Components for Information Extraction: Ontology-Based Information Extractors and Generic Platforms

Daya C. Wimalasuriya
Computer and Information Science
University of Oregon, USA
dayacw@cs.uoregon.edu

Dejing Dou
Computer and Information Science
University of Oregon, USA
dou@cs.uoregon.edu

ABSTRACT

Information Extraction (IE) has existed as a field for several decades and has produced some impressive systems in the recent past. Despite its success, widespread usage and commercialization remain elusive goals for this field. We identify the lack of effective mechanisms for reuse as one major reason behind this situation. Here, we mean not only the reuse of the same IE technique in different situations but also the reuse of information related to the application of IE techniques (e.g., features used for classification).

We have developed a comprehensive component-based approach for information extraction that promotes reuse to address this situation. We designed this approach starting from our previous work on the use of multiple ontologies in information extraction [24]. The key ideas of our approach are "information extractors," which are components of an IE system that make extractions with respect to particular components of an ontology and "platforms for IE," which are domain and corpus independent implementations of IE techniques. A case study has shown that this component-based approach can be successfully applied in practical situations.

Categories and Subject Descriptors

H.4.0 [**Information Systems Applications**]: General

General Terms

Theory, Experimentation

Keywords

Information Extraction, Ontologies, Software Components

1. INTRODUCTION

1.1 IE and OBIE

The objective of Information Extraction (IE) is recognizing and extracting certain types of information from natural language text [17]. Here, the decision to leave out irrelevant information is a conscious one and it reduces the difficulty associated with the task at hand. Because information extraction deals with natural language sources, it is seen as a subfield of Natural Language Processing (NLP). It has existed as a field for a few decades and has experienced a significant development since 1990's partly due to the Message Understanding Conferences (MUC), which provided standard extraction tasks and evaluation criteria.

Recently, Ontology-Based Information Extraction (OBIE) has emerged as a subfield of information extraction. It is based on the use of ontologies to *guide* the information extraction process [25]. An ontology is defined as a formal and explicit specification of a shared conceptualization [20]. Typically, an ontology consists of several components such as classes, properties, individuals and values. OBIE normally takes place by specifying a domain ontology for the domain targeted by an IE system and using an information extraction technique to discover individuals for classes and values for properties. In addition, there are some OBIE systems that construct an ontology for its domain. The details of several OBIE systems such as KIM [16] and Kylin [27] have been published in the recent past.

One of the most important potentials of OBIE is its ability to automatically generate *semantic contents* for the Semantic Web. As envisioned by Berners-Lee et al. [6], the goal of the Semantic Web is to bring meaning to the web, creating an environment where software agents roaming from page to page can readily carry out sophisticated tasks for the users. For this vision to realize, *semantic contents* that can be processed by such agents should be made available. Information contained in ontologies fall under this category because Semantic Web agents are expected to process them automatically. This has been pointed out by several authors including Cimiano et al. [7].

Most OBIE systems use a *single* ontology. However, multiple ontologies exist for most domains and there is no rule that prevents an OBIE system from using more than one ontology. In fact, the use of multiple ontologies can be expected to improve the information extraction process: because multiple ontologies provide different perspectives on a domain, a system that uses multiple ontologies has the potential to make more extractions (with respect to the different perspectives) and the output of the system can be used to provide more accurate answers to queries related to different perspectives. Based on this idea, we designed the principles for using multiple ontologies in information extraction and conducted two case studies [24]. The results from the case studies indicate a significant increase in performance metrics when multiple ontologies are used. The insights provided by this work were crucial in the design of the component-based approach for IE described here.

1.2 Challenges in IE

As mentioned earlier, information extraction has improved rapidly as a research field. The details of several advanced IE systems such as TextRunner [5] and KIM [16] have been published in the recent past. However, it can be seen that information extraction still does not enjoy widespread use or commercialization, especially when compared with the field of information retrieval. Information Retrieval (IR) aims to identify documents related to a user's request from a large collection of documents and this field has given rise to many widely used systems including search engines. It can be argued that IE systems should be more useful than IR system because IE systems provide the required information itself instead of a pointer to a document. Yet, the usage of IE systems is very low when compared with IR systems.

The costs and complexity associated with setting up an IE system in a new domain or a new text corpora can be seen as one main factor hindering the widespread usage of IE. This results in serious difficulties in applying information extraction techniques in new situations as well as in developing large scale IE systems that work on different domains and different text corpora. It can be seen that there are two problems that give rise to this issue.

1. The requirement of *templates*: As described by Wilks and Brewster [23], IE systems normally require a *set of templates* for a particular domain (e.g., those used by MUCs that show *slots* such as "human target name") and merely attempt to fill the templates. Generating these *templates* manually beforehand is not practical in at least some situations.

2. Bundling domain and corpus specific information with the IE techniques: IE systems are often built as monolithic systems where no clear separation is made between domain and corpus specific information used by the implementation and the underlying IE technique. This makes the application of the IE system in a new situation difficult.

The first problem mentioned above is targeted by the emerging paradigm of "open information extraction," which aims to discover relations of interest from text instead of being provided in advance [5]. For instance, the TextRunner system [5], which is based on this paradigm, is capable of extracting relations using some grammatical structures and data tuples that fit into these relations in a single pass over the corpus. Open information extraction avoids the second problem by using IE techniques that do not use any domain or corpus specific information.

While open information extraction is a significant step forward, we believe that further advancement is possible by concentrating on the following issues.

1. Caution has to be exercised in the process of "relation discovery" to ensure that the discovered relations are useful and fit into a coherent conceptual framework.

2. IE techniques that use domain and corpus specific information should be accommodated in a structured approach for IE, while making them reusable.

1.3 Reusable Components for IE

In this paper, we present a comprehensive component-based approach for information extraction that addresses the issues mentioned above. As mentioned earlier, this approach is closely related to our work on using multiple ontologies in information extraction [24].

