DESIGNING A SYSTEM FOR SEMI-AUTOMATIC POPULATION OF KNOWLEDGE BASES FROM UNSTRUCTURED TEXT

Jade Goldstein-Stewart
U.S. Department of Defense, U.S.A.

Ransom K. Winder
The MITRE Corporation, Hanover, MD, U.S.A.

Keywords: Information Extraction, Content Analysis, Knowledge Base Population, Human-Computer Interaction, Graphical User Interface.

Abstract: Important information from unstructured text is typically entered manually into knowledge bases, resulting in limited quantities of data. Automated information extraction from the text could assist with this process, but the technology is still at unacceptable accuracies. This task therefore requires a suitable user interface to allow for correction of the frequent extraction errors and validation of proposed assertions that a user wants to enter into a knowledge base. In this paper, we discuss our system for semi-automatic database population and how it handles the issues arising in content extraction and populating a knowledge base. The main contributions of this work are identifying the challenges in building such a semi-automated tool, the categorization of extraction errors, addressing the gaps in current extraction technology required for databasing, and the design and development of a usable interface and system, FEEDE, to support correcting content extraction output and speeding up the data entry time into knowledge bases. To our knowledge, this is the first effort to populate knowledge bases using content extraction from unstructured text.

1 INTRODUCTION

With the rapid growth of digital documents, it is necessary to be able to extract identified essential information at a particular time and create knowledge bases to allow for retrieval and reasoning about the information. Unfortunately, database entry is time consuming. If automatic processes could extract relevant information, such methods could automatically populate "knowledge" bases based on document information. For such knowledge bases to be useful, the end user must trust the information provided, i.e., it must have a high enough degree of accuracy and/or provide a means to correct and validate the information.

In the past two decades research has been dedicated to the automatic extraction of text entities, relations, and events. While the best precision scores for entity extraction are in the 90s (Grishman, 1996), precision for relations is typically less than 40%, and events have an even lower precision. Through the Automatic Content Extraction (ACE) program, an ontology has been developed to characterize the types of extraction, and annotation guidelines have been developed to cover ambiguous cases. For example, in the sentence, "the family went to McDonald's" is McDonald's a facility, an organization, or both? Is the definition of a facility a place that is locatable on a map?

In ACE, entities can be people, organizations, locations, facilities, geographical/social/political entities, vehicles, or weapons, and their mentions are the textual references to an entity within a document. These can be a name ("Ben Smith"), a representative common noun or noun phrase called a nominal ("the tall man"), or a pronoun ("he"). Although good scores have been achieved in entity tagging, there is cause to doubt the extensibility of systems trained for this task (Vilain, 2007). Also, because an entity can be referred to multiple times, an entity potentially has many mentions, and mentions of the same entity are said to be coreferenced. The best extractor scores for coreferencing entity mentions are in the range of 60-80% (Marsh, 1998). Since

Goldstein-Stewart J. and Winder R. (2009).
DESIGNING A SYSTEM FOR SEMI-AUTOMATIC POPULATION OF KNOWLEDGE BASES FROM UNSTRUCTURED TEXT.
In *Proceedings of the International Conference on Knowledge Engineering and Ontology Development*, pages 88-99
DOI: 10.5220/0002307500880099

relations or events can involve referents of multiple entities, the likelihood of accurately extracting all arguments of a relation or event is low.

Using ACE terminology, a relation is defined as an ordered pair of entities with an asserted relationship of a specific interesting type ("ACE English," 2005). So a relation can be thought of as a four tuple: <entity1, relation, entity2, time>. For example, "Scott was a member of ACM for four years" contains a relation where the first entity is "Scott," the second entity is "ACM," and the time is a duration of "four years." This relation has a type of "Organization Affiliation" and has a subtype of "Membership."

An event is defined as a specific occurrence involving participants and a "trigger" word that best represents the event (e.g., "attacked" in "The rebels attacked the convoy yesterday") ("ACE English," 2005). Despite this broad definition, ACE limits its events to a set of types and subtypes that are most interesting. For example, "Jen flew from Boston to Paris" contains a "travel" event, defined as an event that captures movement of people, that is, a change of location of one or more people. The captured arguments of the event would be the travelling entity "Jen," the origin "Boston," and the destination "Paris." Like relations, events can have associated time values ("Working Guidelines," 2007).

In an examination of a leading rule-based commercial extractor on 230 annotated internal documents, it was able to identify the "ORG-AFF/Membership" relation with a precision of 47% (meaning that 47% of the times it identified this relation, the relation existed in the data). The recall was also 47% meaning that 53% of the membership relations in the data were missed by the system. For those relations that were identified, the first entity, the person, was identified with 71% precision, meaning that 29% of the items that the system returned were incorrect. For the second entity, the organization, the precision was 85%. After the company improved the results, the new relation identification improved to 70% while it remained the same for the two entity arguments. A member of this company suggested that this score was considered "very good" for relations and was unsure that much more improvement could be obtained.

Unfortunately, relations and events are often the key assertions that one needs in a knowledge base in order to identify information about people and/or organizations. Due to the high error rate in extraction technology, rather than introducing errors into the knowledge base, a preferred solution might be semi-automatic population of a knowledge base, involving the presentation of extracted information to users who can validate the information, including accepting, rejecting, correcting, or modifying it before uploading it to the knowledge base. This interface must be designed in a manner that supports the users' workflow when doing this task. Ideally, the interface would speed up significantly the time to enter data in the knowledge base manually. Since extractor recall tends to be less than 60%, besides correcting precision errors that the extractor makes, the interface must have the ability for users to add information missed by the extractor (recall errors).

