HLTCOE Participation in TAC KBP 2017: Cold Start TEDL and Low-resource EDL

Tim Finin
Computer Science and Electrical Engineering
University of Maryland, Baltimore County
Baltimore, MD

Dawn Lawrie
Computer Science
Loyola University of Maryland
Baltimore, MD

James Mayfield and Paul McNamee
Human Language Technology Center of Excellence
Johns Hopkins University
Baltimore, MD

Cash Costello
Applied Physics Laboratory
Johns Hopkins University
Laurel, MD

Abstract

The JHU HLTCOE participated in the Cold Start and the Entity Discovery and Linking tasks of the 2017 Text Analysis Conference Knowledge Base Population evaluation. For our sixth year of participation in Cold Start we continued our research with the KELVIN system. We submitted experimental variants that explore use of linking to Freebase across English and Chinese languages and add relations beyond those required by Cold Start. This is our third year of participation in Tri-lingual EDL and first year for the EDL Pilot for 10 Low-resource Languages. We used KELVIN in the Cold Start and Tri-lingual tasks and a new custom system for the Low-resource EDL task.

1 Introduction

The JHU Human Language Technology Center of Excellence (HLTCOE) has participated in the TAC Knowledge Base Population exercise since its inception in 2009. Our focus over the past year was on developing our KELVIN system (McNamee et al., 2012; McNamee et al., 2013; Mayfield et al., 2014; Finin et al., 2014; Finin et al., 2015; Finin et al., 2016) as a core technology for multiple TAC tasks. We used KELVIN in the Cold Start and Tri-lingual tasks and a new custom system for the Low-resource EDL task.

This is the sixth year that we used KELVIN in the Cold Start task. This year we enhanced our system to accomodate the many small changes required in the output and added a new OpenIE component to discover relations. We empoyed a new custom system for the low-resource language EDL task.

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, BBN SERIF (Ramshaw et al., 2011). SERIF provides a substrate that includes entity recognition, relation extraction, and within-document coreference resolution. In addition to SERIF, significant components that we rely on include: the Stanford CoreNLP OpenIE system (Manning et al., 2014) for extracting potential relations; a maximum entropy trained model for extracting personal attributes (FACETS, also a BBN tool); a cross-document entity coreference resolver (the HLTCOE *Kripke* system); and a procedurally implemented rule system.

KELVIN is organized as a pipeline with three stages: (i) document level processing done in parallel on small batches of documents; (ii) cross-document coreference resolution to produce an initial KB; and (iii) knowledge-base enhancement and refinement through inference and relation analysis. An optional fourth stage loads the knowledge base

into an iPad app to collect human annotations on the document set. The next section describes the major steps in these stages.

3 Cold Start System Description

KELVIN runs from a set of Unix shell scripts 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. Document-level processing
2. Extended Document-level processing
3. Cross-document entity coreference resolution
4. KB cleanup and slot value consolidation
5. Linking entity mention chains to an external background KB
6. Applying inference rules to posit additional assertions
7. KB-level entity clustering
8. KB cleanup and slot value consolidation
9. Selection of the best provenance metadata
10. 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; many of these steps are executed as a single process.

3.1 Document-Level Processing

BBN's SERIF tool[1] (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;[2] they include:

- identifying named entities and classifying them by type and subtype;
- performing intra-document coreference resolution, on named, nominal, and pronominal mention;
- 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.[3] 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 and the relations and events in which it participates. Entities comprising solely nominal mentions were included in 2017 for both Cold Start and EDL, per the task guidelines. Finally, the output from each document is entered into a Concrete object, (Ferraro et al., 2014), which is our standard representation for information extracted from a document.

FACETS 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. 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 a physician, or (academic) affiliation for someone associated with an educational institution.

3.2 Extended Document-Level Processing

Five additional steps are taken once SERIF and FACETS are run. These steps generally address shortcomings in the tools or add additional information that was not found by the primary tools.

The first two steps focus on augmenting relations. In Step 1 relations are identified using pattern matching, which relies on entity type as well as string matches. In Step 2, new relations are found using an open information extraction system. Here facts are aligned to TAC relations using a rule based approach. Step 1 is described in greater detail in a prior system description (Finin et al., 2015). Step 2

[1] Statistical Entity & Relation Information Finding

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

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

was added to the system this year and is described in greater detail in Section 3.2.1.

