Cold Start Knowledge Base Population at TAC 2017
Task Description[1]

Version 1.0 of May 23, 2017

[1] The TAC organizing committee welcomes comments on this Task Description, or on any aspect of the TAC evaluation. Please send comments to `tac-kbp@nist.gov`.

What's New

The 2017 Cold Start KB and SF tasks differ from the 2016 tasks in the following significant ways:

1. The Cold Start KB includes events (analogous to entities) and event arguments (analogous to slot fillers).
2. The Cold Start KB includes sentiment from an entity towards another entity.
3. The Cold Start KB will be evaluated via a composite KB evaluation using queries (as in Cold Start 2016), and a set of component evaluations for Entity Discovery and Linking (EDL), Slot Filling (SF), Event Nugget Detection and Coreference (EN), Event Argument and Linking (EAL), and Sentiment.
4. Mean Average Precision (MAP) will be the main evaluation metric for the composite KB evaluation and the component SF evaluation.
5. Multiple justifications are allowed and encouraged for KB relations (involving SF, event, or sentiment predicates). Justification spans for any single justification in the KB and SF tasks must come from the same document, in order to make justifications easier to differentiate and count.

The 2017 Cold Start KB Construction task builds on the 2016 task by extending the KB schema to include not only entities and Slot Filling relations, but also events and relations involving event arguments and sentiment between entities.

In addition, Cold Start 2017 has the goal of encouraging systems to 1) provide meaningful confidence values for assertions that are made in the KB and 2) return as much evidence for each KB relation as can be found in the document collection. To encourage development of meaningful confidence values, the primary evaluation metric for Cold Start 2017 will be a variant of mean average precision (MAP). To support an evaluation that rewards systems for finding more than one justification for each KB relation, Cold Start 2017 requires that all text spans for a single justification must come from a single document. This simplification allows TAC to define two justifications to be the same if and only if they come from the same document, and gives more credit for finding more documents that each justify relation.[2]

The submitted Cold Start KBs are evaluated by both a ***composite*** query-based evaluation, and a set of ***component*** evaluations. The composite KB evaluation applies a set of Cold Start evaluation queries to each KB and assesses the correctness of the events, sentiment sources and targets, and SF slot fillers found.

The component evaluations are implemented by projecting out the individual components from the submitted KB and evaluating each component output file as though it had been submitted directly to the standalone track for that component.

[2] The TAC Cold Start KB and SF tasks in 2014-2016 allowed a single relation justification to contain provenance spans from multiple documents, in order to encourage inference across wider contexts. While restricting provenance spans to a single document per justification may seem like a step backwards for KBP 2017, the benefit of being able to count justifications outweighs the reduction in allowable inference. Furthermore, because of the types of relations and what is consider correct in Cold Start/SF, most successful cross-document inference in Cold Start has been limited to a handful of hard-coded inference rules (e.g., involving familial relationships and part-whole relationships between GPE's); such inferences could be done by some downstream process, using a separate inference engine and a possibly richer set of inference rules and world knowledge.

This document describes the 2017 composite (end-to-end) Cold Start KB Construction task and the component Slot Filling task, which are evaluated using post-submission assessment of responses to Cold Start evaluation queries.

The standalone EDL, EN, EAL, and sentiment tasks are evaluated using gold standard annotations on a common set of approximately 500 "core" documents, and are described fully on their respective track home pages. The detailed task description for those component tasks are available on their track home pages:

- Entity Discovery and Linking (http://nlp.cs.rpi.edu/kbp/2017/index.html).
- Event Nugget Detection and Coreference (http://cairo.lti.cs.cmu.edu/kbp/2017/event/)
- Event Argument and Linking (https://tac.nist.gov/2017/KBP/Event/Argument/)
- Belief and Sentiment (http://www.cs.columbia.edu/~rambow/best-eval-2017/)

Introduction

Since 2009, TAC KBP has evaluated performance on several important aspects of knowledge base population: entity discovery and linking, slot filling, event nugget detection and coreference, event argument extraction and linking, and belief and sentiment. The goal of the Cold Start track is to exercise technology in each of these areas, and evaluate the ability of a system to use these technologies to actually construct a knowledge base (KB). Cold Start participants build a software system that processes a large text collection and creates a knowledge base that is consistent with and accurately represents the content of that collection. The knowledge base is then evaluated as a single connected resource, using queries that traverse nodes and predicate links in the KB to determine if the KB contains correct relations between entities, events, and strings. In Cold Start 2017, the predicates can be slot filling predicates, sentiment predicates, or event argument predicates.

We call the task *Cold Start* Knowledge Base Population to convey two features of the evaluation: it implies both that a knowledge base schema has been established at the start of the task, and that the knowledge base is initially unpopulated. Thus, we assume that a schema exists for the entities, events, and relations that will compose the knowledge base; it is not part of the task to automatically identify and name relationships present in the text collection. In 2017, Cold Start uses a schema that combines the entity types, event types, and relations from the TAC component tracks of EDL, SF, EN, EAL, and BeSt.

Cold Start also implies that the knowledge base is initially empty. To avoid solutions that rely on verifying content already present in Wikipedia or other large data sources about entities, the queries used in Cold Start will be dominated by entities that are not present in Wikipedia.

This document describes both the end-to-end Cold Start KB construction task and the component Slot Filling task.

1. In the Cold Start *Knowledge Base* task (CSKB), participants submit entire knowledge bases, without prior knowledge of the evaluation queries.
2. In the *Slot Filling* task (SF), the Cold Start evaluation queries that involve only SF predicates are split into Cold Start Slot Filling queries, with one entry point per query, and are distributed at the start of the task evaluation window. Participants do not have to submit entire knowledge bases. Rather, they apply their slot filling system twice, the first time on the entry point for each query, the second time on each of the results of the first round.

Participating systems in both the Cold Start KB and SF tasks will receive the following input:

1. a *document collection*;
2. a *knowledge base schema*

From these, Cold Start KB systems will produce a knowledge base. This KB will be submitted to NIST as a set of augmented subject-predicate-object triples. The Cold Start KB will include various `*mention`, `link`, and `type` triples, as well as a range of triples involving SF, event argument, and sentiment predicates (all triples are described more fully below). Participating KB systems must tie each entity mention and event mention in the document collection to a particular KB node; in this way, the knowledge base can be queried without first aligning it to a reference knowledge base.

Systems participating in the Slot Filling task will also receive:

3. a set of Cold Start Slot Filling (CSSF) evaluation queries (each evaluation query is a sequence of one or two SF queries to be applied in series).