In our previous work we reused *components of information extraction systems* related to different ontologies. Such a component is defined as an *information extractor*, which extracts individuals for a class or values for a property of an ontology. The general idea is to reuse an information extractor for a class or a property of one ontology when making extractions with respect to some other class or property of another ontology that is related the original entity. Such relationships between classes or properties of different ontologies are known as *mappings*. This technique was shown to improve the results of information extraction in our previous case studies. It was applied on the same corpus using more than one ontology instead of a single ontology. Different IE techniques were used with respect to different ontologies and the information extractors related to these different IE techniques were reused.

In the component-based approach presented here, we extend this idea to applying information extractors in *different text corpora and different domains*. Moreover, we did not clearly define what constitutes an information extractor in our previous work. Here, we formalize this using the concept of *platforms for IE*, which are domain and corpus independent implementations of IE techniques. An information extractor consists of a platform for IE and any domain and corpus specific information used with the particular class or property, which we call the *metadata* of the information extractor. The separation between platforms and metadata makes reuse of information extractors structured and straight-forward.

It can be seen that the use of platforms for IE and information extractors addresses the problem of bundling domain and corpus specific information with IE techniques. Further, they allow the use of IE techniques that use domain and corpus specific information unlike open information extraction, which is restricted to domain and corpus independent IE techniques. Regarding the requirement of templates, it can be seen that "ontology construction" (extracting classes and properties from text), undertaken by some OBIE systems [13, 14] addresses this issue. In essence, this can be seen as following the paradigm of open information extraction [25]. Moreover, ontologies guarantee that the extracted concepts fit into a conceptual framework unlike the relation discovery process undertaken by TextRunner [5].

In our component-based approach for IE, we identify components in an IE system based on ontologies. Other types of models such as relational models or UML class diagrams can be used for this purpose. However, we believe that ontologies are the best option because of the following reasons.

1. Since ontologies are based on logic, they provide formal mechanisms to define concepts and mappings and support reasoning.

2. The IE systems that follow this approach can be easily converted into OBIE systems that provide advantages such as the ability to generate semantic contents.

The rest of the paper is organized as follows. Section 2 discusses some related work and section 3 presents the design of our component-based approach. The details of the case study we have conducted on this approach is presented in section 4. Sections 5 and 6 provide a discussion on our work and future work respectively. Some concluding remarks are provided in section 7.

2. RELATED WORK

Details of several IE and OBIE systems have been published in the recent past and these systems use different IE techniques. One such system is Kylin [27], whose information extraction technique is based on classification and operates in two phases: identifying sentences in which interesting information is present and identifying words within sentences that carry the information. Kylin uses the Maximum Entropy model for the first phase and Conditional Random Fields (CRF) for the second phase. It uses a set of Wikipedia pages as its corpus and attempts to extract information presented by the "infoboxes," which provide a summary of the contents of each page. Kylin can also be considered an OBIE system because it constructs an ontology based on the structure of the infoboxes in order to aid its information extraction process.

Another information extraction technique used by many information extraction systems is known as "extraction rules." Here, the idea is to specify regular expressions that can be used to extract certain information from text. For example, the expression `(belonged|belongs) to <NP>`, where `<NP>` denotes a noun phrase, might capture the names of organizations in a set of news articles. This technique has been used by several IE systems including the ontoX [28] system and the implementations by Embley [9].

The Apache UIMA (Unstructured Information Management Architecture) project [1] appears to be the most serious attempt so far to develop a component-based approach for information extraction. It targets analysis on all types of unstructured data including text, audio and video. It defines a common structure, known as Common Annotation Structure (CAS), to store the extracted information and provides frameworks in Java and C++ to deploy the developed components. In terms of analyzing text, UIMA components have been mostly developed for general NLP tasks such as sentence splitting, POS tagging and tokenization although some components have been developed for extracting instances of specific classes such as gene names. UIMA components do not separate the domain and corpus specific information from the underlying IE technique, which is a key idea in our component-based approach. Moreover, UIMA assumes that the developed components are *interoperable* or *reusable* whereas our approach studies the basis for successful reuse and presents methods to improve reusability. It is also interesting to note that UIMA uses UML models (through *type systems*) to relate the extracted information to domain models. As mentioned in section 1.3, we believe that ontologies are a better option for this purpose.

In addition, Embley [9], Maedche et al. [14] and Yildiz and Miksch [28] have independently worked on including extraction rules in ontologies to come up with what has been termed "extraction ontologies" or "concrete ontologies." The general idea here is to include extraction rules related to a class in the ontology itself. As we have pointed out in our previous work [24], the inclusion of these rules, which are known to contain errors, in ontologies appears to violate the requirement that ontologies should be formal. However, this approach can be seen as an attempt to identify *components* that can be used for information extraction. To a certain extent, our work can be seen as an extension of these works because it is based on a similar insight but attempts to accommodate more than one IE technique.

In our component-based approach for IE, we attempt to address the hard problem of information extraction by decomposing it based on ontological concepts. A similar approach has been applied in image retrieval, in an area of study known as ontology-based image retrieval [22] and has produced strong results.

3. THE DESIGN

This section presents the design of our component-based approach. We start with its formal representation and then describe its relationship with component-based software engineering. Next we describe how it operates in practice and move onto presenting the details of two platforms for IE.

3.1 Formal Representation

We provide a formal representation of our component-based approach for information extraction using the Z notation [19], which is a widely used formal specification language. We begin by providing a formal specification for a generic OBIE system and then show that this specification can be refined into a specification that uses the components of our component-based approach. This essentially proves that the component-based approach functions correctly.

In order to provide a specification for a generic OBIE system, we need a formal specification of an ontology. For this, we begin from the definition we have used in our previous work [24], where an ontology is defined as a quintuple consisting of sets of different types of its components, and extend it by defining the nature of these components. This refined definition is shown below.