The third step focuses on refining canonical mentions. This approach uses Freebase to assist in the selection of descriptive names that do not contain ancillary information.

In the fourth step dates identified by SERIF are modified to bring them into compliance with TAC guidelines. Problem with dates include the format when parts of the date are missing (*e.g.* "1948" rather than "1948-XX-XX") and the existence of relative dates rather than absolute dates.

Finally in the fifth step, entities from the headline, dateline, and author fields are extracted. In previous years the lack of identified entities from the fields led to a substantial number of misses in queries where annotators favored these mentions. This year the process of matching named entities was optionally extended to the rest of the document. This means that if exact matches of named entities already identified in the document are found, these new mentions will be added to the mention chain. This is referred to as *Exact Match* when describing the experimental runs.

3.2.1 Relation Extraction using Open IE

Open information extraction is an alternative method of information extraction. An open information extraction system extracts a greater diversity of relations that may or may not align to TAC relations or to entities that have previously been identified. Given that multiple IE approaches are used, the output of the systems must be aligned. In addition to this challenge, the more free-form relations coming from Open IE must be aligned to TAC relations.

In more detail the Stanford CoreNLP system (Manning et al., 2014) was run over each document in the collection. First, the OpenIE output was processed to associate the subjects and objects in the relations to entity mentions identified by CoreNLP. Any extraction whose subject and object aligned to a CoreNLP entity was considered a candidate relation. With the candidates in hand, an alignment to the SERIF mentions is necessary. If a CoreNLP mention did not align a SERIF mention; than a new entity was created in the TAC file. Second, the relations were considered in turn. If the rules associated the relation with a TAC relation, it was accepted as a TAC relation.

The rule based approach was developed by examining common relations found in the TAC 2016 data. A total of 141 rules over twenty-one TAC relations were developed. In order to match a rule, the open ie relation had to be an exact match for a rule. Some open IE extractions are very generic, such as "is." Under certain circumstances some generic extractions can be mapped with good precision to TAC relations. Such rules further constrain the entity to string in the relation, such as identifying a sub-type or specifying the string is a number. For instance the open IE relation "is from" is mapped to per:statesorprovinces_of_residence when the GPE has a subtype State-or-Province.

The rules identified almost 24,000 more relations in the text. The three most common new relations identified were org:top_members_employee with 5944 new instances, per:employee_of with 3290 new instances, and per:age with 3021 new instances.

3.3 Cross-Document Coreference Resolution

In 2013 we developed a tool for cross-document coreference named *Kripke* that takes as input a serialized TAC knowledge base and produces equivalence classes that encode entity coreference relations. *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; and (b) a small set of discriminating contextual features is generally sufficient for disambiguation. 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. While *Kripke*'s approach allows it to work well on many languages, the n-gram length is a language-dependent parameter and we use a smaller value of n for Chinese mentions.

Contextual matching is based exclusively on 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 dis-

criminating names have a weight closer to 1. The top-k (*e.g.,* k=10) weighted names were used, and if the sum of those weights exceeds a cutoff, then the contextual similarity is deemed adequate. This technique can distinguish George Bush the 41st U.S. president from his son, the 43rd U.S. president, through co-occurring names (*e.g.,* Al Gore, Barbara Bush, Kennebunkport, James Baker versus Dick Cheney, Laura Bush, Crawford, Condolezza Rice). The system runs by executing a cascade of clustering passes, where in each subsequent pass the requirements for sufficient name and contextual matching are relaxed. The higher precision matches made in earlier cascade phases facilitate more difficult matches in subsequent phases. Additional details can be found elsewhere (Finin et al., 2014; Finin et al., 2015).

3.4 KB Cleanup and Slot Value Consolidation

This step, which is repeated several times throughout the pipeline, ensures that the inverse of each relation is asserted in the KB, culls relations that violate type or value constraints, and reduces the number of values to match common sense expectations for each type of slot.

3.4.1 Inverses Relations

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.

3.4.2 Predicate Constraints

Some extracted assertions 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. Similarly, 250 is an unlikely value for a person's age. We have procedures to check certain slots to enforce that values are drawn from an 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).