In this paper, we describe the challenges faced in this task and define the design for our system, FEEDE – Fix Extractor Errors before Database Entry. We also discuss the required elements as defined by our end users, the interface's design, and an examination of the extractors used to populate it with initial content to be authenticated. Given the daunting task of manually entering all important information in a knowledge base from unstructured text, we believe this effort is important to save users time, both a valuable commodity in this information age as well as being enterprise cost saving.

To our knowledge, this is the first research effort on developing an interface using content extraction from unstructured text for populating knowledge bases. It has only been in the past year ("Automatic Content Extraction," 2008) that the automatic extraction community has started to focus on text extraction for the purpose of populating databases. In 1996, there was an interface effort for structured data (metadata) (Barclay, 1996). Furthermore, since content extraction efforts have not been focused on the database issue, they are missing certain items that are important for such endeavours. A recent survey of extraction elements important to our users revealed that only 25 out of the 47 requested (53%) were in the ACE guidelines.

2 CONTENT EXTRACTION FOR DATABASING ISSUES

ACE provides specifications for tagging and characterizing entities, relations and events in text, as well as some other features. For entities, the key attributes are type and subtype. Mention categories are also important attributes, determining the specificity of the entities, such as a pronoun referent to an entity name. Relations and events also feature types and subtypes as well as arguments—two for relations, where the order matters, and potentially many different arguments for events where the allowed set depends on the event type. Although quite extensive, the ACE guidelines ("ACE

English," 2005) and temporal annotation guidelines TimeX2 (Ferro, 2005) were not designed for databasing and are missing key items necessary for this purpose. The omissions include:

1. Insufficient data elements from current available extractors to cover what our users want. Therefore new extractors must be developed, which requires requirements gathering, definitions and guidelines. Some have been developed for the highest priority items.
2. In the ACE guidelines, values/traits or contact information for people cannot be associated with people (e.g., "John is 6 ft tall").
3. Nested events do not exist. This is particularly an issue with source attribution. Items attributed to people or a new source are not linked. An example is "According to John, Jane travelled to France." The fact is not necessarily true, but John states it. Since no such extractor existed, we developed one which linked assertions to people.
4. No mechanism in ACE covers group participation. For example, "Anne and John attended Stanford. In spring 2005, the two decided to travel to Europe." ACE does not contain a way to reference "two" to Anne and John, although this is a frequent language pattern.
5. ACE lacks a meaningful primary entity name (PEN) for entities. We define an entity's PEN to be its longest named mention in the document (nominals, titles and pronouns are excluded).
6. ACE lacks descriptors, that is nominals that can define the entity in the context. These important descriptions include titles (e.g., professor), careers (e.g., lawyer), and important roles (e.g., witness). This allows for a distinction between terms that are more of interest than others. We care more that an individual is "prime minister" than about the description of "a man in a green hat." A simple initial solution for distinguishing these is to have an exhaustive gazetteer of all words in each category that are considered descriptors.
7. ACE lacks sufficient time normalization. Databases can allow one to visualize items linked with temporal information and reason over temporal items, if entries have time stamps. The only available temporal normalizer was TIMEXTAG (Ferro, 2005), which did not have sufficient coverage for our purposes. To develop the temporal normalizer, a group of 5 potential users developed grounding rules for key temporal expressions. Users independently mapped all items and then met to come to consensus when there was disagreement. An ambiguous example is "late January," which maps to 21st-31st January. We hope to make this temporal normalizer available to the public soon.
8. While time tagging guidelines include a methodology for sets, they still need to capture the number of members in the set and how often something occurs. For example, a tag for "the past three winters" has no way of representing "three" and a tag for "twice a month" has no way of representing "twice." The knowledge base and database need a way to support this information.

This list indicates that much research and development is required before extraction is at a sufficient level for populating knowledge bases.

Table 1: Results using value-scorer for RB, ST, and ST2 extractors on newswire data.

	ST	RB	ST2
Entities	72.2	72.8	73.1
Entity Mentions	84.6	84.0	84.8
Relations	26.2	24.7	27.3
Events	17.8	N/A	N/A

Table 2: Results for entities in newswire data, where P is precision and R is recall.

	ST	RB	ST2
Unweighted P	49.8	51.9	51.3
Unweighted R	53.5	58.5	57.1
Unweighted F1	51.6	55.0	54.1

Table 3: Results for relations for all three systems and events for ST, where P is precision and R is recall.

	ST Rel.	RB Rel.	ST2 Rel.	ST Events
Unweighted P	34.8	32.8	39.9	2.2
Unweighted R	22.1	24.1	26.3	1.9
Unweighted F1	27.0	27.8	31.7	2.0

Besides the missing key components, as mentioned the accuracy of content extraction is too low for automatic population of databases and perhaps at levels that could frustrate users. The ACE 2005 value results (an official ACE measure) for newswire documents are presented in Table 1 for three participating systems, two statistical (ST and ST2) and one rule-based (RB). The scores for entities and relations/events are presented in Tables 2 and 3, respectively. We present ACE 2005 since it had more participation on the relations task than ACE 2007, and ACE 2008 did not evaluate events. Analysis indicates only slight performance increases for systems in 2007 and 2008. These results were computed using the ACE 2005 evaluation script on each extractor's documents compared to reference documents tagged by humans. In the script's unweighted scores, a relation or event is considered

Table 4: Common extraction errors and how the interface handles them.