Definition 1 ***Ontology***: An ontology O is a quintuple, $O = (C, P, I, V, A)$ where C, P, I, V, and A are the sets of classes, properties, individuals, property values and other axioms (such as constraints) respectively. These sets are defined as follows.
$C = \{c \mid c \text{ } is \text{ } a \text{ } unary \text{ } predicate\}$
$P = \{p \mid p \text{ } is \text{ } a \text{ } binary \text{ } predicate\}$
$I = \{c(i) \mid c \in C \wedge i \text{ } is \text{ } a \text{ } ground \text{ } term\}$
$V = \{p(x, y) \mid p \in P \wedge x \text{ } and \text{ } y \text{ } are \text{ } ground \text{ } terms \wedge (\exists\, c \text{ } c \in C \wedge c(x))\}$
$A = \{a \mid a \text{ } is \text{ } an \text{ } assertion\}$

The important parts of the Z specification of a generic OBIE system is defining a schema for a ontology and showing how this schema is changed by the two main operations of an OBIE system, namely *ontology construction* (identifying classes and properties) and *ontology population* (identifying individuals and values). For the sake of brevity, we only provide these sections of the specification here.

```
┌─ Ontology ────────────────────────────────
│ classes : ℙ UnaryPredicate
│ properties : ℙ BinaryPredicate
│ individuals : ℙ AssertionOnUnaryPredicate
│ values : ℙ AssertionOnBinaryPredicate
│ axioms : ℙ Assertion
└───────────────────────────────────────────
```

$$\begin{array}{|l}
\multicolumn{1}{l}{\textit{PopulateOK}} \\
\Delta \mathit{Ontology} \\
\Xi \mathit{Corpus} \\
r! : \mathit{Response} \\
i! : \mathbb{P}\, \mathit{AssertionOnUnaryPredicate} \\
v! : \mathbb{P}\, \mathit{AssertionOnBinaryPredicate} \\
\hline
\mathit{classes}' = \mathit{classes} \\
\mathit{properties}' = \mathit{properties} \\
\mathit{individuals}' = \mathit{individuals} \cup i! \\
\mathit{values}' = \mathit{values} \cup v! \\
\mathit{axioms}' = \mathit{axioms} \\
r! = \mathit{success}
\end{array}$$

$$\begin{array}{|l}
\multicolumn{1}{l}{\textit{ConstructOK}} \\
\Delta \mathit{Ontology} \\
\Xi \mathit{Corpus} \\
r! : \mathit{Response} \\
c! : \mathbb{P}\, \mathit{UnaryPredicate} \\
p! : \mathbb{P}\, \mathit{BinaryPredicate} \\
a! : \mathbb{P}\, \mathit{Assertion} \\
\hline
\mathit{classes}' = \mathit{classes} \cup c! \\
\mathit{properties}' = \mathit{properties} \cup p! \\
\mathit{individuals}' = \mathit{individuals} \\
\mathit{values}' = \mathit{values} \\
\mathit{axioms}' = \mathit{axioms} \cup a! \\
r! = \mathit{success}
\end{array}$$

This specification can be generalized into an OBIE system that uses n ontologies instead of a single ontology. In this case, there would be n schemata for the n ontologies and n pairs of operations, one pair for each ontology.

In the component-based approach, we keep the operation ConstructOK unchanged but refine the PopulateOK operation into a *pipeline* of three separate operations. These operations perform the tasks of preprocessing the documents into a format that can be used by information extractors (using a *preprocessor* component), deploying *information extractor components* and combining the results produced by individual information extractors (using an *aggregator* component). The refinement of the PopulateOK operation into these three operations can be formally represented as follows. We do not provide the formal specifications of these individual operations (which are based on the intuitive description given above) due to lack of space.

$$\mathit{PopulateOK} \;\widehat{=}\; \mathit{Preprocess} >> \mathit{Extract} >> \mathit{Aggregate}$$

As in the case of specification for an OBIE system, this refinement can be extended into the multiple-ontology case.

To summarize, the components in our approach are *preprocessors*, *information extractors* and *aggregators*. As described in section 1.3, an information extractor consists of a *platform* for an IE technique, which is domain and corpus independent and *metadata* that is used with the particular class or property represented by the information extractor.

3.2 Relationship with Component-Based Software Engineering (CBSE)

Component-Based Software Engineering (CBSE) studies how component-based approaches can be used in developing software systems. This field has been in existence for more than a decade and has been successful in many domains. Since information extraction systems can be viewed as software systems, the field of component-based software engineering can be expected to provide some guidance in developing a component-based approach for IE.

Szyperski [21] provides a simple commonsense justification for the use of a component-based approach in software engineering in his textbook on this subject by stating "Components are the way to go because all other engineering disciplines introduced components as they became mature and still use them." He concedes that from a purely formal view, there is nothing that could be done with components that could not be done without them. The differences are in goal-driven factors such as reusability, time to market, quality and viability. It can be seen that these arguments are more or less valid in the field of information extraction as well: a component-based approach, while not doing anything that cannot be done using existing techniques, has the potential to improve the information extraction process quantitatively and qualitatively.

In CBSE, it is generally agreed that software components have clear interfaces and functionalities. It can be seen that preprocessors, information extractors and aggregators described in section 3.1 satisfy these requirements. Further, Szyperski states that software components may contain "meta-data and resources" and even concedes that some *degenerate* components may only consist of such meta-data and resources. Hence, it can be seen that even the metadata of information extractors can be considered independent components. This implies that an information extractor is a component that consists of two sub-components, namely a platform and a metadata component.

3.3 The Operation of the Approach

Figure 1 presents a schematic diagram of an OBIE system consisting of the components described in section 3.1. We focus our attention on the information extractors because we believe that they contain the most valuable information captured by IE systems. But preprocessors and aggregators are also very important in an IE system. The tasks performed by preprocessors depend on the IE techniques (implemented through platforms) that they are meant for and could include tasks such as analyzing or removing HTML/XML tags, identifying section headings and POS tagging. The aggregators combine the results produced by information extractors and do some adjustments on them. This includes *reference reconciliation* [8], which refers to the problem of identifying whether two individuals refer to the same real world entity.

For our component-based approach to work, there should be standard mechanisms to store the different types of information associated with information extractors and to represent the links between them. We have designed a three-layered architecture, consisting of ontologies, metadata for information extractors and platforms for this purpose. Figure 2 represents this architecture.