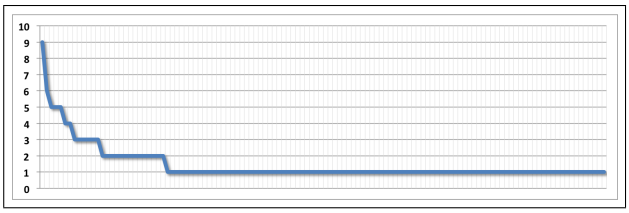

Figure 1: KELVIN initially extracted 121 distinct values for Barack Obama's employer from 26,000 Washington Post articles. The number of attesting documents for each followed a power law, with nine documents for the most popular value only one for the majority.

3.4.3 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 on the popularity of both the entity and slot in the document collection. For example, in processing a collection of 26K articles from the Washington Post, we observed more than fifty entities who had 14 or 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 had 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. Ta-

relation	many	maximum
per:children	8	10
per:countries_of_residence	5	7
per:employee_or_member_of	18	22
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.

ble 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.

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 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 geo-political 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 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.

3.6 Linking to External Knowledge Bases

Entities are linked to one more external knowledge bases. Our current system uses just one external KB, the version of the Freebase KB described in Section 4. Our approach is relatively simple, only comparing an entity's type and mentions to the external KB's entity types, names and aliases.

In linking a collection entity to a KB entity, we start by producing a candidate set by selecting all KB entities whose names or aliases match any of the collection entity's canonical mentions.[4] The candidates are ranked by counting how often each matching mention was used and by the KB entity's *significance score* (see Section 4). We used experimentally derived thresholds to reject all candidates if there were too many or the top score was too low relative to the second highest score.

3.7 Knowledge-Level Clustering

After analyzing our previous Cold Start performance, we observed that KELVIN often undermerged entities. We added additional inference rules for merging entities that were applied at the knowledge-base level. One set of rules merges entities that are linked to the same Freebase entity. Another set merges entities that share the same canonical mention under several entity type specific conditions. For example, two ORG entities with subtype *Educational* are merged if they have the same canonical mention and the mention includes a token implying they are organizations of higher education (*e.g.,* college, university or institute).

A third set merges entities based on "discriminating relations." Our intuition is that it is likely that two people with similar names who have the same spouse or were born on the same date and in the same city should be merged. Similarly, organizations with similar names who share a top-level employee are good candidates for merging.

We maintain three categories of relations, those with high, medium and low discriminating power. Example of highly discriminating relations are per:children, org:date_founded, and gpe:part_of. Medium discriminating relations include per:city_of_birth, gpe:headquarters_in_city,

[4]Matching is done after normalizing strings by downcasing and removing punctuation.

entity	type	significance	inlinks	outlinks
United States	GPE	19.2	452006	162081
India	GPE	15.8	34273	23281
Harvard University	ORG	14.4	11163	11348
UMBC	ORG	7.4	172	192
Barack Obama	PER	11.4	744	1948
Alan Turing	PER	7.6	35	163
Ralph Sinatra	PER	2.8	0	7
Harvard Bridge	FAC	5.1	3	32
Mississippi River	LOC	8.9	242	245

Table 2: This table shows examples of entities, their estimated significance, and their number of incoming and outgoing links.

and org:member_of. Examples of relations with low discriminating power include per:stateorprovince_of_birth, org:students, and gpe:deaths_in_city. The decision to merge two entities with similar names is dependent on their type and the number of high, medium, and low discriminating relations they share.

3.8 Selecting Provenance Metadata

This step selects the provenance strings that will be used to support each relation for the final submission. The Cold Start 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 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.[5] 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.

3.9 Cross-language Entity Linking

The overall processing has three stages: monolingual document processing, multilingual cross-document coreference resolution, and multilingual knowledge-base processing.

The first stage applies KELVIN's standard pipeline to each of the monolingual document collections using the appropriate SERIF language model.[6] For each language, we use just two outputs of the monolingual system: the serialized TAC KB produced by KELVIN's document level processing and the coreference relations produced by *Kripke*.

The second stage starts by creating a multilingual document level KB by concatenating the three monolingual KBs. If the mention-translation option is enabled, English translations of Chinese and Spanish mentions are added. We used the Bing translation service API. This combined, multilingual collection is then processed by *Kripke* to produce cross-document coreference relations.

