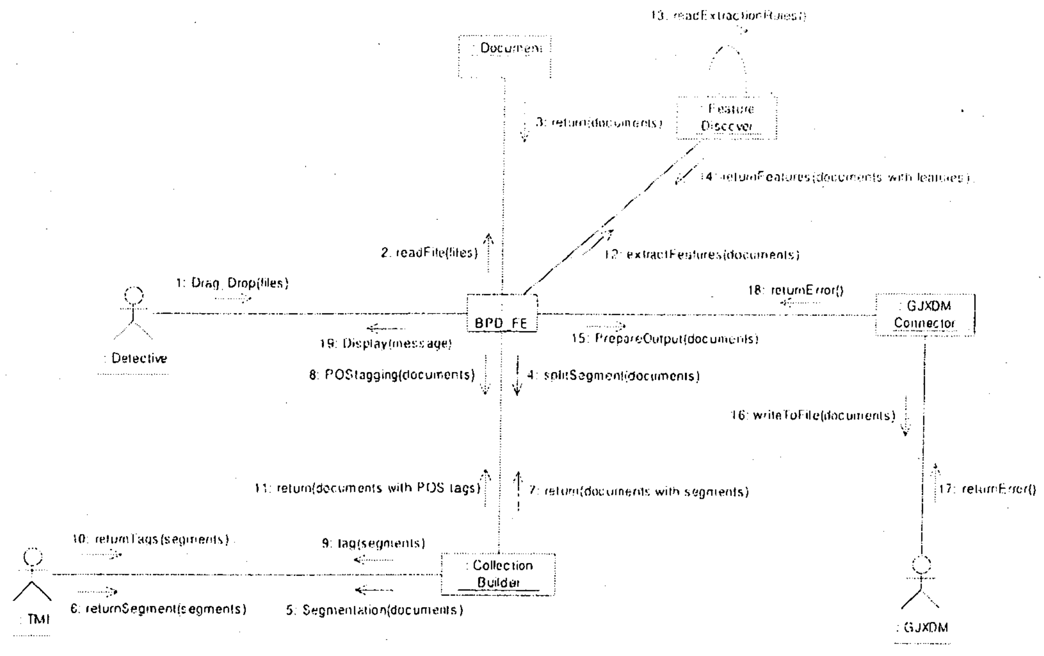

13: readExtractionRules()
: Document
3: return(documents)
: Feature Discover
14: returnFeatures(documents with features)
2. readFile(files)
12: extractFeatures(documents)
1: Drag_Drop(files)
18: returnError()
BPD_FE
: GJXDM Connector
19: Display(message)
15: PrepareOutput(documents)
: Detective
8: POStagging(documents)
4: splitSegment(documents)
16: writeToFile(documents)
11: return(documents with POS tags)
7: return(documents with segments)
17: returnError()
10: returnTags(segments)
9: tag(segments)
: Collection Builder
6: returnSegment(segments)
5: Segmentation(documents)
: TMI
: GJXDM

Figure 6-9: Collaboration diagram of "Drag and Drop Files to Extract Information"

In the next step, either a SerialFeatureExtractor or ParallelFeatureExtractor object is instantiated per the system configuration. The constructor of FeatureDiscover is invoked to load information extraction rules. Next, the Document objects are passed to the SerialFeatureExtractor object or the ParallelFeatureExtractor object, and the extractFeatures() method is invoked to extract features from all the Document objects. All features extracted are saved in the corresponding Document objects that are returned to the main procedure of the BPD_FE object (from link 12 to link 14 in Figure 6.9).

The final step is to store the extracted features in the BPD_IE database or a GJXDM XML file. A GJXDMConnector object or an ODBC object is instantiated in the main procedure of the BPD_FE object. The GJXDMConnector or ODBC object automatically accesses the corresponding data storage. All Document objects with features extracted are passed to the GJXDMConnector or ODBC object and the insertDocument() method is invoked to save extraction results. A notification message is then displayed to the user (from link 15 to link 19 in Figure 6.9).

6.2.3 Modus Operandi Search

To search on modus operandi, a Detective utilizes a user interface to search the features extracted from different reports. There are three methods to search modus operandi. The first is to fill out a search form manually. The most often used feature types are located at the top of the form, and there is a dropdown menu that corresponds to each feature type that contains the most common values of the particular feature type in question. An extended search form will support additional feature types and values.

The second search method is based on a search-engine interface. The detective inputs keywords in a search box. Like a search engine, the keywords can be input in any order. Optionally, keywords may be preceded by a feature type designator in order to narrow the scope of the search. For example, "Age:15" would narrow the scope of search for the number 15 to the Age feature type. Multiple featureType:keyword pairs are permitted within the same query.

The third method is to drag and drop an investigator's report onto the BPD_MO.exe icon. The system will automatically fill out the search form described in the first method using the FeatureDiscover class, which is discussed in Section 6.2.3.2. The Detective can manually refine the search form, and then perform the same kind of search employed in the first approach described above.

In all of these cases the system will return a list of summaries of existing investigative reports sorted according to their similarity with the query. The Detective can zoom in to a given report by double-clicking on the summary.

6.2.3.1 Preconditions

Features extracted from various investigative reports are stored in the BPD_IE database. These features, along with their context in the original reports, are available for Modus Operandi Search.

6.2.3.2 Class Overview

There are 18 classes in the class diagram of the Modus Operandi Search behavior (Figure 6.10): BPD_IE, BPD_MO, Document, DetectiveReport, PoliceIncidentReport, OldWordPerfectReport, Feature, Segment, Configuration, CollectionBuilder, FeatureDiscover, SerialFeatureExtractor, ParallelFeatureExtractor, Rule, Element, Storage, ODBC, and GJXDMConnector.

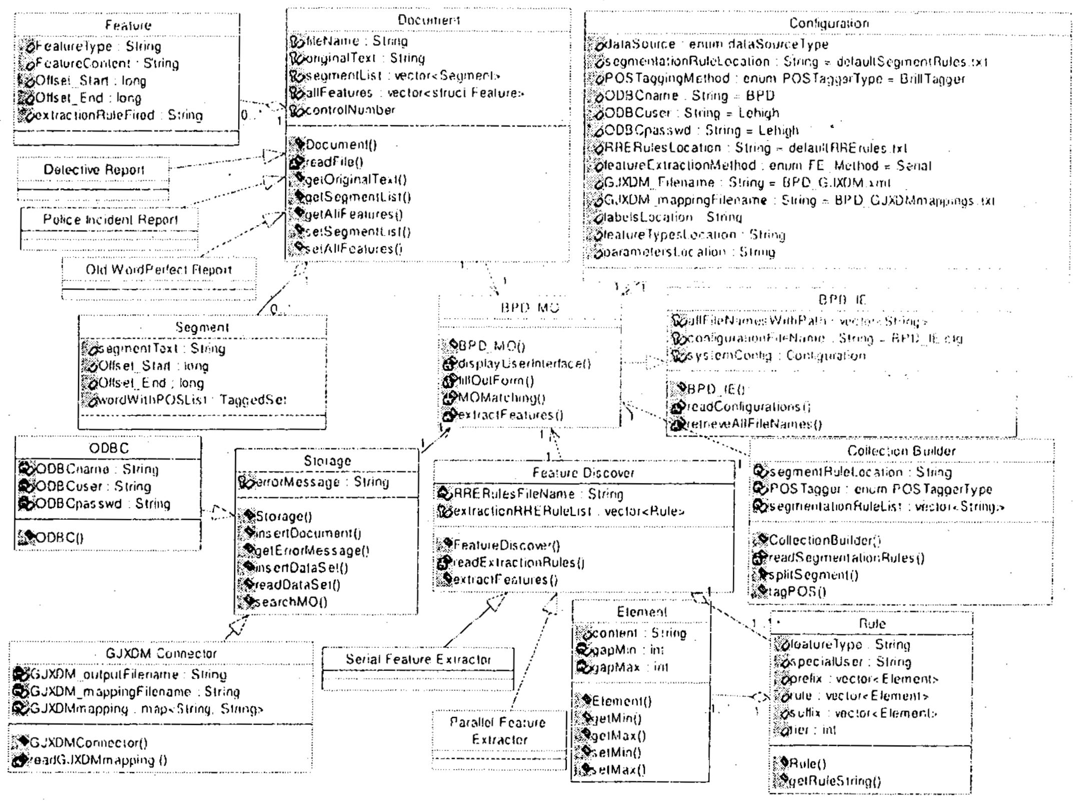


Figure 6-10: Class diagram of Modus Operandi Search

All classes except the BPD_MO have been described in previous sections. All methods related to Modus Operandi Search are included in BPD_MO. BPD_MO provides a graphical user interface to obtain input from the user. As described previously, it has an auto-fill functionality that uses the input report to fill out the search form automatically for the user. A fuzzy search method, MOMatching() in BPD_MO invokes the searchMO() method in the Storage class to complete the search. Search results are sorted and then displayed to the Detective using the system.

6.2.3.3 Interaction View

Figure 6.11 is the sequence diagram for the third method of modus operandi search. An investigator's report is dragged and dropped onto the BPD_MO.exe icon. Next, the system goes through the entire process of information extraction (described in Section 4.2.2) to extract features, which are used to fill out the modus operandi search form. Following this, the user will review the search form and can, at their option, modify some of the search fields. After the Detective clicks the search button on the search form, the search form is converted to an SQL search command that is used to search the BPD_IE database. The search results are documents with features. These documents are sorted and the summaries (feature lists) of the documents are displayed to the Detective to review. The Detective can also retrieve any original document in which they are interested.

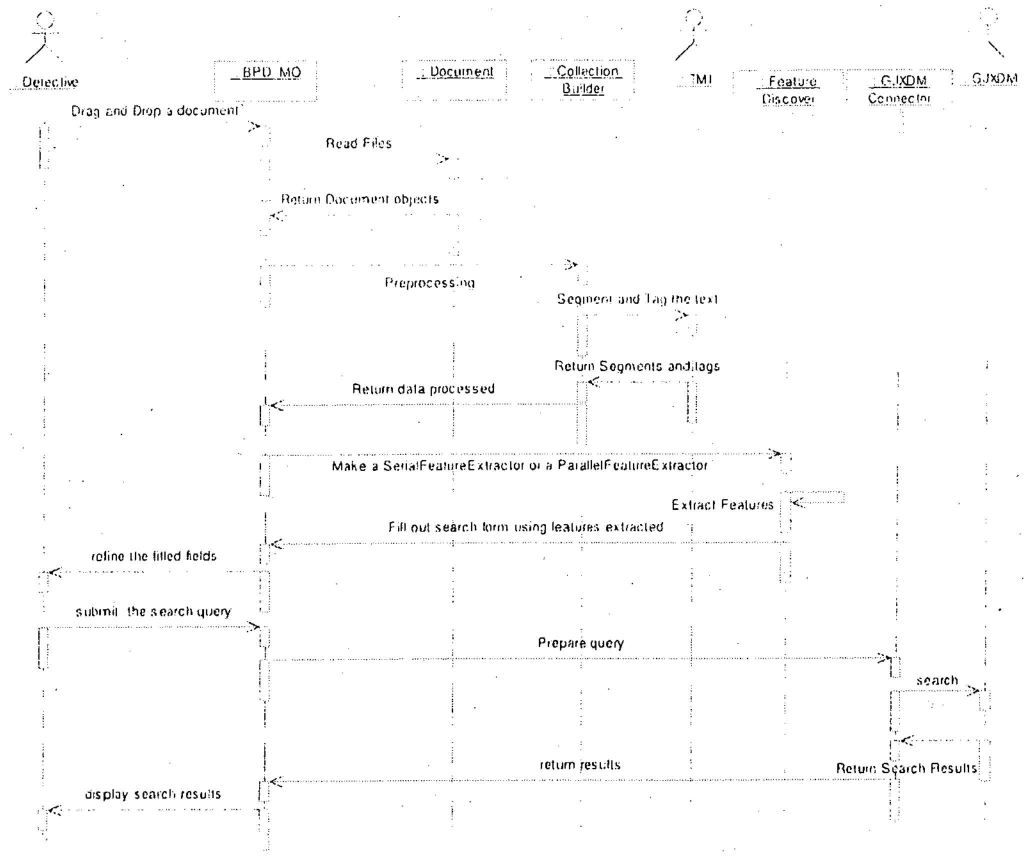


Figure 6-11: Sequence Diagram of the Modus Operandi Search

6.3 *Third Party Components*

6.3.1 Database

The database we selected for the system was MySQL, version 4.0.20d for Windows which is available at http://dev.mysql.com/downloads/mysql/4.0.html. One of the reasons MySQL was selected as the database is because it has a free software/open source GNU General Public License (commonly known as a "GPL"), which allows us to use this database without cost. (The license is located at http://www.mysql.com/company/legal/licensing/opensource-license.html.)

One of the conditions of the license states, "As long as you never distribute the MySQL Software in any way, you are free to use it for powering your application." Therefore, we neither distribute nor modify any version of MySQL. MySQL server must be installed by the end user of our software system on their file server.

The most important reason for the selection of MySQL is that it supports user defined functions through ODBC connections. After initially developing our system with Microsoft Access, we found that, although Access supports user defined modules, they cannot be called through ODBC connections. Therefore, we chose MySQL because both ODBC connection and user defined functions are fundamental parts of our system design and are supported by MySQL.

6.3.2 Regular Expression Library

Regular expression matching is central to the implementation of our system because our algorithm theoretically is based on reduced regular expressions. Without a regular expression library, the system will not function. Fortunately, a free regular expression library for Visual C++ .NET is available, "Regular Expression Component Library for Visual C 7.1." This library is published by Tropic Software East, Inc. under a free license and is available at http://www.tropicsoft.com/Components/RegularExpression/default.htm. This regular expression library supports most Perl regular expression operators. Since our algorithm was originally designed and implemented using Perl, this regular expression library was particularly useful to us.

However, this regular expression library only supports ASCII characters. When a file containing non-ASCII characters such as UNICODE is loaded to the system, our program automatically removes these characters before operating any regular expression upon them.

6.3.3 TMI Library

We also employed functionality contained in the Text Mining Infrastructure (TMI [43]). The TMI is a software system for research in Textual Data Mining that has been developed over the past several years under the direction of Dr. William M. Pottenger. The TMI incorporates both existing and new capabilities in a reusable framework conducive to developing new tools and components for use in text/data mining research and development. It provides core text mining capabilities for a wide variety of applications via a flexible componentized architecture. The TMI adheres to strict guidelines that allow it to run in a wide range of processing environments. As a result, it accommodates the volume of computing and diversity of research occurring in text/data mining.

A unique capability of the TMI is support for optimization. This facilitates text/data mining research by automating the search for optimal parameters in text/data mining algorithms. Another key capability is support for parallelism in both shared-memory and distributed environments using OpenMP and MPI, respectively. The TMI has been licensed by over 125 researchers worldwide, both academic and industrial, and is available online at http://hddi.cse.lehigh.edu.

The TMI is used in our current information extraction system, as information extraction is an important component of text mining. When our modus operandi search system makes use of basic text mining functionality, it employs the TMI. For example, the TMI part of speech tagger is used to assign part of speech tags to words in the investigators' reports. Use of the TMI simplified our system design and implementation.

6.3.4 Other Components

Throughout the development process, several components were downloaded and modified from a variety of sources. "File/Folder Watcher Wrapper" written by Brad Vincent, "DriveComboBox" by Odah, "In-place editing of ListView subitems" by mav.northwind, and "Masked C# TextBox Control" by Jibin Pan were all obtained from http://www.codeproject.com/. "Directory Picker" by Michael Gold was also used from www.c-sharpcorner.com. The Windows version of Antiword from http://www.informatik.uni-frankfurt.de/~markus/antiword/ is also used in our system.

6.3.5 Adding Third Party Programs to the BPD_IE System

If you wish to add a third party program to the BPD_IE System and you have the source code including both .cpp and .h files, you can add them into one or more of the existing projects such as BPD_FE or BPD_MO, and then recompile the project(s).

If you have a .dll or .exe generated under the .NET framework, copy the .dll or .exe file to ./bin directory. Next, add the file under the tab "References" of the current project. Finally, use the correct namespace, and instantiate objects as needed in your code.

For example, if one wanted to use the "FileDownloader" function of "DownloadLibrary", one would need to do the followings:

- Download the source code from http://www.codeproject.com/csharp/CoolDownloader.asp
- Compile and generate "DownloaderLibrary.dll" if necessary.
- Copy and paste DownloaderLibrary.dll to "[TARGETDIR]\bin\"
- Open "BPD_IE.sln"
- Add "DownloaderLibrary.dll" to "BPD_FE"->References
- At the beginning, e.g., of "Form1.h", add "using namespace Batte::CodeProject::Download;"
- Declare the object "FileDownloader * myDownloader = new FileDownloader;"
- Code up a call to the method "myDownloader = Download(url ,fileTarget,cancelEvent);"
- Deal with exceptions

- Compile "BPD_FE" project
- Add the "DownloaderLibrary.dll" file to your setup project
- Recompile the setup project

That's it! You're done.

6.4 *Deployment of the System*

After the alpha version of the software was implemented, we installed it at the Bethlehem Police Department (BPD) with the help of Technician Jonathan Palsi. We also conducted two training sessions to teach police detectives and investigators how to use the system. We are now working with BPD investigators to pilot test the modus operandi search in their regular routine. Meetings are held on a regular basis to discuss the use of the system, to install updates and bug fixes, and to identify ways to improve the usefulness of the system. Currently, BPD investigators are loading tens of thousands of investigative reports into the MySQL database.

6.4.1 System Installation

To distribute the system to users, a setup bootstrapper and an installer (BPD_IE) were created. The setup bootstrapper is a shell that assembles different installers into one single setup executable, Setup.exe. We downloaded the .NET Framework Version 1.1 Setup.exe Bootstrapper Sample Source Code from http://go.microsoft.com/fwlink/?linkid=16824 and modified the bootstrapper to create our own installer shell. The original bootstrapper can only install the .NET Framework and custom programs. Our BPD_IE applications, however, require the MySQL ODBC driver to be installed on the investigator's computer. To make the installation as simple as possible for the end user, we decided to write our own install shell based on the bootstrapper that included the "MySQL ODBC" driver installer.

This install shell first detects whether .NET framework exists on the target computer. If not, it will install .NET framework 1.1. Following this, MySQL ODBC version 3.51.9 is installed. Next, the BPD_IE applications are installed, including BPD_FE.exe and BPD_MO.exe. Finally, the "ODBC data source administrator" is automatically called to allow users to configure their ODBC connection.

A setup project was created using MS Visual Studio .NET. This project was used to generate the BPD_IE installer, which installs the BPD_IE applications on the target computer. The BPD_IE installer copies files, modifies registry keys, and creates shortcuts during installation.

The installation details are described in "Installation Instructions for the BPD Information Extraction System", which is included in Appendix A in this report. The entire installation package, including either the binary executables or the full source for the project as well as project documentation, is available at http://hddi.cse.lehigh.edu.

After a server was deployed at the BPD, we prepared live data for use in training the system. About 2000 investigators' reports were extracted and saved on the server for users to query. The MySQL database performance was optimized to bring the query response time down to less than one second for each user query.

6.4.2 Officer Training

Appendix B, the training tutorial used with the BPD, provides step-by-step instructions on how to launch the system and perform all the necessary user functions. The tutorial provides instruction on the user interface and shows users how to import files, extract features, send data to the server, and retrieve reports. The tutorial also includes two sample police reports used in training scenarios, as well as information on the setup and use of the system.

As noted, we held training sessions for investigators and also visit the BPD on a regular basis to answer any questions, receive feedback, and discuss how the system is being used so that we can properly evaluate and improve it.

6.5 *Using the Tool to Solve Problems*

Our tool is designed to match unsolved cases to solved cases, so that detectives can identify suspects easily. In this section we highlight some of the preliminary successes that have been achieved. As the BPD_IE system continues to be utilized by the Bethlehem Police Department, situations like those illustrated below can be expected to occur with increasing frequency. The benefits provided by our system are illustrated nicely in the two actual examples from live BPD data detailed in sections 6.5.1 and 6.5.2 below.

In addition, the BPD_IE system can link affidavits of probable cause, crime reports and supplements using search terms such as the criminal's or suspect's name, residence, date of birth, etc. This can aid detectives in identifying documents related to a specific criminal or suspect very quickly. Since some of the documents contain inaccurate information, we can also use the linked documents to correct this incorrect information. For example, if a crime report has an incorrect control number and the correct number is found in the arrest record, linking these reports together will allow this error to be corrected quite easily.

6.5.1 Imposter Burglaries

One of the investigators at the BPD provided a summary of imposter burglaries that had been occurring recently within the City of Bethlehem. A modus operandi (MO) from this list is "after diverting the people's attention, suspects break into the home to burglarize it." Based on this MO, we employed the following terms to search our BPD_IE database: "Water department", "yard", "gas", "city work", "Upstairs", "Basement", "Bathroom", "Kitchen", "Living room", "Burglary", "door", and "window."

Following this, we set the edit distance value to two to allow for misspelled terms to be considered as matches. This search focused on investigator's supplement reports. The BPD_IE system identified the following three cases, which are very similar to the MO detailed by the officer.

Case 1 matches the terms "upstairs", "basement", "burglary", "gas", and "living room." The case involved a victim who smelled gas while sleeping. The victim went down to the basement and found a natural gas container leaking gas. When the victim returned upstairs, the victim discovered that the house has been burglarized."

Case 2 matches the terms "basement", "window", "burglary", and "upstairs". In the case, some kids want to rob a friend's house. They went to the friend's home to talk to him, and then invited him out of his home. Meanwhile, another kid burglarized the home.

Case 3 matches terms "bathroom", "burglary", and "kitchen". This case describes some workers that came to help the victim to repair her bathroom. After the work was completed, she discovered her home had been burglarized.

These cases are good matches to the initial MO described above. By alerting investigative detectives to these similar cases they are able to delve deeper into these previous, solved cases and search for clues as well as identify suspects that are repeat offenders.

6.5.2 Serial Car Theft

A second investigator in the BPD investigations unit provided us a set of solved crimes connected to a single serial auto thief. At one point during commission of an auto theft, the criminal was apprehended. After fingerprinting, the detective discovered six other crimes committed by this individual. After analyzing the cases, the investigator found they had a similar MO: a broken side or wing window of the vehicle. This case formed an ideal method of testing our system because we had complete information about the crimes.

To test our MO tool, we first manually entered the cases linked to the solved case through fingerprint analysis into the BPD_IE database. Next, we used the solved case as our query. This produced the following search terms: "incident code" with value range 600 to 699; "incident description" with value "theft"; "type property" with value "broken window", "side window", and "wing window"; "item stolen" value range $200 to $300; and also indicated the presence of a vehicle. With edit distance set to two, we discovered four of the six unsolved related cases in our top five results. This indicates that the BPD_IE system is functioning properly. It is also worth noting that the use of the system would have significantly reduced the time required to solve the crime.

7 Conclusion

In this final report, we have provided extensive background on information extraction (IE) and reviewed several commercial IE systems. Additionally, we have presented a novel semi-supervised, active learning information extraction algorithm developed as part of this project, the Reduced Regular Expression Discovery algorithm. As part of our analysis of the algorithm, we analyzed several scoring metrics, performed a precise time complexity analysis, and reported extensive performance results.

The software system design and implementation of a deployed IE system, the BPD_IE system, has also been presented in this report. This includes an extensive description of the feature extraction and modus operandi subsystems, including the deployment, testing, and training work conducted with the Bethlehem Police Department. A full version of the software has been made available on http://hddi.cse.lehigh.edu, and continue to collaborate with the BPD on the project to evaluate the utility of modus operandi search.

8 Future Work

8.1 *Beam Search*

Currently, our algorithm is a greedy algorithm, which means the algorithm selects the best RRE rule learned from each learning iteration for the next step. This hill climbing approach can become stuck on a local optimal value. Occasionally, this local optimal value produces poor test performance.