The ontologies represent the domain ontologies for which an information extraction system has been developed. Since the Web Ontology Language (OWL) [3] is increasingly being seen as the standard for defining ontologies, we represent ontologies using it. The metadata of information extractors and the details of platforms are stored in *XML files* and made available through *URIs*. The links between classes

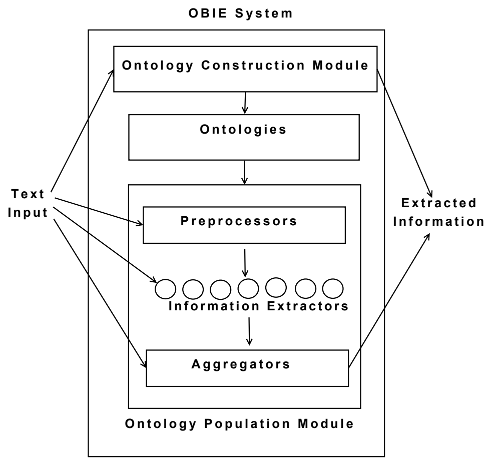

Figure 1: A Component-Based IE System

and properties of ontologies and the information extractors that have been developed for them are represented as *OWL annotation properties* in the ontologies. An XML file for the metadata of an information extractor contains *an element* that represents the URI of the platform that it uses. The XML file for the platform contains a link to the executable file for the platform.

OWL annotation properties provide a mechanism to include the links between ontological concepts and information extractors in ontologies while excluding these links from reasoning. These are typically used to include metadata about concepts such as version, creator and comments and are ignored by software agents that perform reasoning on the ontologies. We consider the information extractors developed for classes and properties of ontologies to be a type of such metadata. Moreover, OWL annotation properties can be used with both classes and properties and can be specified as URIs. These properties match nicely with our requirement to specify the URIs of information extractors for both classes and properties. We have also noticed that OWL annotation properties have been used by Yildiz and Miksch [28] to include extraction rules in the ontologies. We store URIs of information extractors instead of extraction rules but the general idea is the same.

We selected XML as the format for representing metadata of information extractors and platforms because it is a lightweight machine processable format that is widely used for information exchange. We considered using ontologies for this purpose but it was quite clear that advanced features of ontologies are not necessary here. We have designed XML schemata for creating XML files for platforms and metadata but do not include them here due to lack of space.

For our component-based approach to operate, mappings between different ontologies have to be identified. This is not a trivial task and there have been several works on discovering mappings between ontologies in an automatic or semi-automatic manner [11, 12]. In some cases it is possible to identify mappings manually, but automatic mapping discovery techniques are required to make our component-based approach scalable.

Once mappings between ontological concepts are discovered, reuse of information extractors can be performed in different ways as shown below.

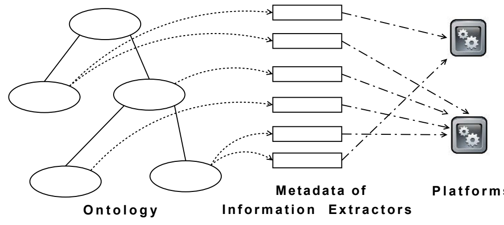

Figure 2: The Operation of Reuse

1. *Black Box Reuse:* This refers to directly reusing an existing information extractor for a particular class or a property in a new system.
2. *Clear Box Reuse:* Here, the metadata of the information extractor, which contain domain and corpus specific information, are changed before applying it in a new corpus or a new domain. The platform is not changed.
3. *Simple Combination:* Here, more than one information extractor is used for the same concept. The results produced by them are combined using set operations. This approach is based on our work on using multiple ontologies in information extraction [24], where we combine the output of different information extractors for the same corpus using the union operation.
4. *Advanced Combination:* More advanced techniques, such as techniques used under ensemble learning can be used to combine different information extractors instead of set operators.

It is interesting to note that the terms *black box reuse* and *clear box reuse* are used to convey similar meanings in the field of component-based software engineering [21].

3.4 Platforms for IE

In this section, we present the details of the domain and corpus independent IE platforms that we have developed.

3.4.1 Two-Phase Classification

As mentioned in section 2, this technique converts the problem of extracting information related to some concept into two classification problems: identifying sentences in which information is present and identifying words within sentences that are related to the information. Following Kylin [27], which uses this technique, we tried two different approaches for combining the two classification phases: a *pipeline* approach, where the word-level classifier only operates on the sentences selected by the sentence-level classifier and a *combination* approach, where the word-level classifier operates on all the sentences but uses the output of the sentence-level classifier as one feature in classification. In the experiments conducted, we got better results using the combination approach and as such we incorporated it into our platform.

Since this IE technique is based on classification, it requires a training set in addition to the corpus on which information extraction is to be performed (test set). Instead of requiring an annotated corpus indicating the positions where the particular concept is found, our platform allows providing key files specified for each file in the corpus. Then it internally annotates the text files with the keys provided based on string matching (allowing some prefixes and suffixes such as " ' " and " 's "). While this reduces the accuracy of the annotations, it also significantly reduces the effort required to create the training corpus. For the test set, it is not necessary to provide keys but the accuracy of the platform can be automatically measured if keys are provided.

We experimented with different classification techniques for the two classification phases and found that Bayesian techniques (specifically Naive Bayes model) used with bagging produced best results in sentence-level and Conditional Random Fields (CRF), which is a sequence tagging technique, produced best results at the word-level. Therefore, we incorporated these two techniques into the platform although the user has the option of selecting a different classification technique for the sentence-level classification. We used the Mallet system [15] for the CRF technique and the Weka system [26] for other classification techniques.

In sentence-level classification, we used several domain and corpus independent features such as the word count for each POS tag. Similarly, we used domain and corpus independent features such as POS tags, stop words, the half of the sentence a word belong to (first or second) and capitalization information at word-level. We adopted some of such features from the Kylin system. In addition, we used specific words, WordNet [10] synsets of some words and some gazetteers[1] as concept-specific information. (Specific examples are provided in section 4.2.) These features represent the metadata of the information extractor and are included in the XML file for the metadata. The platform requires such an XML file with metadata as an input. Currently, this platform runs only in the Windows operating system.

