Hot Stuff at Cold Start: HLTCOE participation at TAC 2014

Tim Finin
University of Maryland,
Baltimore County

Paul McNamee
Johns Hopkins University
HLTCOE

Dawn Lawrie
Loyola University
Maryland

James Mayfield
Johns Hopkins University
HLTCOE

Craig Harman
Johns Hopkins University
HLTCOE

Abstract

The JHU HLTCOE participated in the Cold Start task in this year's Text Analysis Conference Knowledge Base Population evaluation. This is our third year of participation in the task, and we continued our research with the KELVIN system. We submitted experimental variants that explore use of forward-chaining inference, slightly more aggressive entity clustering, refined multiple within-document conference, and prioritization of relations extracted from news sources.

1 Introduction

The JHU Human Language Technology Center of Excellence has participated in the TAC Knowledge Base Population exercise since its inception in 2009. Our focus over the past year was on the Cold Start task. We attempted to enhance our KELVIN system (McNamee et al., 2012; McNamee et al., 2013; Mayfield et al., 2014) by application of forward-chaining inference rules, improved cross-document entity coreference, refined within-document coreference, biasing extraction towards relations identified in news, and through a variety of software engineering and architectural modifications

In the rest of the paper we present our system, which is architecturally very similar to our 2013 submission, and briefly discuss our experimental results.

2 Cold Start KB Construction

The TAC-KBP Cold Start task is a complex task that requires application of multiple layers of NLP software. The most significant tool that we use is a NIST ACE entity/relation/event detection system, the BBN SERIF system. SERIF provides a substrate that includes entity recognition, relation extraction, and within-document coreference analysis. In addition to SERIF, significant components which we relied on include: a maximum entropy trained model for extracting personal attributes (FACETS, also a BBN tool); cross-document entity coreference (the HLTCOE *Kripke* system); and a procedurally implemented rule system.

The system is organized as a pipeline with three stages: (i) document level processing done in parallel on small batches of documents, (ii) cross-document co-reference resolution to produce an initial KB, and (iii) knowledge-base enhancement and refinement through inference and relation analysis. The next section describes the major steps in these stages.

3 System Description

KELVIN runs from two Unix shell scripts[1] that execute a pipeline of operations. The input to the system is a file listing the source documents to be processed; the files are presumed to be plain UTF-8 encoded text, possibly containing light SGML markup. During processing, the system produces a series of tab-separated files, which capture the intermediate state of the growing knowledge base. At the end of the pipeline the resulting file is compliant with the Cold Start guidelines.

Our processing consists of the following steps, which are described in detail below:

[1] Named Margaret and Fanny after Lord Kelvin's wives.

1. Document-level processing
2. Curating intra-document coreference
3. Cross-document entity coreference
4. KB cleanup and slot value consolidation
5. Applying inference rules to posit additional assertions
6. KB cleanup and slot value consolidation
7. Selecting the best provenance metadata
8. Post-processing

The *Margaret* script performs the document-level processing in parallel on our Sun Grid Engine computing cluster. *Fanny* executes the balance of the pipeline, and many of these steps are executed as a single process.

3.1 Document-Level Processing

BBN's SERIF tool[2] (Boschee et al., 2005) provides a considerable suite of document annotations that are an excellent basis for building a knowledge base. The functions SERIF can provide are based largely on the NIST ACE specification,[3] and include:

- identifying named-entities and classifying them by type and subtype;
- performing intra-document coreference analysis, including named mentions, as well as coreferential nominal and pronominal mentions;
- parsing sentences and extracting intra-sentential relations between entities; and,
- detecting certain types of events.

We run each document through SERIF, and extract its annotations.[4] Additionally we run another module named FACETS, described below, which adds attributes about person entities. For each entity with at least one named mention, we collect its mentions, the relations, and events in which it participates. Entities comprised solely of nominal or pronominal mentions are ignored for the Cold Start task, per the task guidelines.

FACETS is an add-on package that takes SERIF's analyses and produces role and argument annotations about person noun phrases. FACETS is implemented using a conditional-exponential learner trained on broadcast news. The attributes FACETS can recognize include general attributes like religion and age (which anyone might have), as well as some role-specific attributes, such as employer for someone who has a job, (medical) specialty for physicians, or (academic) affiliation for someone associated with an educational institution.

3.2 Intra-Document Coreference

Within document conference is one of the areas that leads to numerous errors in downstream applications such as the Cold Start task. For example, we have observed cases where family members or political rivals are mistakenly combined into a single entity cluster. This creates problems in knowledge base population where correct facts from distinct individuals can end up being combined into the same entity. For example, if Bill and Hillary Clinton are mentioned in a document that also mentions that she was born in the state of Illinois, a conjoined cluster might result in a knowledge base incorrectly asserting that Bill Clinton was born in Illinois.[5] Therefore in our prior submission and in this submission, this issue has received specific attention.