An alternative of the greedy algorithm is to use beam search, which can find a better local optimal value than the simple greedy search. A beam search keeps *n* number of best RRE rules learned in each learning iteration for the next step. When *n* equals 1, it is the simple greedy search. But usually *n* is greater than 1. Using the training and testing datasets, an optimal value for *n* can be reached for an application.

We have not designed a beam algorithm for use within our semi-supervised active learning algorithm. Implementing our algorithm with the beam algorithm as a component may lead to even better results.

8.2 *Unsupervised Learning Approach*

Even though semi-supervised learning and active learning can reduce knowledge engineering cost of IE systems, an unsupervised approach is always the ideal solution to the knowledge engineering problem.

If one could define a similarity function to measure the candidate segments generated in our active learning approach (Section 4.5) to the seed, we believe that we can make the active learning fully automatic. In short, we want to replace the active selection of candidate segments with a similarity function plus a threshold. However, such a similarity function usually is difficult to identify and domain specific.

8.3 *Word-based Edit Distance*

Currently, the BPD_IE system uses edit distance for fuzzy string matching. While this method works well for some cases, it does not perform well in cases having long description fields or fields having differently-ordered words. To use an example from police reports, "broken side window" and "passenger side window broken" appear quite different when they are compared using character-based edit distance. In fact, however, they are similar.

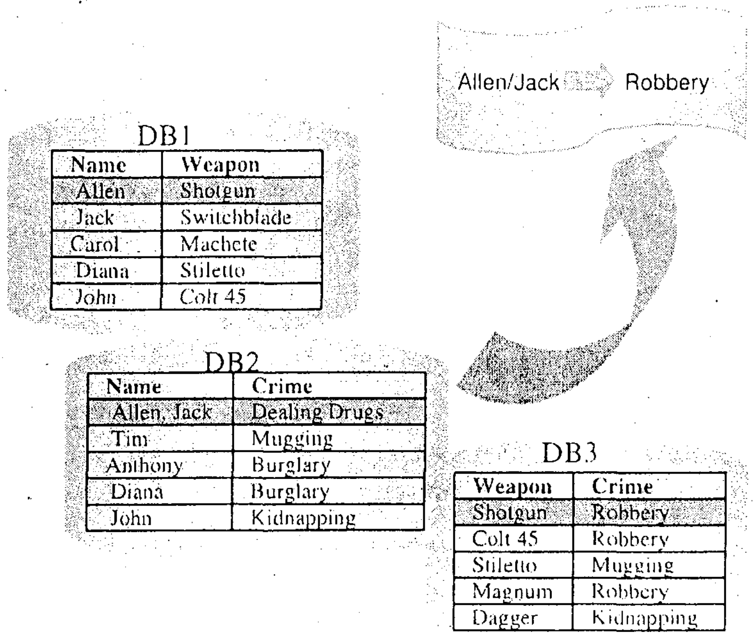
We are planning to modify the character based edit distance to a word based edit distance. If this can be accomplished, instances like those represented above will be able to be matched more accurately with similar fields.

8.4 *Distributed Systems*

The burgeoning amount of textual data in distributed sources combined with the obstacles involved in creating and maintaining central repositories motivates the need for effective distributed information extraction and mining techniques. One example of this is in the criminal justice domain. For instance, there are more than 1,260 police jurisdictions in the Commonwealth of Pennsylvania alone. As was made strikingly clear in the aftermath of the terrorist attack on September 11, different kinds of records on a given individual may exist in different databases -- a type of data fragmentation. In fact, the United States Department of Homeland Security (DHS) recognizes that the proliferation of databases and schemas involving fragmented data poses a challenge to information sharing. As a result, the DHS is promoting a "System of Systems" approach that is based initially on the creation of standards for interoperability and communication in areas where standards are currently lacking [44]. Indeed, efforts are underway to establish standards in schema integration (e.g., OWL [45], GJXDM [46]). Nonetheless, even should there be widespread acceptance of such standards, the ability to integrate schemas automatically is still an open research issue [47, 48, 49, 50].

An important related issue is the fact that current Association Rule Mining (ARM) algorithms for mining distributed data are capable of mining data only when the global schema across all databases is known [51, 52, 53, 54, 55]. Furthermore, existing privacy-preserving techniques for distributed ARM *do not function in the absence of knowledge of the global schema*, but, rather assume that that data integration and record linkage has already been done. In the case of information extracted from distributed textual data, however, no preexisting global schema is available. This is due to the fact that the entities extracted may differ between textual documents at the same or different locations. In short, schemas of textual entities are highly fluid in nature because new input text can contain previously unseen entities. As a result, a fixed global schema cannot be assumed and existing distributed ARM algorithms (whether privacy-preserving or not) cannot be employed.

In addition, these same existing distributed ARM algorithms are able to mine association rules only from data that has been either vertically or horizontally fragmented. It is our contention that the restriction to such data sources is unnecessary, and that useful rules can be mined from diverse databases with different local schemas as long as records can be linked, for example, via a unique key. Many interesting applications emerge if one considers this approach, which we term higher-order distributed association rule mining. Higher-order implies that rules may be inferred between items that do not occur in the same local database. In other words, rules can be inferred based on items (entities) that may never occur together in any record in any of the distributed databases being mined. Figure 8.1 exemplifies the discovery of such rules in a law enforcement context.



DB1

Name	Weapon
Allen	Shotgun
Jack	Switchblade
Carol	Machete
Diana	Stiletto
John	Colt 45

DB2

Name	Crime
Allen, Jack	Dealing Drugs
Tim	Mugging
Anthony	Burglary
Diana	Burglary
John	Kidnapping

DB3

Weapon	Crime
Shotgun	Robbery
Colt 45	Robbery
Stiletto	Mugging
Magnum	Robbery
Dagger	Kidnapping

Figure 8-1: Higher-Order Associations in Law Enforcement

In database DB1 in Figure 8.1, the first record indicates that a criminal suspect Allen was caught with a shotgun. A different jurisdiction's database DB2 contains a record showing that Allen and Jack are involved in dealing drugs. An investigator may surmise that Jack might have some connection with the shotgun also. In a third jurisdiction (in DB3) there is an unsolved robbery case where the same shotgun was used. Since Allen and Jack both have some connection with this shotgun, the robbery could have been committed by either of them, or both. This is precisely the kind of information that investigators need to identify suspects and is a significant challenge facing our law enforcement partners.

To address this challenge, a distributed higher-order text mining framework that requires neither knowledge of the global schema nor schema integration as a precursor to mining rules is necessary. This will build on the work that has already been completed on this project. Furthermore, the framework will be able to mine distributed data in a hybrid form that is neither horizontally nor vertically fragmented but a mixture of both. The framework, termed D-HOTM, extracts entities and discovers rules based on higher-order associations between entities in records linked by a common key.

D-HOTM requires two components: entity extraction and distributed association rule mining. The entity extraction is based on information extraction techniques developed as part of this project. The rules learned are applied to automatically extract entities from textual data that describe, for example, criminal modus operandi. The entities extracted are stored in local relational databases, which are mined using the D-HOTM distributed association rule mining algorithm.

This work will continue the work we have begun with the Bethlehem Police Department by expanding the work presented here to include up to 31 other jurisdictions in Northampton County, Pennsylvania.

D-HOTM is a hybrid approach that combines information extraction and distributed data mining. Employing a novel information extraction technique, we will be able to extract meaningful entities from unstructured text in a distributed environment. The information extracted is stored in local databases and a mapping function is applied to identify globally unique keys. Based on the extracted information, a novel Distributed Higher Order (DiHO) association rule mining (ARM) algorithm will be applied to discover higher-order associations between items (i.e., entities) in records fragmented across the distributed databases.

9 Acknowledgements

This project was supported by grant number 2003-IJ-CX-K003 awarded by the National Institute of Justice, Office of Justice Programs. US Department of Justice. Points of view in this document are those of the authors and do not necessarily represent the official position or policies of the US Department of Justice.

We are grateful for the help of the detectives and investigators, especially Matthew Palmer. at the Bethlehem Police Department as well as Deputy Commissioner Randy Miller, Lieutenant Joe Kimock, Lieutenant Stu Bedics, Detective Matt Palmer and Technician Jonathan Palsi. Additionally, we would like to thank Faisal M. Khan, Lars E. Holzman, Shenzhi Li, Chris Janneck and other co-workers, family members and friends. We also gratefully acknowledge the continuing help of our Lord and Savior, Yeshua the Messiah (Jesus the Christ) in our lives and work. Amen.

10 References

[1] Christopher D. Manning and Hinrich Schütze. "Foundations of Statistical Natural Language Processing". MIT Press 2000.

[2] Van Rijsbergen. "Information Retrieval", Second Edition. Butterworths, London, 1979.

[3] Melnik, S., H. Garcia-Molina, E. Rahm. "Similarity Flooding: A Versatile Graph Matching Algorithm". ICDE 2002.

[4] Ian H. Witten, et al. "Data Mining : Practical Machine Learning Tools and Techniques". Morgan Kaufmann, 1999 (1558605525).

[5] Chia-Hsiang Chang. "From Regular Expressions to DFAs Using Compressed NFAs". PhD Thesis. New York University, October 1992.

[6] M. Crochemore and C. Hancart. "Automata for Matching Patterns" In Handbook of Formal Languages. G. Rozenberg and A. Salomaa, Eds., 2, 399-462, Springer Verlag. 1997.

[7] Tianhao Wu and William M. Pottenger. "A Semi-Supervised Active Learning Algorithm for Information Extraction from Textual Data". Journal of the American Society for Information Science and Technology. JASIST, 2004.

[8] G. Davies and S. Bowsher. "Algorithms for Pattern Matching". Software--Practice and Experience, 16(6):575-601, June 1986.

[9] Reynar, J.C. & Ratnaparkhi, A. "A Maximum Entropy Approach to Identifying Sentence Boundaries". Proceedings of the Fifth Conference on Applied Natural Language Processing, Washington, D.C., 1997.

[10] Eric Brill. "Transformation-Based Error Driven Learning and Natural Language Processing: A Case Study in Part-of-Speech Tagging". Computational Linguistics 21(94): 543-566. 1995.

[11] Grishman, Ralph 1997, "Information Extraction: Techniques and Challenges". Lecture Notes in Computer Science. Vol. 1299. 1997.

[12] A.F. Smeaton 1998. "Retrieving Images of Scanned Text Documents". Proceedings of the Optical Engineering Society of Ireland & Irish Machine Vision and Image Processing Joint Conference, (OESI-IMVIP'98), NUI Maynooth, September 1998, D. Vernon (Ed.), pp 271-286

[13] Sergey Brin and Lawrence Page. "The Anatomy of a Large-Scale Hypertextual Web Search Engine". In 7th International World Wide Web Conference

[14] Adams, Katherine 2001. "The Web as Database: New Extraction Technologies and Content Management." Online. http://www.onlineinc.com/onlinemag/OL2001/adams3_01.html.

[15] AeroText. "Open Source Data Exploitation." Online. http://www.visualanalytics.com/PressRelease/2000-12-05-prnews.html December 5, 2000. Accessed July 28, 2005.

[16] Sugato Basu, Arindam Banerjee and Raymond J. Mooney. "Semi-supervised Clustering by Seeding". In the Proceedings of the Nineteenth International Conference on Machine Learning (ICML-2002). Online. http://www.cs.utexas.edu/users/ml/papers/semi-icml-submitted-02.pdf

[17] C.A.Thompson, M.E.Cali . and R.J.Mooney. "Active Learning For Natural Language Parsing and Information Extraction". In I.Bratko and S.Dzeroski, editors, Proceedings of the Sixteenth International Machine Learning Conference(ICML-99), pages 406-414, Bled, Slovenia, 1999. Online. http://citeseer.nj.nec.com/cache/papers/cs/4878/ftp:zSzzSzftp.cs.utexas.eduzSzpubzSzmooneyzSzpaperszSzactive-nll-ml99.pdf/thompson99active.pdf

[18] Cohn, D., Atlas, L., & Ladner, R.1994. "Improving Generalization With Active Learning". Machine Learning, 15(2).201-221. Online. http://citeseer.nj.nec.com/cache/papers/cs/1895/http:zSzzSzwww.ai.mit.eduzSzpeoplezSzcohnzSzpsychezSzselsampling.pdf/cohn92improving.pdf

[19] Paliouras, G., V. Karkaletsis, C. Papatheodorou and C. Spyropoulos 1999. "Exploiting Learning Techniques for the Acquisition of User Stereotypes and Communities". Proceedings of the Seventh International Conference. Wien, New York: Springer-Verlag. pp. 169-178. Online. http://www.cs.usask.ca/UM99/Proc/karkaletsis.pdf

[20] Regev, Y., Finkelstein-Landau, M., and Feldman R. (2002). Rule-based Extraction of Experimental Evidence in the Biomedical Domain – the KDD Cup 2002 (Task 1). SIGKDD Explorations 4(2):90-92.

[21] NetOwl®. Online. http://www.NetOwl.com/index.html/

[22] ClearForest. Online. http://www.clearforest.com/ Accessed July 28, 2005.

[23] IBM Intelligent Miner for Text. Online. http://www-3.ibm.com/software/data/iminer/fort

[24] NCIC (2000) Code Manual. Online. http://www.leds.state.or.us/resources/ncic_2000/ncic_2000_code_manual.pdf December 2000. Accessed July 28, 2005.

[25] Tianhao Wu and William M. Pottenger. "A Semi-Supervised Active Learning Algorithm for Information Extraction from Textual Data". Journal of the American Society for Information Science and Technology, JASIST, 2004.

[26] Isoquest, Inc: Description of the NetOwl™ Extractor System as Used for MUC-7. In Seventh Message Understanding Conference (MUC-7): Proceedings of a Conference held in Fairfax, Virginia, 29 April – 1 May, 1998.

[27] MUC-7. The Proceedings of the Seventh Message Understanding Conference. Morgan Kaufmann. 1998

[28] Hauck, R. V., Atabakhsh, H., Ongvasith, P., Gupta, H., and Chen, H. (2002). "Using COPLINK® to Analyze Criminal-Justice Data." IEEE Computer, March 2002.

[29] Michael Chau, Jennifer J. Xu, Hsinchun Chen, "Extracting Meaningful Entities from Police Narrative Reports". In Proceedings of the National Conference for Digital Government Research. Los Angeles, California, May 19-22, (2002). Online. http://ai.bpa.arizona.edu/~mchau/papers/COPLINKEE.pdf

[30] William J. Black and Argyrios Vasilakopoulos, Language-Independent Named Entity Classification by Modified Transformation-Based Learning and by Decision Tree Induction. In: Proceedings of CoNLL-2002, Taipei, Taiwan, 2002, pp. 159-162.

[31] Kristie Seymore, Andrew McCallum, and Ronald Rosenfeld. 1999. Learning Hidden Markov Model Structure for Information Extraction. AAAI Workshop on Machine Learning for Information Extraction, p. 37-42.

[32] Hai Leong Chieu , Hwee Tou Ng, A Maximum Entropy Approach to Information Extraction from Semi-structured and Free Text, Eighteenth national conference on Artificial intelligence, p.786-791, July 28-August 01, 2002, Edmonton, Alberta, Canada

[33] Paul McNamee, and James Mayfield. 2002. Entity Extraction without Language-specific Resources. The 6th Conference on Natural Language Learning.

[34] Soderland S. (1999). Learning Information Extraction Rules for SemiStructured and Free Text, Machine Learning: Special Issue on Natural Language Learning, 34, 233-272. Online. http://citeseer.ist.psu.edu/cache/papers/cs/1943/http:zSzzSzwww.cs.washington.eduzSzhomeszSzsoderlanz SzWHISK.pdf/soderland99learning.pdf

[35] Kushmerick, N. (2002)Finite-state approaches to Web information extraction. In Proc. 3rd Summer Convention on Information Extraction (Rome). Online. http://www.cs.ucd.ie/staff/nick/home/research/download/kushmerick-scie2002.pdf

[36] Kushmerick, N. & Thomas, B. (2003) Adaptive Information Extraction: Core Technologies for Information Agents. In Intelligent Information Agents R&D in Europe: An AgentLink perspective (Klusch, Bergamaschi, Edwards & Petta, eds.). Lecture Notes in Computer Science 2586, Springer. Online. http://www.cs.ucd.ie/staff/nick/home/research/download/kushmerick-agentlink-chapter-2003.pdf

[37] Kameyama, M. 1997. Information Extraction Across Linguistic Barriers. In AAAI Symposium on Cross-Language Text and Speech Retrieval. American Association for Artificial Intelligence.

[38] Machine Learning for Information Extraction. In conjunction with 14th European Conference on Artificial Intelligence. August 21, 2000. Online. .http://www.dcs.shef.ac.uk/~fabio/ecai-workshop.html Accessed July 28, 2005

[39] Ciravegna, Dr. Fabio (2001) Adaptive Information Extraction from Text by Rule Induction and Generalisation. In Proceedings 17th International Joint Conference on Artificial Intelligence (IJCAI 2001), Seattle. Online. http://eprints.aktors.org/archive/00000118/01/IJCAI01.pdf

[40] Automation in Information Extraction and Integration. Online. http://www.cs.ust.hk/vldb2002/program-info/tutorial-slides/T1sarawagi.pdf

[41] Tianhao Wu and William M. Pottenger. "A Supervised Learning Algorithm for Information Extraction from Textual Data". In the proceeding of the workshop on Text Mining, Third SIAM International Conference on Data Mining, May (2003).

[42] Levenshtein, V.I. Binary Codes Capable of Correcting Deletions, Insertions and Reversals. Sov. Phys. Dokl., 6:707-710, 1966

[43] TMI: Text Mining Infrastructure Online. http://hddi.cse.lehigh.edu/ Accessed July 28, 2005.

[44] Boyd D., Director of the Department of Homeland Security's new Office of Interoperability and Compatibility, in a presentation at the Technologies for Public Safety in Critical Incident Response Conference and Exposition 2004, New Orleans, LA, September.

[45] Dean M. and Schreiber G. OWL Web Ontology Language Reference. Editors, W3C Recommendation, 10 February 2004. Online. http://www.w3.org/TR/2004/REC-owl-ref-20040210/ February 10, 2004. Accessed Nov. 17, 2004.

[46] GJXDM. Global Justice XML Data Model. Online. http://www.it.ojp.gov/gjxdm Accessed Nov. 17, 2004.

[47] Genesereth M. R., Keller A. M., and Duschka O. M. Infomaster: An Information Integration System. In Proceedings of the ACM SIGMOD Conference, May 1997.

[48] Draper D., Halevy A. Y., Weld D. S.. The Nimble XML Data Integration System. Proceedings of the 17th International Conference on Data Engineering, p.155-160, April 02-06, 2001.

[49] Papakonstantinou Y. and Vassalos V. Architecture and Implementation of an Xquery-based Information Integration Platform. IEEE Data Engineering Bulletin, vol 25, n. 1, pg 18-26, 2002.

[50] Rahm E., Bernstein P.A. A Survey of Approaches to Automatic Schema Matching. VLDB J. 10:4 (2001), pp. 334-350.

[51] Ashrafi M. Z., Taniar D. and Smith K. ODAM: an Optimized Distributed Association Rule Mining Algorithm. IEEE Distributed Systems, vol. 5, No 3, March 2004.

[52] Cheung D. W., Han J., Ng VAT., Fu AWE. and Fu Y. A Fast Distributed Algorithm for Mining Association Rules. Proc. Parallel and Distributed Information Systems, IEEE CS Press, 1996, pp. 31-42.

[53] Vaidya J. and Clifton C. Privacy Preserving Association Rule Mining in Vertically Partitioned Data, Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, July 23-26, 2002, Edmonton, Alberta, Canada.

[54] McConnell S. and Skillicorn D. B. Building Predictors from Vertically Distributed Data, Proceedings of the 2004 Conference of the Centre for Advanced Studies Conference on Collaborative Research, p.150-162, October 04-07, 2004, Markham, Ontario, Canada

[55] Evfimievski A., Srikant R., Agrawal R. and Gehrke J. Privacy Preserving Mining of Association Rules. SIGKDD'02, Edmonton, Alberta, Canada, 2002.

[56] Childs, L.C., and Weir, C.E. (2002). Drug Discovery through Information Extraction Technology. 2002 NIH Biomedical Computing Interest Group (BCIG) seminar.

11 Appendices

Appendix A: Installation Instructions for the BPD Information Extraction System

System Requirements

Operating System

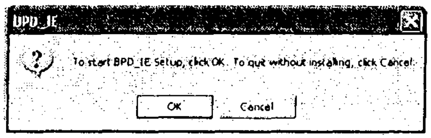
The system and installation files are tested only under Windows XP. Other Windows operating systems have not been tested, and successful installation and use of the system is not guaranteed on any operating system except XP.

Minimum Hardware Requirements

- Pentium III processor
- 256 MB RAM
- CD-ROM or USB port
- Keyboard and Mouse
- Internet or Intranet connection

Installing the Client System

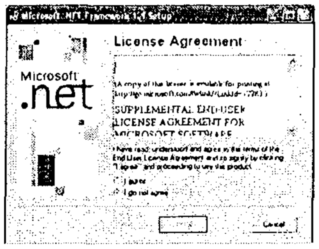
For a full installation, click "Setup.exe" to start the installation process. (This will install the .NET framework, MySQL ODBC, and the BPD_IE system.) For installation of the BPD_IE system only, click "Setup.msi" to start the installation process. Click "OK" to continue the installation after you see the following pop-up window:


To start BPD_IE Setup, click OK. To quit without installing, click Cancel.

.NET Framework

The Setup.exe program will automatically check whether the .NET framework (which is required to run BPD_FE.exe and BPD_MO.exe) has been installed on the target computer system. If the .NET framework has not been installed on the target computer, Setup.exe will install the .NET framework 1.1 English version.

When you see the following window, please follow the instructions to complete the .NET framework installation.



Microsoft .NET Framework 1.1 Setup

License Agreement

[illegible]

SUPPLEMENTAL END-USER LICENSE AGREEMENT FOR MICROSOFT SOFTWARE

I have read, understood and agree to the terms of the End User License Agreement and so signify by clicking "I agree" and proceeding to use the product.

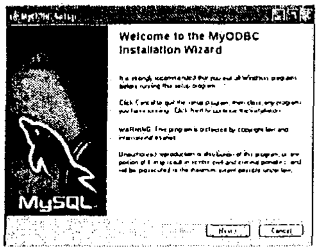
I agree

I do not agree

MySQL ODBC Driver

After the .NET framework has been installed, Setup.exe will launch a MySQL ODBC driver installer, which will install MySQL ODBC version 3.51.9 for Windows on the target computer. If the target computer has an installed MySQL ODBC driver, you can simply skip this step by clicking the "Cancel" button. The "Cancel" button affects the MySQL ODBC driver installation only. Please do not select the "Remove" option.

When you see the following window, please follow the instructions to finish the MySQL ODBC driver installation.



MyODBC Setup

Welcome to the MyODBC Installation Wizard

[illegible]

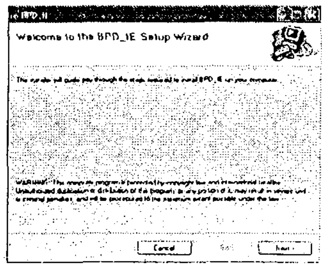
[illegible]

WARNING: This program is protected by copyright law and international treaties.

[illegible]

BPD_IE

Setup.exe will install the BPD_IE system after the MySQL ODBC driver installer is finished (or cancelled). When you see the following window, please follow the instructions to finish the BPD_IE system installation.



Welcome to the BPD_IE Setup Wizard

The wizard will guide you through the steps required to install BPD_IE on your computer.