3.4.2 Extraction Rules

This platform uses the IE technique of extraction rules described in section 2 and considers the extraction rules developed to make extractions for a particular concept to be the metadata for that concept. Unlike the two-phase classification platform, this does not use any domain and corpus independent information. Further, this platform does not require a training set. (But in practice, a training set is required to identify the extraction rules.) As in the case of the platform for two-phase classification, keys for the test set can be provided to evaluate the accuracy although it is not required.

We implemented this platform using the General Architecture for Text Engineering (GATE)[2], which a widely used NLP toolkit that can be used to directly deploy extraction rules. Here, the rules have to written in a format known as Java Annotation Patterns Engine (JAPE), which is interpreted by GATE. As such, these JAPE rules are included in the metadata of the information extractor. In addition, the metadata also include gazetteers, which can be used by the JAPE rules. As in the case of the platform for two-phase classification, the platform requires an XML file with metadata as an input. This platform does not depend on a particular operating system and we have successfully tested it in Windows and Unix environments.

In developing these platforms, we observed that the implementation is much neater than a regular IE system because we had to consider only one type of extractions instead of a set of different types. However, the platforms have to be executed separately for each concept. Parallel processing can be used to improve the efficiency on this regard.

4. A CASE STUDY

4.1 Overview

In our previous work on using multiple ontologies in information extraction [24], we conducted a case study on a set of news articles related to terrorist activities using two ontologies that provide different perspectives on the domain of terrorist attacks. The corpus consists of 200 news articles (160 for the training set, 40 for the test set) selected from the corpus of the 4th Message Understanding Conference. These news articles are on terrorist activities in Latin American countries in late 1980's and early 1990's. One ontology has been derived from the structure of the key files provided by the conference (which is called the *MUC4 ontology*) while the other is a terrorism ontology defined by the Mindswap group at the University of Maryland (which is called the *Mindswap ontology*). We performed information extraction using classification with the MUC 4 ontology and using extraction rules with the Mindswap ontology. However, we did not perform full information extraction and stopped at the phase of identifying sentences related to the concepts in concern. Based on the mappings between concepts of different ontologies, we combined the extractions made by different information extractors using the union operation and the results showed that this leads to an improvement in performance measures.

In the case study described here, we first applied our platforms for information extraction in the corpus used by the above case study, which we call the *MUC 4 corpus*. We used the two-phase classification platform with the MUC4 ontology and the extraction rules platform with the Mindswap ontology. We selected a set of classes and properties from each ontology to use in information extraction. In order to apply the two platforms, we identified words, WordNet synsets and gazetteers for each concept used with the two-phase classification platform and extraction rules (in JAPE format) and gazetteers for each concept used with the extraction rules platform. Some features and rules of our previous work were reused in this exercise. We also developed techniques for identifying these information, the details of which are presented in section 4.2. We then wrote these features into XML files conforming to the XML schema for metadata, executed the platforms using the XML files and obtained results as well as performance measures.

The next step of our case study was reusing the information extractors developed for the MUC4 corpus in a different corpus and a different ontology. For this, we compiled a corpus of Wikipedia pages on terrorist attacks. We selected 100 Wikipedia pages and randomly split it into a training set of 70 pages and a test set of 30 pages. We also constructed a simple ontology for terrorist attacks based on the fields of infoboxes of the selected Wikipedia pages (which we call *Wikipedia ontology*). The keys for the files were also derived from the infoboxes. Then, we identified mappings between

[1]Gazetteers provide a list of known entities for a particular category, such as states of the U.S. and are often used in information extraction.

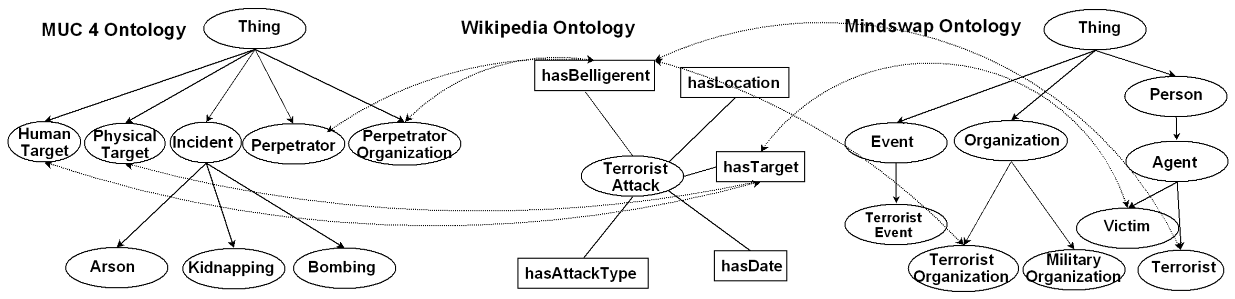

Figure 3: The Ontologies Used

the concepts of this Wikipedia ontology and the MUC4 and Mindswap ontologies described above. Based on these mappings, we reused the XML files containing the metadata of the information extractors together with the platforms to perform information extraction on the Wikipedia corpus.

Figure 3 shows sections of the ontologies used and some mappings between them.

4.2 MUC 4 Corpus

In order to discover the words and WordNet synsets to be used with the two-phase classification platform, we used human knowledge as well as a statistical approach. We selected certain words and their WordNet synsets based on our understanding of the concepts in concern. For instance, we selected the words "Kill" and "Kidnap," as well as their WordNet synsets as features for the class "Human Target." In addition, we employed the following statistical technique. First we selected the sentences that contain key values from the training files. Then we identified the words that are found in more than a predefined fraction of these sentences (5% was selected as the threshold after some experiments). Next we performed correlation analysis (using the frequency of these words among *all* sentences) to ensure that the selected words have a positive correlation with keys instead of just being frequent words. We used the statistical measure of *lift* for this purpose and words having a lift of more than 1 were selected (which mean that they have a positive correlation with key sentences). Still, common words such as "the" and "to" were often included in the selected set of words and we excluded them from the final set of features. There was some overlap between the words selected based on human knowledge and words mined using the statistical technique but many new words were also discovered by the statistical technique.