For both CSKB and SF tasks, the results will then be evaluated by NIST:

- Systems participating in the Slot Filling task return slot fillers directly in response to the given CSSF evaluation queries, and the fillers are then assessed and scored.
- Evaluation of the Knowledge Base variant will start by applying the Cold Start evaluation queries to the submitted knowledge base. Each query will start at a named entity mention in a document (identified by the query's <beg> and <end> tags), identify the knowledge base entity that corresponds to that mention, follow a sequence of one or more relations within the knowledge base, and end in a "slot fill" (which could be an entity, event, or string node, depending on the relation predicate). The resulting slot fills will be assessed and scored in the same way as in the Slot Filling variant. For example, a CSSF evaluation query might ask 'what are the ages of the siblings of the *Bart Simpson*[4] mentioned in Document 42?' A system that correctly identified descriptions of Bart's siblings in the document collection, linked them to the appropriate node in the KB, and also found evidence for and correctly represented the ages of those siblings would receive full credit.

[4] Many of the examples used to illustrate the Cold Start task are drawn from *The Simpsons* television show. Readers lacking a detailed working knowledge of genealogical relationships in the Bouvier/Simpson family need not agonize over what they have been doing with their lives for the past quarter century, but may simply visit http://simpsons.wikia.com/wiki/Simpson_Family.

Input

Schema

The KB schema for Cold Start 2017 consists of:

- Entities: Entities and entity mentions for five entity types (person, organization, geopolitical entity, facility, and location) as defined in the trilingual EDL task of the 2017 EDL track.
- SF Relations: Entity attributes ("slots") as defined in the SF track. These comprise forty-one relation types and their inverses.
- Events: Events and event mentions for 18 event subtypes. Cold Start defines an event as a Rich ERE cross-document event hopper, and an event mention as a Rich ERE event trigger.
- Event (Argument) Relations: Event roles and arguments for the 18 event subtypes, as defined in the EAL track. These comprise 85 event argument relation types having event as the predicate subject, and 56 inverses having an entity as the predicate subject and an event as the predicate object.
- Sentiment Relations: Positive and negative sentiment from a source entity toward a target entity, as defined in the BeSt track.

The schema for Cold Start 2017 combines the entity and mention types from TAC KBP 2016 Entity Discovery and Linking, the SF relation types from TAC KBP 2016 Cold Start Knowledge Base Population, the event roles and argument types from TAC KBP 2016 Event Argument and Linking, the event mentions from TAC KBP 2016 Event Nugget Detection, and sentiment relations from TAC KBP 2016 Belief and Sentiment (BeSt). Annotation/assessment guidelines are available on the TAC web site (http://www.nist.gov/tac/2017/KBP/ColdStart/guidelines.html), and are more fully documented in the data packages that can be requested from the LDC upon completion of TAC KBP track registration.

For relations whose objects ("slot fills") are entities (such as per:siblings or per:likes) or objects (such as per:confict.attack_attacker), Cold Start KBs will be required to link that slot to the node in the submitted KB representing the correct entity[5] or event. Slots whose fills are strings (such as per:title or org:website) must be linked to a specially created string node to represent the object for that relation.

Cold Start entities and entity mentions are defined by DEFT Rich ERE. Full annotation guidelines for DEFT Rich ERE entities are included in the DEFT Rich ERE annotation packages, available from the LDC, but a high-level summary of the five entity types and their mentions are available in *Rich ERE Annotation Guidelines Overview*. For Cold Start, all ***named and nominal mentions*** must be extracted, and the entities must be ***specific individual entities*** (as described in *Annotation Guidelines for Individuality of Specific Entities*). Pronominal entity mentions may be extracted (for use in the sentiment component evaluation in BeSt), but are not required for the composite evaluation of the Cold Start KB. A Cold Start *named entity mention* is the same as a named entity mention in Rich ERE; i.e., a Cold Start named entity mention is a mention that uniquely refers to an entity by its proper name, acronym, nickname, alias, abbreviation, or other alternate name, and includes post author names found in the metadata of discussion forum documents. The extent of

[5] Because facility and location entities are not included in the slot definitions, only person, organization, and geopolitical entity nodes must be linked to the SF slots.

the named entity mention is the entire string representing the name, excluding the preceding definite article and any other pre-posed or post-posed modifiers. A Cold Start *nominal entity mention* is the head of the nominal entity mention in Rich ERE; i.e., a Cold Start nominal entity mention is a mention not including the entity's proper name, referring to it by a common noun phrase (but for Cold Start, the nominal mention is only the head noun of the nominal phrase). Entity mentions are allowed to nest or overlap; for example, the string "Philadelphia Eagles" might be a mention of an ORG (the football team), while the first word might simultaneously be a mention of a GPE (the city of Philadelphia).

Cold Start defines an event as a Rich ERE cross-document event hopper, and an event mention as a Rich ERE event trigger (which is also the same as an event nugget in the TAC KBP Event Nugget track). The criteria for determining whether two or more event mentions belong in the same hopper are essentially the same regardless of whether the mentions are in the same document or different documents. The Cold Start inventory of events and event roles/arguments is a subset of the event types in Rich ERE, and is described in the EAL 2016 Task description.

The Cold Start inventory of SF slots is described thoroughly in *TAC KBP 2015 Slot Descriptions* and *TAC KBP 2015 Assessment Guidelines* available on the TAC Web site. Forty-one slots and their inverses are used for the evaluation. Twenty-six of these have fills that are themselves entities, as shown in Table 1 of the Appendix. The remaining fifteen slots have string fills, as shown in Table 2 of the Appendix. Each SF predicate having an entity as object will have an inverse.[6]

Cold Start sentiment relations are also given in a table in the Appendix, and include an inverse for each sentiment predicate.

Cold Start event relations are given in the last two tables in the Appendix. The first shows event predicates having an event as subject, and an entity or string as object. The second table shows the inverse event predicates, which have an entity (but not a string) as subject, and an event as object.

All inverse relations must be explicitly identified in the submitted knowledge base. That is, if the KB asserts that relation R holds between entities A and B, then it must also assert that relation R^{-1} holds between B and A. As a convenience, the Cold Start KB validation script can be used to introduce missing inverses into a KB, except that the validator will not infer any inverse relations for event predicates having event as subject; all event relations having an event as subject must be explicitly included in the KB.

Document Collection

The Cold Start 2017 evaluation document collection will be the ***TAC KBP 2017 Evaluation Source Corpus***, which comprises approximately 90,000 documents, roughly equally distributed between English, Spanish, and Chinese, and balanced between newswire (NW) and multi-post discussion forum (MPDF) documents. These documents will be new (previously unreleased) documents that will be distributed by NIST via Web download at the beginning of the Cold Start evaluation window. There will be exactly one file per document, and all files will be parsable as XML. Each file will begin

[6] Some SF slots, such as per:siblings, are symmetric. Others, such as per:parents, have inverses that were already in the 2014 English Slot Filling track (in this case, per:children). The remaining SF slots (e.g., org:founded_by) had no corresponding slot in the 2014 English Slot Filling track; Cold Start specifies new slot names for these inverses. All such slots are list-valued.

with the opening tag of the <DOC> element (<doc> for MPDF); [7] note that <DOC> can be spelled with either upper case or lower case letters, depending on the genre, and may optionally include additional attributes (such as "type" for some newswire data).