The coreference relations from each of the monolingual collections and from the combined collection are integrated using a simple algorithm to combine equivalence relations, yielding a single coreference clustering file for the entire collection. The monolingual document-level KBs are then combined (without any translated mentions) and the cross-document coreference relations used to gen-

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

[6]Our version of SERIF has models for English, Chinese, Spanish and Arabic

erate the KB for subsequent KB-level processing by the rest of KELVIN's pipeline. Combining the results of using *Kripke* to cluster the mono-lingual runs and separately their combination with mentions translated into English achieves a greater reduction in the number of entities.

3.10 Post-Processing

The final steps in our pipeline produce several outputs, including a submission file that complies with Cold Start task guidelines and an RDF version that is can be loaded into a triple store for inspection and querying.

We start by normalizing temporal expressions, ensuring that all entities have mentions, insisting that relations are consistent with the types of their subjects and objects, confirming that logical inverses are asserted, and checking that entities have mentions in the provenance documents.

We then translate the KB from TAC format to RDF using an OWL ontology 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 queries (Prud'Hommeaux and Seaborne, 2008), Pubby for browsing, and the Yasgui SPARQL GUI (Rietveld and Hoekstra, 2013).

4 External Knowledge Base

We created an external knowledge base derived from the BaseKB version of Freebase that was distributed by the LDC for use in the 2015 TAC KBP EDL tasks.[7] This external KB supported both our Cold Start and EDL submissions.

The full BaseKB dataset is quite large, containing more than a billion facts (counting each triple as a fact) about more than 40 million subjects. Much of this information is not relevant to the KBP tasks, such as information about musical groups, films or fictional characters.

We started by identifying entities that might be relevant to the TAC KBP tasks and removing any triples whose subjects were not in this set. An initial step was to identify those subjects that mapped to one of the five standard TAC types (PER, ORG, GPE, LOC and FAC) or represented what Freebase calls a Compound Value Type (CVT). The TAC ontology assumes that its five types are disjoint, but relevant Freebase entities can have types that map to several TAC types. For example, the Freebase entity with canonical name *Oval Office* (m.01hhz7) has subtypes associated with both a LOC and an ORG. We used various heuristics to assign such entities to only one TAC type.

We kept information about any CVTs that were linked to a TAC-relevant entity. CVTs are used in Freebase to represent reified relations, such as relations with associated units (for measurements), time or location.

Triples with literal values (i.e., strings) for objects are tagged with an XSD data type (e.g., integer or date) or a language tag (e.g., @EN for English or @ZH for Chinese). We discarded any string values whose language tag was not in the English, Chinese, or Spanish families.

We computed a measure of an entity's *significance* based on the number on triples in which it was the subject or object. The significance was set as the base-2 log of the total number of links, which produced values between 1.0 and 20.0 for the reduced KB Table 2 shows data for a few example entities.

Finally, we added additional assertions to record an entity's TAC type and normalized versions of an entity's names and aliases by downcasing, removing punctuation, entity significance, number of in- and out-links, etc. The reduced KB has 146M triples over more than 4.5M TAC entities: 3074k PERs, 686k ORGs, 539k GPEs, 161k FACs and 85k LOCs.

5 Cold Start and TEDL Submissions and Results

To Be supplied

[7] The dataset is available from the Linguistic Data Consortium as LDC2015E42

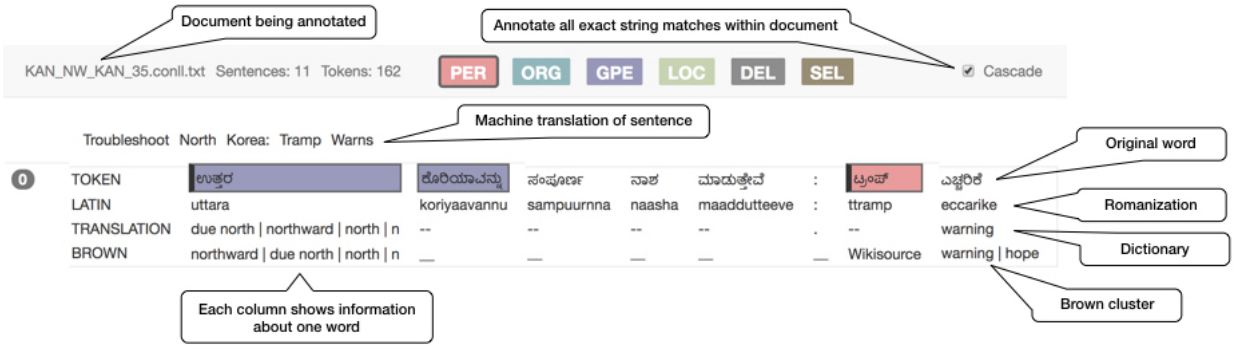

Figure 2: The Dragonfly tool in use annotating a Kannada document.