Error Type	Error	Solution Type	Solution
Entity Label	Type/Subtype Error	Preprocessing	Perform string matches for the primary entity name (or any named mention) of entities. If one matches with an entity of interest of different type/subtype, present to User.
Entity Error	Misspelled by Doc Author	Interface	User corrects by typing in the interface. This applies to punctuation and capitalization errors too.
Entity Error	Too Much/Little Extracted	Preprocessing	Compare entity of interest primary entity names to text of other entities. If there is significant crossover, offer the entity of interest as possibility to User for assertions involving such entities.
Entity Error	Too Much/Little Extracted	Interface	User can add missing information or delete extraneous information.
Relation/ Event	Argument Error	Preprocessing	All proximate entities (mentions within X words) to relation/events should also be offered as alternatives for the actual relation/event.
Relation/ Event	Argument Error	Interface	User examines this relation/event and evidence to recognize an incorrect argument and must either ignore, modify or add a new relation/event. Modifying includes selecting a new argument from a drop-down menu list with possible entities for that argument.
Relation/ Event	Type/Subtype Error	Interface	User observes this error in the evidence and must either ignore, modify or add new relation/event. Relation/event types can be selected by drop-down menu.
Relation/ Event	Spurious Relation/Event	Interface	User observes this relation is spurious in the evidence and can ignore (hide) the relation/event.
Relation/ Event	Missing Relation/Event	Interface	If User can recognize this error by viewing the document, new relation/event can be added in the interface. Adding information is supported in the interface by menus with a list of allowed relations/events.
Coreference	Spurious Coreferences	Interface	User must recognize in evidence that entity mention and primary entity name are different and must either ignore, modify or add new relation/event.
Coreference	Missing Coreferences	Preprocessing	All proximate entities (mentions within X words) to relations/events should be offered as possibilities (with low confidence) for the actual relation/event arguments.
Coreference	Split Entities	Interface	When validating, the user can assign the same knowledge base id to the two entity chains and then the data will be merged in the knowledge base.

to be located in a system-tagged document if there are no missing reference arguments and no attribute errors. Precision equals the number of these mappable pairs over the total number the extractor found. Recall equals the number of these mappable pairs over the total number in the reference text. These are combined to produce an F1 score.

Note that any error present in the data would need to be fixed by a user that chose to utilize that piece of information in the knowledge base. Thus the trade-off between precision and recall has great significance for the task, as a higher precision would imply less user correction of errors in extracted data, but also requires more manual entry of missing assertions, while a higher recall implies the reverse.

Given the low precision and recall as shown by the ACE 2005 results for relations, we believe it is essential to have an effective user interface to allow users to correct the information extracted incorrectly from the documents (precision errors) as well as to enter missed information (recall errors). Though the results available for events are not as comprehensive, these scores are even lower than those for relations.

In terms of the interface, we define effective as (1) intuitive, (2) easy to use, (3) minimizing mouse clicks, (4) following the workflow, (5) faster than manual entry, (6) tailored to user requirements and preferences, and (7) assisting and guiding the user in completing the task of creating entries for the knowledge base.

Since there are so many potential extraction errors and the knowledge base requires vetted information, the user must validate all information before it is uploaded. Table 4 displays a list of common problems encountered in extraction. Solutions to these issues either require action to be performed by the interface in pre-processing before the information is presented to the end user or in the interface itself, essentially assisting the end user in

making corrections based on observations of the evidence.

Something else important to the interface that is absent from ACE is confidence levels in the information. We use an extractor confidence, if provided, as a factor into the confidence score presented to the user. Pronominal references lower the confidence since their accuracy is only approximately 80%. We also use evaluation knowledge about relations and events as a component in the final confidence score. Other factors that could be included might be based on the contextual source; this includes weak attribution, conditional terms, hypothetical statements, or future tense.

Confidence levels indicate to the user whether this assertion has a good chance of being correct, i.e., it helps to focus items they might choose to validate. It is important to note that as we add more information that essentially second-guesses what the extractor has produced, it becomes necessary to distinguish between what is believed and what is not, and confidence plays a role here too.

3 ERROR ANALYSIS

Because relations and events are the most commonly desired pieces of information to be gleaned from a document, we provide examples of the types of errors observed involving relations and events, drawing from results achieved on the 2005 ACE evaluation. Since the most interesting attributes are type and subtype, in this section we only record attribute errors in these. This change would increase the results in Table 3 by less than 13%. These results are still low and indicate that there are many items that need to be corrected for total accuracy. Here we further examine a leading statistical extractor (ST) and a leading rule-based extractor (RB). Event results were only available for ST.

In the tables below, we examine the frequency of specific error types independently. Consider the sentence "British Prime Minister Tony Blair left Hong Kong." This contains a relation of type/subtype "PHYS/Located." The extent of the first argument is "British Prime Minister Tony Blair," while the head of the first argument is "Tony Blair." The extent and head of the second argument are both "Hong Kong." It is the head—a more specific piece of text—that determines whether two mentions in separately tagged documents map to one another. There is potential for error with any of these elements. Table 5 shows the cases where relations (and events) are tagged with the wrong type or subtype, while Table 6 shows span errors for the full arguments of relations and their heads.

Table 5: Relation and event type/subtype error rates observed for newswire documents if other requirements for finding relation or event are filled.

	Relations Tagged w/ Incorrect Type/Subtype	Events Tagged w/ Incorrect Type/Subtype
ST	16.6%	11.4%
RB	18.5%	N/A

Table 6: Argument span error rates observed across relation mentions. The upper results exclude cases of mismatched relation type/subtype, while the lower results ignore the relation type/subtype and just evaluate the head and extent spans.

	Arg1 Head Span	Arg2 Head Span	Arg1 Extent Span	Arg2 Extent Span
ST	2.5%	4.6%	26.9%	13.1%
RB	4.9%	8.0%	22.2%	14.8%
ST	3.4%	3.8%	25.3%	12.2%
RB	5.4%	7.8%	22.9%	16.1%

Considering cases where relations are potentially identified but are tagged with an incorrect type or subtype, the extractors comparably misidentify the types of relations in the reference corpus at rates of 16.6% and 18.5% for ST and RB, respectively.