Following the generally accepted practice on the use of gazetteers, we selected standard sets of lists for certain concepts as gazetteers. For instance, we used a set of Spanish first names provided by a website[2] as a gazetteer. In addition, we used some lists provided by the support material of the MUC 4 conference.

The extraction rules to be used with the Mindswap ontology were discovered by first separating the sentences containing keys from the training files and then manually going through these key sentences to identify extraction patterns. Correlation analysis was used, in a manner similar to its application described above, to ensure that the discovered patterns are useful. In addition, gazetteers were identified in the manner described above. Most of these gazetteers were the same ones used with the classification platform.

[2]`http://www.zelo.com/firstnames/names/spanish.asp`

4.3 Wikipedia Corpus

The Wikipedia pages on terrorist attacks were identified from a list of terrorist incidents provided by Wikipedia. The majority of selected pages were on terrorist activities in the decade of 2000.

It was seen that infobox structure is not uniform among different Wikipedia pages (since authors of pages can add their own attributes for the infoboxes) and as such we only included attributes that are found in at least 20% of the pages in the Wikipedia terrorism ontology. A similar approach has been adopted by Kylin [27]. In addition, it was seen that the infoboxes of most of the pages had to be manually refined before being used as gold standards. Often this included removing descriptions such as "(as reported)" and removing fields such as "Belligerent: Unknown." In some situations, we also manually filled missing information.

As mentioned earlier, we need the mappings between ontologies in order to reuse the information extractors. Here, we need the mappings between the MUC4 and Wikipedia ontologies as well as the mappings between the Mindswap and Wikipedia ontologies. In order to discover these mappings, we tried to use Anchor Flood [11] and Falcon-AO [12] systems, which are two recently developed mapping discovery systems. The precision of the discovered mappings were quite high (close to 80%) but although the recall was also high (close to 70%), we observed that some important mappings we were planning to use were not discovered by these systems. For instance, both systems failed to discover the mappings between *Belligerent* class of the Wikipedia ontology and *Perpetrator* and *Perpetrator Organization* classes of the MUC4 ontology shown in figure 3 above. As we discuss in the following sections, our component-based approach can detect incorrect mappings to a certain extent but in the case of missing mappings there is no alternative other than manually reviewing the entire ontologies to discover mappings.

In addition to the mappings shown in figure 3, we used the following mappings.

- Between class *Location* of MUC4 and class *Location* of Wikipedia
- Between classes *City*, *Country* and *Location* of Mindswap and class *Location* of Wikipedia
- Between classes *Instrument* and *Instrument Type* of MUC4 and class *Weapon* of Wikipedia

Concept	MUC4			Wikipedia-BlackBox			Wikipedia-ClearBox		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
Location	25.86	52.94	34.75	15.83	40.00	22.68	18.26	46.32	26.19
Human Target	27.71	36.51	31.51	1.83	5.26	2.72	1.35	5.26	2.15
Physical Target	38.89	67.74	49.41	1.34	5.26	2.14	1.80	5.26	2.68
Perpetrator Organization	8.52	37.50	13.89	21.59	28.36	24.52	22.88	40.30	29.19
Perpetrator	26.67	27.27	26.97	19.39	28.36	23.03	22.69	40.30	29.03
Instrument	53.33	66.67	59.26	25.40	40.00	31.07	25.81	40.00	31.38
Instrument Type	60.00	46.15	52.17	25.71	45.00	32.72	25.42	37.50	30.30

Table 1: Results for the Two-Phase Classification Platform (%)

Concept	Mindswap			Wikipedia-BlackBox			Wikipedia-ClearBox		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
City	53.85	90.32	67.47	34.69	17.89	23.61	22.39	31.58	26.20
Country	41.27	66.67	50.98	34.69	17.89	23.61	32.88	25.26	28.57
Location	35.21	56.39	43.35	8.93	5.26	6.62	20.88	20.00	20.43
Terrorist Organization	55.79	42.74	48.40	31.25	22.39	26.09	25.49	38.81	30.77
Military Organization	39.86	79.73	53.15						
Victim	34.86	45.86	39.61	0.82	5.26	1.42	0.82	5.26	1.42
Givernment Agent	33.80	55.17	41.92						
Terrorist	22.81	52.00	31.71	17.92	28.36	21.96	17.72	41.79	24.89

Table 2: Results for the Extraction Rules Platform (%)

In reusing the information extractors of the MUC4 corpus with the Wikipedia corpus, we used the black-box reuse, clear-box reuse and simple combination techniques described in section 3.3. When reusing clear-box reuse, we identified words and extraction rules that can be used as features using the statistical analysis techniques described above. In addition, we changed the gazetteers used to take the new domain into consideration. For instance, we replaced the gazetteer of Spanish first names with a list of common Indian, Arabic, U.S. and Spanish first names. We did not apply the platforms on the Wikipedia corpus while ignoring the MUC4 information extractors, because it was seen that this would be quite similar to clear-box reuse in most cases.

4.4 Results

In evaluating the results, we used a scorer that compares extractions made with the gold standard provided in the key files for the test set. It operates based on string matching (while allowing some prefixes and suffixes such as " ' " and " 's ") and counting words. The figures calculated using this scorer for precision, recall and F1 are shown in tables 1 and 2. The first column of each table shows the concept of the MUC4 or Mindswap ontology used with the MUC4 corpus. Results are shown separately for the MUC4 corpus and black-box and clear-box reuse in the Wikipedia corpus.

From the results shown in tables 1 and 2, it can be seen that the reuse of information extractors has been generally successful. While the performance measurements have recorded a drop when the information extractors are reused, they have normally recorded a F1 in the range of 25%- 30%. Not surprisingly, clear-box reuse has shown better results than black-box reuse (because it adds some features better suited for the Wikipedia corpus).