WARNING: This computer program is protected by copyright law and international treaties. Unauthorized duplication or distribution of this program, or any portion of it, may result in severe civil or criminal penalties, and will be prosecuted to the maximum extent possible under the law.

This installation includes the following steps.

Copy files: Files needed to run "BPD_FE.exe" and "BPD_MO.exe" are copied to the "[ApplicationFolder]\bin" folder of the target computer. In addition, some support files are copied to the "C:\res" folder. Please do not delete anything in the "C:\res" folder. Otherwise, the BPD_IE system will not function properly.

Edit Registry Keys: The program requires that an environment variable be set. Therefore, the installer modifies the following key in registry during installation:

```
HKEY_LOCAL_MACHINE
        →SYSTEM
            →CurrentControlSet
                    →Control
                            →Session Manager
                                    →Environment
                                            →ANTIWORDHOME= "c:\res\format"
```

Note: You must restart your computer after the installation to make the key visible to application programs.

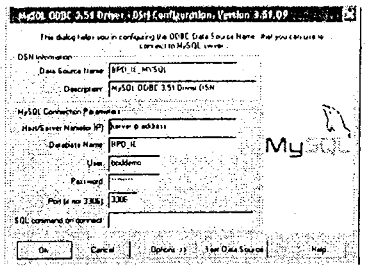
Create Shortcuts: Two shortcuts, "Shortcut to BPD_FE.exe" and "Shortcut to BPD_MO.exe", are created on the Desktop. Users can double click these shortcuts to launch the respective applications.

Help Documents: This document (Installation.doc) and a tutorial (BPD_IE_Tutorial.ppt) on the use of the BPD_IE system components BPD_FE.exe and BPD_MO.exe, are copied into the "[ApplicationFolder]\doc" folder on the target computer.

ODBC Connection Configuration

Following completion of the installation of the BPD_IE system, the "ODBC data source administrator" is launched. Under "System DSN", click "Add", and then select "MySQL ODBC 3.51 Driver". Use the values below to finish installation.

[Data Source Name] = BPD_IE_MYSQL
[Host/Server Name (or IP)] = <Your MySQL server's IP address>
[Database Name] = BPD_IE
[User] = bpddemo
[Password] = demopass


MySQL ODBC 3.51 Driver - DSN Configuration, Version 3.51.09

This dialog helps you in configuring the ODBC Data Source Name, that you can use to connect to MySQL server.

DSN Information
Data Source Name: BPD_IE_MYSQL
Description: MySQL ODBC 3.51 Driver DSN

MySQL Connection Parameters
Host/Server Name (or IP): server ip address
Database Name: BPD_IE
User: bpddemo
Password: ********
Port (if not 3306): 3306
SQL command on connect:

Restart your computer

After the installation, you ***must*** restart your computer before running BPD_FE.exe or BPD_MO.exe.

How to run the programs

Before you run either BPD_FE.exe or BPD_MO.exe, you must start the MySQL server program on the server computer. and your personal firewall or system firewall must be configured so that it does not block traffic to/from the server computer.

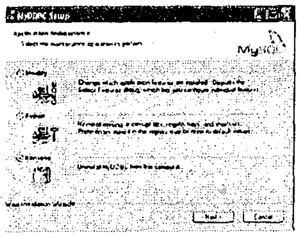
Once your system restarts, there are two shortcuts on your computer desktop: "Shortcut to BPD_FE.exe" and "Shortcut to BPD_MO.exe". Drag and drop plain text (.txt) or MS Word (.doc) files onto the "Shortcut to BPD_FE.exe" or double click the shortcut to start the Information Extraction program. Drag and drop plain text (.txt) MS Word (.doc) files onto the "Shortcut to BPD_FE.exe" or double click the shortcut to start the Modus Operandi Search program.

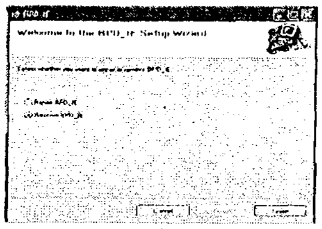
Please consult the tutorial in Appendix B for additional detail on these applications.

How to Uninstall the Programs

You can uninstall BPD_IE using the "Add and Remove Programs" tool. Simply locate BPD_IE in the list, and then click the "Remove" button to uninstall it. "MyODBC" or "Microsoft .NET framework 1.1" can be removed through the "Add and Remove Programs" tool, as well.

Another way to remove BPD_IE and MyODBC is to run Setup.exe again. Select the "Remove" option and follow the instructions to complete the uninstall process.





Questions or comments on the TMI BPD_IE may be directed to support@tmi.lehigh.edu. Additional documentation is available online at http://hddi.cse.lehigh.edu.

Install the Server System

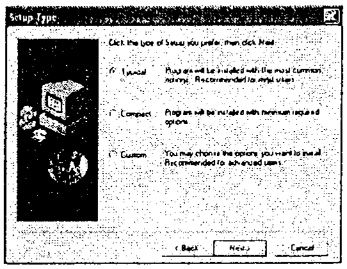
You need to install MySQL database, configure your personal firewall, create database user account, and initialize the database. The server system can be run in the same computer with the client, or in a different computer. The server system has been tested under Windows XP.

Download MySQL

Please download mysql-4.0.20d-win.zip from HERE. You can also get MySQL manual from http://dev.mysql.com/doc/mysql/en/index.html.

Install MySQL

After the download, please unzip it into a fold of your local hard drive. Click "SETUP.EXE" in that fold to start the installation. Follow the instructions and use the "Typical" option to finish the installation.



Setup Type

Click the type of Setup you prefer, then click Next.

[✓ Typical] Programs will be installed with the most common options. Recommended for most users.

Compact Programs will be installed with minimum required options.

Custom You may choose the options you want to install. Recommended for advanced users.

Make sure that your MySQL bin fold is included in the Windows environment variable $PATH.

Look in your Windows fold (i.e. C:\WINDOWS) and see if there is a file called "my.ini".

If there is, open it in an editor, and make sure the path information is correct. i.e.:

```
basedir=C:/mysql/
datadir=C:/mysql/data/
```

If you cannot find "my.ini" in your Windows folder, please create a new file named "my.ini" with the following content under that folder, and make sure the path information is correct. i.e.:

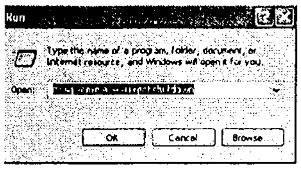
```
[mysqld]
basedir=C:/mysql/
datadir=C:/mysql/data/
```

Shutdown / Start MySQL database server

After your installation, the MySQL server will be automatically started. However, you must shut it down and restart it in order to make it work properly.

To shutdown the current server, please run the following command:

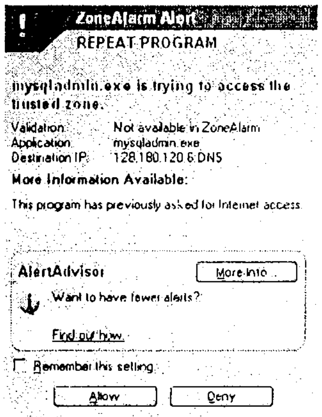
mysqladmin.exe -u root shutdown



Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.

Open: mysqladmin.exe -u root shutdown

To be noticed, you must configure your personal firewall to allow mysqladmin.exe to access DNS and "127.0.0.1:3306".



mysqladmin.exe is trying to access the trusted zone.

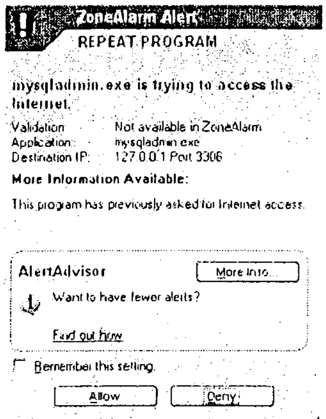
Validation: Not available in ZoneAlarm
Application: mysqladmin.exe
Destination IP: 128.180.120.6:DNS

More Information Available:

This program has previously asked for Internet access.

Want to have fewer alerts?

Remember this setting



mysqladmin.exe is trying to access the Internet.

Validation: Not available in ZoneAlarm
Application: mysqladmin.exe
Destination IP: 127.0.0.1:Port 3306

More Information Available:

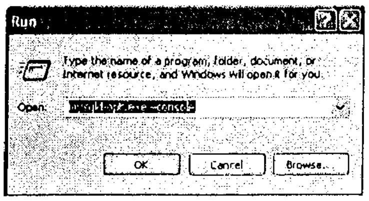
This program has previously asked for Internet access.

Want to have fewer alerts?

Remember this setting

To start the MySQL server, please run the following command:

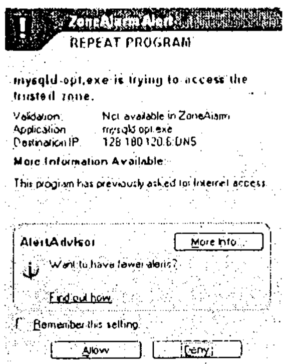
mysqld-opt.exe –console



Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.

Open: mysqld-opt.exe --console

You must configure your personal firewall to allow mysqladmin.exe to access DNS and "0.0.0.0:3306".


ZoneAlarm Alert

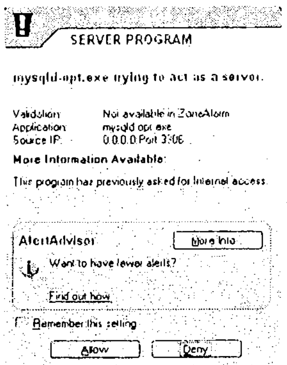
REPEAT PROGRAM

mysqld-opt.exe is trying to access the trusted zone.

Validation: Not available in ZoneAlarm
Application: mysqld-opt.exe
Destination IP: 128.180.120.6:DNS

More Information Available:

This program has previously asked for Internet access.


SERVER PROGRAM

mysqld-opt.exe trying to act as a server.

Validation: Not available in ZoneAlarm
Application: mysqld-opt.exe
Source IP: 0.0.0.0:Port 3306

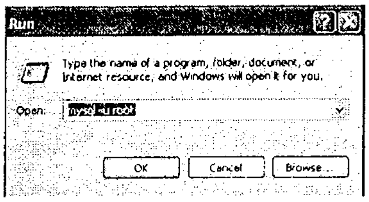
More Information Available:

This program has previously asked for Internet access.

In case you have "Do you already have another mysqld server running on port: 3306" error, you should shutdown the current server, and restart it.

Create and Initialize the database

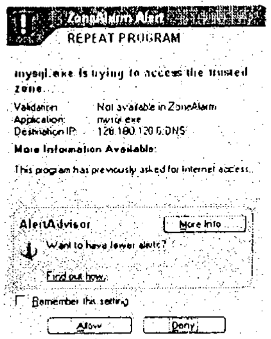
Use the following command to start a MySQL client as a root user.


Run

Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.

Open: mysql -u root

you must configure your personal firewall to allow mysqladmin.exe to access DNS and "127.0.0.1:3306".


ZoneAlarm Alert

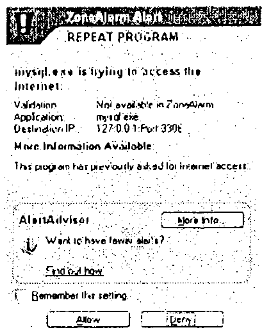
REPEAT PROGRAM

mysql.exe is trying to access the trusted zone.

Validation: Not available in ZoneAlarm
Application: mysql.exe
Destination IP: 128.180.120.6:DNS

More Information Available:

This program has previously asked for Internet access.


ZoneAlarm Alert

REPEAT PROGRAM

mysql.exe is trying to access the Internet.

Validation: Not available in ZoneAlarm
Application: mysql.exe
Destination IP: 127.0.0.1:Port 3306

More Information Available:

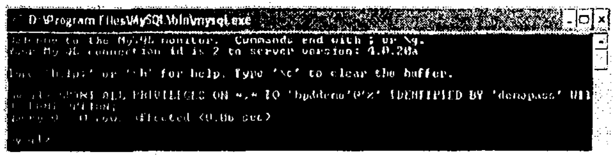
This program has previously asked for Internet access.

All the following commands are in "commandToCreateTables.txt", which comes with the source code package of the system. After you finish section 5.4.4, the server setup is done. You are ready to use the BPD_IE clients (BPD_FE.exe and BPD_MO.exe).

Create a new user:

Through the command line, you can create a new user for the BPD_IE system:

```
GRANT ALL PRIVILEGES ON *.* TO 'bpddemo'@'%' IDENTIFIED BY 'demopass'
                    WITH GRANT OPTION;
```



```
D:\Program Files\MySQL\bin\mysql.exe
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2 to server version: 4.0.20a

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> GRANT ALL PRIVILEGES ON *.* TO 'bpddemo'@'%' IDENTIFIED BY 'demopass' WIT
H GRANT OPTION;
Query OK, 0 rows affected (0.06 sec)

mysql>
```

The user name is "bpddemo", and its password to access the database is "demopass". If you use the BPD client and the server on the same computer, you can create this account for a local user only:

```
GRANT ALL PRIVILEGES ON *.* TO 'bpddemo'@'localhost' IDENTIFIED BY
                 'demopass' WITH GRANT OPTION;
```

Link the user defined function

Copy and paste "editDistance.dll" from "[ApplicationFolder]\bin to a folder that is in your $PATH, and then run the following command in mysql:

```
CREATE FUNCTION editDistance RETURNS INTEGER SONAME "editDistance.dll";
```

Create the database

```
create database bpd_ie;
```

Initialize the database

```
use bpd_ie;
DROP TABLE IF EXISTS Features;
create table Features (
id int unsigned not null  AUTO_INCREMENT primary key,
featureType varchar(255) not null,
featureContent mediumtext,
startValueSTR text,
endValueSTR text,
certainty double(9,8),
startValueNum decimal(18,0),
endValueNum decimal(18,0),
documentName varchar(250),
localHostName varchar(250),
INDEX (featureType(50),startValueSTR(225),endValueSTR(225))
);

DROP TABLE IF EXISTS Documents ;
create table Documents (
documentName varchar(250),
originalText mediumtext,
localHostName varchar(250),
primary key (documentName, localHostName)
);

DROP TABLE IF EXISTS featureCandidate;
create table featureCandidate(
FeatureTypeName varchar(250) not null,
FeatureCandidate varchar(250) not null,
primary key (FeatureTypeName, FeatureCandidate)
);

insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'EyeColor','Black' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'EyeColor','Brown');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'EyeColor','Green');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'EyeColor','Maroon');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'EyeColor','Pink');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'EyeColor','Blue');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'EyeColor','Gray');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'EyeColor','Hazel');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'EyeColor','Multicolored' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'EyeColor','Dark');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'HairColor','Bald' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'HairColor','Black' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'HairColor','Blond' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'HairColor','Strawberry' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'HairColor','Blue' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'HairColor','Brown' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'HairColor','Gray' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'HairColor','Partially Gray' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'HairColor','Green' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'HairColor','Orange' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'HairColor','Pink' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'HairColor','Purple' );
```

```
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'HairColor','Red ' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'HairColor','Auburn' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'HairColor','Sandy' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'HairColor','White' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'HairColor','Dark' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Ambulance' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Coach' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Convertible' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Coupe' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Sedan' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Hardtop' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Hatchback' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Hearse' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Limousine' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Roadster' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Wagon' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','SUV' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Hammer' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Motorcycle' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Van' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Tanker' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Trailer' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Truck' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Bus' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Cab' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Dump' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Hopper' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Pallet' );
```

```
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Rack' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Stake' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Wrecker' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Vanette' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Crane' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Vehicle','Car' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Weapon','Ammunition' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Weapon','Bomb' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Weapon','Cannon' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Weapon','Disguised gun' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Weapon','Electric shock gun' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Weapon','Grenade' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Weapon','Machine gun' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Weapon','gun' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Weapon','Mine' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Weapon','Missile' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Weapon','Mortar' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Weapon','Pistol' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Weapon','Rifle' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Weapon','shotgun' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Weapon','Rocket' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Weapon','Silencer' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Weapon','Carbine' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Weapon','Derringer' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Weapon','Flare' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Weapon','Flintlock' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Weapon','Revolver ' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Weapon','Recoilless' );
```

```
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Weapon','Tear gas' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Weapon','Knife' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Daughter' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Son' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Brother' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Brother-in-law' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Sister' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Sister-in-law' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Mother' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Father' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Dad' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Grand-mother' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Mom' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Grand-father' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Grand-daughter' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Grand-son' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Niece' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Nephew' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Aunt' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Uncle' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Cousin' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Stepmother' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Stepfather' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Stepsister' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Stepbrother' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Mother-in-law' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Father-in-law' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Daughter-in-law' );
```

```
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Son-in-law' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Spouse' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Husband' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Wife' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Boyfriend' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Girlfriend' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Parent' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Relationship','Grandchild' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Gender','Male' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Gender','Female' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Race','American Indian' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Race','Asian' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Race','Pacific Islander' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Race','Black' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Race','White' );
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Race','Hispanic');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Drugs','Alcohol');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Drugs','Amphetamines');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Drugs','Stimulants');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Drugs','Barbiturates');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Drugs','Cocaine');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Drugs','Glue');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Drugs','Hallucinogens');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Drugs','Marijuana');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Drugs','Narcotics');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Drugs','Heroin');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Drugs','Morphine');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Drugs','Dilaudid');
```

```
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Drugs','Methadone');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Drugs','Paint');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Drugs','Thinner');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Drugs','Ritalin');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Drugs','Rohypnol');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Drugs','Flunitrazepam');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Drugs','Rophies');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Drugs','Roofies');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Drugs','Ruffies');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'Drugs','Roche');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Treason');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Misprision');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Espionage');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Sabotage');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Sedition');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Sovereignty');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Desertion');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Illegal Entry');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','False Citizenship');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Smuggling');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Kidnap');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Abduct');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Rape');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Sex Asslt');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Homicide');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Robbery');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Carjacking');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Threat');
```

```
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Arson');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Aggrav Asslt');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Extort');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Burglary');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Abortion');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','LARCENY');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Pocketpicking');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Shoplifting');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Theft');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Fraud');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Embezzle');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Forgery');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Damage');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Counterfeiting');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Obscenity');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Bigamy');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Bookmaking');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Gambling');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Lottery');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Bestiality');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Seduction');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Prostitution');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Liquor');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Perjury');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Anarchism');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Trespassing');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Eavesdropping');
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Antitrust');
```

```
insert into featureCandidate(FeatureTypeName,FeatureCandidate ) values
( 'OffenseType','Conservation');

DROP TABLE IF EXISTS featureType;
create table featureType (
FeatureTypeName varchar(255) not null primary key
);

insert into featureType (FeatureTypeName) values ( 'AccountNumber' );
insert into featureType (FeatureTypeName) values ( 'Age' );
insert into featureType (FeatureTypeName) values ( 'ArrestNumber' );
insert into featureType (FeatureTypeName) values ( 'BadgeNumber' );
insert into featureType (FeatureTypeName) values ( 'Classification' );
insert into featureType (FeatureTypeName) values ( 'Clothing' );
insert into featureType (FeatureTypeName) values ( 'CompanyName' );
insert into featureType (FeatureTypeName) values ( 'ControlNumber' );
insert into featureType (FeatureTypeName) values ( 'DayOfWeek' );
insert into featureType (FeatureTypeName) values ( 'DateOfBirth' );
insert into featureType (FeatureTypeName) values ( 'Height' );
insert into featureType (FeatureTypeName) values ( 'Location' );
insert into featureType (FeatureTypeName) values ( 'Date' );
insert into featureType (FeatureTypeName) values ( 'OfficerName' );
insert into featureType (FeatureTypeName) values ( 'PhoneNumber' );
insert into featureType (FeatureTypeName) values ( 'Race' );
insert into featureType (FeatureTypeName) values ( 'ReportDate' );
insert into featureType (FeatureTypeName) values ( 'Reward' );
insert into featureType (FeatureTypeName) values ( 'Weapon' );
insert into featureType (FeatureTypeName) values ( 'Weight' );
insert into featureType (FeatureTypeName) values ( 'ItemStolen' );
insert into featureType (FeatureTypeName) values ( 'PersonName' );
insert into featureType (FeatureTypeName) values ( 'Drugs' );
insert into featureType (FeatureTypeName) values ( 'OffenseType' );
insert into featureType (FeatureTypeName) values ( 'Time' );
insert into featureType (FeatureTypeName) values ( 'Vehicle' );
insert into featureType (FeatureTypeName) values ( 'HairColor' );
insert into featureType (FeatureTypeName) values ( 'EyeColor' );
insert into featureType (FeatureTypeName) values ( 'Relationship' );
insert into featureType (FeatureTypeName) values ( 'Residence' );
insert into featureType (FeatureTypeName) values ( 'SSN' );
insert into featureType (FeatureTypeName) values ( 'DriverLicenseNumber' );
insert into featureType (FeatureTypeName) values ( 'Gender' );
insert into featureType (FeatureTypeName) values ( 'ID' );
insert into featureType (FeatureTypeName) values ( 'BLocation' );
insert into featureType (FeatureTypeName) values ( 'EQUIPMENT');
insert into featureType (FeatureTypeName) values ( 'TARGET');
insert into featureType (FeatureTypeName) values ( 'GRAM');
```

Uploading a New Release of the BPD_IE System to the Server

Login to "hddi.cse.lehigh.edu" and modify the following files as necessary:

- "/export/www/htdocs/source.html"
- "/export/www/htdocs/bpd_ie_binary-1.1-zip.html"
- "/export/www/cgi-bin/gen/bpd_ie_binary_1.1.zip.cgi" (Add executable permissions to this file.)

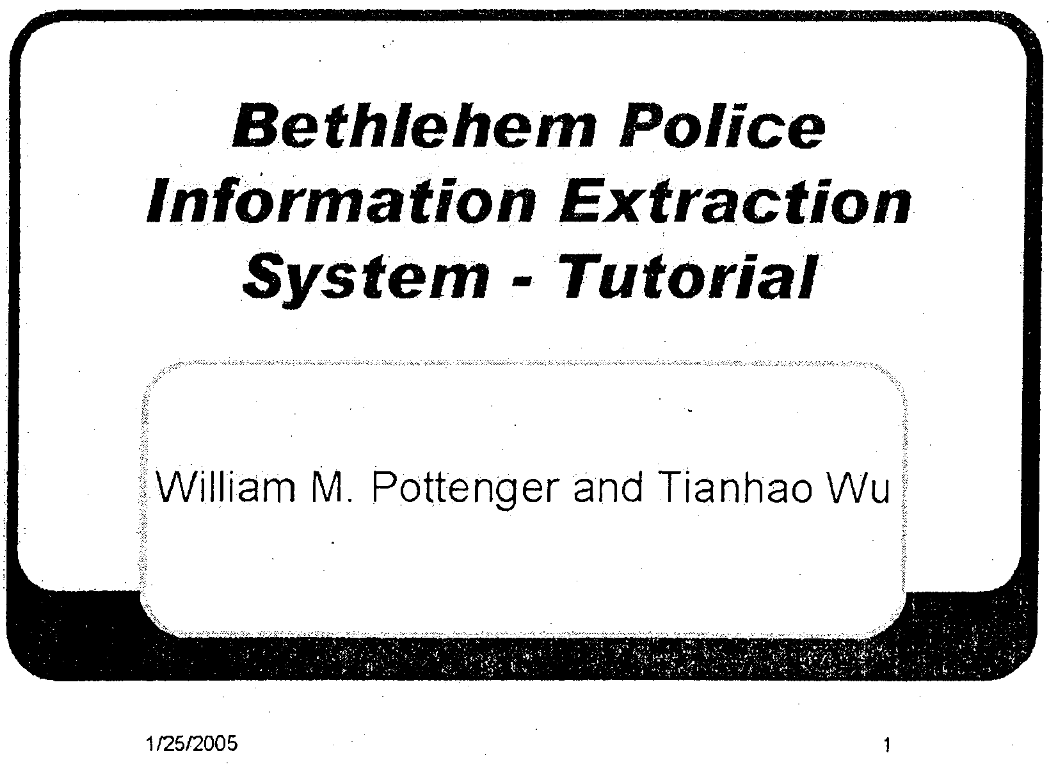
- "/export/www/htdocs/source/BPDIE_binary_release_1.1.zip"

Note that you must edit the cgi file using a unix editor.