Newswire data will use the following markup framework:

```
<DOC id="{doc_id_string}" type="{doc_type_label}">

<HEADLINE>

...

</HEADLINE>

<DATELINE>

...

</DATELINE>

<TEXT>

<P>

...

</P>

...

</TEXT>

</DOC>
```

where the HEADLINE and DATELINE tags are optional (not always present), and the TEXT content may or may not include "<P> ... </P>" tags (depending on whether or not the "doc_type_label" is "story").

Multi-Post Discussion Forum files (MPDFs) are derived from Discussion Forum threads. They consist of a continuous run of posts from a thread but they are only approximately 800 words in length (excluding metadata and text within <quote> elements). When taken from a short thread, a MPDF may comprise the entire thread. However, when taken from longer threads, a MPDF is a truncated version of its source, though it will always start with the preliminary post. The MPDF files will use the following markup framework, in which there may also be arbitrarily deep nesting of quote elements, and other elements may be present (e.g. "<a...>...</a>" anchor tags):

```
<doc id="{doc_id_string}">

<headline>

...

</headline>

<post ...>

...
```

[7] In contrast to some of the KBP source corpora prior to 2016, the TAC KBP 2017 Source Corpus will *not* contain any files that begin with xml declarations such as <?xml version="1.0" encoding="utf-8"?>. This is to ensure that offsets align across the various KBP 2017 tracks that are using this same evaluation source corpus, regardless of whether offsets are counted from the beginning of the file, or the beginning of the <DOC> tag.

```
<quote ...>
...
</quote>
...
</post>
...
</doc>
```

All provenance/justifications for Cold Start KB/SF 2017 tasks must be drawn from the documents in the TAC KBP 2017 Evaluation Source Corpus. Each document is represented as a UTF-8 character array and begins with the <DOC> tag, where the "<" character has index 0 for the document. Thus, offsets for provenance are counted *before* XML tags are removed. Start offsets must be the index of the first character in the corresponding string, and end offsets must be the index of the last character of the string (therefore, the length of the corresponding string is endoffset – startoffset + 1).

All KBP 2017 systems should return extractions from anywhere in the document, including <quote> regions of MPDF documents. However, for the following component KBP tracks, in which evaluation is by comparison with gold standard Rich ERE annotations (which will not include annotations of <quote> regions), the track coordinator will automatically filter out <quote> regions from submitted runs before scoring, so as to avoid penalizing runs that include <quote> regions:

(a) EDL
(b) Belief and Sentiment
(c) Event Nuggets
(d) Event Arguments

For the following KBP tasks, in which evaluation is by assessment, assessment and scoring will allow provenance and extractions from anywhere in the document, including <quote> regions:

(a) Slot Filling
(b) Cold Start KB Construction

Evaluation Queries

CSKB and CSSF systems are evaluated by the same set of Cold Start evaluation queries. A Cold Start evaluation query begins with one or more mentions of the same entity, followed by a sequence of slots to be filled for the entity. Each mention in the query is called an *entry point* because it can be used to select (at most) one entity node in a KB that is being evaluated; multiple entry points are included for each Cold Start evaluation query in order to increase the chances that the KB will have a response to the query even if it misses one entry point. Each Cold Start evaluation query is split into multiple Cold Start Single Entrypoint (CSSE) queries, with one entry point per CSSE query (the CSSE queries will request the same slots, but each will have a different entry point).

At the beginning of the CSSF evaluation window, participants in the Slot Filling task will receive a set of CSSF queries, which is the subset of the CSSE evaluation queries that involve only SF slots, and will apply a script to incrementally convert those queries to a form that looks similar to queries from the 2014 English Slot Filling task. Participants in the Knowledge Base variant will not receive the queries; rather, NIST will apply the evaluation queries to each submitted knowledge base and assess the results. An outline of the NIST assessment process for both Cold Start variants is given below.

All CSSE evaluation queries start with an *entry point* into the knowledge base being evaluated. The entry point is defined by a named entity mention (name, docid, begin offset, and end offset), and is followed by the entity type and either one or two slots to be extracted for the entity. The query may request any of the SF, Sentiment, or Event relations that have the query entity as the predicate subject.

Evaluation queries could take one of two forms: single-hop or multiple-hop. For example, here is a sample single-hop CSSE evaluation query that asks "What is the age of the *June McCarthy* mentioned at offsets 16931-16943 in Document 42?":

```
<query id="CSSF16_ENG_00243754cd">
  <name>June McCarthy</name>
  <docid>42</docid>
  <beg>16931</beg>
  <end>16943</end>
  <enttype>PER</enttype>
  <slot>per:age</slot>
  <slot0>per:age</slot0>
</query>
```

This single-hop query looks very much like a query from the 2014 English Slot Filling task, except that each query in Cold Start asks for a specific slot, rather than all slots for which there is information in the document collection. [8]

A more complex "two-hop" query might ask, "What are the ages of the children of the *June McCarthy* mentioned at offsets 16931-16943 in Document 42":

```
<query id="CSSF16_ENG_002109743e">
  <name>June McCarthy</name>
  <docid>42</docid>
  <beg>16931</beg>
  <end>16943</end>
  <enttype>PER</enttype>
  <slot>per:children</slot>
  <slot0>per:children</slot0>
  <slot1>per:age</slot1>
</query>
```

The above queries are homogeneous SF queries in the sense that each query asks for only SF relations. An example of a mixed query, involving both a sentiment predicate (in slot0) and an event predicate (in slot1) is below ("What are the attack events in which the target is a person whom June McCarthy dislikes?"):

```
<query id="CSSF16_ENG_002109347b">
    <name>June McCarthy</name>
    <docid>42</docid>
    <beg>16931</beg>
    <end>16943</end>
    <enttype>PER</enttype>
```

[8] Participants in the Slot Filling variant should treat all other slots as if they appear in the <ignore> field of a Slot Filling query from 2013 or earlier.

```
    <slot>per:dislikes</slot>
    <slot0>per:dislike</slot0>
    <slot1>per:conflict.attack_target</slot1>
  </query>
```

In general, two-hop queries will start from an entry point (selecting the corresponding KB entity of a CSKB submission), follow a single entity-valued relation, then ask for a single slot value. [9] Such queries will verify that the knowledge base is well-formed in a way that goes beyond basic entity linking and slot filling, without allowing combinations of errors to drive scores to zero.