Turning to the relation's arguments, their mentions can have errors in the span of their extent (text tagged as being the full argument entity) and head (the key text that defines the argument entity). For both extractors the error rates for extent spans are higher for the first argument than for the second. Head span errors are lower for both extractors, but because relations that do not have overlapping heads will be classified as spurious, it is not surprising that the feature which is the criterion for mapping relations between reference and system-tagged documents has a low error rate when examined.

As Table 7 shows, ST misses 66.9% of these specific relation mentions and RB misses 72.8%. If the relations themselves are considered, as opposed to their specific mentions, then 74.0% of relations are missed by ST and 70.7% of relations are missed by RB. When considering relations as opposed to relation mentions, some of this error is propagated from errors in entity coreferencing. If perfect entity coreferencing is assumed, then the number of missing relations drops to 60.6% for ST and 59.3% for RB, which is still a high number in both cases. These numbers are still quite high if we permit relations to be recognized that are unmatched with a reference relation and have the appropriate

arguments but a different type/subtype, 52.1% for ST and 50.1% for RB. While the advantage in terms of recall belongs to RB, the extraction results on relations are highly errorful, and even when accounting for errors, the precision and recall are quite low, indicating that human validation is necessary for relations. Examining the spurious relations, these make up a significant portion of the returned results. Even presuming perfect entities and ignoring tag mismatches, more than a fourth of returned relations are spurious for either ST or RB.

Table 7: Rates of missing and spurious relations. STa/RBa results consider cases of mismatched type/subtype missing or spurious, while STb/RBb results ignore type/subtype.

	Missing Relations (Perfect-Coref.)	Missing Relations	Missing Relation Mentions
STa	60.6%	74.0%	66.9%
RBa	59.3%	70.7%	72.8%
STb	52.1%	68.9%	60.2%
RBb	50.1%	64.1%	65.6%
	Spurious Relations (Perfect-Coref)	Spurious Relations	Spurious Relation Mentions
STa	40.1%	59.1%	44.8%
RBa	52.9%	60.2%	65.5%
STb	27.2%	51.0%	33.6%
RBb	42.2%	51.1%	56.3%

Table 8: Rates of missing and spurious events. The STa results consider cases of mismatched type/subtype missing or spurious, while the STb results ignore type/subtype.

	Missing Events (Perfect-Coref.)	Missing Events	Missing Event Mentions
STa	84.8%	87.6%	87.8%
STb	82.9%	86.0%	87.6%
	Spurious Events (Perfect-Coref.)	Spurious Events	Spurious Event Mentions
STa	82.4%	85.7%	84.0%
STb	80.1%	83.8%	83.7%

Table 9: Error rates (in %) for spurious and missing arguments for all ST events where "a" results consider cases of mismatched type/subtype missing or spurious, while "b" results ignore type/subtype. Arguments assigned the wrong role are considered found in the results marked with an *. Numbers in the parentheses include arguments of missing or spurious events in their counts.

	Missing Args	Missing Args*	Spurious Args	Spurious Args*
a	54.6 (76.2)	51.0 (74.4)	30.1 (52.3)	24.7 (48.6)
b	62.4 (74.1)	55.2 (69.1)	41.7 (48.1)	30.4 (38.1)

Table 8 displays the results for missing and spurious events. These occur at very high rates, with 87.8% of specific event mentions missed and 87.6% of events missed. Even when perfect entity coreferencing is assumed, the percent of events missed only drops to 84.8%. As for spurious event mentions, these make up 84.0% of tagged event mentions and 85.7% of tagged events. Once again assuming no errors with entity coreferencing only drops the percentage of spurious events to 82.4%. Table 5 reveals that only 11.4% of events are potentially tagged with the wrong type/subtype. If this restriction is ignored, scores for missing and spurious events improve only marginally.

These numbers are so high mainly due to the difficulty in capturing all reference arguments, a requirement for finding events. Note that events are chiefly defined by their arguments. Examining event arguments more closely, we discover that the error rate for missing arguments is about 54.6%. This error rate increases dramatically though if the totals are allowed to include the missing arguments of completely missing events, rising into the 70s. With regard to spurious arguments, they make up 30.1% of the arguments identified. This number can rise into the 50s if the arguments of completely spurious events are included in the totals. These numbers are presented in Table 9. While results improve when restrictions on event type/subtype and argument role are slackened, they still remain significantly high.

Apart from type, subtype, and arguments, events in ACE are defined by attributes ignored in these results such as tense, genericity, modality, and, perhaps most importantly, polarity. The last of these tells whether or not the event is positive or negative, which means essentially whether or not the event happened. Excluding tense, the scores for these are high, but this is due to the tagger consistently tagging all events with a particular value (its most common). Therefore, the scores for these values are meaningless, which is particularly significant for polarity, as it essentially changes the entire meaning of an event. Considering this as well as the low recall and precision of both events and their constituent parts, it is clear that these also cannot be accurately mediated by automatic means alone but require human validation and correction.

4 SYSTEM, DATABASE, AND INTERFACE

An analysis of the issues involved in bringing together these different approaches gave rise to a list of challenges that must be dealt with in order to

achieve a successful outcome for the project. Table 10 lists these challenges and a description of each. Among these are issues directly related to content extraction discussed earlier (Challenges #1-5) as well as extraction issues for text sets that present special challenges and require special extractors to be trained to handle them (Challenge #6). An example of the latter might be text that is completely capitalized. This section addresses other challenges related to the interface needed for correcting the extracted information as well as database population.