FAQ

- MySQL and My ODBC version
  The BPD_IE System has been tested using mysql-4.0.20d-win and MyODBC-3.51.10-x86-win-32bit.
- ODBC connection error 10061
  You may need to reinstall the ODBC driver and then restart your computer.
- MySQL setup.exe has the "parameter incorrect error"
  This is because your installation directory contains some undesirable special characters in the path. Please move the installation folder to a simpler directory such as "C:\", and try setup.exe again.
- Cannot find "mfc71d.dll" or "msvcr71d.dll" or "msvcp71d.dll"
  Copy and paste the above files from your "[ApplicationFolder]\bin" folder to your MySQL/bin folder.

Appendix B – Training Tutorial



Bethlehem Police Information Extraction System - Tutorial

William M. Pottenger and Tianhao Wu

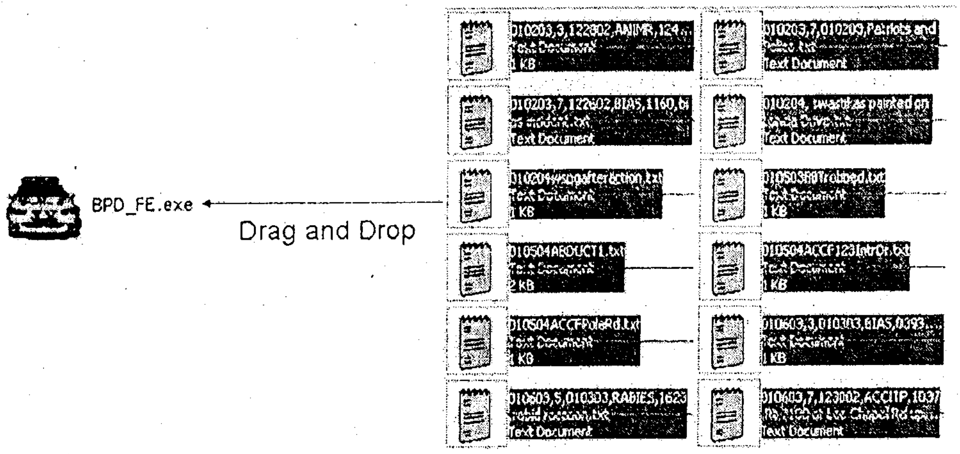
1/25/2005 1

Sub Systems

- BPD_FE
  - Extract meaningful entities from BPD investigator's reports using predefined rules
- BPD_MO
  - Search existing investigator's reports using user's input or features extracted from the current unsolved investigator's report
    - Range search
    - Fuzzy search

1/25/2005 2

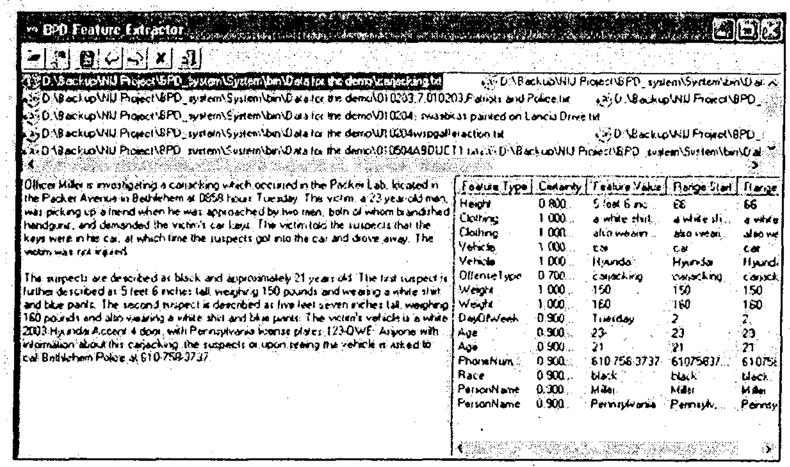
BPD_FE


BPD_FE.exe
Drag and Drop
010203,3,122802,ANIMR,124...
Text Document
1 KB
010203,7,010209,Patriots and Pelted.txt
Text Document
010203,7,122602,BIAS,1160,bias incident.txt
Text Document
010204, swastikas painted on [illegible]
Text Document
010204wsopafteraction.txt
Text Document
1 KB
010503RBTrobbed.txt
Text Document
1 KB
010504ABDUCT1.txt
Text Document
2 KB
010504ACCF123IntrDr.txt
Text Document
1 KB
010504ACCFPoleRd.txt
Text Document
1 KB
010603,3,010303,BIAS,0393...
Text Document
1 KB
010603,5,010303,RABIES,1623 rabid raccoon.txt
Text Document
010603,7,123002,ACCITP,1037 [illegible]
Text Document

Limit: about thirty documents

1/25/2005 3

BPD_FE GUI



D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\carjacking.txt D:\Backup\NIJ Project\BPD_system\System\bin\Dat
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010203.7.010203.Patriots and Police.txt D:\Backup\NIJ Project\BPD_
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010204; swastikas painted on Lancia Drive.txt
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010204wspgalleraction.txt D:\Backup\NIJ Project\BPD_
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010604ABDUCT1.txt D:\Backup\NIJ Project\BPD_system\System\bin\Dat

Officer Miller is investigating a carjacking which occurred in the Packer Lab, located in the Packer Avenue in Bethlehem at 0858 hours Tuesday. The victim, a 23 year old man, was picking up a friend when he was approached by two men, both of whom brandished handguns, and demanded the victim's car keys. The victim told the suspects that the keys were in his car, at which time the suspects got into the car and drove away. The victim was not injured.

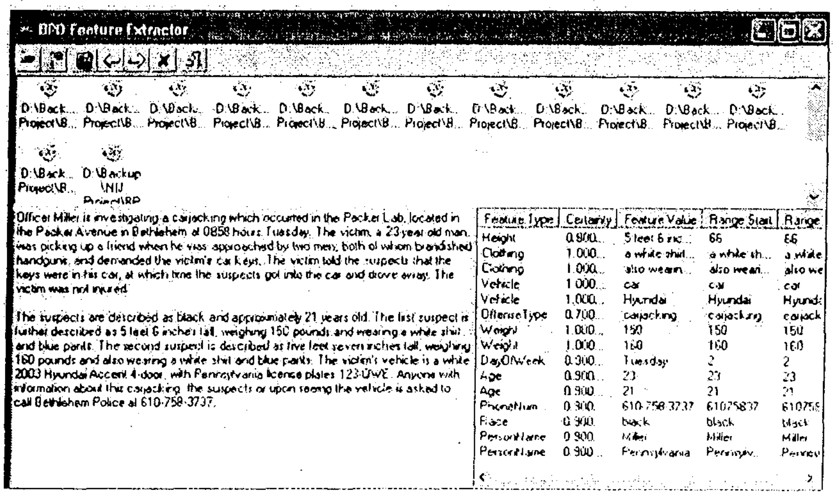
The suspects are described as black and approximately 21 years old. The first suspect is further described as 5 feet 6 inches tall, weighing 150 pounds and wearing a white shirt and blue pants. The second suspect is described as five feet seven inches tall, weighing 160 pounds and also wearing a white shirt and blue pants. The victim's vehicle is a white 2003 Hyundai Accent 4 door, with Pennsylvania license plates 123-QWE. Anyone with information about this carjacking, the suspects or upon seeing the vehicle is asked to call Bethlehem Police at 610-758-3737.

Feature Type	Certainty	Feature Value	Range Start	Range
Height	0.800...	5 feet 6 inc...	66	66
Clothing	1.000...	a white shirt...	a white sh...	a white
Clothing	1.000...	also wearin...	also wear...	also we
Vehicle	1.000...	car	car	car
Vehicle	1.000...	Hyundai	Hyundai	Hyundai
OffenseType	0.700...	carjacking	carjacking	carjack
Weight	1.000...	150	150	150
Weight	1.000...	160	160	160
DayOfWeek	0.900...	Tuesday	2	2
Age	0.900...	23	23	23
Age	0.900...	21	21	21
PhoneNum	0.900...	610-758-3737	61075837...	61075
Race	0.900...	black	black	black
PersonName	0.300...	Miller	Miller	Miller
PersonName	0.900...	Pennsylvania	Pennsylv...	Pennsy

The first document is selected automatically. The original text and features of the document are displayed.

1/25/2005

Style – large Icon



Officer Miller is investigating a carjacking which occurred in the Packer Lab, located in the Packer Avenue in Bethlehem at 0858 hours Tuesday. The victim, a 23 year old man, was picking up a friend when he was approached by two men, both of whom brandished handguns, and demanded the victim's car keys. The victim told the suspects that the keys were in his car, at which time the suspects got into the car and drove away. The victim was not injured.

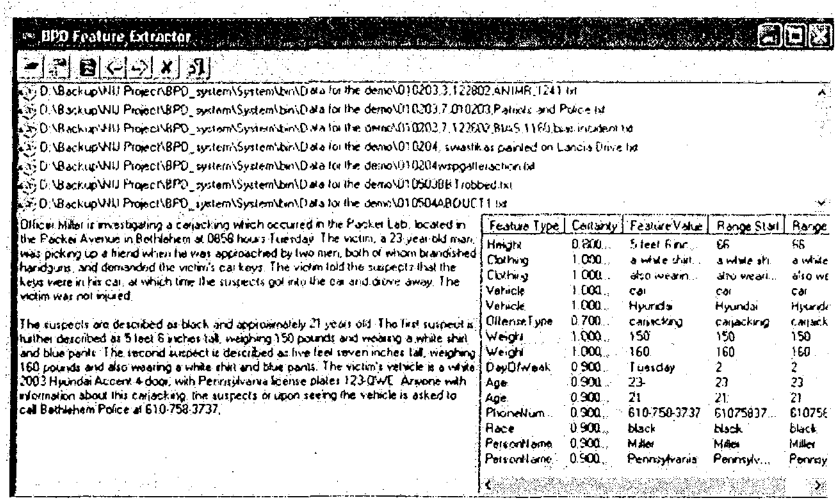
The suspects are described as black and approximately 21 years old. The first suspect is further described as 5 feet 6 inches tall, weighing 150 pounds and wearing a white shirt and blue pants. The second suspect is described as five feet seven inches tall, weighing 160 pounds and also wearing a white shirt and blue pants. The victim's vehicle is a white 2003 Hyundai Accent 4-door, with Pennsylvania license plates 123-QWE. Anyone with information about this carjacking, the suspects or upon seeing the vehicle is asked to call Bethlehem Police at 610-758-3737.

Feature Type	Certainty	Feature Value	Range Start	Range
Height	0.800...	5 feet 6 inc...	66	66
Clothing	1.000...	a white shirt...	a white sh...	a white
Clothing	1.000...	also wearin...	also wear...	also we
Vehicle	1.000...	car	car	car
Vehicle	1.000...	Hyundai	Hyundai	Hyundai
OffenseType	0.700...	carjacking	carjacking	carjack
Weight	1.000...	150	150	150
Weight	1.000...	160	160	160
DayOfWeek	0.900...	Tuesday	2	2
Age	0.900...	23	23	23
Age	0.900...	21	21	21
PhoneNum	0.900...	610-758-3737	61075837	61075
Race	0.900...	black	black	black
PersonName	0.900...	Miller	Miller	Miller
PersonName	0.900...	Pennsylvania	Pennsylv...	Pennsy

Change the display style of the input documents

1/25/2005

Style – Small Icon



```
D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010203.3,122802,ANIMR,1241.txt
D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010203.7,010203,Patrols and Police.txt
D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010203.7,122602,BIAS,1160,bias incident.txt
D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010204, swastikas painted on Lancia Drive.txt
D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010204wspgalleraction.txt
D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010503BBTrobbed.txt
D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010504ABOUCT1.txt
```

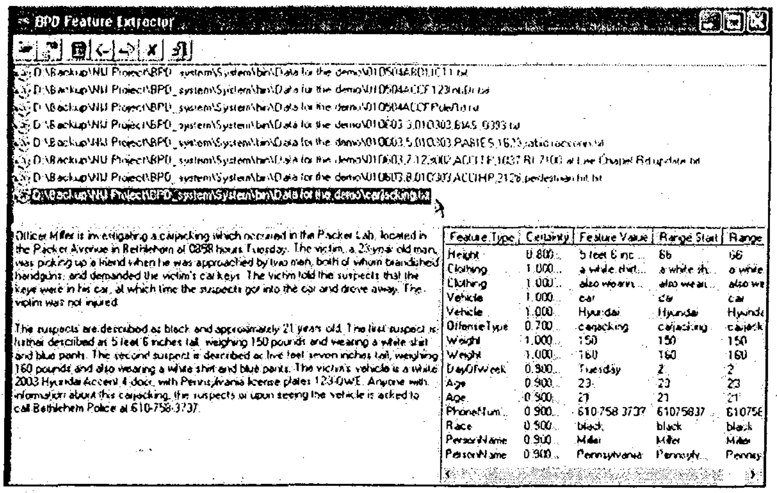
Officer Miller is investigating a carjacking which occured in the Packer Lab, located in the Packer Avenue in Bethlehem at 0858 hours Tuesday. The victim, a 23-year-old man, was picking up a friend when he was approached by two men, both of whom brandished handguns, and demanded the victim's car keys. The victim told the suspects that the keys were in his car, at which time the suspects got into the car and drove away. The victim was not injured.

The suspects are described as black and approximately 21 years old. The first suspect is further described as 5 feet 6 inches tall, weighing 150 pounds and wearing a white shirt and blue pants. The second suspect is described as five feet seven inches tall, weighing 160 pounds and also wearing a white shirt and blue pants. The victim's vehicle is a white 2003 Hyundai Accent 4-door, with Pennsylvania license plates 123-QWE. Anyone with information about this carjacking, the suspects or upon seeing the vehicle is asked to call Bethlehem Police at 610-758-3737.

Feature Type	Certainty	Feature Value	Range Start	Range
Height	0.800...	5 feet 6 inc...	66	66
Clothing	1.000...	a white shirt...	a white sh...	a white
Clothing	1.000...	also wearin...	also wear...	also we
Vehicle	1.000...	car	car	car
Vehicle	1.000...	Hyundai	Hyundai	Hyunda
OffenseType	0.700...	carjacking	carjacking	carjack
Weight	1.000...	150	150	150
Weight	1.000...	160	160	160
DayOfWeek	0.900...	Tuesday	2	2
Age	0.900...	23-	23	23
Age	0.900...	21	21	21
PhoneNum...	0.900...	610-758-3737	61075837...	610758
Race	0.900...	black	black	black
PersonName	0.900...	Miller	Miller	Miller
PersonName	0.900...	Pennsylvania	Pennsylv...	Pennsy

1/25/2005 6

Select Document – Mouse Click



```
D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010504ABOUCT1.txt
D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010504ACCF123robDr.txt
D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010504ACCFPoleRd.txt
D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010303.3,010303,BIAS,0393.txt
D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010303.5,010303,RABIES,1623,rabid raccoon.txt
D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010303.7,12,2002,ACCTF,1037 Rt 7100 at Lee Chapel Rd update.txt
D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010303.8,010303,ACCIHP,2126,pedestrian hit.txt
D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\carjacking.txt
```

Officer Miller is investigating a carjacking which occured in the Packer Lab, located in the Packer Avenue in Bethlehem at 0858 hours Tuesday. The victim, a 23-year-old man, was picking up a friend when he was approached by two men, both of whom brandished handguns, and demanded the victim's car keys. The victim told the suspects that the keys were in his car, at which time the suspects got into the car and drove away. The victim was not injured.

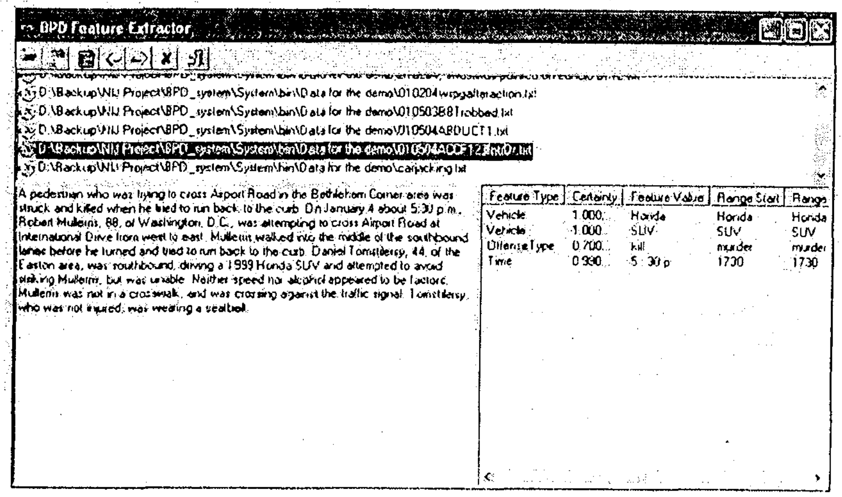
The suspects are described as black and approximately 21 years old. The first suspect is further described as 5 feet 6 inches tall, weighing 150 pounds and wearing a white shirt and blue pants. The second suspect is described as five feet seven inches tall, weighing 160 pounds and also wearing a white shirt and blue pants. The victim's vehicle is a white 2003 Hyundai Accent 4-door, with Pennsylvania license plates 123-QWE. Anyone with information about this carjacking, the suspects or upon seeing the vehicle is asked to call Bethlehem Police at 610-758-3737.

Feature Type	Certainty	Feature Value	Range Start	Range
Height	0.800...	5 feet 6 inc...	66	66
Clothing	1.000...	a white shirt...	a white sh...	a white
Clothing	1.000...	also wearin...	also wear...	also we
Vehicle	1.000...	car	car	car
Vehicle	1.000...	Hyundai	Hyundai	Hyunda
OffenseType	0.700...	carjacking	carjacking	carjack
Weight	1.000...	150	150	150
Weight	1.000...	160	160	160
DayOfWeek	0.900...	Tuesday	2	2
Age	0.900...	23-	23	23
Age	0.900...	21	21	21
PhoneNum...	0.900...	610-758-3737	61075837...	610758
Race	0.900...	black	black	black
PersonName	0.900...	Miller	Miller	Miller
PersonName	0.900...	Pennsylvania	Pennsylv...	Pennsy

Using Mouse to select a document to display its original text and features extracted

1/25/2005 7

Previous Document



BPD Feature Extractor

```
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010204wrpgalteraction.txt
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\01050388Trobbed.txt
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010504ABDUCT1.txt
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010504ACCF123InfOr.txt
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\carjacking.txt
```

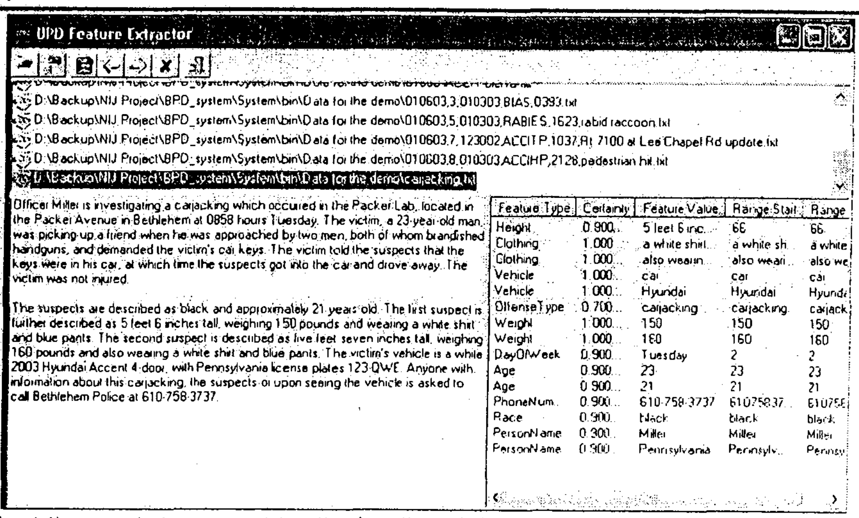
A pedestrian who was trying to cross Airport Road in the Bethlehem Corner area was struck and killed when he tried to run back to the curb. On January 4 about 5:30 p.m., Robert Mulleins, 88, of Washington, D.C., was attempting to cross Airport Road at International Drive from west to east. Mulleins walked into the middle of the southbound lanes before he turned and tried to run back to the curb. Daniel Tomstlersy, 44, of the Easton area, was southbound, driving a 1999 Honda SUV and attempted to avoid striking Mulleins, but was unable. Neither speed nor alcohol appeared to be factors. Mulleins was not in a crosswalk, and was crossing against the traffic signal. Tomstlersy, who was not injured, was wearing a seatbelt.

Feature Type	Certainty	Feature Value	Range Start	Range
Vehicle	1.000	Honda	Honda	Honda
Vehicle	1.000	SUV	SUV	SUV
OffenseType	0.700	kill	murder	murder
Time	0.930	5 : 30 p	1730	1730

Select the previous document to display its original text and features extracted

1/25/2005 8

Next Document



UPD Feature Extractor

```
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603,3,010303,BIAS,0393.txt
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603,5,010303,RABIES,1623,rabid raccoon.txt
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603,7,123002,ACCITP,1037,Rt 7100 at Lee Chapel Rd update.txt
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603,8,010303,ACCIHP,2128,pedestrian hit.txt
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\carjacking.txt
```

Officer Miller is investigating a carjacking which occured in the Packer Lab, located in the Packer Avenue in Bethlehem at 0858 hours Tuesday. The victim, a 23-year-old man, was picking up a friend when he was approached by two men, both of whom brandished handguns, and demanded the victim's car keys. The victim told the suspects that the keys were in his car, at which time the suspects got into the car and drove away. The victim was not injured.

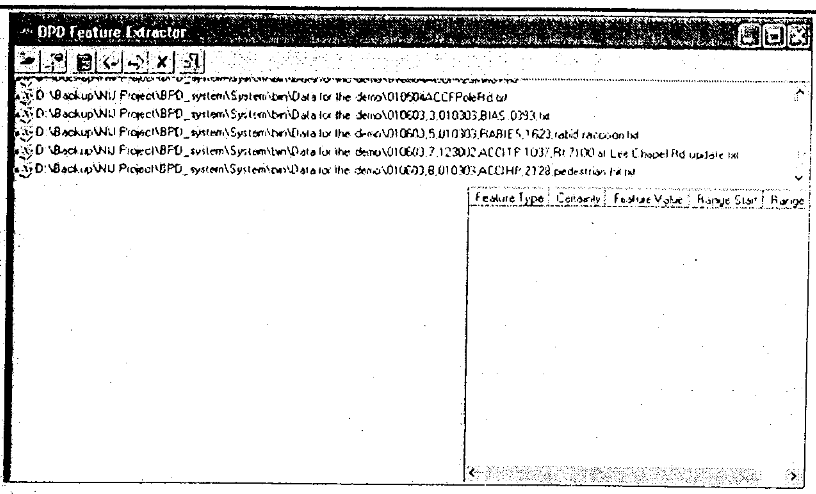
The suspects are described as black and approximately 21 years old. The first suspect is further described as 5 feet 6 inches tall, weighing 150 pounds and wearing a white shirt and blue pants. The second suspect is described as five feet seven inches tall, weighing 160 pounds and also wearing a white shirt and blue pants. The victim's vehicle is a white 2003 Hyundai Accent 4-door, with Pennsylvania license plates 123-QWE. Anyone with information about this carjacking, the suspects or upon seeing the vehicle is asked to call Bethlehem Police at 610-758-3737.