Because two-hop queries do not look like any slot filling queries from KBP 2009-2014, participants in the Cold Start Slot Filling variant must process the CSSF queries in two "rounds" using the `CS-GenerateCSQueries.pl` script from NIST, which adds the `<slot>` entry to the NIST-distributed CSSF queries. Participants in the Slot Filling variant must treat `<slot>` as the slot to be filled. During the first round, `<slot>` will be identical to `<slot0>`. The `CS-GenerateCSQueries.pl` script will then convert a first round output file to a second round query file. Second round queries generated by this script will bear `<slot>` entries equivalent to `<slot1>`. Though some of the CSSF queries will differ only in having different mentions (possibly for the same entity) as their entry points, participating CSSF systems are prohibited from using information about one query to inform the processing of another query.

For the Knowledge Base variant, the following rules are applied to map from a CSSE evaluation query to a knowledge base entry: First, form a candidate set of all KB node mentions that have at least one character in common with the evaluation query mention and that have the same type. If this set is empty, the submission does not contain any answers for the evaluation query. Otherwise, for each mention K in the candidate set, calculate:

- COMMON, the number of characters in K that are also in the query mention Q.
- K_ONLY, the number of characters in K that are not in Q.

Execute each the following eliminations until the candidate set is size one, and select that candidate as the KB node that matches the query:

- Eliminate any candidate that does not have the maximal value of COMMON
- Eliminate any candidate that does not have the minimal value of K_ONLY
- Eliminate all but the candidate that appears first in the submission file

The proper specification of entity `mentions` in a KB is therefore important for scoring well; CSKB participants should therefore take care to ensure that every named entity mention in the evaluation collection serves as a `mention` for a node in the KB.

The NIST evaluation of a KB will proceed by finding all entries in the KB that fulfill an evaluation query. For example, if the evaluation query 'schools attended by the siblings of *Bart Simpson*' finds two siblings for the node specified by the entry point, and the KB indicates that those siblings attended two and one schools respectively, then three results would be assessed by NIST. These results will be converted to the same form as the output for the Slot Filling variant. Results will be pooled across all CSKB and CSSF submissions, and assessors will judge the validity of each result. Finally, a scoring script will report a variety of statistics for each submitted run.

[9] In principle, multiple-hop queries could include more than two relations, but we currently limit ourselves to two.

In creating evaluation queries, LDC will exercise a range of SF, sentiment, and event predicates and strive to balance even distribution across predicate types with productivity of those slots. This means that the queries in the composite evaluation will not necessarily follow the distribution of mention-level occurrences of facts, so even if there are 10 times as many negative sentiments as positive sentiments, the number of queries asking for positive sentiment vs negative sentiment will be about the same. Systems that have been optimized for the component evaluations by assuming a particular distribution of mention-level phenomena in the component evaluation documents, may need to recalibrate to take into account less frequent phenomena (event types, cognitive states, etc.).

Single hop queries will in many cases ask for multiple slots for a given entity regardless of whether fillers for those slots are attested in the document collection. Multiple hop queries will favor entities and slot sequences that are attested in the document collection (although here too, availability of answers is not guaranteed at any hop level).

Because coreference of strings and events is still difficult (for both humans and automatic systems), Cold Start 2017 will mitigate errors and inter-annotator disagreement about how to coreference events and strings (especially cross-document coreference) by avoiding queries that involve an event or string as the predicate subject. Cold Start 2017 queries will always have an entity as the predicate subject ***at each hop level***. This means that all event queries will be of the form: Find all events that have entity X in role Y (e.g., "Find all conflict.attack events that have the entity "Homer Simpson" in the role "attacker", but ***not*** "Find all targets of conflict.attack events where Homer is the attacker").

Single-hop queries will request some event or string object in slot0; two-hop queries will request an entity object in slot0, and an entity, event, or string object in slot1. The targeted distribution of predicates across queries that are assessed in the final evaluation is given in the following table.

Approximate distribution of predicate types across Cold Start evaluation queries								
	Single-hop queries		Two-hop queries					
Slot0	event	SF (string object)	SF	SF	SF	sentiment	sentiment	sentiment
Slot1	--	--	SF	sentiment	event	SF	sentiment	event
	3/12	1/12	3/12	1/12	1/12	1/12	1/12	1/12

Cold Start KB Task Output

The Cold Start knowledge base is represented as a directed labeled multigraph: entities, events, and other predicate arguments (i.e., "string" arguments, that are neither entities nor events) are represented as nodes in the knowledge graph, while binary relations between the entities/events/strings are represented as edges labeled with SF, sentiment, and event predicate names. Generally, the subject and the object of the predicate are both nodes in the KB (except that the `type`, `link`, and `*mention` predicates take a quoted string as object). The KB is grounded to the document collection via various `*mention` predicates that connect each entity, event, or string node to its mentions in the document collection. The Cold Start KB is connected to external KBs via a `link` predicate that indicates coreference between an entity in the Cold Start KB and an entity node in the external KB.

CSKB systems must produce a knowledge base as output. The first line of the KB output file must contain a unique run ID. The remainder of the KB output file is a set of assertions, or augmented triples (subject, predicate, object). Assertions will appear, one-per-line, in tab-separated format. The KB output file will be automatically converted to RDF statements during evaluation. All KB output must be encoded in UTF-8.

Each triple appears in the KB output file in subject-predicate-object order. For example, to indicate that Entity4 has Entity7 as a sibling, the triple might be:

```
:Entity4     per:siblings  :Entity7
```

If Entity4 has siblings in addition to Entity7, these relations should be entered as separate triples.

Each triple in the CSKB submission will include a set of augmentations (again using tabs as separators). Except for the `type` predicate (which does not require explicit support from a document) the first augmentation will describe the provenance (a justification) of the triple, and the second augmentation will provide the confidence for the triple and justification. If there is more than one justification for a triple, each justification appears in a separate assertion (line), along with its confidence.

At least one assertion for each unique subject-predicate-object triple will be evaluated. If more than one assertion of a given triple appears in the output (with each triple having different provenance), LDC will assess the assertion with the highest confidence value (see below), and will assess additional assertions if resources allow. If more than one such assertion shares the same confidence value, the assertion that appears earlier in the output will be considered to have higher confidence.

Nodes

The KB contains three different kinds of nodes: Entity, Event, and String. Each node specification begins with one of ":Entity", ":Event", or ":String", followed by a sequence of letters, digits, and underscores. Examples of legal entity, event, and string node specifications include `:Entity42`, `:Event_056`, and `:String74_R29`, respectively. No meaning is ascribed to this sequence by the evaluation software; it is used only as a unique identifier. Any subsequent use of the same colon-preceded sequence will be taken as a reference to the same entity, event, or string node.