In last year's submission, we built a classifier to detect such instances and then remove document entities identified as problematic from the KB. Our classifier uses name variants from the American English Nickname Collection[6] and lightweight personal name parsing to identify acceptable variants (*e.g.,* Francis Albert Sinatra and Frank Sinatra). If our rules for name equivalence are not satisfied, then string edit distance is computed using a dynamic time warping approach to identify the least cost match; two entity mentions that fail to meet a closeness threshold by this measure are deemed to be mistakenly conflated. Organizations and GPEs are handled similarly. Name variants for GPEs include capital cites and nationalities for known countries. In addition, both are permitted to match with acronyms.

[2] Statistical Entity & Relation Information Finding

[3] `http://www.itl.nist.gov/iad/mig/tests/ace/2008/doc/ace08-evalplan.v1.2d.pdf`

[4] We used an in-house version of SERIF, not the annotations available from LDC.

[5] He was born in Arkansas.

[6] LDC2012T11

In this year's submission, we decided to merge the opinions of two separate intra document coreference systems, SERIF and the Stanford CoreNLP tools. The hypothesis was that the two systems might fail in different ways so a combined system might outperform either independent tool.

In order to accomplish this task, the first step is to align the two separate mention chains. This is done by aligning each mention independently, whether it is a named, nominal, or pronominal reference. One of the main challenges with alignment is that the different systems could identify different entities from within the same span of text (e.g "Christian" and "the Christian church"). We use a rule-based approach to identify the one best alignment for each mention.

Once all mentions are aligned, new entities are created that represent merged entities that were identified by both systems. In addition there are other mentions only identified by one of the systems. Initially, these form separate entities. At this point each entity is given a score by the classifier developed last year. These scored entities are considered for remerging based not the classifier scores.

If merging two entities that were unified by one of the systems does not negatively impact the classifier score, then the merging will occur. After entity merging, mentions only identified by one of the systems are considered for merging into a merged entity. Merging of named mentions occurs if it does not negatively impact the merged score. Nominal mentions are also considered for merging; however, a different classifier was developed based purely on the string edit distance. Nominal mentions are merged if the classifier does not report a negatively impact to this classifier's score. With this process, new entities are produced for cross-document coreference.

3.3 Cross-document entity coreference

In 2013 we developed a tool for cross-document coreference named *Kripke*. Our motivation for a new tool were that we wanted an easy-to-run, efficient, and precision-focused clusterer; previously (*i.e.,* in 2012) we had used string-matching alone, or a Wikipedia-based entity linker. We produced runs that ran *Kripke* with standard settings, and also a variant with somewhat more aggressive clustering which we thought might improve (entity-clustering) recall.

Kripke is an unsupervised, procedural clusterer based on two principles: (a) to combine two clusters each must have good matching of both names and contextual features; (b) a small set of discriminating contextual features is generally sufficient for disambiguation. Additional details can be found in Section 4.

3.4 KB cleanup and slot value consolidation

This step, which is repeated several times in the pipeline ensures that all relations have their inverses in the KB, culls relations that violate type or value constraints, and reduces the number of values to match expectations for each type of slot.

Inverses. Producing inverses is an entirely deterministic process that simply generates Y *inverse* X in *Doc D* from an assertion of X *slot* Y in *Doc D*. For example, inverse relations like per:parent and per:children, or per:schools_attended and org:students. While straightforward, this is an important step, as relations are often extracted in only one direction during document-level analysis, yet we want both assertions to be explicitly present in our KB to aid with downstream reasoning.

Predicate Constraints. Some assertions extracted from SERIF or FACETS can be quickly vetted for plausibility. For example, the object of a predicate expecting a country (*e.g.,* per:countries_of_residence) must match a small, enumerable list of country names; Massachusetts is not a reasonable response.[7] Similarly, 250 is an unlikely value for a person's age. We have procedures to check certain slots to enforce that values must come from a accepted list of responses (*e.g.,* countries, religions), or cannot include responses from a list of known incorrect responses (*e.g.,* a girlfriend is not allowed as a slot fill for per:other_family).

Consolidating Slot Values. Extracting values for slots is a noisy process and errors are more likely for some slots than for others. The likelihood of finding incorrect values also depends the popularity of both the entity and slot. For example, in processing a collection of 26K articles from the Washington Post, we observed more than fifty entities who had 14 or

[7] And in 2014 neither is Texas.

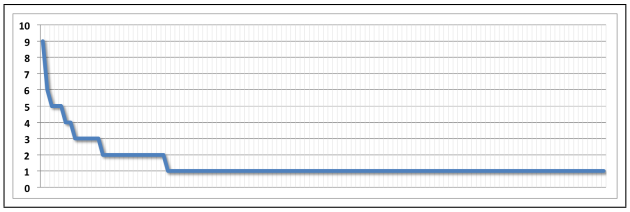

Figure 1: Kelvin initially extracted 121 distinct values for Barack Obama's employer from 26,000 Washington Post articles. The number of attesting documents for each followed a power law, with nine documents for the most popular value only one for the majority.

more employers. One entity was reported as having had 122 employers (per:employee_of)!