Feature Type	Certainty	Feature Value	Range Start	Range
Height	0.900	5 feet 6 inc...	66	66
Clothing	1.000	a white shirt...	a white sh	a white
Clothing	1.000	also wearin...	also weari...	also we
Vehicle	1.000	car	car	car
Vehicle	1.000	Hyundai	Hyundai	Hyundai
OffenseType	0.700	carjacking	carjacking	carjack
Weight	1.000	150	150	150
Weight	1.000	160	160	160
DayOfWeek	0.900	Tuesday	2	2
Age	0.900	23	23	23
Age	0.900	21	21	21
PhoneNum	0.900	610-758-3737	61075837	61075E
Race	0.900	black	black	black
PersonName	0.300	Miller	Miller	Miller
PersonName	0.900	Pennsylvania	Pennsylv...	Pennsy

Select the next document to display its original text and features extracted

1/25/2005 9

Delete



```
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010504ACCFPoleRd.txt
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603,3,010303,BIAS,0393.txt
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603,5,010303,RABIES,1623,rabid raccoon.txt
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603,7,123002,ACCITP,1037,Rt 7100 at Lee Chapel Rd update.txt
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603,8,010303,ACCIHP,2128,pedestrian hit.txt
```

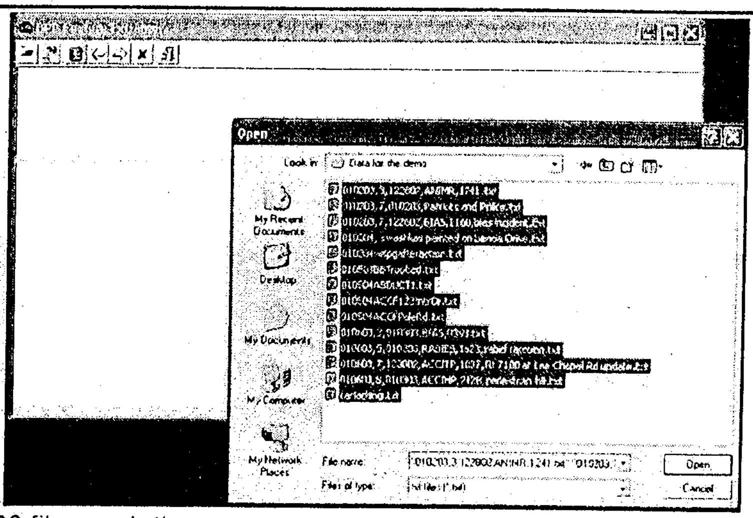
Feature Type	Certainty	Feature Value	Range Start	Range

Delete the current document selected. One document each time. The files on the hard drive are not deleted.

1/25/2005

10

Open file



Look in: Data for the demo

```
010203,3,122802,ANIMR,1741.txt
010203,7,010203,Patricks and Prince.txt
010203,7,122802,BIAS,1160,bias incident.txt
010304, swastikas painted on Lenox Drive.txt
010304wagafterbotan.txt
010504BBTrucked.txt
010504ASDUCT1.txt
010504ACCF122mrDr.txt
010504ACCFPoleRd.txt
010603,3,010303,BIAS,0393.txt
010603,5,010303,RABIES,1623,rabid raccoon.txt
010603,7,123002,ACCITP,1037,Rt 7100 at Lee Chapel Rd update.txt
010603,8,010303,ACCIHP,2128,pedestrian hit.txt
carlocking.txt
```

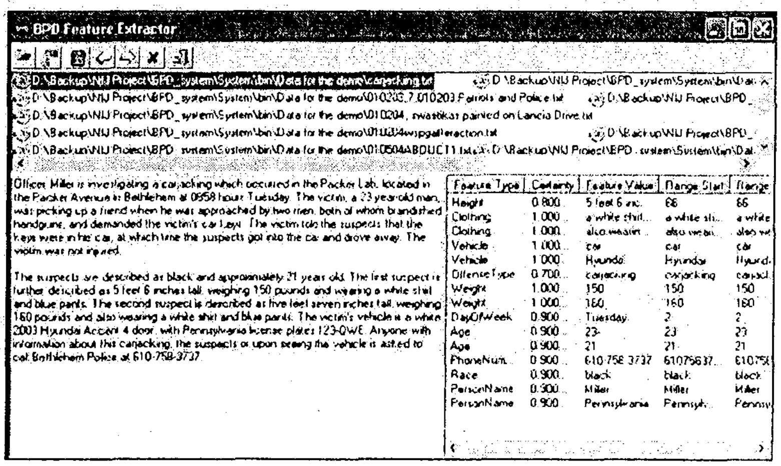
File name: "010203,3,122802,ANIMR,1741.txt" "010203,
Files of type: txt files (*.txt)

Limit: 200 files each time

1/25/2005

11

Open file - result



BPD Feature Extractor

D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\carjacking.txt D:\Backup\NIJ Project\BPD_system\System\bin\Dat...
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010203,7,010203,Patrols and Police.txt D:\Backup\NIJ Project\BPD_
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010204, swastikas painted on Lancia Drive.txt
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010204wspgafteraction.txt D:\Backup\NIJ Project\BPD_
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010504ABDUCT1.txt D:\Backup\NIJ Project\BPD_system\System\bin\Dat...

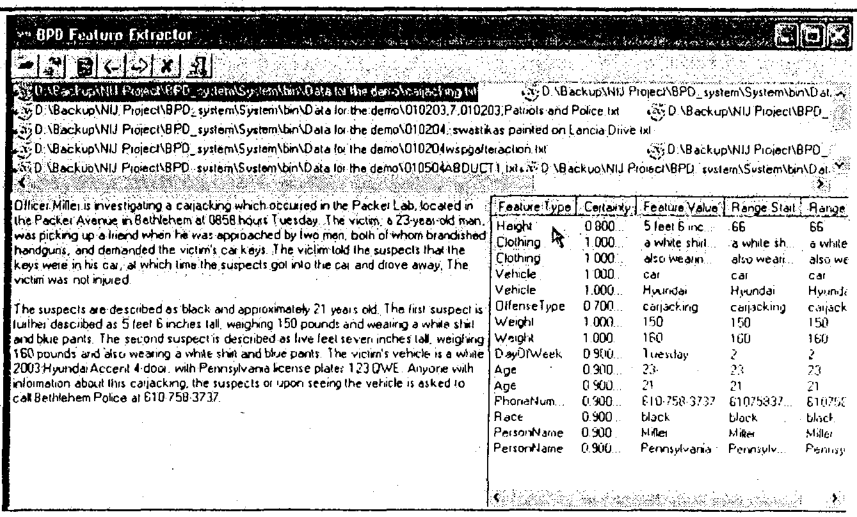
Officer Miller is investigating a carjacking which occurred in the Packer Lab, located in the Packer Avenue in Bethlehem at 0858 hours Tuesday. The victim, a 23-year-old man, was picking up a friend when he was approached by two men, both of whom brandished handguns, and demanded the victim's car keys. The victim told the suspects that the keys were in his car, at which time the suspects got into the car and drove away. The victim was not injured.

The suspects are described as black and approximately 21 years old. The first suspect is further described as 5 feet 6 inches tall, weighing 150 pounds and wearing a white shirt and blue pants. The second suspect is described as five feet seven inches tall, weighing 160 pounds and also wearing a white shirt and blue pants. The victim's vehicle is a white 2003 Hyundai Accent 4-door, with Pennsylvania license plate: 123-QWE. Anyone with information about this carjacking, the suspects or upon seeing the vehicle is asked to call Bethlehem Police at 610-758-3737.

Feature Type	Certainty	Feature Value	Range Start	Range
Height	0.800...	5 feet 6 inc...	66	66
Clothing	1.000...	a white shirt...	a white sh...	a white
Clothing	1.000...	also wearin...	also wear...	also we
Vehicle	1.000...	car	car	car
Vehicle	1.000...	Hyundai	Hyundai	Hyundai
OffenseType	0.700...	carjacking	carjacking	carjacl
Weight	1.000...	150	150	150
Weight	1.000...	160	160	160
DayOfWeek	0.900...	Tuesday	2	2
Age	0.900...	23	23	23
Age	0.900...	21	21	21
PhoneNum...	0.900...	610-758-3737	61075837...	61075
Race	0.900...	black	black	black
PersonName	0.900...	Miller	Miller	Miller
PersonName	0.900...	Pennsylvania	Pennsylv...	Pennsy

The first document will be selected automatically. Its text and features are displayed

Highlight Features



BPD Feature Extractor

D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\carjacking.txt D:\Backup\NIJ Project\BPD_system\System\bin\Dat...
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010203,7,010203,Patrols and Police.txt D:\Backup\NIJ Project\BPD_
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010204, swastikas painted on Lancia Drive.txt
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010204wspgafteraction.txt D:\Backup\NIJ Project\BPD_
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010504ABDUCT1.txt D:\Backup\NIJ Project\BPD_system\System\bin\Dat...

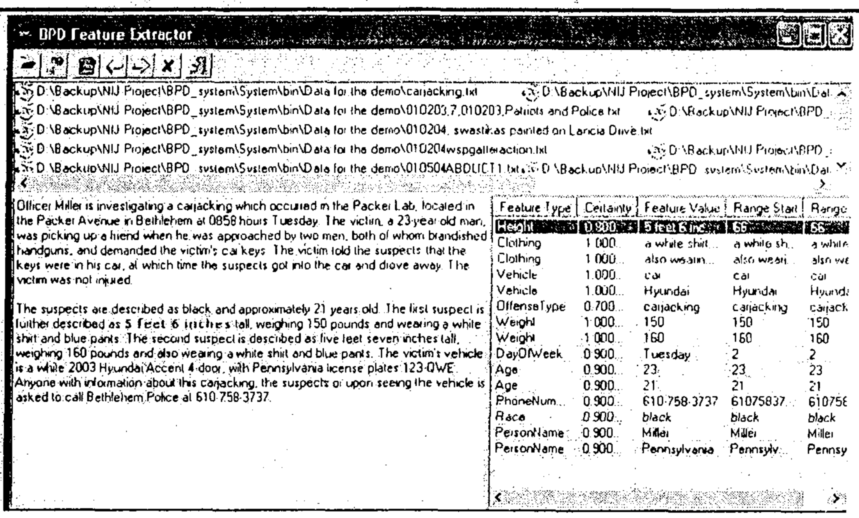
Officer Miller is investigating a carjacking which occurred in the Packer Lab, located in the Packer Avenue in Bethlehem at 0858 hours Tuesday. The victim, a 23-year-old man, was picking up a friend when he was approached by two men, both of whom brandished handguns, and demanded the victim's car keys. The victim told the suspects that the keys were in his car, at which time the suspects got into the car and drove away. The victim was not injured.

The suspects are described as black and approximately 21 years old. The first suspect is further described as 5 feet 6 inches tall, weighing 150 pounds and wearing a white shirt and blue pants. The second suspect is described as five feet seven inches tall, weighing 160 pounds and also wearing a white shirt and blue pants. The victim's vehicle is a white 2003 Hyundai Accent 4-door, with Pennsylvania license plate: 123 QWE. Anyone with information about this carjacking, the suspects or upon seeing the vehicle is asked to call Bethlehem Police at 610-758-3737.

Feature Type	Certainty	Feature Value	Range Start	Range
Height	0.800...	5 feet 6 inc...	66	66
Clothing	1.000...	a white shirt...	a white sh...	a white
Clothing	1.000...	also wearin...	also wear...	also we
Vehicle	1.000...	car	car	car
Vehicle	1.000...	Hyundai	Hyundai	Hyunda
OffenseType	0.700...	carjacking	carjacking	carjack
Weight	1.000...	150	150	150
Weight	1.000...	160	160	160
DayOfWeek	0.900...	Tuesday	2	2
Age	0.900...	23	23	23
Age	0.900...	21	21	21
PhoneNum...	0.900...	610-758-3737	61075837...	61075
Race	0.900...	black	black	black
PersonName	0.900...	Miller	Miller	Miller
PersonName	0.900...	Pennsylvania	Pennsylv...	Pennsy

Locate the feature selected in the document's original text

Highlight Features Result



D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\carjacking.txt D:\Backup\NIJ Project\BPD_system\System\bin\Dat
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010203.7,010203,Patriots and Police.txt D:\Backup\NIJ Project\BPD_
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010204. swastikas painted on Lancia Drive.txt
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010204wspgalleraction.txt D:\Backup\NIJ Project\BPD_
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010504ABDUCT1.txt D:\Backup\NIJ Project\BPD_system\System\bin\Dat

Officer Miller is investigating a carjacking which occurred in the Packer Lab, located in the Packer Avenue in Bethlehem at 0858 hours Tuesday. The victim, a 23-year old man, was picking up a friend when he was approached by two men, both of whom brandished handguns, and demanded the victim's car keys. The victim told the suspects that the keys were in his car, at which time the suspects got into the car and drove away. The victim was not injured.

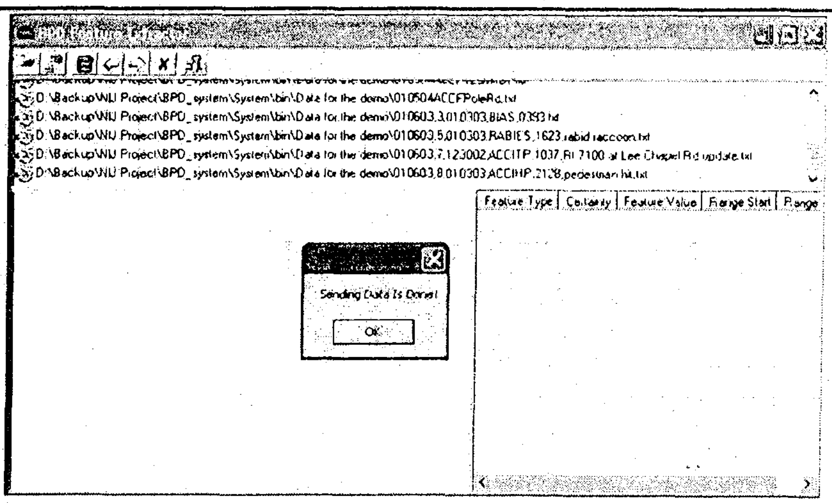
The suspects are described as black and approximately 21 years old. The first suspect is further described as 5 feet 6 inches tall, weighing 150 pounds and wearing a white shirt and blue pants. The second suspect is described as five feet seven inches tall, weighing 160 pounds and also wearing a white shirt and blue pants. The victim's vehicle is a white 2003 Hyundai Accent 4-door, with Pennsylvania license plates 123-QWE. Anyone with information about this carjacking, the suspects or upon seeing the vehicle is asked to call Bethlehem Police at 610-758-3737.

Feature Type	Certainty	Feature Value	Range Start	Range
Height	0.900	5 feet 6 inc...	66	66
Clothing	1.000...	a white shirt...	a white sh...	a white
Clothing	1.000...	also wearin...	also weari...	also we
Vehicle	1.000...	car	car	car
Vehicle	1.000...	Hyundai	Hyundai	Hyunda
OffenseType	0.700...	carjacking	carjacking	carjack
Weight	1.000...	150	150	150
Weight	1.000...	160	160	160
DayOfWeek	0.900...	Tuesday	2	2
Age	0.900...	23	23	23
Age	0.900...	21	21	21
PhoneNum...	0.900...	610-758-3737	61075837...	610758
Race	0.900...	black	black	black
PersonName	0.900...	Miller	Miller	Miller
PersonName	0.900...	Pennsylvania	Pennsylv...	Pennsy

The feature will be highlighted

1/25/2005 14

Send to server



D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010504ACCFPoleRd.txt
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603.3,010303,BIAS.0393.txt
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603.5,010303,RABIES.1623.rabid raccoon.txt
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603.7.123002,ACCITP.1037,Rt 7100 at Lee Chapel Rd update.txt
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603.8.010303,ACCIHP.2128,pedestrian hit.txt

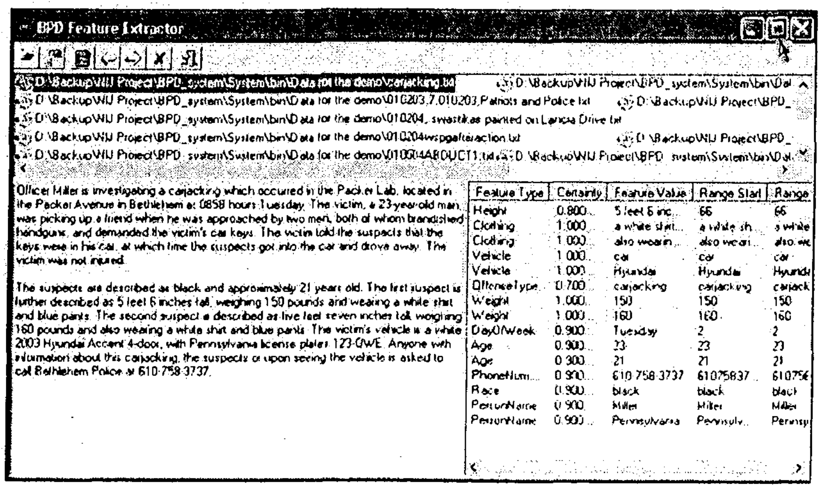
Feature Type	Certainty	Feature Value	Range Start	Range

Sending Data Is Done!

Send all documents with text and features to the BPD_IE database on the server

1/25/2005 15

Resize - Maximize



BPD Feature Extractor

```
D:\Backup\WU Project\BPD_system\System\bin\Data for the demo\carjacking.txt          D:\Backup\WU Project\BPD_system\System\bin\Dal
D:\Backup\WU Project\BPD_system\System\bin\Data for the demo\010203,7,010203,Patriots and Police.txt     D:\Backup\WU Project\BPD_
D:\Backup\WU Project\BPD_system\System\bin\Data for the demo\010204, swastikas painted on Lancia Drive.txt
D:\Backup\WU Project\BPD_system\System\bin\Data for the demo\010204wspgafteraction.txt                D:\Backup\WU Project\BPD_
D:\Backup\WU Project\BPD_system\System\bin\Data for the demo\010004ABDUCT1.txt  D:\Backup\WU Project\BPD_system\System\bin\Dal
```

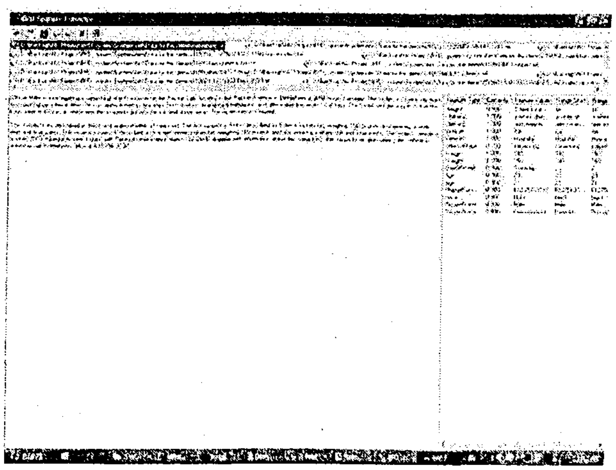
Officer Miller is investigating a carjacking which occured in the Packer Lab, located in the Packer Avenue in Bethlehem at 0858 hours Tuesday. The victim, a 23-year-old man, was picking up a friend when he was approached by two men, both of whom brandished handguns, and demanded the victim's car keys. The victim told the suspects that the keys were in his car, at which time the suspects got into the car and drove away. The victim was not injured.

The suspects are described as black and approximately 21 years old. The first suspect is further described as 5 feet 6 inches tall, weighing 150 pounds and wearing a white shirt and blue pants. The second suspect is described as five feet seven inches tall, weighing 160 pounds and also wearing a white shirt and blue pants. The victim's vehicle is a white 2003 Hyundai Accent 4-door, with Pennsylvania license plate: 123-QWE. Anyone with information about this carjacking, the suspects or upon seeing the vehicle is asked to call Bethlehem Police at 610-758-3737.

Feature Type	Certainty	Feature Value	Range Start	Range
Height	0.800...	5 feet 6 inc...	66	66
Clothing	1.000...	a white shirt...	a white sh...	a white
Clothing	1.000...	also wearin...	also wear...	also we
Vehicle	1.000...	car	car	car
Vehicle	1.000...	Hyundai	Hyundai	Hyundai
OffenseType	0.700...	carjacking	carjacking	carjack
Weight	1.000...	150	150	150
Weight	1.000...	160	160	160
DayOfWeek	0.900...	Tuesday	2	2
Age	0.900...	23	23	23
Age	0.300...	21	21	21
PhoneNum...	0.900...	610-758-3737	61075837...	610756
Race	0.900...	black	black	black
PersonName	0.900...	Miller	Miller	Miller
PersonName	0.900...	Pennsylvania	Pennsylv...	Pennsy

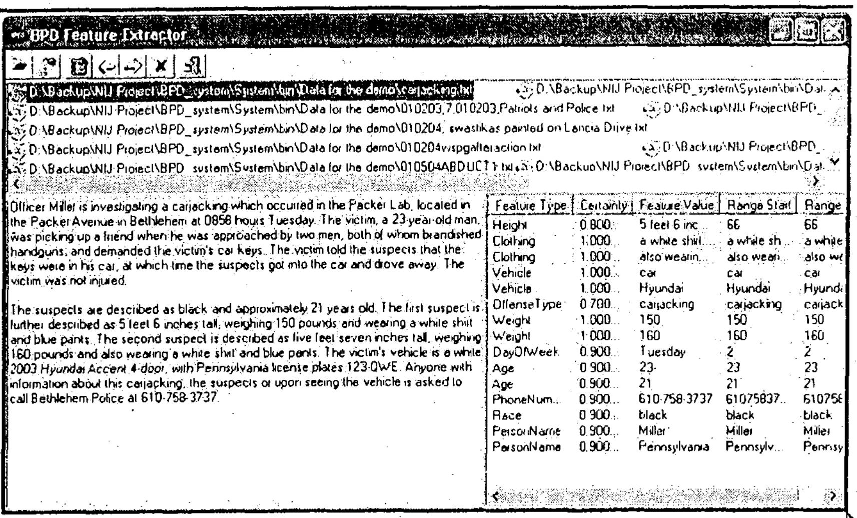
To Maximize the application window.

Resize – Maximize Result



All components are resized automatically

Resize - Manual



BPD Feature Extractor

```
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\carjacking.txt          D:\Backup\NIJ Project\BPD_system\System\bin\Dat
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010203,7,010203,Patriots and Police.txt     D:\Backup\NIJ Project\BPD_
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010204, swastikas painted on Lancia Drive.txt
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010204vvspgalteraction.txt          D:\Backup\NIJ Project\BPD_
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010504ABDUCT1.txt   D:\Backup\NIJ Project\BPD_system\System\bin\Dat
```

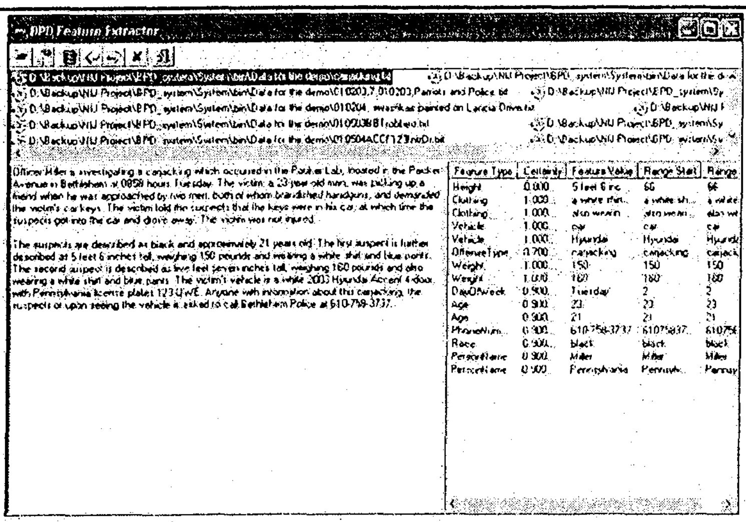
Officer Miller is investigating a carjacking which occurred in the Packer Lab, located in the Packer Avenue in Bethlehem at 0858 hours Tuesday. The victim, a 23-year-old man, was picking up a friend when he was approached by two men, both of whom brandished handguns, and demanded the victim's car keys. The victim told the suspects that the keys were in his car, at which time the suspects got into the car and drove away. The victim was not injured.