Each specific individual entity or event that appears in the document collection must be represented by exactly one entity or event node. Two separate entity (or event) nodes in the KB will be interpreted as representing two different entities (or events).

The string node is a catch-all structure to represent SF and Event predicate arguments that are not specific individual entities. The string node allows the KB to represent multiple justifications for the same subject-predicate-object triple when the object is string-valued (e.g., per:cause_of_death), even when the literal strings are different (e.g., "cardiac arrest" vs. "heart attack"); if a given subject and predicate has two separate string nodes as objects, they will be interpreted as representing two different slot fillers for that subject and predicate. However, in TAC 2017 there is no requirement that the same string node be used for triples having *different* subjects or predicates (i.e., for "John and Mary died of heart disease", the KB may contain two different string nodes for "heart disease", where one node is the cause of death of John, and the other node is the cause of death of Mary).[10]

[10] The KB may elect to represent each real-world concept or value by exactly one node (e.g., for dates and numeric values), but such a global grounding of nodes to the real world is not a requirement in TAC 2017 (beyond the entities and events that are defined explicitly in the 2017 TAC KBP ontology). A richer ontology of concepts and values is left for future work.

For the 2017 evaluation, the string object only needs to include one mention for each document provided as justification for that subject-predicate-object triple.

In particular, each event argument that is not a specific individual entity (e.g., unnamed aggregates like "3 people") must be represented in the Cold Start KB as a string node. When the event argument is not a valid Cold Start entity (i.e., a specific individual PER, ORG, GPE, LOC, or FAC), the KB may represent each argument string as a separate string node, even if it's clear from context that the strings are coreferential; string valued event arguments are ignored in the composite KB evaluation (Cold Start 2017 event queries will always have an entity node as subject, and an event node as object), and are used only to produce the EAL output files for the component EAL evaluation (which does not require explicit coreference of event argument strings).

Predicates

The legal predicates are the Slot Filling predicates, sentiment predicates, and event predicates shown in the Appendix, plus `type`, `link`, `mention`, `nominal_mention`, `pronominal_mention`, `normalized_mention` and `canonical_mention`.

SF predicates found in Table 1 must have entity specifications in both the subject and object positions; predicates found in Table 2 must specify an entity node in the subject position, and a string node in the object position; the string node in the object position will exactly correspond with the slot fill for that relation in the Slot Filling task.

type

Each entity, event, and string node will be the subject of exactly one `type` triple. The object of that triple will be one of the allowable types listed in Table 1 below. It is up to submitting systems to correctly identify and report the type of each entity and event; all string nodes must have type STRING.

Table 1 Allowable values for type predicate

Node	Allowable type
String	STRING
Entity	PER
	ORG
	GPE
	LOC
	FAC
Event	CONFLICT.ATTACK
	CONFLICT.DEMONSTRATE
	CONTACT.BROADCAST

	CONTACT.CONTACT
	CONTACT.CORRESPONDENCE
	CONTACT.MEET
	JUSTICE.ARREST-JAIL
	LIFE.DIE
	LIFE.INJURE
	MANUFACTURE.ARTIFACT
	MOVEMENT.TRANSPORT-ARTIFACT
	MOVEMENT.TRANSPORT-PERSON
	PERSONNEL.ELECT
	PERSONNEL.END-POSITION
	PERSONNEL.START-POSITION
	TRANSACTION.TRANSACTION
	TRANSACTION.TRANSFER-MONEY
	TRANSACTION.TRANSFER-OWNERSHIP

*mention predicates

Each entity, event, and string node will be the subject of one or more predicates from {`mention`, `nominal_mention`, `pronominal_mention`, `canonical_mention`, `normalized_mention`}; the term "`*mention`" is used to refer to these predicates. Together with the provenance information (see below), these `*mention` triples indicate how the knowledge base is tied to the document collection. The object of a `*mention` triple is the double-quoted mention string; document ID and offset appear under provenance information (see below).

`mention and nominal_mention and pronominal_mention`

Each entity will be the subject of one or more `mention`, `nominal_mention` or `pronominal_mention` triples. The definition of what constitutes a named, nominal, or pronominal entity mention for Cold Start is described in the Cold Start schema above. Each named entity mention in the collection must be submitted as the object of a `mention` triple, while each nominal entity mention in the collection must be submitted as the object of a `nominal_mention` triple. For example, if an entity is mentioned by name five times in a document, five `mention` triples should be generated. The `pronominal_mentions` are used only for the component BeSt evaluation, and are not referenced or evaluated in the composite KB evaluation or any of the other component evaluations besides BeSt; therefore, the KB only needs to include pronominal mentions that serve as provenance for sentiment assertions. An example is shown below to demonstrate the usage of the assertion:

```
:Entity_0007 type   PER
:Entity_0007 canonical_mention   "Dzhokhar Tsarnaev" NYT_ENG_20131113.0264:434-450 1.0
:Entity_0007 mention             "Dzhokhar Tsarnaev" NYT_ENG_20131113.0264:434-450 1.0
:Entity_0007 pronominal_mention  "he"                NYT_ENG_20131113.0264:546-547 1.0
```

Each event will be the subject of one or more `mention` triples. The definition of what constitutes an event mention is described in the Cold Start schema above. Each mention of the event (i.e., each event nugget or ERE event trigger for the event) must be submitted as the object of a `mention` triple. Event mentions need to be exhaustive in order to support evaluation of the component Event Nugget Detection and Coreference task.

Each string node will be the subject of one or more more `mention` triples. The string node only needs to include one mention for each document provided as justification for that subject-predicate-object triple.

`canonical_mention`

For each document that mentions an entity or event, one of the `mentions` (or `nominal_mentions` if it's an entity) must be identified as the *canonical mention* for that entity/event in that document; it is the string that will be seen by the assessor if that entity/event appears as a slot fill, supported by that document (in Slot Filling task terms, it is the content of Column 5 of a CSSF 2017 submission, and its provenance will serve as Column 7 of the CSSF submission).[13] This implies that a document attesting to a relation must contain `mentions` or `nominal_mentions` of both the subject and the object of the relation. Canonical mentions are expressed using a `canonical_mention` triple. The arguments for `canonical_mention` are the same as for `mention`. Note that there is no requirement that submissions select a single, global canonical mention for an entity. While such a mention might be useful, here we require that a canonical mention be provided within each *document* for the assessor to use during assessment.

Each `canonical_mention` is also a `mention` (or `nominal_mention` or `pronominal_mention` if the node is an entity). As a convenience, if a submitted KB does not contain a `mention` (or `nominal_mention/pronominal_mention`) triple for each `canonical_mention` triple, the missing relations will be inferred (perhaps incorrectly) as `mentions` (albeit with a warning). This shortcut is provided to make submitted KBs easier to view, and does not relieve submitters from the requirement to provide each of the required `mentions`, `nominal_mentions`, and `canonical_mentions`.