Our system is designed around the extraction of assertions (relations/events) about people from unstructured text (e.g., newswire documents). These assertions can come from a batch of documents or a single document. In the case of a batch, there may be redundant repetitive data. For example, in a batch of documents about UK politics, many may state that Gordon Brown is a member of the Labour Party in Britain. Users may (1) want to be able to enter this information (with source) only once to allow them to concentrate on the entry of novel information or (2) enter this data many times for use as supporting information. If the user wants to enter the data only once, the system tries to first present the extracted candidates mostly likely to be accurate. This eliminates text snippets that have a pronominal reference since coreference chains can be determined with an F1 score of approximately 80% (Harabagiu, 2001). For example, "he is a member of the Labour Party" would be automatically be assigned a lower confidence score than "Gordon Brown is a member of the Labour Party."

Another issue is that all events and relations are treated as facts by ACE. We prefer to label them assertions and to assign a source to the assertion, if provided. In a case like "Michael said" or "BBC reported," Michael or BBC would be tagged as the source. When presenting content to a user in the interface, it is important for them to know the trustworthiness of a relation or event, and thus it is important in populating the knowledge base. So in this interface and database definition, a field for source attribution to an entity is available for all events and relations. Since no software existed for source attribution, code was written to provide this functionality.

The system is designed to allow users to set preferences to specify which types of data to extract, correct, and validate. For example, one user may be interested in social network information, such as family and friends. Another user might be interested in the travel of key sports figures.

Since extracted information is often incorrect, the system must also have a mode for correction and validation of information. This is done through drop down text menus in the hope that the user can quickly select a correct entity or relation. If the entity was missed by the extractor or misspelled due to author error, the user has the option to edit it. Since extractors frequently miss information ("NIST 2005," 2005), the system also requires a mode for entering information missed by the extractors.

Table 10: List of challenges.

Challenge	Description
1. Accuracy of content extraction	The accuracy of extractors unaided is prohibitively low on key assertions.
2. Precision/recall trade-off	High precision means less User correction, but more manual entry. High recall means the reverse.
3. Current extractors lack appropriate inventory	47 data elements map to entities, relations, and events, but 20 elements are missing from or lack suitable ACE version.
4. User definitions don't match ACE	Definitions differ between terms in ACE and how Users define them.
5. Temporal anchoring and normalization	Extracting time tags and resolving them with source date. Some cases are complex or highly ambiguous.
6. Data specific challenges	Extracting from data sources that present special challenges when compared to the majority of texts.
7. Primary entity name selection	Detecting the most appropriate primary entity name, and allowing Users to choose primary entity names and effectively use them.
8. Disambiguating entities with KB	Assisting User in merging entity with existing record or creating a new one.
9. Accuracy of coreference resolution	Detecting if two phrases refer to the same entity, including a related issue of coreferencing "sets" of people.
10. Mapping from one schema to another	Converting User specification to extractor output and this output to the knowledge base data model.
11. Building successful interface & prototype	Developing an interface that allows easy entry, efficient correction and data verification.
12. Measurement of success	Measuring usability aspects of effectiveness, efficiency and User satisfaction.

The system must be able to select for any entity its primary entity name (PEN) from the list of names in the document (Challenge #7). As far as we are aware, there have been no evaluations or research on how easy this task is, but here we define the PEN as the longest named mention of an entity in the document. For example, the PEN for David Beckham, the English footballer, would be "David Robert Joseph Beckham." This name then has to be

resolved with names currently existing in the knowledge base. The user has the option of creating a new entry for this name or adding the information to entities that already exist. Once this is determined, the system stores this name (and knowledge base id) in case it should arise in future documents. This software has been written but has not yet been evaluated. The user may want to change the display name to be different from the PEN, so the system has a method to allow for this and to link the PEN with the display name.

The user also has the option of stating a specific interest in information that appears about a particular person. Suppose a person is interested in Michael Jackson, the English footballer. Once Michael Jackson has a knowledge base identification number, the user can select him as an entity of interest (EOI). EOIs are separated in the user interface for the user. Documents contain many entities (typically about 50-100 per document) and if the user is only focusing on a few, having them as EOIs, makes them easy to find. Other entities that are assigned a knowledge base id, but not chosen as an EOI are referred to as resolved entities and are accessible from the interface as well. A resolved entity can be easily converted to an EOI, especially since it already has a knowledge base id.

The system stores a list of names that it has found in documents associated with a user's EOIs. For Beckham it could have both "David Beckham" and "David Robert Joseph Beckham." It compares these names with new documents to offer knowledge base ids for entities that have previously been seen by the system and assigned an id by the user.

With a content related database, entity disambiguation is required when adding information from new documents (Challenge #8). If Michael Jackson, the English footballer, is an entity of interest, the user will have to determine that the information in the document being presented is his or her Michael Jackson. For example, "Michael Jackson" could be the singer, the American linebacker, the English footballer, the British television executive, or the former Deputy Secretary of the U.S. Department of Homeland Security.

The first time that the user has an entity Michael Jackson and goes to validate an assertion about Michael Jackson, the system returns the various named entities already in the knowledge base as well as any stored information that would assist the user to determine a match for this person. The user chooses whether to add the information to one of these existing entities or to create a new one. Selection or the creation of a knowledge base id is required for every entity in the assertion when the user chooses to validate an assertion as every entry in the knowledge base must have an identification number to determine its uniqueness.

Related to this disambiguation issue is entity coreferencing, an arena where extractors experience difficulty (good performance with names, moderate with pronouns, and poorest with nominals). There are also some inherent limitations with ACE in this regard, such as an inability to deal with sets of people (Challenge #9). For example, in "Ron met Joy after class, and they went to the store," ACE cannot coreference "they" to "Ron" and "Joy." This requires the development of software to extract possibilities other than those currently available.

All data is stored in an intermediary database before being uploaded to the main knowledge base. This allows the user to stop in a middle of a session and return before committing the data to the knowledge base (which is designed for manual entry). The architecture of our system that makes use of the FEEDE service is shown in Figure 1. Additionally because the schemas used for ACE and the corporate knowledge base are different, they must be mapped to one another (Challenge #10).