Slot value consolidation involves selecting the best value in the case of a single valued slot (e.g., per:city_of_birth) and the best set of values for slots that can have more than one value (e.g, per:parents). In both cases, we use the number of attesting documents to rank candidate values, with greater weight given to values that were explicitly attested rather than implicitly attested via inference rules. See Figure 1 for the number of attesting documents for each of the values for the entity that have 122 distinct values for employer.

For slots that admit only a single value, we select the highest ranked candidate. However, for list-valued slots, it is difficult to know how many, and which values to allow for an entity. We made the pragmatic choice to limit list-values responses in a predicate-sensitive fashion, preferring frequently attested values. We associate two thresholds for selected list-valued predicates on the number of values that are reasonable – the first represents a number that is suspiciously large and the second is an absolute limit on the number of values reported. Table 1 shows the thresholds we used for some predicates. For predicates in our table, we accepted the nth value on the candidate list if n did not exceed the first threshold and rejected it if n exceeded the second. For n between the thresholds, a value is accepted only if it has more than one attesting document.

relation	many	maximum
per:children	8	10
per:countries_of_residence	5	7
per:employee_of	8	10
per:member_of	10	12
per:parents	5	5
per:religion	2	3
per:schools_attended	4	7
per:siblings	9	12
per:spouse	3	8

Table 1: The number of values for some multi-valued slots were limited by a heuristic process that involved the number of attesting documents for each value and two thresholds.

3.5 Inference

We apply a number of forward chaining inference rules to increase the number of assertions in our KB. To facilitate inference of assertions in the Cold Start schema, we introduce some unofficial slots into our KB, which are subsequently removed prior to submission. For example, we add slots for the sex of a person, and geographical subsumption (*e.g.,* Gaithersburg is part-of Maryland). The most prolific inferred relations were based on rules for family relationships, corporate management, and geopolitical containment.

Many of the rules are logically sound and follow directly from the meaning of the relations. For example, two people are siblings if they have a parent in common and two people have an "other_family" relation if they one is a grandparent of the other. Our knowledge of geographic subsumption produced a large number of additional relations, e.g., knowing that a person's *city_of_birth* is Gaithersburg and that it is part of Maryland and that Maryland is a state supports the inference that the person's *stateorprovince_of_birth* is Maryland.

For TAC 2014, we only used inference rules we considered to be *sound* and did not use heuristic rules. For example, in 2013 we we inferred that if a person attended a school S, and S has headquarters in location L, then the person has been a resident of L. In general, we do not add an inferred fact that is already in our knowledge base. Some of the heuristic rules are default rules in that they only add a value for a slot for which we have no values. For example, we know that person P1 is the spouse of person P2 and that the sex of P1 is male and we have no value for the sex of P2, we infer that P2 is female. In this case, the rule is both a default rule and one whose conclusion is very often, but not always, true.

The 2013 TAC-KBP guidelines stipulated that relations must be attested in a single document, which constrained severely the number of inferred assertions allowed, and required removing relations not evidenced entirely in a single document. For example, consider learning that Lisa is Homer's child in one document and that Bart is Homer's child in another. Assuming that the two Homer mentions corefer, it follows that Lisa and Bart are siblings. This year the 2014 guidelines do allow such inferences, but required systems to keep a set of provenance descriptions (document and offsets) for each inferred relation. Table 6 shows the distribution of the additional relations that we inferred using only sound rules from the facts extracted in the evaluation corpus.

We ran the inference step over the entire knowledge base which had been loaded into memory, since in general, a rule might have any number of antecedent relations. However, we realized that many of our inference rules do not require arbitrary joins and could be run in parallel on subsets of the knowledge base if we ensure that all facts about any entity are in the same subset. The fraction of rules for which this is true can be increased by refactoring them. For example, the rule for *per:sibling* might normally be written as

X per:parent P $\wedge$ Y per:parent P $\rightarrow$ X per:siblings Y

but can also be expressed as

P per:child X $\wedge$ P per:child Y $\rightarrow$ X per:siblings Y

assuming that we materialize inverse relations in the knowledge base (e.g, asserting a child relation for every parent relation and vice versa). A preliminary analysis of our inference rules shows that all could be run in at most three parallelizable inference steps using a Map/Reduce pattern.

3.6 Selecting provenance metadata

This step selects the provenance strings to support each relation for the final submission. The 2014 evaluation rules allow for up to four provenance strings to support a relation, none of which can exceed 150 characters. For simple attested values, our initial provenance strings are spans selected from the sentence from which we extracted the relation, e.g., *"Homer is 37 years old"* for a per:age relation. Inferred relations can have more than one provenance string which can come from the different documents, e.g., *"His daughter Lisa attends Springfield Elementary"* and *"Maggie's father is Homer Simpson"* for a per:siblings relation.