[13] In the Slot Filling task of KBP 2009-2014 (and in the Slot Filling variant of Cold Start), all slot fills are strings. Assessors verify the validity of a slot fill by looking for that string in the specified document, using the provenance information provided in the system response. In a submitted KB, slots that are filled with entities or events hold not strings, but pointers to the KB structure for the appropriate entity/event. Thus, a canonical mention must be identified by the Cold Start KB for each entity in each document, so that the assessor can be presented with a string that represents the entity during assessment.

`normalized_mention`

In order to allow normalized dates (and other normalized strings in future) in the KB, a string node for a normalized string value must be the subject of a `normalized_mention` predicate. An example of the usage of a `normalized_mention` assertion is shown below:

```
:String_0001  type   STRING

:String_0001  mention      "April 15"    NYT_ENG_20131113.0264:624-631    1.0

:String_0001  normalized_mention  "2013-04-15"  NYT_ENG_20131113.0264:624-631    1.0
```

The string provided as the object of `normalized_mention` would not be verified against text in the source document; however, it is a requirement that for a given string node the provenance of the `normalized_mention` assertion should be the same as the provenance of another (non-normalized) `mention` of that string.

link

Each entity may be the subject of up to one `link` predicate. The object of the predicate is a quoted string of the form "ExternalKBID:ExternalNodeID" and indicates that the Cold Start entity is the same as the entity with ID "ExternalNodeID" in an external reference KB. For TAC 2017, the external reference KB is the same as that used in 2015-2017 for the TAC Trilingual EDL track, namely LDC2015E42 (TAC KBP Knowledge Base II – BaseKB). The `link` predicate is ignored in the composite KB evaluation and is used only for the EDL component evaluation.

The following example shows how to use link (assuming the external KB is the reference KB in LDC2015E42: TAC KBP Knowledge Base II - BaseKB):

```
DEMO

:Entity_0001  type   GPE

:Entity_0001  canonical_mention   "Boston"     NYT_ENG_20131113.0264:402-407    1.0

:Entity_0001  mention      "Boston"     NYT_ENG_20131113.0264:402-407    1.0

:Entity_0001  link   "LDC2015E42:m.050v43"
```

This will produce the following EDL output:

```
DEMO :Entity_0001_M00001 Boston NYT_ENG_20131113.0264:402-407 m.050v43 GPE NAM  1.0
```

SF, sentiment, and event predicates

The KB must include all triples involving SF predicates, sentiment predicates, and event predicates in the Appendix.

Event Realis

The KB must specify realis for

- event predicates, and
- *mention predicates that have an event node as subject.

Realis may take one of the following values:

- actual

- generic
- other

In order to support the Event Nugget and Event Argument and Linking component evaluations of the KB, the KB must specify the realis of event mentions and event argument assertions, and include events and argument assertions of all three realis values. The realis of an event mention in the Cold Start KB follows the definition of realis in Rich ERE, while the realis of an event argument assertion follows the definition in the Event Argument and Linking task (EAL).

From the perspective of a Cold Start KB user, both completed and planned events are of interest, but not generic events; therefore, when a Cold Start query requesting an event is applied to the KB, it will consider only event nodes that have an "actual" or "other" mention, where the query entity is an "actual" or "other" argument for the event. This means, for example, that the query "Find all attack events that have Homer as an attacker" means "Find all ACTUAL or OTHER events that have Homer as an ACTUAL or OTHER attacker". GENERIC events (that have been mis-classified as ACTUAL or OTHER in the KB) will be assessed as Wrong.

Events that have "generic" realis in the KB are ignored in the composite query-based evaluation, and are used only for the component event nugget and event argument evaluations.

The realis value should be appended at the end of the predicate name, using "." to separate the two. For example,

```
:Event_0001 type    CONFLICT.ATTACK

:Event_0001 mention.actual  "bombing"   NYT_ENG_20131113.0264:418-424   1.0

:Event_0001 canonical_mention.actual    "bombing"   NYT_ENG_20131113.0264:418-424  1.0

:Event_0001 conflict.attack:attacker.actual :Entity_0007    NYT_ENG_20131113.0264:492-
681;NYT_ENG_20131113.0264:546-547;NIL 1.0
```

Provenance

Each assertion (except for type assertions) must contain a single justification (provenance) immediately after the subject-predicate-object triple. Provenance is a set of justification spans; each span may comprise at most 200 UTF-8 characters. Each justification span will include a document ID, followed by a colon, followed by two dash-separated offsets (begin and end offsets). The offsets that show the provenance of an extracted relation are used to narrow the assessor's focus within the documents when assessing the correctness of that relation.

Provenance spans for a single justification must come from a single document. This is a new restriction in 2017, to allow justifications to be countable based on the justification documents.

Provenance spans can be partitioned into four different groups:

- FILLER_STRING (must have exactly 1 span)
- PREDICATE_JUSTIFICATION (may have 1-3 spans; multiple spans are separated by a comma)
- BASE_FILLER (must have exactly 1 span)
- ADDITIONAL_JUSTIFICATION (may have any number of spans; multiple spans are separated by a comma)

Provenance for the assertion consists of some subset of the four span groups above, depending on the predicate and object; multiple groups are separated by semicolon.

a) If the predicate is `type`:
    - No provenance should be provided
b) Otherwise, if the predicate is any of the `*mention` predicates:
    - Provenance consists of only PREDICATE_JUSTIFICATION, containing exactly one span, representing the exact location of the mention in the document collection
c) Otherwise, if the predicate is a sentiment predicate:
    - Provenance consists of only PREDICATE_JUSTIFICATION, containing exactly one span
    - The provenance must be a mention of the entity that is the target of the sentiment, and must be the mention closest to where the sentiment is expressed
    - N.B.: The target of the sentiment could be either the subject or object of the predicate, depending on the predicate:
    - for likes and dislikes predicates, the target of the sentiment is the object
    - for is_liked_by and is_disliked_by predicates, the target of the sentiment is the subject
d) Otherwise, if the predicate is an SF predicate with a non-string object:
    - Provenance consists of only PREDICATE_JUSTIFICATION, containing 1-3 spans
e) Otherwise, if the predicate is an SF predicate with a string object:
    - Provenance consists of FILLER_STRING;PREDICATE_JUSTIFICATION
    - FILLER_STRING must be one of the mentions of the string object
    - PREDICATE_JUSTIFICATION contains 1-3 spans
f) Otherwise, if the predicate is an event predicate with a non-string object
    - Provenance consists of PREDICATE_JUSTIFICATION;BASE_FILLER;ADDITIONAL_JUSTIFICATION
    - PREDICATE_JUSTIFICATION contains 1-3 spans
    - ADDITIONAL_JUSTIFICATION may be "NIL" or any number of spans
g) Otherwise, if the predicate is an event predicate with a string object
    - Provenance consists of FILLER_STRING;PREDICATE_JUSTIFICATION;BASE_FILLER;ADDITIONAL_JUSTIFIC ATION
    - FILLER_STRING must be one of the mentions of the string object
    - PREDICATE_JUSTIFICATION contains 1-3 spans
    - ADDITIONAL_JUSTIFICATION may be "NIL" or any number of spans

For predicates with a string object, the first justification span (FILLER_STRING) must represent the document ID and offsets of the string fill. (Slot Filling variant participants are already providing this information in Column 7 of their submissions.) This is the text that will be shown to assessors instead of a canonical_mention for the string node.