The exception to this are the results for reuse based on mappings for the *Target* class of Wikipedia. For this mapping, the F1 is lower than 5%. In analyzing the reasons for this, we found out that targets specified by Wikipedia infoboxes are very different from the human targets (victims) and physical targets specified for MUC4 corpus. They contained very few person names, effectively invalidating the reuse of information extractors for the *Human Target* class of MUC4 and the *Victim* class of Mindswap. On the first glance, it appeared that there was a somewhat stronger relationship between the *Target* class of Wikipedia and the *Physical Target* class of MUC4 but a closer inspection revealed that even this relationship is questionable. Targets of Wikipedia often contained phrases such as "Moscow Metro," which were quite different from the physical targets identified in the MUC4 corpus. In other words, the mappings identified between the *Target* class of the Wikipedia ontology and classes of other ontologies appear to be contradicted by the actual data of the corpora although the mappings make sense intuitively. In other words, even manually identified mappings can not be considered 100% accurate.

For the application of each platform in each ontology, aggregate performance measures can be computed as follows.

Assume that $C = \{C_1, C_2, ..., C_n\}$ denotes the set of concepts (classes and properties) for which extractions are made. For each concept C_i, let $correct(C_i), total(C_i)$ and $all(C_i)$ denote the number of correct extractions, total number of extractions and the number of extractions in the gold standard respectively. Then aggregate measures for precision and recall can be calculated as follows. F1 will be the harmonic mean between precision and recall as usual.

$$Precision = \frac{\sum_{i=1}^{n} correct(C_i)}{\sum_{i=1}^{n} total(C_i)} \quad Recall = \frac{\sum_{i=1}^{n} correct(C_i)}{\sum_{i=1}^{n} all(C_i)}$$

Tables 3 and 4 present these results. The *Target* class has been excluded from the calculations for the Wikipedia corpus. These results further highlight that the reuse of information extractors has been generally successful. It can also be seen that the reusability of information extractors is higher with the classification platform. This can be expected

Ontology/Method	Precision	Recall	F1
MUC4/Direct	23.59	42.77	30.41
Wikipedia/Black-Box	19.68	35.60	25.35
Wikipedia/Clear-Box	21.54	41.75	28.42

Table 3: Aggregate Results for Classification

Ontology/Method	Precision	Recall	F1
Mindswap/Direct	39.22	57.75	46.71
Wikipedia/Black-Box	23.70	17.42	20.08
Wikipedia/Clear-Box	22.76	30.31	26.00

Table 4: Aggregate Results for Extraction Rules

because it uses words and WordNet synsets as features instead of hand-crafted rules for a particular corpus.

As mentioned earlier, we also attempted to use the simple combination technique described in section 3.3. For instance, the output of the information extractors for the *Perpetrator* and *Perpetrator Organization* classes from the MUC4 ontology can be combined to get the results for the *Belligerent* class of Wikipedia and the output of this combination can be combined again with the output produced by combining the output for *Terrorist Organization* and *Terrorist* classes of Mindswap. When union was used for combination, it was observed that F1 measure drops by around 2% - 5%, when compared with the best original information extractor. The possible reason for this is the increase of errors when the information extractors are reused in a different corpus. (We obtained better results by combining results in this manner for the corpus for which information extractors are developed in our previous work [24]). We also experimented with the intersection operator and it was seen that this results in unpredictable behavior ranging from a slight gain in F1 measure to a drastic drop in F1 (when individual information extractors identify sub-concepts of a broader concept, such as countries and cities falling under the *Location* class).

In discussing the implementation work, we have concentrated on the use of platforms and information extractors. Currently, these platforms do not require any preprocessors but we are considering to delegate tasks such as POS tagging to such a preprocessor component. We have developed an aggregator component that can be used with the Wikipedia corpus, where all the extractions are related to a main class (in this case, Terrorist Attack). In literature, this form of information extraction is often known as *attribute-based IE* [18]. The information extraction using the Mindswap and MUC4 ontologies represent a more complex type of information extraction, where extractions are related to multiple classes (known as *relation-based IE* [18]). Developing a generic aggregator component for this case a complex problem. Currently, we are developing an aggeragator using the simplifying assumption that *all object properties should have a single main class as the domain.* This assumption is valid with the MUC4 and Mindswap ontologies as well as in many other ontologies used in information extraction.

All the details of this case study including executable platforms and XML files will be made available from our project web site[3]. The executable platforms in particular should be useful tools for researchers.

[3]`http://aimlab.cs.uoregon.edu/obie/`

5. DISCUSSION

It can be seen that the performance measures we have obtained are somewhat lower than those of some other systems. For instance, the Kylin system [27] has recorded F1 measures in the range of 40% - 45% for similar Wikipedia pages. For the classification platform, the main reason for this appears to be the insufficiency of training data. Kylin has faced the same problem and has employed special mechanisms to add new training data to improve performance [27]. For the extraction rules platform, the only way to improve results is to identify better extraction rules. On the other hand, these results mark an improvement over the results of our previous case study [24] (F1 measures of 30% - 35% for the MUC4 corpus) because we only performed information extraction at the *sentence-level* in that case study.

In conducting our case study, we tried to reuse the extraction rules used by the FASTUS system [4], which has participated in the MUC4 conference and which is one of the first IE systems to use this technique. By communicating with the authors of this system, we learnt that it has been superseded by another system and that even the developers of the system do not have easy access to the extraction rules, which are considered *source* of the program. While conceding that the situation might be different in some other IE systems we believe that this provides an indication of what happens to the useful information captured by IE systems. Such information (extraction rules in this case), get mixed up with source code of the programs and often it is difficult to retrieve them to be used in a new application. A component-based approach with structured mechanisms to store such information would correct this situation.

Even though there has been a surge in interest in information extraction, experts appear to have very different opinions on its future. Some are convinced that it is the way to go while others suspect whether it would be good for anything. Such diametrically opposing views were expressed in a panel discussion in the CIKM 2009 conference[4]. Much of the criticisms of information extraction target its inflexibility and apparent brittleness. We believe that this situation shows the need to seriously look at radically different approaches for information extraction in addition to attempting to make incremental improvements in existing techniques.