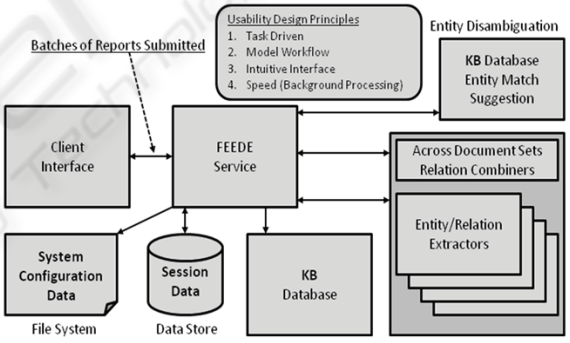

Figure 1: System architecture.

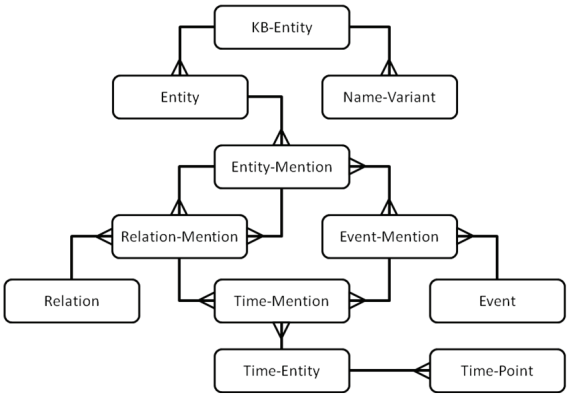

Figure 2: Overview of basic intermediary database structure.

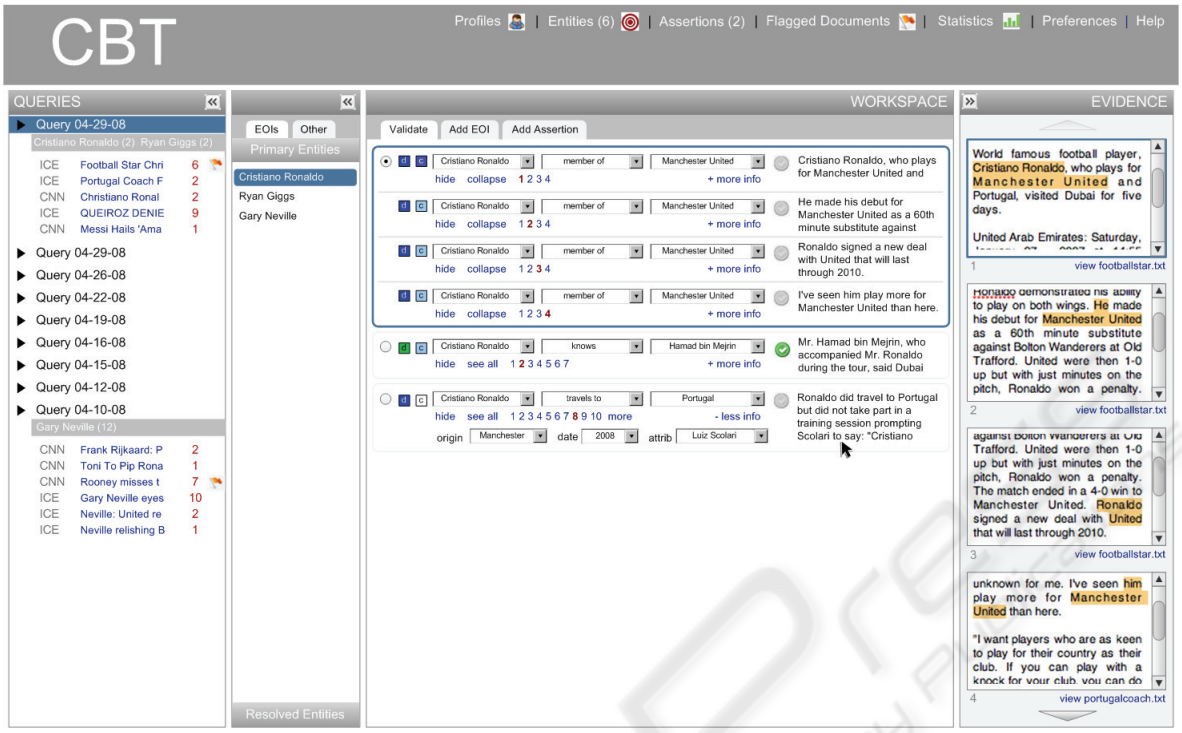

Figure 3: Example mock-up of the correction interface where a relation between two entities is selected and different mentions of this relation are presented to the user with corresponding textual evidence in the panels to the right. Note: The actual interface exists, and its appearance is very similar to this mock-up.

Although extraction from unstructured text has low accuracy, extraction from the headers can be done with high accuracy. Extraction of items such as the document source, title, date and other key information save the user from manually entering this information. This data is extracted and presented to the user when uploading validated data. In particular, the document date is also required for temporal normalization, to resolve items such as "in February" if the year is not provided.

5 INTERMEDIARY DATABASE

The intermediary database is a relational database. When the documents are processed, the information extraction is stored in this database awaiting validation by the user. Thus it is populated with entities, relations, events, and time expressions, each with varying degrees of confidence. It is not intended for long-term but rather temporary storage of extracted information from documents so that a user can verify, correct, and validate content extraction results. Only the user validated or entered results are used to populate the main knowledge base after the user chooses to upload them to the knowledge base. A basic schema for the database is shown in Figure 2. Links without prongs indicate "has one" while crow's feet indicate "has many." Entity-mentions have one entity, relation-mentions have two entity-mentions, and event-mentions and entity-mentions have a many-to-many relationship. Following this schema, each entity entry in the knowledge base (KB-Entity) can have any number of entities (and variants on its name) associated with it across document sets. Each of these entities in turn can have any number of mentions in a document. Each mention can be part of any number of relation-mentions or event-mentions, each of which refers to a single relation or event. Relation and event-mentions can also be associated with any number of time-mentions, each referring to a single time-entity, grounded to a timeline with defining time-points.