Confidence Measure

To promote research into probabilistic knowledge bases and confidence estimation, each assertion in the KB (or slot fill in the CSSF submission) must have an associated confidence score. Confidence scores will be used to compute a variant of Mean Average Precision (MAP) in the composite KB evaluation. Confidence scores will be used to induce a total order over the relations being evaluated (ties are broken when two scores are equal by assuming that the assertion appearing earlier in the submission has a higher score). Any submitted confidence score must be a positive real number between 0.0 (exclusive, representing the lowest confidence) and 1.0 (inclusive, representing the highest confidence), and must include a decimal point (no commas, please) to clearly distinguish it from a document offset. Confidence scores, if present, will appear at the end of each output line, separated from the provenance information with a tab. If no confidence score is provided for an assertion, the confidence will be inferred to be 1.0 for the purposes of evaluation. Confidence scores may not be used to qualify two incompatible fills for a single slot; submitter systems must decide amongst such possibilities and submit only one. For example, if the system believes that Bart's only sibling is Lisa with confidence 0.7 and Milhouse with confidence 0.3, it should submit only one of these possibilities. If both are submitted, it will be interpreted as Bart having two siblings.

Comments

Output files may contain comments, which begin at any occurrence of a pound sign (#) and continue through (but do not include) the end of the line. Comments and blank lines will be ignored. The first line of a KB variant output file must contain the unique run ID (i.e., it may not be blank). Submitters may like to add a comment to this line giving further details about the run.

Slot Filling Task Output

Output for the Slot Filling variant will be in the form of a tab-separated file. The columns of the submitted file are as follows:

Column 1	Query ID. For the first round, this is taken directly from the <query> XML tag. For the second round, this is drawn from the <query> tag of the query generated from the first round output.
Column 2	The name of the slot being filled.
Column 3	A unique run ID for the submission.
Column 4	Provenance for the relation between the query entity and slot filler, consisting of up to 3 docid:startoffset-endoffset triples separated by commas. Individual spans may comprise at most 200 UTF-8 characters. Unlike the 2014 Slot Filling task, there is no requirement to generate NIL entries when no information about the target entity is available.
Column 5	A slot filler (possibly normalized, e.g., for dates). This is used both to populate the <name> entry of the next round query, and by the assessor to judge the slot fill. The string should be *extracted* from the filler provenance in Column 7, except that any embedded tabs or newline characters should be converted to a space character and dates must be normalized (therefore, slot fillers should *not* be translated across languages). If a nominal mention is returned as a slot filler, only the head word of the nominal phrase should be returned (consistent with the EDL definition of nominal mentions). For dates, systems must normalize document text strings to standardized month, day, and/or year values, following the TIMEX2 format of yyyy-mm-dd (e.g., document text "New Year's Day 1985" would be normalized as "1985-01-01"); if a full date cannot be inferred using document text and metadata, partial date normalizations are allowed using "X" for the missing information.
Column 6	A filler type, selected from {PER, ORG, GPE, STRING}. The STRING filler is

	used for string-valued slots shown in Table 2.
Column 7	Provenance for the slot filler string. This is either a single span (docid:startoffset-endoffset) from the document where the **canonical** slot filler string was extracted, or (in the case when the slot filler string in Column 5 has been normalized) a set of up to two comma-separated docid:startoffset-endoffset spans for the base strings that were used to generate the normalized slot filler string. The documents used for the slot filler string provenance must be a subset of the documents provided in Column 4. This column serves two purposes. First, LDC will judge Correct vs. Inexact with respect to the document(s) provided in the slot filler string provenance. Second, this column is used to fill the `<docid>`, `<beg>` and `<end>` entries in second round queries. If more than one provenance triple is provided here, the first one will be used to fill the second round query.
Column 8	Confidence score.

The process for constructing a Slot Filling variant submission is as follows:

- Download the following from the NIST Web site:
  - The Cold Start evaluation documents`CS-GenerateQueries.pl` script
  - `CS-PackageOutput.pl` script
  - `CS-ValidateSF.pl` script
- Send an email to tac-manager@nist.gov to request the following:
  - The CSSF evaluation queries
- Configure your system to produce results only from the Cold Start evaluation documents.
- Run the `CS-GenerateQueries.pl` script on the evaluation queries to produce the first round queries your system will run on. Note that the raw evaluation queries might differ from the format given above, so you should not assume that you can use them as input to your system without running this script.
- Run your system, producing a slot-filling submission for the first round queries.
- Run the `CS-ValidateSF.pl` script on your first round output to verify that it is formatted correctly.
- Run the `CS-GenerateQueries.pl` script on the evaluation queries and your first round output to produce the second round queries.
- Run your system on the second round queries to produce a second output file.
- Run the `CS-PackageOutput.pl` script on the two output files to produce your submission.
- Run the `CS-ValidateSF.pl` script on your submission to verify that it is formatted correctly.
- Upload the submission to NIST.

Slot filling systems that participated in the 2014 Slot Filling task will need to handle the following differences to successfully participate in the 2017 CSSF task:

- Only the slot specified by the `<slot>` entry is to be filled; all other slots should be ignored. The `<slot>` entry is added to the queries received from NIST by running the `CS-GenerateQueries.pl` script.
- Participants will need to do one round of slot filling, run the `CS-GenerateQueries.pl` script to create the second round queries, then run slot filling again on the new queries. The

results of rounds one and two are to be concatenated before submission using the `CS-PackageOutput.pl` script.

- CSSF requires that participants be able to fill all slots in both directions. For example, the 2014 Slot Filling task required detection of the `per:cities_of_residence` slot. CSSF also requires systems to be able to detect the inverse of that slot, `gpe:residents_of_city`.
- Each slot filler must be assigned a type, selected from {PER, ORG, GPE, STRING}. This field represents an additional output column not found in the 2014 Slot Filling or CSSF tasks.
- NIL entries, indicating that no information about a particular slot is available, are not required in CSSF.
- Nominal mentions of slot fillers may be return if no named entity mention is available in the document collection. (Returning nominal entity mentions is not required, but may improve system recall if done correctly.)
- To conform with requirements in the Cold Start KB task, provenance for each SF relation is limited to only 3 spans (instead of 4), and each span may have up to 200 UTF-8 characters (instead of 150).