An initial step is to minimize the length of any overly-long provenance strings is to select a substring that spans both the subject and object. Candidate provenance strings whose length exceeds the maximum allowed after minimization are discarded[8]. If there are multiple provenance candidates, a simple greedy bin packing algorithm is used to include as many as possible into the four slots available. Preference is given for attested values over inferred values and provenance sources with higher certainty over a those with lower.

3.7 Post-processing

The final steps in our pipeline ensure compliance with the task guidelines. We normalize temporal expressions, ensure that all entities have mentions, insist that relations are consistent with the types of their subjects and objects, confirm that logical inverses are asserted, and check that entities have mentions in the provenance documents. so forth.

4 Kripke

The *Kripke* system[9] takes a set of document-level entities and performs agglomerative clustering on them to produce cross-document entity clusters. The tool is written in approximately 2000 lines of Java source code. The intent is for the system to have a precision bias, which we feel is appropriate for knowledge base population.

The principles on which *Kripke* operations are:

- Coreferential clusters should match well in their names.
- Coreferential clusters should share contextual features.
- Only a few, discriminating contextual features should be required to disambiguate entities.

[8]This could result in a relation being discarded if it has no legal provenance strings after minimization

[9]Named after Princeton philosopher Saul Kripke, who wrote a book on naming entities in the 1970s.

Size	*PER*	*ORG*	*GPE*
1	75660	44481	16506
2	15485	7750	3036
3	6319	3009	1222
4	3384	1603	673
5	2137	969	424
6	1570	690	314
7	1086	517	223
8	853	351	214
9	692	272	140
10-20	2839	1367	650
21-30	814	463	256
31-40	344	240	164
41-50	186	156	76
51-60	101	104	71
61-70	76	83	48
71-80	46	77	38
81-90	42	59	30
91-100	13	43	25
101-200	65	153	136
201-300	9	48	83
301-400	3	15	41
401-500	2	3	24
501-600	3	4	17
601-700	0	4	14
701-800	2	4	10
801-900	0	2	6
901-1000	0	1	3
1001-2000	0	4	26
>2000	1	2	6

Table 2: A histogram of cluster sizes for each entity type for run hltcoe3 shows that Kripke is effective at producing large clusters but that most entity clusters are small.

To avoid the customary quadratic-time complexity required for brute-force pairwise comparisons, *Kripke* maintains an inverted index of names used for each entity. Only entities matching by full name, or some shared words or character n-grams are considered as potentially coreferential.[10] Related indexing techniques are variously known as blocking (Whang et al., 2009) or canopies (McCallum et al., 2000).

At present, contextual matching is accomplished solely by comparing named entities that co-occur in the same document. Between candidate clusters, the sets of all names occurring in any document forming each cluster are intersected. Each name is weighted by normalized Inverse Document Frequency, so that rare, or discriminating names have a weight closer to 1. The top-k (*i.e.,* k=10) weighted names were used, and if the sum of those weights exceeds a cutoff, then the contextual similarity is deemed adequate. Such a technique should be able to tease apart George Bush (41st president) and his son (43rd president) through co-occurring names (*e.g.,* Al Gore, Barbara Bush, Kennebunkport, James Baker versus the entities Dick Cheney, Laura Bush, Crawford, Condolezza Rice).

The system runs by executing a cascade of clustering passes, where in each subsequent pass conditions are relaxed in the requirements for good name and contextual matching. The hope is that higher precision matches are made in earlier phases of the cascade, and these will facilitate more difficult matches later on.

An example of Kripke's performance on the 2014 evaluation corpus is given in Table 2, which show results from the hltcoe3 run. The number of clusters by size shows a power law distribution, with larger cluster being more common for entities of type GPE and ORG. The largest clusters for each type were the PER *Barack Obama* with 2641 document entities, the ORG *Associated Press* with 2341 and the GPE *United States* with 6719. Associated with each of these were many smaller entity clusters with one or two entities that had not been merged.

5 Development Tools

Evaluation is an essential step in the process of developing and debugging a knowledge base population system. We briefly describe several of the evaluation tools used by the KELVIN system. Two were aimed at comparing the system's output from two different versions: *entity-match,* which focuses on differences in entities found and linked; and *kbdiff*, which identifies differences in relations among those entities. Together, these tools support assessment of relative KB accuracy by sampling the parts of two KBs that disagree (Lawrie et al., 2013). *Tac2Rdf* produces an RDF representation of a TAC KB supported by an OWL ontology and loads it into a standard triple store, making it available for browsing, inference and querying using standard RDF tools. *KB_Annotator* allows developers to browse the system output and annotate entities and relations as either supported or not by the document text provided as provenance.

[10] Support for orthographically dissimilar name variants (*i.e.,* aliases) was planned, but not implemented in time for this year.