6 Low resource EDL pilot

The Low Resource EDL pilot asked participants to create entity discovery and linking systems for ten low resource languages. The HLTCOE generated training and test data for five of the languages, and made submissions in those languages using the RPI Blender system.

6.1 Dragonfly Non-speaker Annotation

Dragonfly is an annotation tool from the Johns Hopkins University Applied Physics Laboratory designed to allow a non-speaker of a language to annotate text in that language. Each sentence to be annotated is laid out in a single row. Each word in the sentence is augmented with a variety of information about the word. Given rich enough information about the words in the sentence, it may be possible to annotate the sentence for named entities or other linguistic phenomena without fully comprehending the sentence.

Figure 2 shows a screenshot of a portion of the Dragonfly tool being used to annotate text written in the Kannada language (a Dravidian language spoken in India). Each word of the sentence being annotated appears in its own column. The top entry in the column is the word in its native language. Next is a Romanization of the word. We use the ISI uroman software to generate romanizations of all languages that are not natively written in Latin script. The third entry is one or more dictionary translations, if available. Hovering over this entry will display all known translations, if all of them don't fit into the allotted column width. The fourth entry is a set of dictionary translations of other words in the Brown cluster (Brown et al., 1992) for the word. While these tend to be less accurate than dictionary translations of the word itself, they can give a strong signal that a word falls into a particular category. For example, a Brown cluster containing translations such as 'Paris,' 'Rome,' and 'Vienna' is likely to refer to a city, even if no translation exists to indicate which city. Finally, if automated labels for the sentence have been generated, e.g., by a trained NER system, those labels are displayed at the bottom of the column for the word.

In addition to word-specific information, Dragonfly can present sentence-level information. In Figure 2, an automatic translation of the sentence into English is shown above the words of the sentence. In this example, the translation was produced by Google Translate. Translations might also be available when annotating a parallel document collection. Other sentence-level information that might prove useful in this slot includes a topic model description, or a bilingual embedding of the entire sentence.

Given this presentation of information about the sentence and its component words, an annotator must decide which words and word sequences to annotate as named entities. In Figure 2, we see a short sentence that has been annotated with two named entities. The first is 'North Korea.' The first word of the sentence (Romanization 'uttara') has translations of 'due north,' 'northward,' 'north,' etc. The second word has no direct translations or translations of words in its Brown cluster. However, its Romanization, 'koriyaavannu,' begins with a sequence

Language	#sents	#tokens	GPE	LOC	ORG	PER	Total
Albanian	1,652	41,785	1,153	229	542	759	2,683
Kannada	535	8,158	314	64	192	330	900
Nepali	959	16,036	815	76	224	298	1,413
Polish	1,933	26,924	425	61	501	369	1,356
Swahili	1,714	42,715	1,120	177	513	959	2,769

Table 3: Five annotated languages, with number of sentences, number of tokens, and number of named entity mentions found for each type of entity across fifty annotated documents per language.

that suggests the word 'Korea' together with a morphological ending. Even without the presence of the phrase 'North Korea' in the machine translation output, an annotator likely has enough information to draw the conclusion that the GPE 'North Korea' is being mentioned here. The presence of the phrase 'North Korea' in the machine translation output confirms this choice.

The sentence also contains a word whose Romanization is 'ttramp.' This is a harder call. There is no translation, and the Brown cluster translations are no help. Knowledge of world events, examination of other sentences in the document, the translation of the following word, and the machine translation output together suggest that this is a mention of Donald Trump; it can thus be annotated as a person.

6.2 Generating Training and Test Data

The HLTCOE used Dragonfly to generate both ground truth and training data for five of the ten TAC 2018 EDL Pilot languages shown in Table 3. For each language, we annotated fifty documents. Table 3 provides statistics on the annotated documents, including the number of sentences and tokens in each language, and the number of each type of entity we discovered. Ten of the fifty documents in each language were selected by the track coordinators as test data for the task; we used the remaining forty documents as training data for our runs. Four of the five languages we annotated were annotated by a single person, while Kannada annotation was split up among two people. Thus, for most of our runs, the same person annotated both the training data and some or all of the ground truth. This conveys some advantage to our runs, as inter-annotator disagreements are minimized.