6 INTERFACE DESIGN

The correction/validation interface (a mock-up shown in Figure 3) that lies atop the intermediary database must allow for easy entry of missed information as well as efficient correction and verification of extracted data. The interface must

make it as easy as possible for users to enter items not found by extractors into the database. This requires an examination of user workflow and the minimization of time required for the critical steps in extracting assertions (Challenge #11). Of paramount importance here is maximizing the efficiency, efficacy, and satisfaction of the users with the interface (Challenge #12), three properties of usability that should be independently considered (Frokjaer, 2000).

This interface has the following basic flow of content that is extracted from a corpus of documents. Because information presented to the user is "entity centric," meaning that a user specifies entities of interest (EOIs) and the interface provides the relevant entities as well as related relations and events that involve those entities, the first part of this content flow from a set of documents is the list of entities (EOIs if the user has already specified these). Additionally, taken from a list of entities that appear as arguments to relations and events involving the EOIs, a list of secondary entities is also populated.

When a specific entity is selected in the workspace, the flow progresses to information about the entity from the document corpus. This information requires user validation. These are relations and events, with associated fields for arguments and other attributes (time, source-attribution, et al). Each of these pieces of evidence is also given a confidence (from high to low depending on textual and source factors) and an indication of whether the content is already validated and present in the KB (the two boxes to the left of relations in Figure 3). The shading of the boxes allows the user to quickly scan the data. Dark blue in the first box indicates that the information is not present in the main knowledge base and dark blue in the second box indicates that there is high extractor confidence in that information, suggesting that the user might want to examine that item first for validation. Tracking and displaying the presence or absence of the information in the knowledge base is important, as the users are often only interested in entering new information.

In the second piece of evidence, "he" is coreferenced with "Ronaldo." Because the user is validating at an entity level, the PEN is the name present for each field that represents an entity, not the entity mention's referent in the particular piece of text. The user then verifies relations and events by checking the textual evidence to the right of the relations, as well as the larger context for each on the far right if necessary. In the cases where the referent in the text evidence does little to clarify who the entity is (as with a pronoun), then other mentions of the entity can be indicated, as shown with the underlining in the 2nd evidence example in Figure 4.

Because there are potentially multiple mentions of the same relation or event in the document corpus, the user can specify whether to see one or all of these at once. Each item the user desires to have entered in the main knowledge base must be checked and validated (the check circle on the right of the text). This information can be ignored if incorrect or corrected to form an accurate relation/event. Each of the arguments to the relation/event must be present in the knowledge base. The interface is structured so that arguments to relations and events can be modified via drop-down menus or typing in text fields. Some of these are accessed by clicking "more info." When the relation or event has been corrected, if necessary, and is present in the material, then the user validates it for entry into the knowledge base. Otherwise there is no validation or a different relation or event can be validated if corrections significantly changed the relation or event.

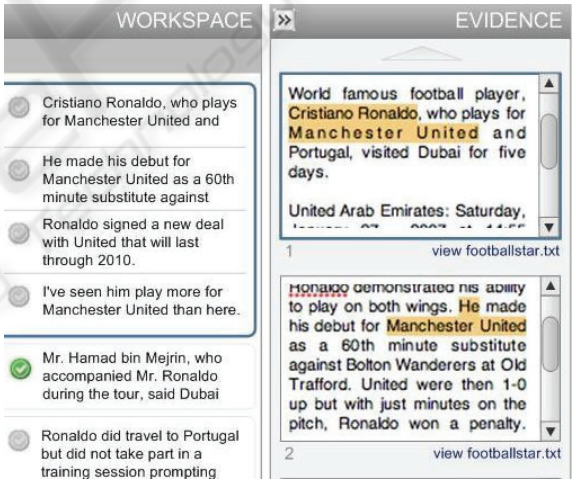

Figure 4: Close-up example of evidence.

Figure 3 also shows the implicit task flow as one looks from left (document sets and entity lists) to the right (extracted content to be corrected and textual evidence). The selected relation can be expanded (as in Figure 3 with the "member of" relation) to reveal the different instances of the relation in the text and the evidence that support it.

The interface also provides a convenient way to find out more information about entities, helping in disambiguating them. By simply leaving the mouse cursor over an entity in the workspace, the interface will generate a pop-up display about the entity,

including information on its knowledge base identity, its type, and when it was last updated.

7 DATABASE FIELDS FOR RELATIONS

Given the attributes present in ACE in addition to those we add to extend it, we can present a picture of what fields are necessary to store sufficient information in our database. As an example, the list of fields, along with definitions, for the relation-mention type follows. First we describe the fields necessary for the relations to interact with the knowledge base. Fields that must contain a value are marked as (R) for "Required."

- account_id (R) = Unique identification of the user looking at the relation, which is used to identify who validated and committed it.
- validated_date = Time the relation was validated by the user.
- committed_date = Time the relation was committed to the knowledge base by the user.
- modified_date (R) = Last time the relation was modified by the user or an updated extraction.
- comments = Comments by the user on the information.
- user_confidence = How certain the user is that this relation holds based on trustworthiness and ambiguity of the source text.

What follows next is a list of fields necessary for thorough relation-mention definitions in the database.