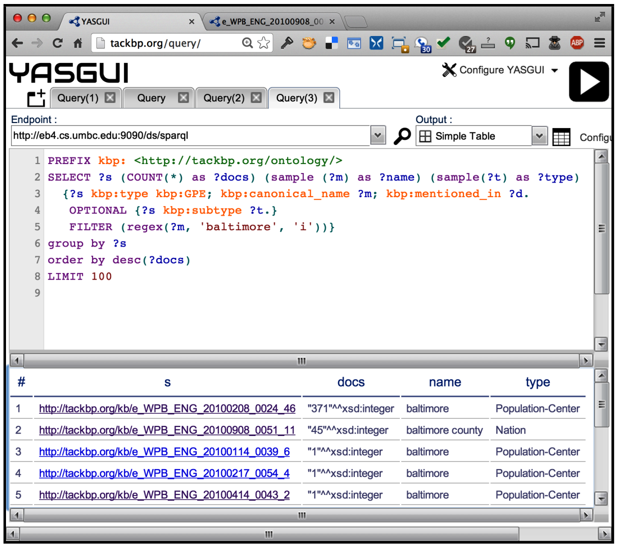

Figure 2: The RDF KB encoding can be queried via SPARQL (here using the Yasgui interface) to find anomalies or collect data for analysis or training.

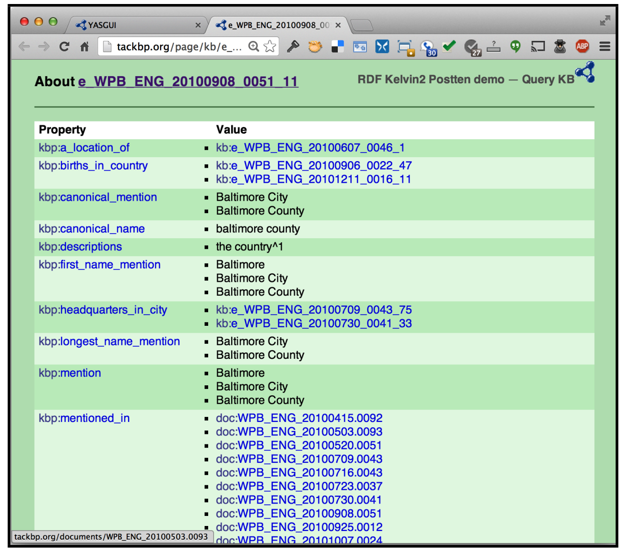

Figure 3: Pubby provides a simple way to browse the RDF version of the extracted knowledge as a graph via a Web browser.

Entity-match defines a KB entity as the set of its mentions. From the perspective of an entity in one KB, its mentions might be found within a single entity in the other KB, spread among multiple entities, or missing altogether from the other KB. In the first case there is agreement on what makes up the entity. In the second case, there is evidence either that multiple entities have been conflated in the first KB, or that a single entity has been incorrectly split in the second. In the third case, the entity has gone undetected. The tool reports shows the entities and cases into which they fall. If there is disagreement between the KBs, it reports each corresponding entity in the second KB and the number of mentions that map to it.

Kbdiff identifies assertions in one KB that do not appear in the other. The challenge here is to identify which entities are held in common between the two KBs. Provenance is again useful; relations from different KBs are aligned if they have the same predicates and the provenance of their subjects and objects match. The algorithm works by first reading all the assertions in both KBs and matching them based on provenance and type. The output includes assertions in the first KB lacking a match in the second (prefixed by $<$) and those in the second but not the first (prefixed by $>$.)

Tac2Rdf translates a KB in TAC format to RDF using an OWL ontology (available at http://tackbp.org/static/tackbp.ttl) that encodes knowledge about the concepts and relations, both explicit and implicit. For example, the Cold Start domain has an explicit type for geo-political entities (GPEs), but implicitly introduces disjoint GPE subtypes for cities, states or provinces, and countries through predicates like *city_of_birth*. Applying an OWL reasoner to this form of the KB detects various logical problems, e.g., an entity is being used as both a city and a country. The RDF KB results are also loaded into a triple store, permitting access by an integrated set of standard RDF tools including Fuseki for SPARQL (Prud'Hommeaux and Seaborne, 2008) querying, Pubby for browsing, and the Yasgui SPARQL GUI.

Figure 2, for example, shows the results of an ad hoc SPARQL query for GPEs with the string "baltimore" in their canonical mention along with the number of documents in which they were mentioned and their subtype. Such queries are useful in identifying possible cross-document coreference mistakes (e.g., GPEs with mentions matching both *"X County" X*) and likely extraction errors (e.g., pairs of people connected by more than one relation in the set $\{$*spouse*, *parent* and *other-family*$\}$).