The suspects are described as black and approximately 21 years old. The first suspect is further described as 5 feet 6 inches tall, weighing 150 pounds and wearing a white shirt and blue pants. The second suspect is described as five feet seven inches tall, weighing 160 pounds and also wearing a white shirt and blue pants. The victim's vehicle is a white *2003 Hyundai Accent 4-door*, with Pennsylvania license plates 123-QWE. Anyone with information about this carjacking, the suspects or upon seeing the vehicle is asked to call Bethlehem Police at 610-758-3737.

Feature Type	Certainty	Feature Value	Range Start	Range
Height	0.800...	5 feet 6 inc...	66	66
Clothing	1.000...	a white shirt...	a white sh...	a white
Clothing	1.000...	also wearin...	also wear...	also we
Vehicle	1.000...	car	car	car
Vehicle	1.000...	Hyundai	Hyundai	Hyund
OffenseType	0.700...	carjacking	carjacking	carjack
Weight	1.000...	150	150	150
Weight	1.000...	160	160	160
DayOfWeek	0.900...	Tuesday	2	2
Age	0.900...	23	23	23
Age	0.900...	21	21	21
PhoneNum...	0.900...	610-758-3737	61075837...	610758
Race	0.900...	black	black	black
PersonName	0.900...	Miller	Miller	Miller
PersonName	0.900...	Pennsylvania	Pennsylv...	Pennsy

To manually resize the application window.

Resize – Manual Result



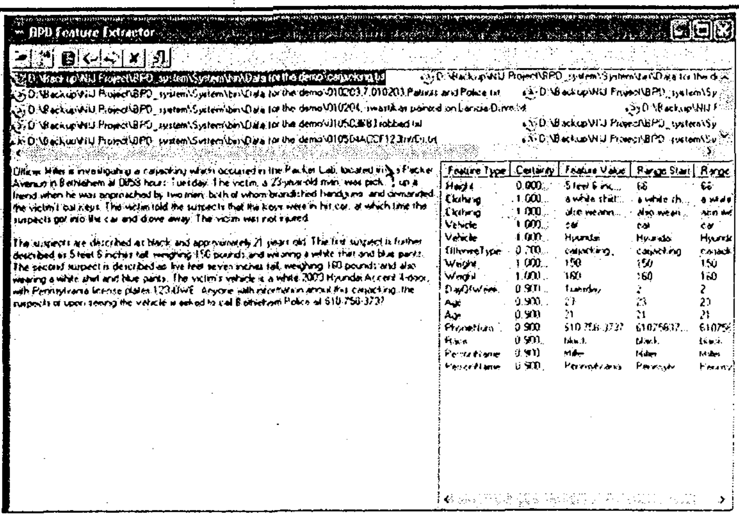
BPD Feature Extractor

Officer Miller is investigating a carjacking which occurred in the Packer Lab, located in the Packer Avenue in Bethlehem at 0858 hours Tuesday. The victim, a 23-year-old man, was picking up a friend when he was approached by two men, both of whom brandished handguns, and demanded the victim's car keys. The victim told the suspects that the keys were in his car, at which time the suspects got into the car and drove away. The victim was not injured.

The suspects are described as black and approximately 21 years old. The first suspect is further described as 5 feet 6 inches tall, weighing 150 pounds and wearing a white shirt and blue pants. The second suspect is described as five feet seven inches tall, weighing 160 pounds and also wearing a white shirt and blue pants. The victim's vehicle is a white 2003 Hyundai Accent 4-door, with Pennsylvania license plates 123-QWE. Anyone with information about this carjacking, the suspects or upon seeing the vehicle is asked to call Bethlehem Police at 610-758-3737.

All components are resized automatically

Resize – Horizontal Splitter



BPD Feature Extractor

```
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\carjacking.txt        D:\Backup\NIJ Project\BPD_system\System\bin\Data for the d
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010203,7,010203,Patriots and Police.txt        D:\Backup\NIJ Project\BPD_system\Sy
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010204, swastikas painted on Lancia Drive.txt        D:\Backup\NIJ P
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\0105030B1 robbed.txt        D:\Backup\NIJ Project\BPD_system\Sy
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010504ACCF123nnDr.txt        D:\Backup\NIJ Project\BPD_system\Sy
```

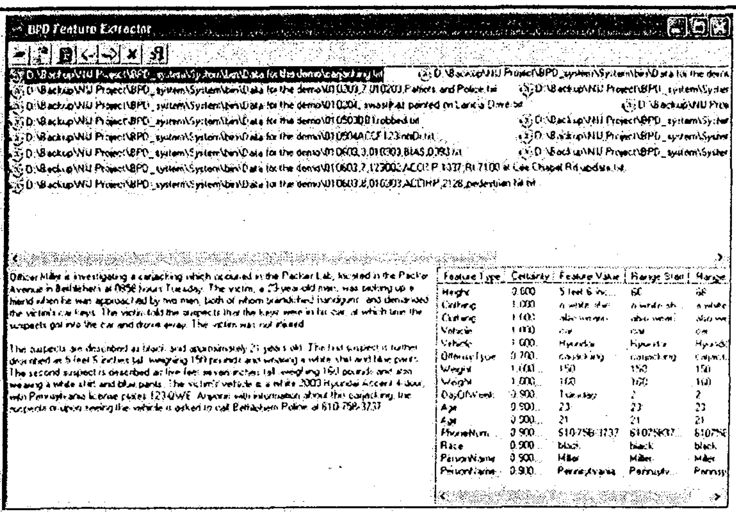
Officer Miller is investigating a carjacking which occurred in the Packer Lab, located in the Packer Avenue in Bethlehem at 0858 hours Tuesday. The victim, a 23-year-old man, was pick up a friend when he was approached by two men, both of whom brandished handguns, and demanded the victim's car keys. The victim told the suspects that the keys were in his car, at which time the suspects got into the car and drove away. The victim was not injured.

The suspects are described as black and approximately 21 years old. The first suspect is further described as 5 feet 6 inches tall, weighing 150 pounds and wearing a white shirt and blue pants. The second suspect is described as five feet seven inches tall, weighing 160 pounds and also wearing a white shirt and blue pants. The victim's vehicle is a white 2003 Hyundai Accent 4-door, with Pennsylvania license plates 123-QWE. Anyone with information about this carjacking, the suspects or upon seeing the vehicle is asked to call Bethlehem Police at 610-758-3737.

Feature Type	Certainty	Feature Value	Range Start	Range
Height	0.900...	5 feet 6 inc...	66	66
Clothing	1.000...	a white shirt...	a white sh...	a white
Clothing	1.000...	also wearin...	also wear...	also we
Vehicle	1.000...	car	car	car
Vehicle	1.000...	Hyundai	Hyundai	Hyundai
OffenseType	0.700...	carjacking,	carjacking	carjack
Weight	1.000...	150	150	150
Weight	1.000...	160	160	160
DayOfWeek	0.900...	Tuesday	2	2
Age	0.900...	23	23	23
Age	0.900	21	21	21
PhoneNum	0.900	510-758-3737	61075837...	610758
Race	0.900...	black	black	black
PersonName	0.900	Miller	Miller	Miller
PersonName	0.900...	Pennsylvania	Pennsylv...	Pennsy

Change the size of the document name listview

Resize – Horizontal Splitter Result



BPD Feature Extractor

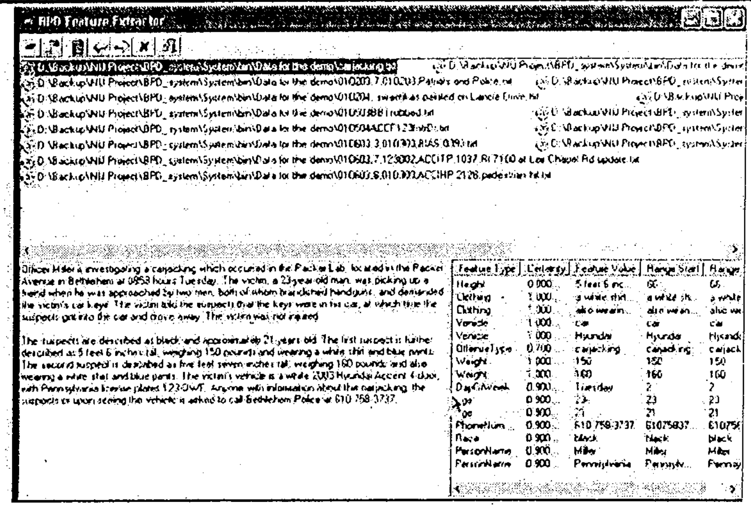
```
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\carjacking.txt        D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010203,7,010203,Patriots and Police.txt        D:\Backup\NIJ Project\BPD_system\Syster
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010204, swastikas painted on Lancia Drive.txt        D:\Backup\NIJ Proje
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\01050301 robbed.txt        D:\Backup\NIJ Project\BPD_system\Syster
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010504ACCF123nnDr.txt        D:\Backup\NIJ Project\BPD_system\Syster
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603,3,010303,BIAS,0393.txt        D:\Backup\NIJ Project\BPD_system\Syster
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603,7,123002,ACCP,P1037,Rt 7100 at Cee Chapel Rd update.txt
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603,8,010303,ACCIHP,2128,pedestrian hit.txt
```

Officer Miller is investigating a carjacking which occurred in the Packer Lab, located in the Packer Avenue in Bethlehem at 0858 hours Tuesday. The victim, a 23-year-old man, was picking up a friend when he was approached by two men, both of whom brandished handguns, and demanded the victim's car keys. The victim told the suspects that the keys were in his car, at which time the suspects got into the car and drove away. The victim was not injured.

The suspects are described as black and approximately 21 years old. The first suspect is further described as 5 feet 6 inches tall, weighing 150 pounds and wearing a white shirt and blue pants. The second suspect is described as five feet seven inches tall, weighing 160 pounds and also wearing a white shirt and blue pants. The victim's vehicle is a white 2003 Hyundai Accent 4-door, with Pennsylvania license plates 123-QWE. Anyone with information about this carjacking, the suspects or upon seeing the vehicle is asked to call Bethlehem Police at 610-758-3737.

Feature Type	Certainty	Feature Value	Range Start	Range
Height	0.900	5 feet 6 inc...	66	66
Clothing	1.000	a white shirt	a white sh...	a white
Clothing	1.000	also wearin...	also wear...	also we
Vehicle	1.000	car	car	car
Vehicle	1.000.	Hyundai	Hyundai	Hyundai
OffenseType	0.700.	carjacking	carjacking	carjack
Weight	1.000...	150	150	150
Weight	1.000...	160	160	160
DayOfWeek	0.900.	Tuesday	2	2
Age	0.900...	23	23	23
Age	0.900...	21	21	21
PhoneNum	0.900	510-758-3737	61075837...	610758
Race	0.900...	black	black	black
PersonName	0.900...	Miller	Miller	Miller
PersonName	0.900	Pennsylvania	Pennsylv...	Pennsy

Resize – Vertical Splitter


BPD Feature Extractor

```
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\carjacking.txt        D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010203,7,010203,Patriots and Police.txt        D:\Backup\NIJ Project\BPD_system\Syster
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010204, swastikas painted on Lancia Drive.txt        D:\Backup\NIJ Proj
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\01050388 I robbed.txt        D:\Backup\NIJ Project\BPD_system\Syster
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010504ACCF123tnbDr.txt        D:\Backup\NIJ Project\BPD_system\Syster
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603,3,010303,BIAS,0393.txt        D:\Backup\NIJ Project\BPD_system\Syster
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603,7,123002,ACCITP,1037,Rt 7100 at Lee Chapel Rd update.txt
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603,8,010303,ACCIHP,2128,pedestrian hit.txt
```

Officer Miller is investigating a carjacking which occurred in the Packer Lab, located in the Packer Avenue in Bethlehem at 0858 hours Tuesday. The victim, a 23-year-old man, was picking up a friend when he was approached by two men, both of whom brandished handguns, and demanded the victim's car keys. The victim told the suspects that the keys were in his car, at which time the suspects got into the car and drove away. The victim was not injured.

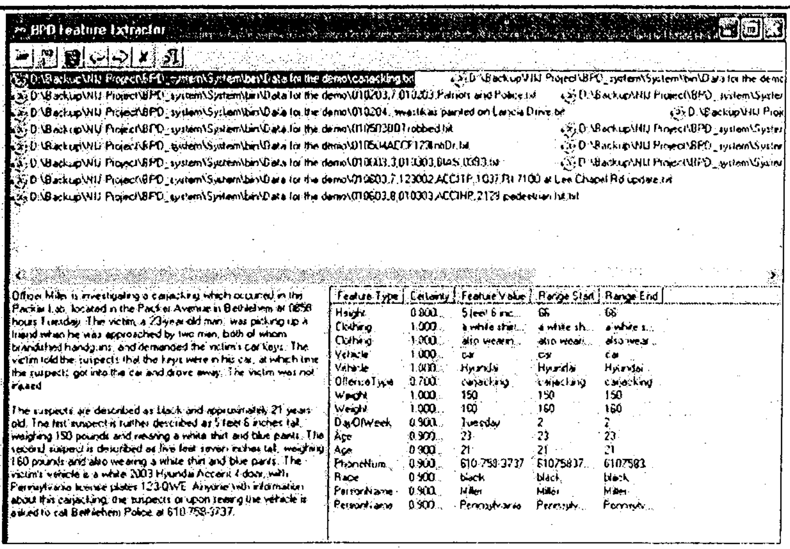
The suspects are described as black and approximately 21 years old. The first suspect is further described as 5 feet 6 inches tall, weighing 150 pounds and wearing a white shirt and blue pants. The second suspect is described as five feet seven inches tall, weighing 160 pounds and also wearing a white shirt and blue pants. The victim's vehicle is a white 2003 Hyundai Accent 4-door, with Pennsylvania license plates 123-QWE. Anyone with information about this carjacking, the suspects or upon seeing the vehicle is asked to call Bethlehem Police at 610-758-3737.

Feature Type	Certainty	Feature Value	Range Start	Range
Height	0.000...	5 feet 6 inc...	66	66
Clothing	1.000...	a white shirt...	a white sh...	a white
Clothing	1.000...	also wearin...	also wean...	also we
Vehicle	1.000...	car	car	car
Vehicle	1.000...	Hyundai	Hyundai	Hyund
OffenseType	0.700...	carjacking	carjacking	carjack
Weight	1.000...	150	150	150
Weight	1.000...	160	160	160
DayOfWeek	0.900...	Tuesday	2	2
Age	0.900...	23	23	23
Age	0.900...	21	21	21
PhoneNum...	0.900...	610-758-3737	61075837...	610758
Race	0.900...	black	black	black
PersonName	0.900...	Miller	Miller	Miller
PersonName	0.900...	Pennsylvania	Pennsylv...	Pennsy

Change the size of the feature listview

1/25/2005 22

Resize – Vertical Splitter Result


BPD Feature Extractor

```
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\carjacking.txt        D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010203,7,010203,Patriots and Police.txt        D:\Backup\NIJ Project\BPD_system\Syster
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010204, swastikas painted on Lancia Drive.txt        D:\Backup\NIJ Proj
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\01050388 I robbed.txt        D:\Backup\NIJ Project\BPD_system\Syster
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010504ACCF123tnbDr.txt        D:\Backup\NIJ Project\BPD_system\Syster
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603,3,010303,BIAS,0393.txt        D:\Backup\NIJ Project\BPD_system\Syster
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603,7,123002,ACCITP,1037,Rt 7100 at Lee Chapel Rd update.txt
D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603,8,010303,ACCIHP,2129,pedestrian hit.txt
```

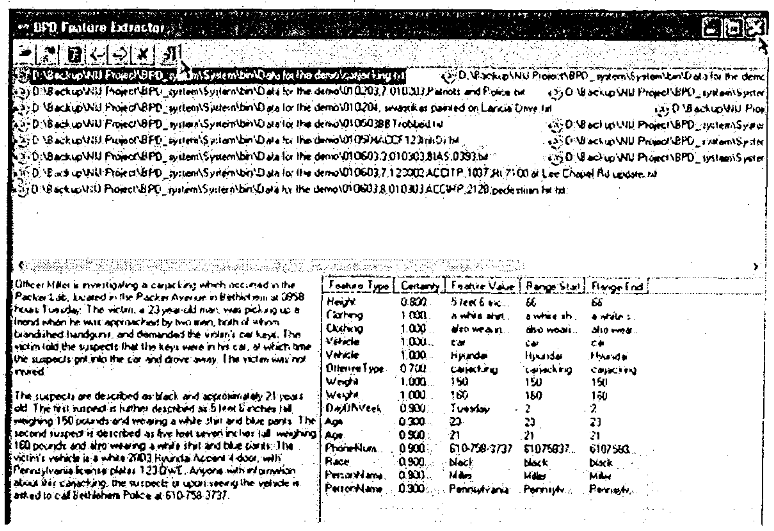
Officer Miller is investigating a carjacking which occurred in the Packer Lab, located in the Packer Avenue in Bethlehem at 0858 hours Tuesday. The victim, a 23-year-old man, was picking up a friend when he was approached by two men, both of whom brandished handguns, and demanded the victim's car keys. The victim told the suspects that the keys were in his car, at which time the suspects got into the car and drove away. The victim was not injured.

The suspects are described as black and approximately 21 years old. The first suspect is further described as 5 feet 6 inches tall, weighing 150 pounds and wearing a white shirt and blue pants. The second suspect is described as five feet seven inches tall, weighing 160 pounds and also wearing a white shirt and blue pants. The victim's vehicle is a white 2003 Hyundai Accent 4-door, with Pennsylvania license plates 123-QWE. Anyone with information about this carjacking, the suspects or upon seeing the vehicle is asked to call Bethlehem Police at 610-758-3737.

Feature Type	Certainty	Feature Value	Range Start	Range End
Height	0.800...	5 feet 6 inc...	66	66
Clothing	1.000...	a white shirt...	a white sh...	a white s...
Clothing	1.000...	also wearin...	also weari...	also wear...
Vehicle	1.000...	car	car	car
Vehicle	1.000...	Hyundai	Hyundai	Hyundai
OffenseType	0.700...	carjacking	carjacking	carjacking
Weight	1.000...	150	150	150
Weight	1.000...	160	160	160
DayOfWeek	0.900...	Tuesday	2	2
Age	0.900...	23	23	23
Age	0.900...	21	21	21
PhoneNum...	0.900...	610-758-3737	61075837...	6107583
Race	0.900...	black	black	black
PersonName	0.900...	Miller	Miller	Miller
PersonName	0.900...	Pennsylvania	Pennsylv...	Pennsylv...

1/25/2005 23

Exit



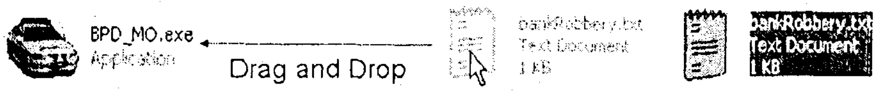
```
D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\carjacking.txt      D:\Backup\NU Project\BPD_system\System\bin\Data for the demo
D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010203.7.010303.Patriots and Police.txt      D:\Backup\NU Project\BPD_system\Syster
D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010204, swastikas painted on Lancia Drive.txt      D:\Backup\NU Proj
D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\01060388 Trobbed.txt      D:\Backup\NU Project\BPD_system\Syste
D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\0109HACCF123nhD.txt      D:\Backup\NU Project\BPD_system\Syster
D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010603.2.010303.BIAS.0393.txt      D:\Backup\NU Project\BPD_system\Syster
D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010603.7.122002.ACCITP.1007.Rt 7100 at Lee Chapel Rd update.txt
D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010603.8.010303.ACCWP.2128.pedestrian hit.txt
```

Officer Miller is investigating a carjacking which occurred in the Packer Lab, located in the Packer Avenue in Bethlehem at 0958 hours Tuesday. The victim, a 23 year-old man, was picking up a friend when he was approached by two men, both of whom brandished handguns, and demanded the victim's car keys. The victim told the suspects that the keys were in his car, at which time the suspects got into the car and drove away. The victim was not injured.

The suspects are described as black and approximately 21 years old. The first suspect is further described as 5 feet 6 inches tall weighing 150 pounds and wearing a white shirt and blue pants. The second suspect is described as five feet seven inches tall weighing 160 pounds and also wearing a white shirt and blue pants. The victim's vehicle is a white 2003 Hyundai Accent 4-door, with Pennsylvania license plates 123DWE. Anyone with information about this carjacking, the suspects or upon seeing the vehicle is asked to call Bethlehem Police at 610-758-3737.

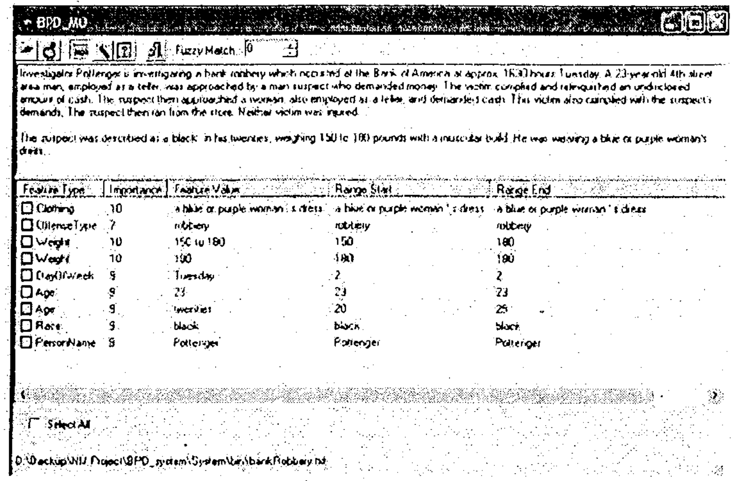
Feature Type	Certainty	Feature Value	Range Start	Range End
Height	0.800	5 feet 6 inc...	66	66
Clothing	1.000	a white shirt	a white sh...	a white s...
Clothing	1.000	also wearin...	also weari...	also wear...
Vehicle	1.000	car	car	car
Vehicle	1.000	Hyundai	Hyundai	Hyundai
OffenseType	0.700	carjacking	carjacking	carjacking
Weight	1.000	150	150	150
Weight	1.000	160	160	160
DayOfWeek	0.900	Tuesday	2	2
Age	0.300	23	23	23
Age	0.900	21	21	21
PhoneNum	0.900	610-758-3737	61075837...	6107583...
Race	0.900	black	black	black
PersonName	0.900	Miller	Miller	Miller
PersonName	0.300	Pennsylvania	Pennsylv...	Pennsylv...

BPD_MO



One document each time

BPD_MO GUI



BPD_MO

Fuzzy Match 0

Investigator Pottenger is investigating a bank robbery which occurred at the Bank of America at approx. 1630 hours Tuesday. A 23-year-old 4th street area man, employed as a teller, was approached by a man suspect who demanded money. The victim complied and relinquished an undisclosed amount of cash. The suspect then approached a woman, also employed as a teller, and demanded cash. This victim also complied with the suspect's demands. The suspect then ran from the store. Neither victim was injured.

The suspect was described as a black, in his twenties, weighing 150 to 180 pounds with a muscular build. He was wearing a blue or purple woman's dress.