- doc_id (R) = Unique identification of the document where relation was found.
- extractor_info (R) = Features used to do the extraction, which determine the extractor used, the version, its parameter settings, etc.
- evidence (R) = Text of the document where the relation was found.
- evidence_start (R) = Beginning index of the relation in the evidence.
- evidence_end (R) = End index of the relation in the evidence.
- paragraph_start = Beginning index of the paragraph snippet.
- paragraph_end = End index of the paragraph snippet.
- arg1_entity_mention_id (R) = Unique mention of the subject of the relation in the relevant document.
- arg2_entity_mention_id (R) = Unique mention of the object of the relation in the relevant document.
- relation_type_id (R) = Uniquely determined by extracted type and subtype, this corresponds to a specific relation type that can be inserted into the knowledge base.
- tense = When the relation occurs with respect to the document. This can be past, present, future, or unspecified.
- attributed_to = A reference to a person source or document source in the document for this relation mention.
- polarity = TRUE if assertion is so stated and FALSE when it does not hold (NOTE: this is not part of the ACE guidelines for relations). Most users prefer to enter positive assertions only, so the default value is TRUE.
- source_confidence = Likelihood that this relation holds given conditional statements (e.g. may, believe, probably, etc.) and can be attributed to the writer of the document or some entity in the document or a combination of the two.
- extractor_confidence = How certain the extractor is that this relation holds based on its own metrics.
- system_confidence = A measure calculated by the system based on extractor confidence and its own internal knowledge.
- hidden = Set by the user, TRUE when the user wishes to hide the relation and FALSE otherwise.
- inKB = TRUE when the relation is in the knowledge base, FALSE otherwise.

With these fields, the relations are fully defined and prepared to be inserted into the final knowledge base.

8 DISCUSSION

In this paper we have discussed a design for an interface to aid in the process of populating a knowledge base with corrected and validated knowledge originating from text sources that combines the benefits of both human users and automated extractors in order to make the process more efficient. We also examined the basic structure needed for the interface's underlying temporary database and the properties necessary for the information to possess in order to disambiguate and fully describe entities.

Also examined was the performance of two extractors, one rule-based and the other using statistical methods. The key point of this examination is to demonstrate that the extractors are very prone to errors and that in order to populate a knowledge base with what they produce a human

being is still required to examine and validate text, even if facilitated by the interface and extractors' guidance. This interaction in the interface remains the most crucial element in ensuring that the process will be brisk, successful, and have minimal errors.

We expect that this system will save users significant time over manual entry of information into the knowledge base. Users have experimented with the prototype system and really like it. Their stated expectation is that this will save them lots of time. Our intention is to speed up the data entry process by 50%, which preliminary studies indicate is the case. We are currently working to improve our interface based on the user feedback and performing experiments on the amount of time saved using this process.

ACKNOWLEDGEMENTS

We would like to thank all of our team members and consultants for their advice and numerous contributions to this project.

REFERENCES

ACE (automatic content extraction) English annotation guidelines for entities version 5.6.1. (2005). Retrieved May 7, 2008 from: http://projects.ldc.upenn.edu/ace/docs/English-Entities-Guidelines_v5.6.1.pdf

ACE (automatic content extraction) English annotation guidelines for events version 5.4.3. (2005). Retrieved May 7, 2008 from: http://projects.ldc.upenn.edu/ace/docs/English-Events-Guidelines_v5.4.3.pdf

ACE (automatic content extraction) English annotation guidelines for relations version 5.8.3. (2005). Retrieved May 7, 2008 from: http://projects.ldc.upenn.edu/ace/ docs/English-Relations-Guidelines_v5.8.3.pdf

Automatic content extraction 2008 evaluation plan. (2008). Retrieved 2009 from: http://www.nist.gov/speech/tests/ace/2008/doc/ace08-evalplan.v1.2d.pdf

Barclay, C., Boisen, S., Hyde, C., & Weischedel, R. (1996). The Hookah information extraction system, *Proc. Workshop on TIPSTER II* (pp. 79-82). Vienna, VA: ACL.

Evaluation scoring script, v14a. (2005). Retrieved September, 25, 2008, from: ftp://jaguar.ncsl.nist.gov/ace/resources/ace05-eval-v14a.pl

Ferro, L., Gerber, L., Mani, I., Sundheim, B., & Wilson, G. (2005). TIDES-2005 standard for the annotation of temporal expressions, Technical Report, MITRE. Retrieved June 3,2008 from:http://timex2.mitre.org/ annotation_guidelines/2005_timex2_standard_v1.1.pdf

Frokjaer, E., Hertzum, M., & Hornbaek, K. (2000). Measuring usability: are effectiveness, efficiency, and satisfaction really correlated? *Proc. ACM CHI 2000 Conference on Human Factors in Computing Systems* (pp. 345-352). The Hague: ACM Press.

Grishman, R., & Sundheim, B. (1996). Message understanding conference – 6: A brief history. *Proc. 16th International Conference on Computational Linguistics (COLING)* (pp. 466-471). Copenhagen: Ministry of Research, Denmark.

Harabagiu, S., Bunescu, R., & Maiorano, S. (2001). Text and knowledge mining for coreference resolution. *Proc. 2nd Meeting of the North America Chapter of the Association for Computational Linguistics (NAACL-2001)* (pp. 55–62). Pittsburgh: ACL.

Marsh, E., & Perzanowsi, D. (1998). MUC-7 evaluation of IE technology: overview of results. Retrieved 2009 from: http://www.itl.nist.gov/iaui/894.02/related_projects/muc/proceedings/muc_7_toc.html

NIST 2005 automatic content extraction evaluation official results. (2006). Retrieved May 7, 2008 from: http://www.nist.gov/speech/tests/ace/2005/doc/ace05eval_official_results_20060110.html

Vilain, M., Su, J., & Lubar, S. (2007). Entity extraction is a boring solved problem—Or is it? *HLT-NAACL – Short Papers* (pp. 181-184). Rochester: ACL.

Working guidelines ACE++ events. (2007). Unpublished Internal Report.