Language	Precision	Recall	F_1
Albanian	0.800	0.722	0.759
Kannada	0.679	0.512	0.584
Nepali	0.739	0.580	0.650
Polish	0.587	0.530	0.557
Swahili	0.756	0.728	0.742

Table 4: Precision, recall, and F_1 values for our five language submissions.

6.3 Results

Table 4 shows our system performance on the five languages. The metric in this table is *strong typed mention match.* This metric assesses the ability of a system to identify the extent and type of named entities in text.

7 Conclusion

The JHU Human Language Technology Center of Excellence has participated in the TAC Knowledge Base Population exercise since its inception in 2009, in the Cold Start task since 2012, and in Entity Discovery and Linking the last three years. We modified the KELVIN system used in the 2016 Cold Start and EDL tasks by adding an open information extraction compoent to discover additional relations. We also participated in the Low Resource EDL pilot task, generating training and test data for five of the languages and making submissions in those languages using the RPI Blender system.

Acknowledgments

This project uses the universal romanizer software 'uroman' written by Ulf Hermjakob, USC Information Sciences Institute (2015-2016).

References

Adrian Benton, Jay Deyoung, Adam Teichert, Mark Dredze, Benjamin Van Durme, Stephen Mayhew, and Max Thomas. 2014. Faster (and better) entity linking with cascades. In *NIPS Workshop on Automated Knowledge Base Construction*.

E. Boschee, R. Weischedel, and A. Zamanian. 2005. Automatic information extraction. In *Int. Conf. on Intelligence Analysis*, pages 2–4.

Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n-gram models of natural language. *Comput. Linguist.*, 18(4):467–479, December.

Mark Dredze, Paul McNamee, Delip Rao, Adam Gerber, and Tim Finin. 2010. Entity disambiguation for knowledge base population. In *Conf. on Computational Linguistics*.

Francis Ferraro, Max Thomas, Matt Gormley, Travis Wolfe, Craig Harman, and Benjamin Van Durme. 2014. Concretely annotated corpora. In *4th Workshop on Automated Knowledge Base Construction*.

Tim Finin, Paul McNamee, Dawn Lawrie, James Mayfield, and Craig Harman. 2014. Hot stuff at cold start: HLTCOE participation at TAC 2014. In *7th Text Analysis Conf.*, Nov.

Tim Finin, Dawn Lawrie, Paul McNamee, James Mayfield, Doug Oard, Nanyun Peng, Ning Gao, Yiu-Chang Lin, Josh MacKin, and Tim Dowd. 2015. HLTCOE participation at TAC KBP 2015: Cold start and TEDL. In *8th Text Analysis Conf.*, Nov.

Tim Finin, Dawn Lawrie, James Mayfield, Paul McNamee, Jessa Laspesa, and Micheal Latman. 2016. HLTCOE Participation in TAC KBP 2016: Cold Start and EDL . In *Proceedings of the 2016 Text Analysis Workshop*. National Institute of Standards and Technology, November.

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, Nov.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The stanford corenlp natural language processing toolkit. In *52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60.

James Mayfield and Tim Finin. 2012. Evaluating the quality of a knowledge base populated from text. In *Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction*. ACL.

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.

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 *5th Text Analysis Conf.*, Gaithersburg, MD, Nov.

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 *NAACL-HLT*, volume 10, page 32.

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

Lance Ramshaw, Elizabeth Boschee, Marjorie Freedman, Jessica MacBride, Ralph Weischedel, and Alex Zamanian. 2011. Serif language processing effective trainable language understanding. *Handbook of Natural Language Processing and Machine Translation: DARPA Global Autonomous Language Exploitation*, pages 636–644.

Laurens Rietveld and Rinke Hoekstra. 2013. Yasgui: Not just another sparql client. In Philipp Cimiano, Miriam Fernández, Vanessa Lopez, Stefan Schlobach, and Johanna Völker, editors, *The Semantic Web: ESWC 2013 Satellite Events*, volume 7955 of *Lecture Notes in Computer Science*, pages 78–86. Springer Berlin Heidelberg.

V. Stoyanov, J. Mayfield, T. Xu, D.W. Oard, D. Lawrie, T. Oates, T. Finin, and B. County. 2012. A context-aware approach to entity linking. *Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction, NAACL-HLT*.