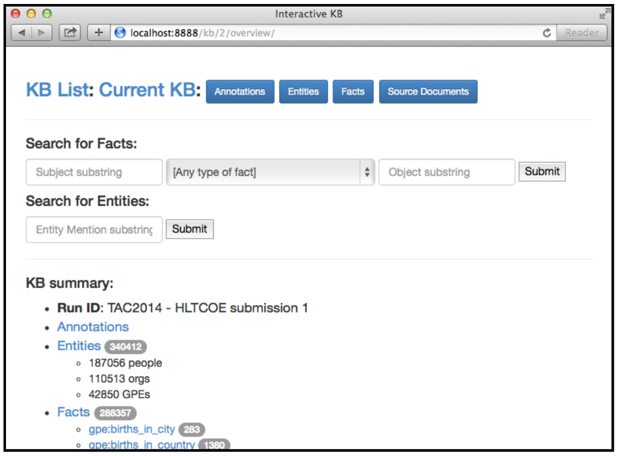

Figure 4: Overview of a KB including counts on entity type and relations.

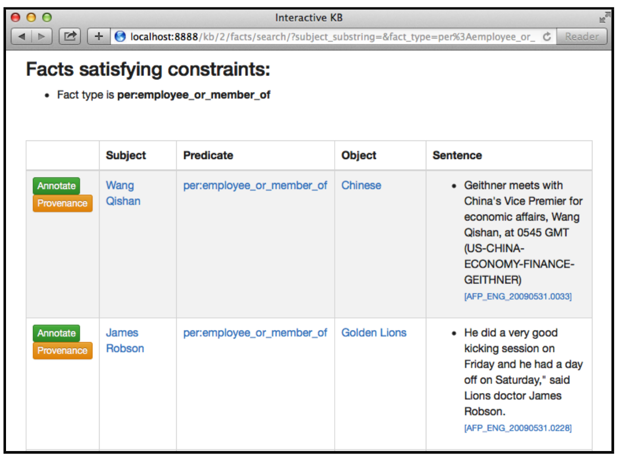

Figure 5: Yhe results of a relation search provides access to relations and their associated entities.

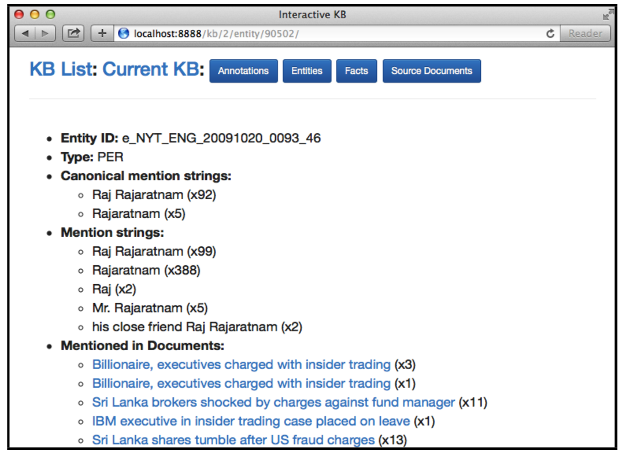

Figure 6: The entity view shows the docments, mentions and facts associated with an entity.

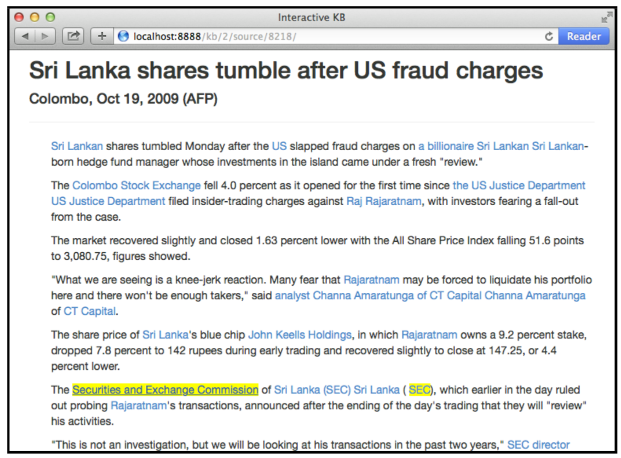

Figure 7: A document view where hovering over an entity mention reveals strings that are co-referent.

Clicking on the second entity in the table of results opens the entity in the Pubby linked data browser, as shown in Figure 3.

The *Interactive KB Browser* system loads triples from a TAC submission into a KB and provides a Web-based interface allowing one to view a document's text, see the entities, entity mentions and relations found in it, and give feedback on their correctness. Figure 4 details an overview of the KB, by listing entities by their type with numeric counts overall and for each type. It also allows searching with subjects and objects found using string matches. Either all relations can be included or a particular relation. Search results are presented in a table as in shown in Figure 5. The light blue text represents links to entities and relationships. In this example, the object of the relation is a string rather than an entity, so the object is not a link. While a relation is viewed as a list of facts, an entity consists of a type, canonical mentions, mention strings, documents in which the entity is mentioned, and facts pertaining to the entity. Figure 6 show an example entity and 7 shows document text with entity mentions identified in a contrasting color as well as the extracted facts. Mousing over a mention highlights it and any co-referential mentions.