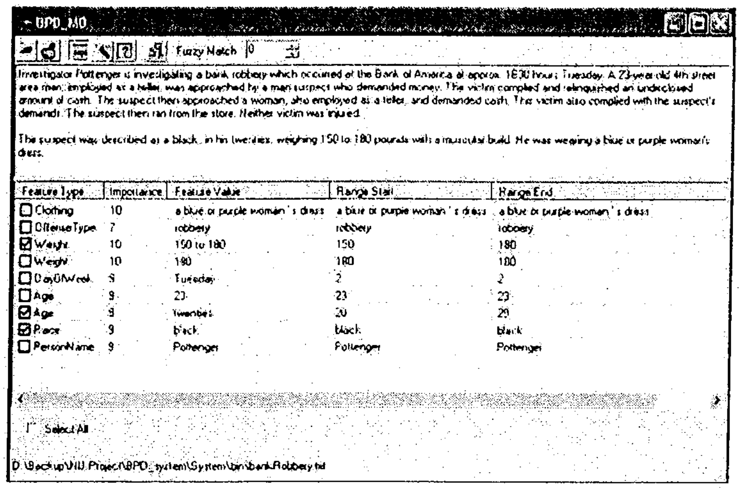
Feature Type	Importance	Feature Value	Range Start	Range End
☐ Clothing	10	a blue or purple woman ' s dress	a blue or purple woman ' s dress	a blue or purple woman ' s dress
☐ OffenseType	7	robbery	robbery	robbery
☐ Weight	10	150 to 180	150	180
☐ Weight	10	180	180	180
☐ DayOfWeek	9	Tuesday	2	2
☐ Age	9	23	23	23
☐ Age	9	twenties	20	29
☐ Race	9	black	black	black
☐ PersonName	9	Pottenger	Pottenger	Pottenger

☐ Select All

D:\Backup\NIJ Project\BPD_system\System\bin\bankRobbery.txt

The document's original text and features extracted are displayed

Search



BPD_MO

Fuzzy Match 0

Investigator Pottenger is investigating a bank robbery which occurred at the Bank of America at approx. 1630 hours Tuesday. A 23-year-old 4th street area man, employed as a teller, was approached by a man suspect who demanded money. The victim complied and relinquished an undisclosed amount of cash. The suspect then approached a woman, also employed as a teller, and demanded cash. This victim also complied with the suspect's demands. The suspect then ran from the store. Neither victim was injured.

The suspect was described as a black, in his twenties, weighing 150 to 180 pounds with a muscular build. He was wearing a blue or purple woman's dress.

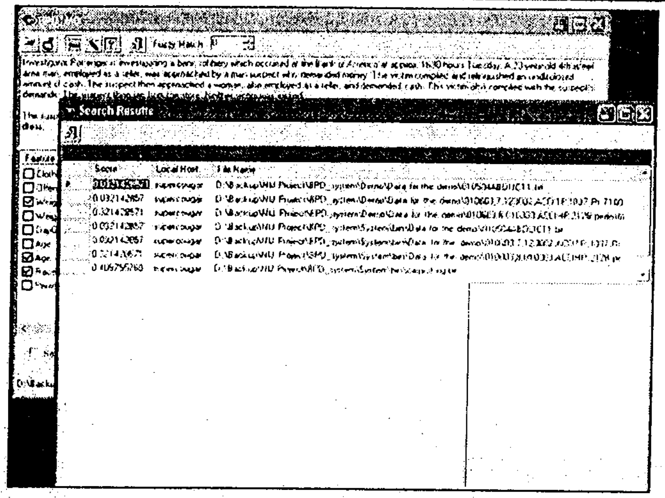
Feature Type	Importance	Feature Value	Range Start	Range End
☐ Clothing	10	a blue or purple woman ' s dress	a blue or purple woman ' s dress	a blue or purple woman ' s dress
☐ OffenseType	7	robbery	robbery	robbery
☑ Weight	10	150 to 180	150	180
☐ Weight	10	180	180	180
☐ DayOfWeek	9	Tuesday	2	2
☐ Age	9	23	23	23
☑ Age	9	twenties	20	29
☑ Race	9	black	black	black
☐ PersonName	9	Pottenger	Pottenger	Pottenger

☐ Select All

D:\Backup\NIJ Project\BPD_system\System\bin\bankRobbery.txt

Select features as search terms

Search Result



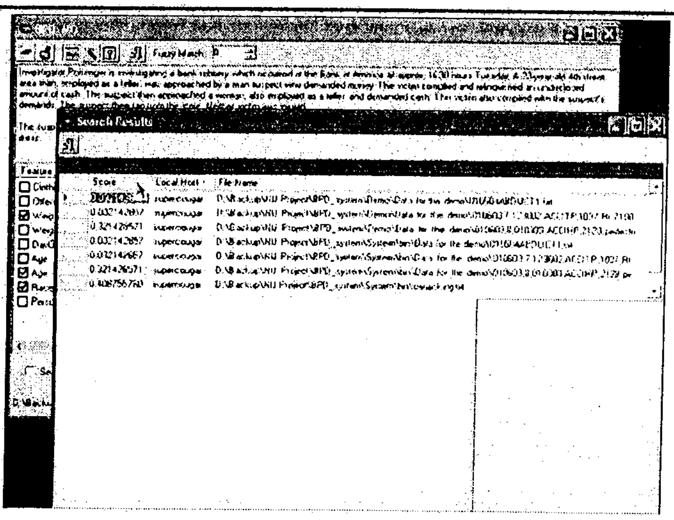
Investigator Potenger is investigating a bank robbery which occurred at the Bank of America at approx. 1630 hours Tuesday. A 23 year old 4th street area man, employed as a teller, was approached by a man suspect who demanded money. The victim complied and relinquished an undisclosed amount of cash. The suspect then approached a woman, also employed as a teller, and demanded cash. The victim also complied with the suspect's demands. [illegible]

Score	Local Host	File Name
[illegible]	supercougar	D:\Backup\NIU Project\BPD_system\Demo\Data for the demo\0105044BDU0C11.txt
0.032142857	supercougar	D:\Backup\NIU Project\BPD_system\Demo\Data for the demo\010603 7.12 2002 ACD1P 1037 Pr 7100
0.321428571	supercougar	D:\Backup\NIU Project\BPD_system\Demo\Data for the demo\010603 8.010303 ACD1HP 2129 pedestri
0.032142857	supercougar	D:\Backup\NIU Project\BPD_system\System\bin\Data for the demo\0105044BDU0C11.txt
0.032142857	supercougar	D:\Backup\NIU Project\BPD_system\System\bin\Data for the demo\010603 7.12 2002 ACD1P 1037 Pr
0.321428571	supercougar	D:\Backup\NIU Project\BPD_system\System\bin\Data for the demo\010603 8.010303 ACD1HP 2129 pe
0.406756760	supercougar	D:\Backup\NIU Project\BPD_system\System\bin\[illegible]

Click the search button to search the BPD_IE database on the server

1/25/2005 28

Search Result - Sort



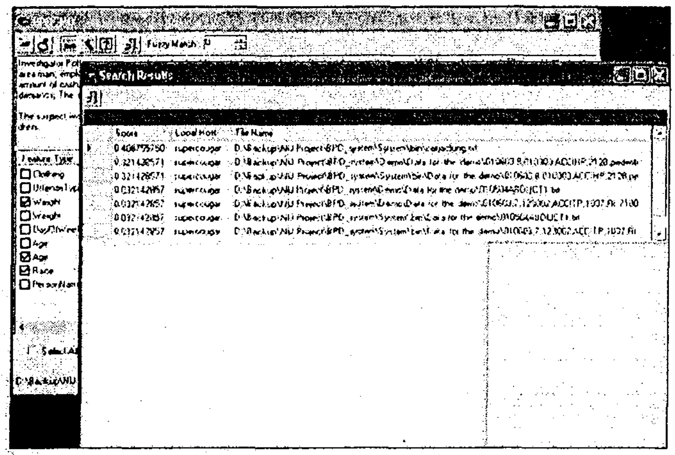
Investigator Potenger is investigating a bank robbery which occurred at the Bank of America at approx. 1630 hours Tuesday. A 23 year old 4th street area man, employed as a teller, was approached by a man suspect who demanded money. The victim complied and relinquished an undisclosed amount of cash. The suspect then approached a woman, also employed as a teller, and demanded cash. The victim also complied with the suspect's demands. [illegible]

Score	Local Host	File Name
[illegible]	supercougar	D:\Backup\NIU Project\BPD_system\Demo\Data for the demo\0105044BDU0C11.txt
0.032142857	supercougar	D:\Backup\NIU Project\BPD_system\Demo\Data for the demo\010603 7.12 2002 ACD1P 1037 Pr 7100
0.321428571	supercougar	D:\Backup\NIU Project\BPD_system\Demo\Data for the demo\010603 8.010303 ACD1HP 2129 pedestri
0.032142857	supercougar	D:\Backup\NIU Project\BPD_system\System\bin\Data for the demo\0105044BDU0C11.txt
0.032142857	supercougar	D:\Backup\NIU Project\BPD_system\System\bin\Data for the demo\010603 7.12 2002 ACD1P 1037 Pr
0.321428571	supercougar	D:\Backup\NIU Project\BPD_system\System\bin\Data for the demo\010603 8.010303 ACD1HP 2129 pe
0.406756760	supercougar	D:\Backup\NIU Project\BPD_system\System\bin\[illegible]

Click the "Score" header to sort the results

1/25/2005 29

Search Result – Sort Result



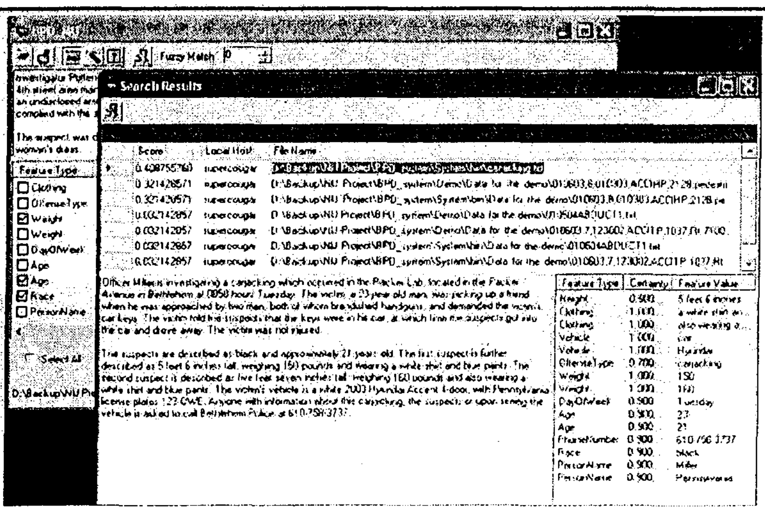
Search Results

Score	Local Host	File Name
0.406795750	supercougar	D:\Backup\NU Project\BPD_system\System\bin\carjacking.txt
0.321428571	supercougar	D:\Backup\NU Project\BPD_system\Demo\Data for the demo\010603 8,010303 ACCIHP 2128.pedestr
0.321428571	supercougar	D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010603 8 010303 ACCIHP 2128.pe
0.032142857	supercougar	D:\Backup\NU Project\BPD_system\Demo\Data for the demo\010504ABDUCT1.txt
0.032142857	supercougar	D:\Backup\NU Project\BPD_system\Demo\Data for the demo\010603 7,123002 ACCIP 1037 Rt 7100
0.032142857	supercougar	D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010504ABDUCT1.txt
0.032142857	supercougar	D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010603 7 123002 ACCIP 1037 Rt

The document with the highest score will be the most related document. In this case, the suspect's physical description in the carjacking and bank robbery incidents are similar.

1/25/2005 30

Search Result – Document Content



Search Results

Score	Local Host	File Name
0.406795760	supercougar	D:\Backup\NU Project\BPD_system\System\bin\carjacking.txt
0.321428571	supercougar	D:\Backup\NU Project\BPD_system\Demo\Data for the demo\010603,8,010303,ACCIHP,2128.pedestri
0.321420571	supercougar	D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010603,8,010303,ACCIHP,2128.pe
0.032142857	supercougar	D:\Backup\NU Project\BPD_system\Demo\Data for the demo\010504ABDUCT1.txt
0.032142057	supercougar	D:\Backup\NU Project\BPD_system\Demo\Data for the demo\010603,7,123002,ACCIP,1037,Rt 7100
0.032142867	supercougar	D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010504ABDUCT1.txt
0.032142857	supercougar	D:\Backup\NU Project\BPD_system\System\bin\Data for the demo\010603,7,123002,ACCIP,1037,Rt

Officer Miller is investigating a carjacking which occurred in the Packer Lab, located in the Packer Avenue in Bethlehem at 0050 hours Tuesday. The victim, a 23-year old man, was picking up a friend when he was approached by two men, both of whom brandished handguns, and demanded the victim's car keys. The victim told the suspects that the keys were in his car, at which time the suspects got into the car and drove away. The victim was not injured.

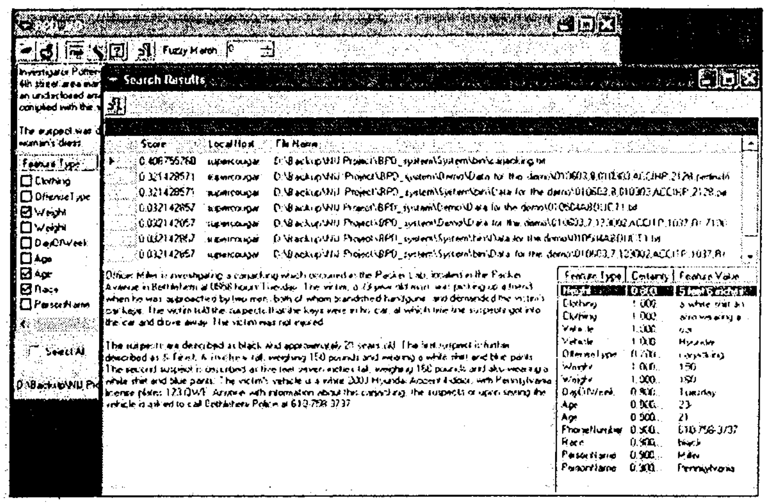
The suspects are described as black and approximately 21 years old. The first suspect is further described as 5 feet 6 inches tall, weighing 150 pounds and wearing a white shirt and blue pants. The second suspect is described as five feet seven inches tall, weighing 160 pounds and also wearing a white shirt and blue pants. The victim's vehicle is a white 2003 Hyundai Accent 4-door, with Pennsylvania license plates 123-CWE. Anyone with information about this carjacking, the suspects or upon seeing the vehicle is asked to call Bethlehem Police at 610-758-3737.

Feature Type	Certainty	Feature Value
Height	0.900	5 feet 6 inches
Clothing	1.000	a white shirt an...
Clothing	1.000	also wearing a...
Vehicle	1.000	car
Vehicle	1.000	Hyundai
OffenseType	0.700	carjacking
Weight	1.000	150
Weight	1.000	160
DayOfWeek	0.900	Tuesday
Age	0.900	23
Age	0.900	21
PhoneNumber	0.900	610 758 3737
Race	0.900	black
PersonName	0.900	Miller
PersonName	0.900	Pennsylvania

Click a document to show its original text and features

1/25/2005 31

Search Result
- Highlight Features



Fuzzy Match 0

Investigator Potter… 4th street area mar… an undisclosed am… complied with the…

The suspect was d… woman's dress

Search Results

Score	Local Host	File Name
0.406755760	supercougar	D:\Backup\NIJ Project\BPD_system\System\bin\carjacking.txt
0.321428571	supercougar	D:\Backup\NIJ Project\BPD_system\Demo\Data for the demo\010603,8,010303 ACCIHP 2128.pertinent
0.321428571	supercougar	D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603,8,010303 ACCIHP 2128.txt
0.032142857	supercougar	D:\Backup\NIJ Project\BPD_system\Demo\Data for the demo\0105044B0UC11.txt
0.032142857	supercougar	D:\Backup\NIJ Project\BPD_system\Demo\Data for the demo\010603,7,123002 ACCI1P 1037 R: 2130
0.032142857	supercougar	D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\0105044B0UC11.txt
0.032142857	supercougar	D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603,7,123002 ACCI1P 1037 R:

Feature Type
- [ ] Clothing
- [ ] OffenseType
- [✓] Weight
- [ ] Weight
- [ ] DayOfWeek
- [ ] Age
- [✓] Age
- [✓] Race
- [ ] PersonName

Select All

D:\Backup\NIJ Pr…

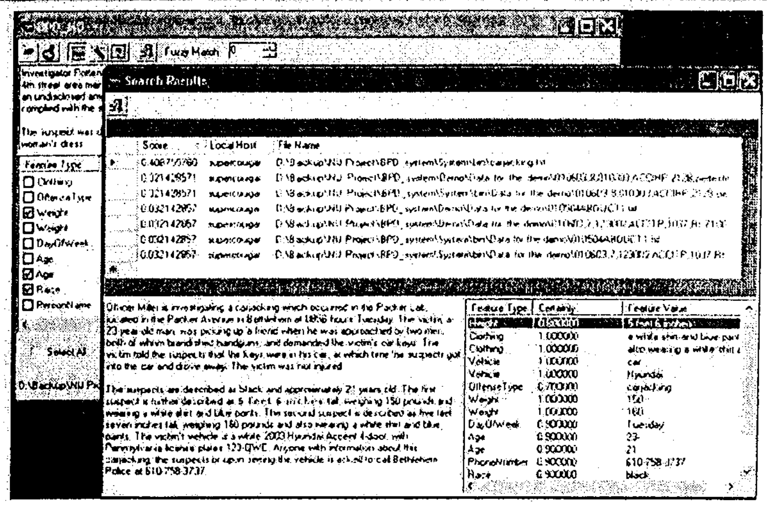
Officer Miles is investigating a carjacking which occurred in the Packer Lab, located in the Packer Avenue in Bethlehem at 0858 hours Tuesday. The victim, a 23-year-old man, was picking up a friend when he was approached by two men, both of whom brandished handguns, and demanded the victim's car keys. The victim told the suspects that the keys were in his car, at which time the suspects got into the car and drove away. The victim was not injured.

The suspects are described as black and approximately 21 years old. The first suspect is further described as 5 feet, 6 inches tall, weighing 150 pounds and wearing a white shirt and blue pants. The second suspect is described as five feet seven inches tall, weighing 160 pounds and also wearing a white shirt and blue pants. The victim's vehicle is a white 2003 Hyundai Accent 4-door, with Pennsylvania license plates 123-QWE. Anyone with information about this carjacking, the suspects or upon seeing the vehicle is asked to call Bethlehem Police at 610-758-3737.

Feature Type	Certainty	Feature Value
Height	0.800	5 feet 6 inches
Clothing	1.000	a white shirt an
Clothing	1.000	also wearing a
Vehicle	1.000	car
Vehicle	1.000	Hyundai
OffenseType	0.700	carjacking
Weight	1.000	150
Weight	1.000	160
DayOfWeek	0.900	Tuesday
Age	0.900	23
Age	0.900	21
PhoneNumber	0.900	610-758-3737
Race	0.900	black
PersonName	0.900	Miles
PersonName	0.900	Pennsylvania

Click a feature to highlight it in the document's original text

Search Result
- Splitters (Resize)



Fuzzy Match 0

Investigator Potter… 4th street area mar… an undisclosed am… complied with the…

The suspect was d… woman's dress

Search Results

Score	Local Host	File Name
0.406755760	supercougar	D:\Backup\NIJ Project\BPD_system\System\bin\carjacking.txt
0.321428571	supercougar	D:\Backup\NIJ Project\BPD_system\Demo\Data for the demo\010603,8,010303 ACCIHP 2128.pertinent
0.321428571	supercougar	D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603,8,010303 ACCIHP 2128.txt
0.032142857	supercougar	D:\Backup\NIJ Project\BPD_system\Demo\Data for the demo\0105044B0UC11.txt
0.032142857	supercougar	D:\Backup\NIJ Project\BPD_system\Demo\Data for the demo\010603,7,123002 ACCI1P 1037 R: 2130
0.032142857	supercougar	D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\0105044B0UC11.txt
0.032142857	supercougar	D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603,7,123002 ACCI1P 1037 R:

Feature Type
- [ ] Clothing
- [ ] OffenseType
- [✓] Weight
- [ ] Weight
- [ ] DayOfWeek
- [ ] Age
- [✓] Age
- [✓] Race
- [ ] PersonName

Select All

D:\Backup\NIJ Pr…

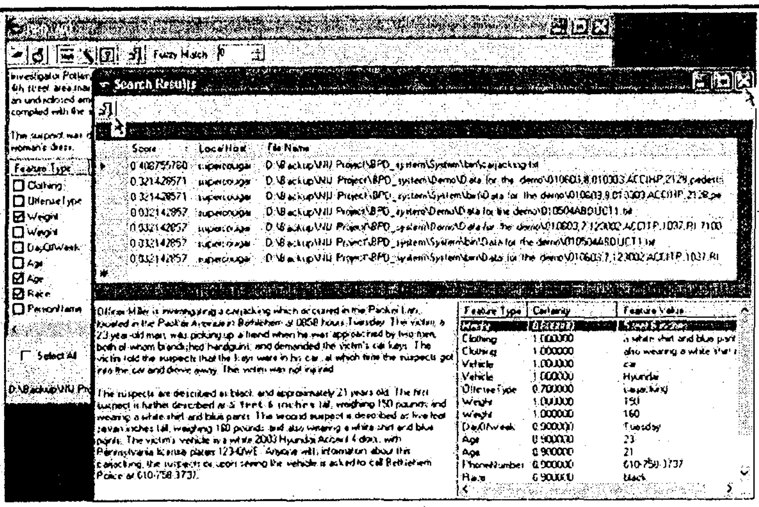
Officer Miles is investigating a carjacking which occurred in the Packer Lab, located in the Packer Avenue in Bethlehem at 0858 hours Tuesday. The victim, a 23-year-old man, was picking up a friend when he was approached by two men, both of whom brandished handguns, and demanded the victim's car keys. The victim told the suspects that the keys were in his car, at which time the suspects got into the car and drove away. The victim was not injured.

The suspects are described as Black and approximately 21 years old. The first suspect is further described as 5 feet, 6 inches tall, weighing 150 pounds and wearing a white shirt and blue pants. The second suspect is described as five feet seven inches tall, weighing 160 pounds and also wearing a white shirt and blue pants. The victim's vehicle is a white 2003 Hyundai Accent 4-door, with Pennsylvania license plates 123-QWE. Anyone with information about this carjacking, the suspects or upon seeing the vehicle is asked to call Bethlehem Police at 610-758-3737.

Feature Type	Certainty	Feature Value
Height	0.800000	5 feet 6 inches
Clothing	1.000000	a white shirt and blue pant
Clothing	1.000000	also wearing a white shirt a
Vehicle	1.000000	car
Vehicle	1.000000	Hyundai
OffenseType	0.700000	carjacking
Weight	1.000000	150
Weight	1.000000	160
DayOfWeek	0.900000	Tuesday
Age	0.900000	23
Age	0.900000	21
PhoneNumber	0.900000	610-758-3737
Race	0.900000	black

All components can be resized manually

Search Result - Close


Fuzzy Match 0

Search Results

Score	LocalHost	File Name
0.408755760	supercougar	D:\Backup\NIJ Project\BPD_system\System\bin\carjacking.txt
0.321428571	supercougar	D:\Backup\NIJ Project\BPD_system\Demo\Data for the demo\010603,8,010303,ACCIHP,2129,pedestr
0.321428571	supercougar	D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603,9,010303,ACCIHP,2128,pe
0.032142857	supercougar	D:\Backup\NIJ Project\BPD_system\Demo\Data for the demo\010504ABDUCT1.txt
0.032142857	supercougar	D:\Backup\NIJ Project\BPD_system\Demo\Data for the demo\010603,7,123002,ACCITP,1037,RI 7100
0.032142857	supercougar	D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010504ABDUCT1.txt
0.032142857	supercougar	D:\Backup\NIJ Project\BPD_system\System\bin\Data for the demo\010603,7,123002,ACCITP,1037,RI

Officer Miller is investigating a carjacking which occured in the Packer Lot, located in the Packer Avenue in Bethlehem at 0858 hours Tuesday. The victim, a 23-year-old man, was picking up a friend when he was approached by two men, both of whom brandished handguns and demanded the victim's car keys. The victim told the suspects that the keys were in his car, at which time the suspects got into the car and drove away. The victim was not injured.