6. FUTURE WORK

The following can be identified as the main work to be carried out to make our component-based approach for information extraction fully usable.

- **Developing platforms for more IE techniques:** IE systems that use different techniques such as constructing parse trees, web-based search and using visual components of webpages have been developed. The component-based approach would be more effective if platforms are available for such techniques.

- **Improving the developed platforms:** We will investigate how the results can be improved by using larger training sets and better extraction rules as mentioned in section 4.4. We will also investigate whether different classification techniques and new features can be used to improve the classification platform.

[4]`http://www.comp.polyu.edu.hk/conference/cikm2009/program/panels.htm`

- **Conducting more case studies:** It is necessary to conduct more case studies in order to comprehensively evaluate the effectiveness of our component-based approach. Suitable domains include bioinformatics and business intelligence, where IE systems are widely used.
- **Exploring new ways to reuse information extractors:** As mentioned in section 3.3, ensemble learning techniques can be used to combine several information extractors. In addition, it may be possible to use *uncertainty of mappings* to improve the reuse of information extractors.
- **Developing components other than information extractors:** As described in section 4.4, we are currently developing preprocessor and aggregator components. These, as well as ontology construction components, are necessary to make our component-based approach complete.

7. CONCLUSION

In this paper, we have presented the design of a comprehensive component-based approach for information extraction. As a part of this approach, we have presented two generic platforms for information extraction which have the potential to be applied in different situations. We have also conducted a case study that shows that the presented component-based approach functions as expected.

Due to the current state of affairs in information extraction - showing a lot of potential but still failing to achieve widespread usage or commercialization - we believe that research community should seriously consider new approaches for the field in addition to making incremental changes in the existing techniques. We believe that our component-based approach constitutes such an approach and thus deserves the attention of the research community.

8. ACKNOWLEDGEMENTS

We would like to thank Dr. Michal Young and Dr. Arthur Farley of the Computer and Information Science Department of University of Oregon for their help in this work.

9. REFERENCES

[1] Apache UIMA. `http://uima.apache.org/`.

[2] General Architecture for Text Engineering (GATE). `http://www.gate.ac.uk/`.

[3] OWL Web Ontology Language. `http://www.w3.org/TR/owl-ref/`.

[4] D. E. Appelt, J. R. Hobbs, J. Bear, D. J. Israel, and M. Tyson. FASTUS: A finite-state processor for information extraction from real-world text. In *IJCAI*, pages 1172–1178, 1993.

[5] M. Banko, M. J. Cafarella, S. Soderland, O. Etzioni, and M. Broadhead. Open information extraction from the web. In *IJCAI*, pages 2670–2676, 2007.

[6] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, 284(5), May 2001.

[7] P. Cimiano, S. Handschuh, and S. Staab. Towards the self-annotating web. In *WWW*, pages 462–471, 2004.

[8] X. Dong, A. Y. Halevy, and J. Madhavan. Reference reconciliation in complex information spaces. In *SIGMOD Conference*, pages 85–96, 2005.

[9] D. W. Embley. Toward semantic understanding: an approach based on information extraction ontologies. In *ADC*, pages 3–12, 2004.

[10] C. Fellbaum, editor. *WordNet: An Electronic Lexical Database*. MIT Press, 1998.

[11] M. S. Hanif and M. Aono. Alignment results of Anchor-Flood algorithm for OAEI-2008. In *OM*, 2008.

[12] W. Hu and Y. Qu. Falcon-AO: A practical ontology matching system. *J. Web Sem.*, 6(3):237–239, 2008.

[13] C. H. Hwang. Incompletely and imprecisely speaking: Using dynamic ontologies for representing and retrieving information. In *KRDB*, pages 14–20, 1999.

[14] A. Maedche, G. Neumann, and S. Staab. Bootstrapping an ontology-based information extraction system. *Intelligent exploration of the web*, pages 345–359, 2003.

[15] A. K. McCallum. Mallet: A machine learning for language toolkit. http://mallet.cs.umass.edu, 2002.

[16] B. Popov, A. Kiryakov, D. Ognyanoff, D. Manov, and A. Kirilov. KIM – a semantic platform for information extraction and retrieval. *Nat. Lang. Eng.*, 10(3-4):375–392, 2004.

[17] E. Riloff. Information extraction as a stepping stone toward story understanding. *Understanding language understanding: computational models of reading*, pages 435–460, 1999.

[18] S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Englewood Cliffs, NJ, 2nd edition edition, pages 848-850, 2003.

[19] J. M. Spivey. *The Z notation: a reference manual*. Prentice-Hall, Inc., NJ, USA, 1989.

[20] R. Studer, V. R. Benjamins, and D. Fensel. Knowledge engineering: principles and methods. *Data Knowledge Engineering*, 25(1-2):161–197, 1998.

[21] C. Szyperski. *Component Software*. Addison-Wesley Professional. pages xv-xix, 3-12, 35-47, 2002.

[22] H. Wang, S. Liu, and L.-T. Chia. Does ontology help in image retrieval?: a comparison between keyword, text ontology and multi-modality ontology approaches. In *ACM Multimedia*, pages 109–112, 2006.

[23] Y. Wilks and C. Brewster. Natural language processing as a foundation of the semantic web. *Foundations and Trends in Web Science*, 1(3-4):199–327, 2009.

[24] D. C. Wimalasuriya and D. Dou. Using multiple ontologies in information extraction. In *CIKM*, pages 235–244, 2009.

[25] D. C. Wimalasuriya and D. Dou. Ontology-based information extraction: An introduction and a survey of current approaches. *Journal of Information Science*, 36(3):306–323, 2010.

[26] I. H. Witten and E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann, 1999.

[27] F. Wu, R. Hoffmann, and D. S. Weld. Information extraction from wikipedia: moving down the long tail. In *KDD*, pages 731–739, NY, USA, 2008. ACM.

[28] B. Yildiz and S. Miksch. ontox - a method for ontology-driven information extraction. In *ICCSA (3)*, pages 660–673, 2007.