Whenever a fact is encountered, the developer or assessor has the option of providing an annotation or judgment. As shown in Figure 8, the judgment is solicited by stating the relationship as a fact, show-

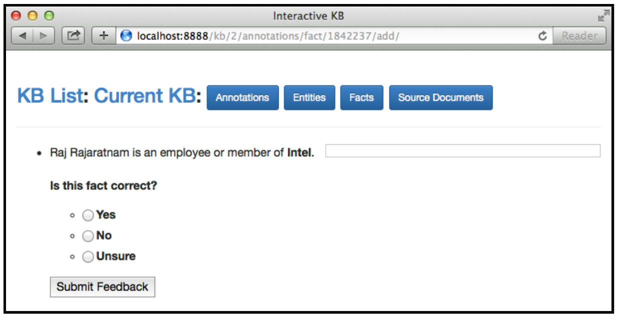

Figure 8: For each fact extracted from a document, a developer or assessor can express a judgment on its correctness.

Name	Inference	Clustering	InDoc	NewsBias
hltcoe1		Standard		
hltcoe2	Yes	Standard		
hltcoe3		Aggressive		
hltcoe4		Standard	Yes	
hltcoe5		Standard		Yes

Table 3: Experimental variables for submitted runs.

ing the sentence with relevance parts of the text in contrasting colors, and a simple "yes", "no", "unsure" feedback on the fact. This allows an assessor to annotate errors in named entity resolution, in-document coreference, and relation detection with respect to particular facts. An annotator may alway annotate all the facts concerning a particular entity, wich can then be used for error analysis and as training data.

6 Submissions and results

We submitted the maximum of five experimental conditions that started with a simplistic baseline pipeline, and which added individually: inference rules, more aggressive entity coreference, within-document mention-chain purification, and a filter that required relations to be attested in newswire sources. KELVIN does not access the Internet during processing. Table 3 summarizes the various conditions and Tables 4 and 5 give the key performance metrics. The number of times each slot was asserted for run hltcoe3 is given in Table 6.

Table 7 lists the number of entities of each type which are included in each of our runs. Note that as entities having no asserted relations cannot improve scores in the ColdStart task, we did not include such "mention only" entities in our submis-

Slot	Read	Inferred	Queried
per:employee_or_member_of	56107	0	yes
org:employees_or_members	43660	0	yes
org:alternate_names	38368	0	yes
per:title	28378	0	yes
per:countries_of_residence	12641	1500	yes
gpe:residents_of_country	12641	1500	yes
gpe:employees_or_members	12447	0	yes
per:origin	8714	0	yes
org:parents	5888	0	yes
org:city_of_headquarters	5338	0	yes
gpe:headquarters_in_city	5338	0	yes
per:cities_of_residence	4826	0	yes
gpe:residents_of_city	4826	0	yes
gpe:headquarters_in_country	4756	1749	yes
org:country_of_headquarters	4756	1729	yes
org:subsidiaries	3494	0	yes
gpe:subsidiaries	2394	0	no
per:spouse	2345	0	yes
per:date_of_death	1933	0	yes
per:age	1916	0	yes
gpe:residents_of_stateorprovince	1676	1994	yes
per:statesorprovinces_of_residence	1676	1994	yes
per:country_of_birth	1366	78	yes
gpe:births_in_country	1366	78	yes
per:alternate_names	1251	0	yes
per:children	1194	0	yes
per:parents	1194	0	yes
gpe:headquarters_in_stateorprovince	1161	2449	yes
org:stateorprovince_of_headquarters	1161	2449	yes
org:founded_by	1080	0	yes
per:schools_attended	1018	0	yes
org:students	1018	0	yes
per:organizations_founded	980	0	yes
per:siblings	838	426	yes
per:top_member_employee_of	715	7911	yes
org:top_members_employees	715	7911	yes
per:date_of_birth	497	0	yes
org:date_founded	452	0	yes
per:charges	429	0	yes
gpe:deaths_in_city	353	0	yes
per:city_of_death	353	355	yes
per:other_family	345	355	yes
org:members	302	0	yes
per:city_of_birth	280	0	yes
gpe:births_in_city	280	0	yes
org:member_of	195	0	no
per:religion	178	0	yes
gpe:member_of	107	0	yes
gpe:deaths_in_country	102	85	no
per:country_of_death	102	85	no
org:organizations_founded	87	0	no
org:date_dissolved	67	0	yes
org:shareholders	47	0	no
gpe:deaths_in_stateorprovince	46	182	yes
per:stateorprovince_of_death	46	182	yes
per:holds_shares_in	43	0	yes
gpe:births_in_stateorprovince	39	139	yes
per:stateorprovince_of_birth	39	139	yes
gpe:organizations_founded	13	0	no
org:holds_shares_in	4	0	yes
org:website	0	0	yes
org:number_of_employees_members	0	0	yes
per:employee_of	0	1209	no
per:cause_of_death	0	0	yes

Table 6: This table shows the number of assertions for each slot that were read or inferred for run hltcoe3 and whether or not the slot was used in any evaluation queries. Slots not listed were neither asserted nor queried.