The suspects are described as black and approximately 21 years old. The first suspect is further described as 5 feet, 6 inches tall, weighing 150 pounds and wearing a white shirt and blue pants. The second suspect is described as five feet seven inches tall, weighing 160 pounds and also wearing a white shirt and blue pants. The victim's vehicle is a white 2003 Hyundai Accent 4-door, with Pennsylvania license plates 1234DWE. Anyone with information about this carjacking, the suspects or upon seeing the vehicle is asked to call Bethlehem Police at 610-758-3737.

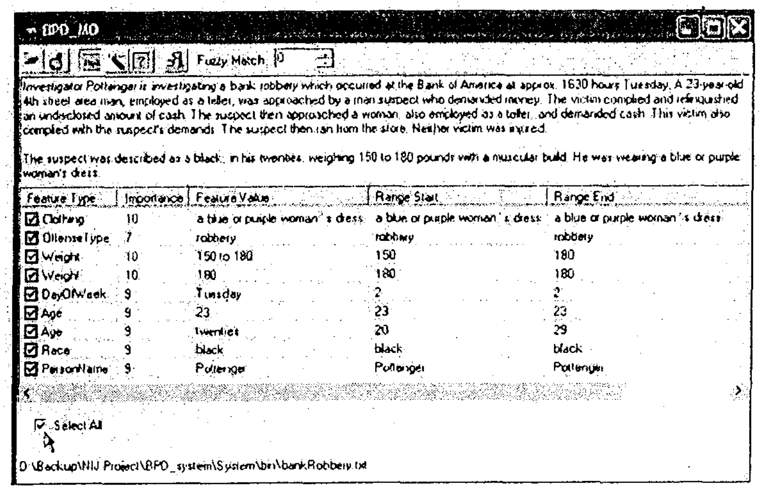
Feature Type	Certainty	Feature Value
[illegible]	[illegible]	[illegible]
Clothing	1.000000	a white shirt and blue pant
Clothing	1.000000	also wearing a white shirt
Vehicle	1.000000	car
Vehicle	1.000000	Hyundai
OffenseType	0.700000	carjacking
Weight	1.000000	150
Weight	1.000000	160
DayOfWeek	0.900000	Tuesday
Age	0.900000	23
Age	0.900000	21
PhoneNumber	0.900000	610-758-3737
Race	0.900000	black

Feature Type: ☐ Clothing, ☐ OffenseType, ☑ Weight, ☐ Weight, ☐ DayOfWeek, ☐ Age, ☑ Age, ☑ Race, ☐ PersonName

☐ Select All

Two ways to close the result window

Select All Features


BPD_MO

Fuzzy Match 0

Investigator Pottenger is investigating a bank robbery which occurred at the Bank of America at approx. 1630 hours Tuesday. A 23-year-old 4th street area man, employed as a teller, was approached by a man suspect who demanded money. The victim complied and relinquished an undisclosed amount of cash. The suspect then approached a woman, also employed as a teller, and demanded cash. This victim also complied with the suspect's demands. The suspect then ran from the store. Neither victim was injured.

The suspect was described as a black, in his twenties, weighing 150 to 180 pounds with a muscular build. He was wearing a blue or purple woman's dress.

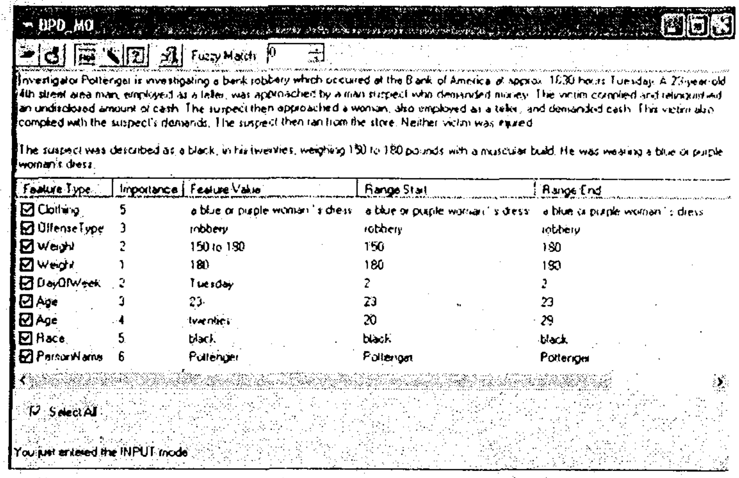
Feature Type	Importance	Feature Value	Range Start	Range End
☑ Clothing	10	a blue or purple woman ' s dress	a blue or purple woman ' s dress	a blue or purple woman ' s dress
☑ OffenseType	7	robbery	robbery	robbery
☑ Weight	10	150 to 180	150	180
☑ Weight	10	180	180	180
☑ DayOfWeek	9	Tuesday	2	2
☑ Age	9	23	23	23
☑ Age	9	twenties	20	29
☑ Race	9	black	black	black
☑ PersonName	9	Pottenger	Pottenger	Pottenger

☑ Select All

D:\Backup\NIJ Project\BPD_system\System\bin\bankRobbery.txt

All features can be selected and deselected easily

Search – Change Importance



Investigator Pottenger is investigating a bank robbery which occured at the Bank of America at approx. 1630 hours Tuesday. A 23-year-old 4th street area man, employed as a teller, was approached by a man suspect who demanded money. The victim complied and relinquished an undisclosed amount of cash. The suspect then approached a woman, also employed as a teller, and demanded cash. This victim also complied with the suspect's demands. The suspect then ran from the store. Neither victim was injured.

The suspect was described as a black, in his twenties, weighing 150 to 180 pounds with a muscular build. He was wearing a blue or purple woman's dress.

Feature Type	Importance	Feature Value	Range Start	Range End
☑ Clothing	5	a blue or purple woman 's dress	a blue or purple woman 's dress	a blue or purple woman 's dress
☑ OffenseType	3	robbery	robbery	robbery
☑ Weight	2	150 to 180	150	180
☑ Weight	1	180	180	180
☑ DayOfWeek	2	Tuesday	2	2
☑ Age	3	23	23	23
☑ Age	4	twenties	20	29
☑ Race	5	black	black	black
☑ PersonName	6	Pottenger	Pottenger	Pottenger

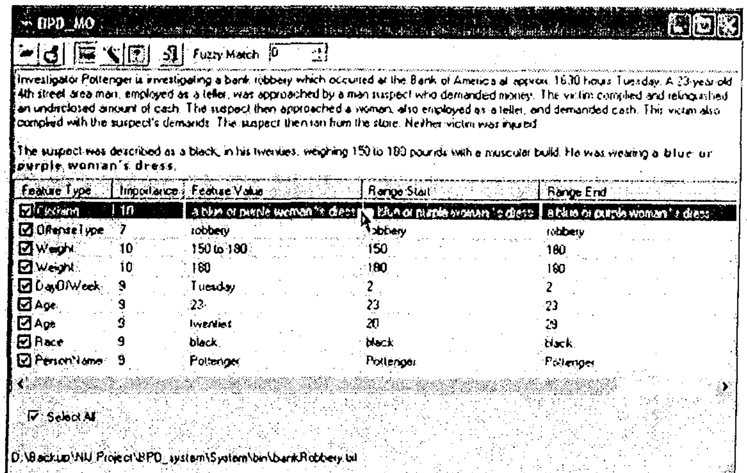
☑ Select All

You just entered the INPUT mode

Different features can have different importance during search.
The default values are derived from the rule developer.

1/25/2005 36

Highlight Features



Investigator Pottenger is investigating a bank robbery which occurred at the Bank of America at approx. 1630 hours Tuesday. A 23-year-old 4th street area man, employed as a teller, was approached by a man suspect who demanded money. The victim complied and relinquished an undisclosed amount of cash. The suspect then approached a woman, also employed as a teller, and demanded cash. This victim also complied with the suspect's demands. The suspect then ran from the store. Neither victim was injured.

The suspect was described as a black, in his twenties, weighing 150 to 180 pounds with a muscular build. He was wearing a **blue or purple woman's dress.**

Feature Type	Importance	Feature Value	Range Start	Range End
☑ Clothing	10	a blue or purple woman 's dress	a blue or purple woman 's dress	a blue or purple woman 's dress
☑ OffenseType	7	robbery	robbery	robbery
☑ Weight	10	150 to 180	150	180
☑ Weight	10	180	180	180
☑ DayOfWeek	9	Tuesday	2	2
☑ Age	9	23	23	23
☑ Age	9	twenties	20	29
☑ Race	9	black	black	black
☑ PersonName	9	Pottenger	Pottenger	Pottenger

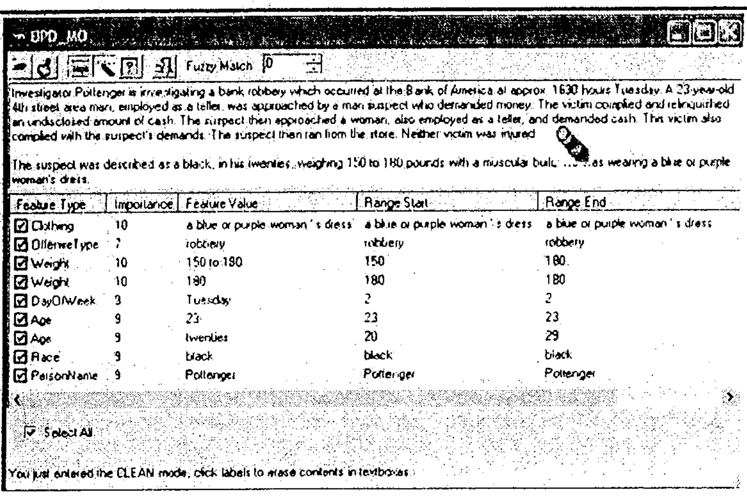
☑ Select All

D:\Backup\NIJ Project\BPD_system\System\bin\bankRobbery.txt

Click a feature to highlight it in the original text.
Different features have different colors.

1/25/2005 37

Clear Highlights



Investigator Poltenger is investigating a bank robbery which occurred at the Bank of America at approx. 1630 hours Tuesday. A 23-year-old 4th street area man, employed as a teller, was approached by a man suspect who demanded money. The victim complied and relinquished an undisclosed amount of cash. The suspect then approached a woman, also employed as a teller, and demanded cash. This victim also complied with the suspect's demands. The suspect then ran from the store. Neither victim was injured.

The suspect was described as a black, in his twenties, weighing 150 to 180 pounds with a muscular buil... was wearing a blue or purple woman's dress.

Feature Type	Importance	Feature Value	Range Start	Range End
☑ Clothing	10	a blue or purple woman ' s dress	a blue or purple woman ' s dress	a blue or purple woman ' s dress
☑ OffenseType	7	robbery	robbery	robbery
☑ Weight	10	150 to 180	150	180
☑ Weight	10	180	180	180
☑ DayOfWeek	3	Tuesday	2	2
☑ Age	9	23	23	23
☑ Age	9	twenties	20	29
☑ Race	9	black	black	black
☑ PersonName	9	Poltenger	Poltenger	Poltenger

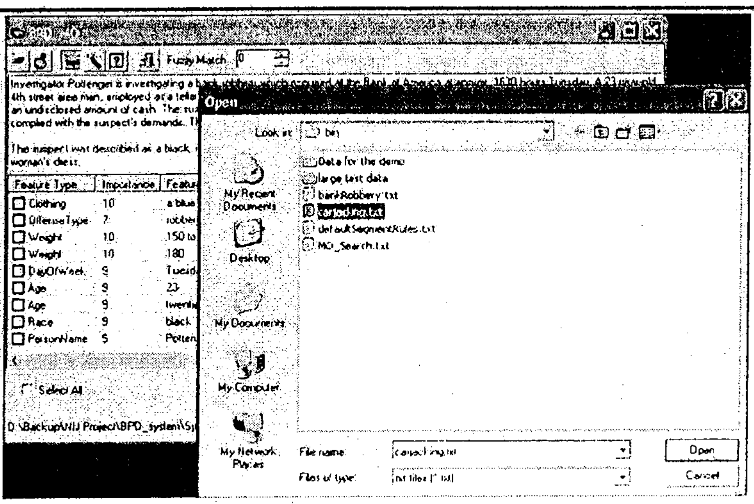
☑ Select All

You just entered the CLEAN mode, click labels to erase contents in textboxes

By selecting the eraser, the cursor becomes an eraser.
Click the document's original text to remove all highlighting.

1/25/2005 38

Open file



Investigating Poltenger is investigating a bank robbery which occurred at the Bank of America at approx. 1630 hours Tuesday. A 23-year-old 4th street area man, employed as a teller
an undisclosed amount of cash. The su
complied with the suspect's demands. T

The suspect was described as a black,
woman's dress.

Feature Type	Importance	Featu
☐ Clothing	10	a blue
☐ OffenseType	7	robber
☐ Weight	10	150 to
☐ Weight	10	180
☐ DayOfWeek	9	Tuesd
☐ Age	9	23
☐ Age	9	twentie
☐ Race	9	black
☐ PersonName	5	Potten

☐ Select All

D:\Backup\NIJ Project\BPD_system\S

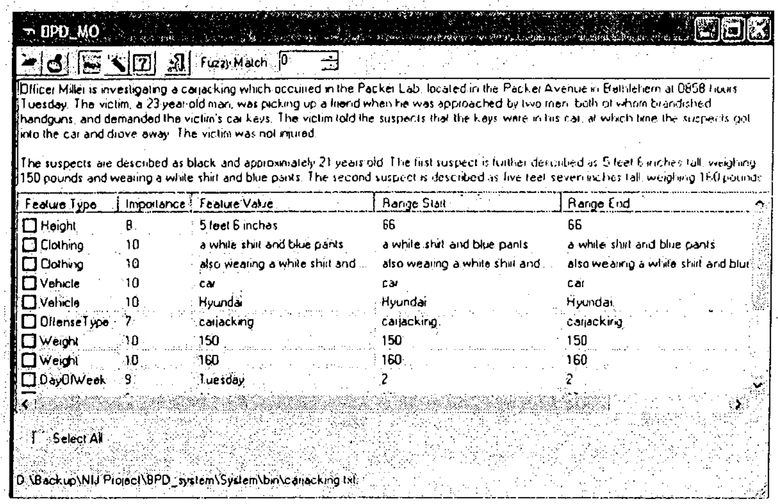
Data for the demo
large test data
bankRobbery.txt
carjacking.txt
defaultSegmentRules.txt
MO_Search.txt

File name: carjacking.txt
Files of type: txt files (*.txt)

Another document can be opened for a new search

1/25/2005 39

Open file - result



Officer Miller is investigating a carjacking which occurred in the Packer Lab. located in the Packer Avenue in Bethlehem at 0858 hours Tuesday. The victim, a 23-year-old man, was picking up a friend when he was approached by two men, both of whom brandished handguns, and demanded the victim's car keys. The victim told the suspects that the keys were in his car, at which time the suspects got into the car and drove away. The victim was not injured.

The suspects are described as black and approximately 21 years old. The first suspect is further described as 5 feet 6 inches tall, weighing 150 pounds and wearing a white shirt and blue pants. The second suspect is described as five feet seven inches tall, weighing 160 pounds

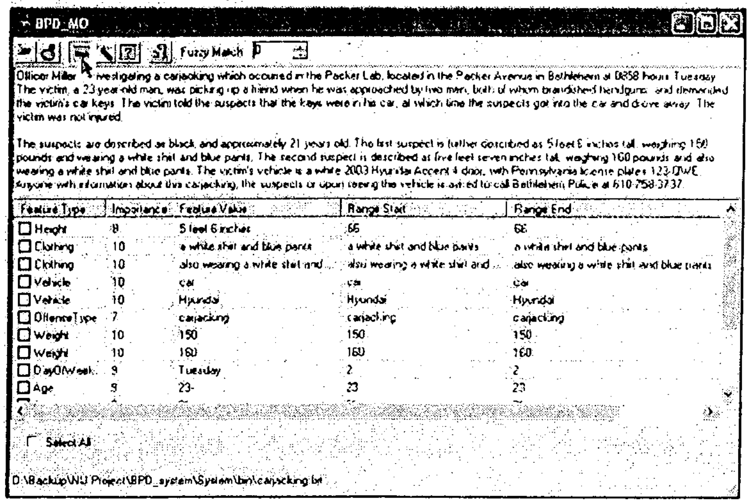
Feature Type	Importance	Feature Value	Range Start	Range End
☐ Height	8	5 feet 6 inches	66	66
☐ Clothing	10	a white shirt and blue pants	a white shirt and blue pants	a white shirt and blue pants
☐ Clothing	10	also wearing a white shirt and ...	also wearing a white shirt and ...	also wearing a white shirt and blue
☐ Vehicle	10	car	car	car
☐ Vehicle	10	Hyundai	Hyundai	Hyundai
☐ OffenseType	7	carjacking	carjacking	carjacking
☐ Weight	10	150	150	150
☐ Weight	10	160	160	160
☐ DayOfWeek	9	Tuesday	2	2

☐ Select All

D:\Backup\NIJ Project\BPD_system\System\bin\carjacking.txt

1/25/2005 40

Switch Search Method Features → Form



Officer Miller is investigating a carjacking which occurred in the Packer Lab. located in the Packer Avenue in Bethlehem at 0858 hours Tuesday. The victim, a 23-year-old man, was picking up a friend when he was approached by two men, both of whom brandished handguns, and demanded the victim's car keys. The victim told the suspects that the keys were in his car, at which time the suspects got into the car and drove away. The victim was not injured.

The suspects are described as black and approximately 21 years old. The first suspect is further described as 5 feet 6 inches tall, weighing 150 pounds and wearing a white shirt and blue pants. The second suspect is described as five feet seven inches tall, weighing 160 pounds and also wearing a white shirt and blue pants. The victim's vehicle is a white 2003 Hyundai Accent 4 door, with Pennsylvania license plates 123-QWE. Anyone with information about this carjacking, the suspects or upon seeing the vehicle is asked to call Bethlehem Police at 610-758-3737.

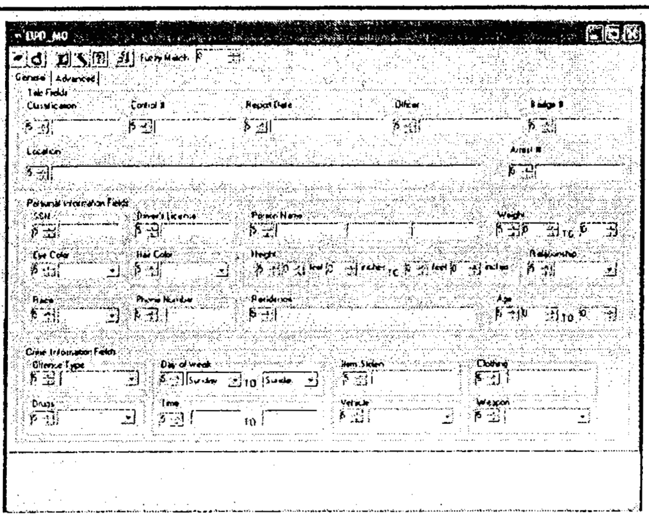
Feature Type	Importance	Feature Value	Range Start	Range End
☐ Height	8	5 feet 6 inches	66	66
☐ Clothing	10	a white shirt and blue pants	a white shirt and blue pants	a white shirt and blue pants
☐ Clothing	10	also wearing a white shirt and ...	also wearing a white shirt and ...	also wearing a white shirt and blue pants
☐ Vehicle	10	car	car	car
☐ Vehicle	10	Hyundai	Hyundai	Hyundai
☐ OffenseType	7	carjacking	carjacking	carjacking
☐ Weight	10	150	150	150
☐ Weight	10	160	160	160
☐ DayOfWeek	9	Tuesday	2	2
☐ Age	9	23-	23	23

☐ Select All

D:\Backup\NIJ Project\BPD_system\System\bin\carjacking.txt

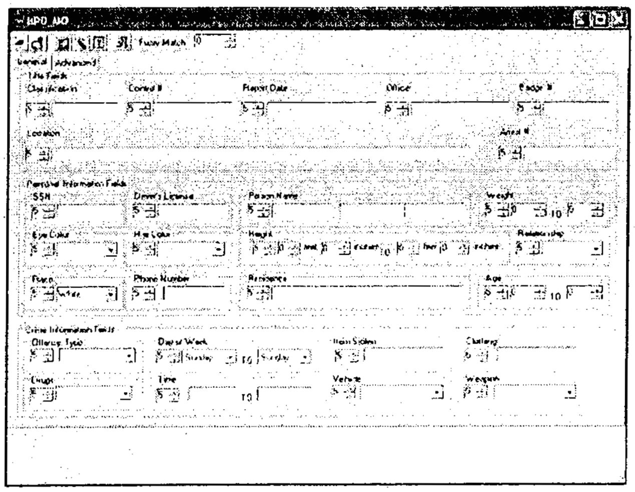
1/25/2005 41

Switch Search Method Simple → Features → Form (continued)

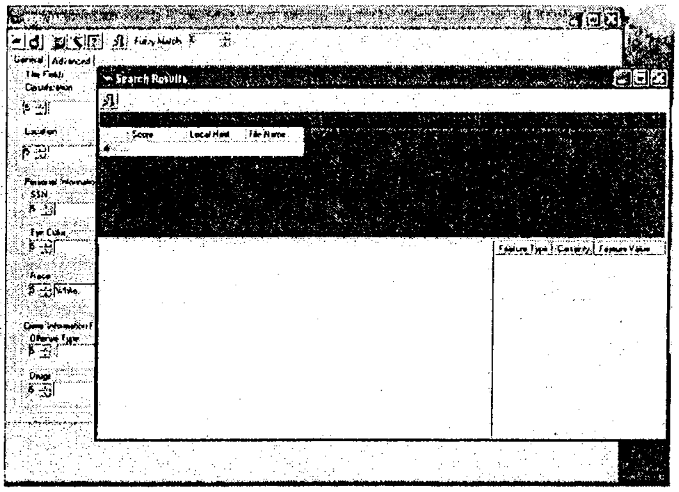


All feature types are from Investigator's reports and BPD discussions

Form Search

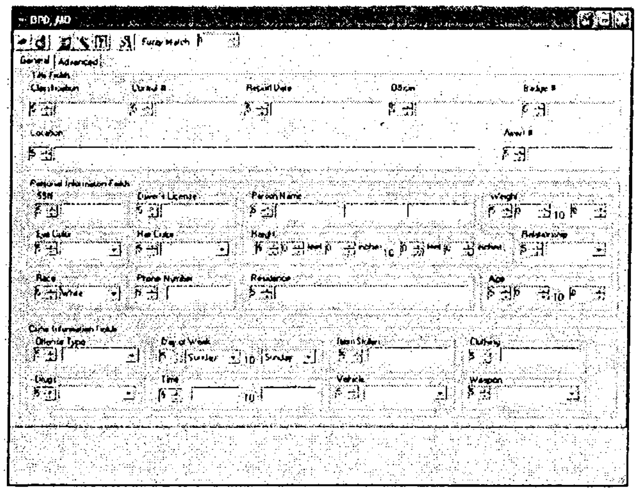


Form Search – Result



1/25/2005 44

Fuzzy Match – Edit Distance <= 1



1/25/2005 45

171