	0-hop				1-hop				All-hop			
Run	GT	R	W	D	GT	R	W	D	GT	R	W	D
1	1073	136	133	46	1364	64	57	11	2437	200	190	57
2	1073	121	113	37	1364	66	71	10	2437	187	184	47
3	1073	156	112	33	1364	71	61	7	2437	227	173	40
4	1073	84	82	29	1364	40	30	7	2437	124	112	36
5	1073	107	113	40	1364	53	43	7	2437	160	156	47

Table 4: Ground-truth, right, wrong and duplicate answers for our submitted 2014 runs.

	0-hop			1-hop			All-hop		
Run	P	R	F1	P	R	F1	P	R	F1
1	0.5056	0.1267	0.2027	0.5289	0.0469	0.0862	0.5128	0.0821	0.1415
2	0.5171	0.1128	0.1852	0.4818	0.0484	0.0879	0.5040	0.0767	0.1332
3	**0.5821**	**0.1454**	**0.2327**	0.5379	**0.0521**	**0.0949**	**0.5675**	**0.0931**	**0.1600**
4	0.5060	0.0783	0.1356	**0.5714**	0.0293	0.0558	0.5254	0.0509	0.0928
5	0.4864	0.0997	0.1655	0.5521	0.0389	0.0726	0.5063	0.0657	0.1162

Table 5: Micro precision, recall and F_1 scores for our submitted 2014 runs.

run	entities	PER	ORG	GPE	facts
1	340412	187056	110506	42850	288358
2	340420	187109	110445	42866	311019
3	306499	171951	98234	36314	283582
4	372509	203779	120045	48685	236550
5	340593	187134	110569	42890	244283

Table 7: Number of entities mentions and facts identified in the evaluation corpus for each run.

sions. The number of reported entities is generally similar in each run, with differences likely attributable to changes in cross-document entity coreference.

6.1 Discussion

Comparing our various experimental conditions, we make the following observations.

It appears that more aggressive cross-document coreference does improve recall, as was hoped for; 0-hop recall rises from 0.127 in hltcoe1 to 0.145 in hltcoe3. Precision also improves.

Use of inference rules (contrast hltcoe2 to hltcoe1) does not appear to improve performance. To date we have not been able to conduct an analysis of the reasons for this.

Requiring posited relations to be supported from news documents (*i.e.,* hltcoe5 vs. hltcoe1 baseline) seems to have lowered recall, and therefore F_1.

7 Conclusion

The JHU Human Language Technology Center of Excellence has participated in the TAC Knowledge Base Population exercise since its inception in 2009 and in Cold Start task since 2012. We modified the KELVIN system used in the 2012 and 2013 Cold Start task by enhancing inference, cross-document entity coreference, and application of inference rules.

References

E. Boschee, R. Weischedel, and A. Zamanian. 2005. Automatic information extraction. In *Proceedings of the 2005 International Conference on Intelligence Analysis, McLean, VA*, pages 2–4.

Dawn Lawrie, Tim Finin, James Mayfield, and Paul McNamee. 2013. Comparing and Evaluating Semantic Data Automatically Extracted from Text. In *AAAI 2013 Fall Symposium on Semantics for Big Data*. AAAI Press, November.

James Mayfield, Paul McNamee, Craig Harmon, Tim Finin, and Dawn Lawrie. 2014. KELVIN: Extracting Knowledge from Large Text Collections. In *AAAI Fall Symposium on Natural Language Access to Big Data*. AAAI Press, November.

Andrew McCallum, Kamal Nigam, and Lyle Ungar. 2000. Efficient clustering of high-dimensional data sets with application to reference matching. In *Knowledge Discovery and Data Mining (KDD)*.

Paul McNamee, Veselin Stoyanov, James Mayfield, Tim Finin, Tim Oates, Tan Xu, Douglas W. Oard, and Dawn Lawrie. 2012. HLTCOE participation at TAC 2012: Entity linking and cold start knowledge base construction. In *Text Analysis Conference (TAC)*, Gaithersburg, Maryland, November.

Paul McNamee, James Mayfield, Tim Finin, Tim Oates, Baltimore County, Dawn Lawrie, Tan Xu, and Douglas W Oard. 2013. KELVIN: a tool for automated knowledge base construction. In *Human Language Technologies: The 2013 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, volume 10, page 32.

E Prud'Hommeaux and A. Seaborne. 2008. SPARQL query language for RDF. Technical report, World Wide Web Consortium, January.

Steven Euijong Whang, David Menestrina, Georgia Koutrika, Martin Theobald, and Hector Garcia-Molina. 2009. Entity resolution with iterative blocking. In *SIGMOD 2009*, pages 219–232. ACM.