Here are example lines from a Slot Filling submission:

```
Q4 org:city_of_headquarters myrun1 Doc42:3-8,Doc8:3-11 Baltimore GPE Doc8:3-11 1.0
Q5 per:siblings myrun1 Doc124:283-288,Doc885:173-179 Lisa PER Doc124:283-286  0.7
Q6 per:age myrun1 Doc124:180-181,Doc885:173-179 10 STRING Doc124:180-181  0.9
```

Evaluation

The submitted Cold Start KBs are evaluated by both a ***composite*** query-based evaluation, and a set of ***component*** evaluations. The composite KB evaluation applies a set of Cold Start evaluation queries to each KB and assesses the correctness of the events, sentiment sources and targets, and SF slot fillers found.

Because the composite evaluation may hide many factors contributing to the performance of the end-to-end KB system, each submitted KB also undergoes a set of component evaluations. The component evaluations are implemented by projecting out the individual components from the submitted KB, such that each component output file is formatted in the same way as a submission to the KBP track for that component, and evaluating each output file as though it had been submitted directly to the standalone track for that component.

Except for the SF task, all component tasks are evaluated using gold standard annotations on a common set of approximately 500 "core" documents.

Component Evaluations

The following component files are projected from each Cold Start KB for the component evaluations:

1. Entity discovery and linking (EDL): An EDL file consisting of name and nominal mentions and links for PER, ORG, GPE, FAC, and LOC entities from the "core" documents used to evaluate submissions to the EDL track. Links can be to either a node in the reference KB (TAC KBP Knowledge Base II - BaseKB) or (if the entity does not exist in the reference KB) a

NIL node corresponding to an entity node in the submitted KB. Evaluation of the Entity Discovery and Linking component of submitted Cold Start KBs will be identical to scoring for the 2017 TAC Trilingual Entity Discovery and Linking task. Please see *TAC KBP2017 Entity Discovery and Linking Task Description* for complete details on scoring.

2. Slot Filling: An SF file consisting of slot fillers and justifications found in the KB by applying Cold Start evaluation queries that involve only SF predicates. The component SF evaluation is identical to the composite KB evaluation, except that the SF evaluation includes only CS queries that involve only SF slots. Because SF systems are allowed to submit only one justification per relation, only the top ranked justification per relation will be considered for the SF component evaluation of KBs.
3. Event Nugget Detection and Coreference: An EN file consisting of event mentions and within-document coreference from the "core" documents. Evaluation of the Event Nugget component of submitted Cold Start KBs will be identical to scoring for the 2017 TAC Event Nugget Detection and Coreference task. Please see *TAC KBP2017 Event Nugget Detection and Coreference Task Description* for complete details on scoring.
4. Event Argument and Linking: A set of "arguments" files, each file consisting of event argument assertions (including justifications) from a "core" document; a set of "linking" files, each file consisting of coreference of assertions in the corresponding "arguments" file. Evaluation of the Event Argument component of submitted Cold Start KBs will be identical to scoring for the 2017 TAC Event Argument and Linking task. Please see *TAC KBP2017 Event Argument and Linking Task Description* for complete details on scoring.
5. Sentiment: A set of predicted ERE xml files, each file consisting of name, nominal, and pronominal mentions and coreference for PER, ORG, GPE, FAC, and LOC entities from a "core" document; a set of BeSt xml files, each file consisting of sentiment (including provenance) from a source towards a target entity in the corresponding predicted ERE file. Evaluation of the sentiment component of submitted Cold Start KBs will be identical to scoring for the 2017 TAC Belief and Sentiment task, except that only sentiment towards entities will be evaluated. Please see *TAC KBP2017 Belief and Sentiment Task Description* for complete details on scoring.

Composite Evaluation Assessment

Cold Start 2017 assessment and scoring will proceed as follows: The responses for each evaluation query (from both CSKB and CSSF systems and from human-generated results) will be pooled, and each response will be assessed by a person. The result of following the first relation will be assessed as if it were a Slot Filling query. The second relation in the query will also be assessed as a Slot Filling query, but only if the fill for the first relation is correct. ***If the fill for the first relation is not correct, each fill for the second relation is automatically counted as Wrong.*** For example, if the query asks for the ages of the siblings of "Bart Simpson," and the submitted knowledge base gives "Lisa age 8" and "Milhouse age 10" as siblings, then only the reported age of Lisa will be assessed (Milhouse is not Bart's sibling), and the reported age of Millhouse will automatically be counted as Wrong.

Cold Start uses *pseudo-slot* scoring to evaluate multiple-hop queries, in which each evaluation query is treated as if it selects a single indivisible slot. For example, an evaluation query that asks for the children of the siblings of an entity will be scored as if it were a query about a virtual `per:nieces_and_nephews` slot.[15] The guidelines in *TAC KBP 2015 Slot Descriptions* specify whether each of the component slots of a pseudo-slot is single-valued (*e.g.,* `per:date_of_birth`) or list-valued (*e.g.,* `per:employee_of`, `per:children`). A pseudo slot is single-valued if each of its component slots is single-valued, and list-valued otherwise. In contrast to the Slot Filling task, Cold Start KB submissions may contain multiple fills for single-valued slots. If such are present in the submission, LDC will assess the slot fill with the highest confidence value, and will assess additional slot fills if resources allow. If more than one such slot fill shares the same confidence value, the slot fill that appears earlier in the output will be considered to have higher confidence.

Each CSSF slot filler response (or CSKB object of each component relation that makes up a single evaluation query response) is assessed as Correct, ineXact, or Wrong. A response is inexact if it either includes only a part of the correct answer or includes the correct answer plus extraneous material. Inexact answers are counted as Wrong for the purposes of scoring. If the relation object is an event, the canonical_mention should be an ERE event trigger but (given the difficulty of determining exact extents of event mentions), assessors in 2017 will be lenient when assessing the extent of an otherwise correct event mention.

For each query, all system responses in which the slot filler is assessed as Correct or ineXact will be partitioned into equivalence classes, where slot fillers in the same equivalence class represent the same entity, event, or value (as in the case of dates). Each Correct or ineXact response will receive an annotation for filler mention type (either NAM or NOM), and each equivalence class will receive an annotation for equivalence class mention type (NAM if the assessor can find a named mention for the filler anywhere in the provenances in any of the responses; otherwise, NOM if only nominal mentions appear in the provenances of all responses).

Pseudo-slots will be scored just as slots in the Slot Filling task, with the additional constraint that both the slot fill and the path leading to that fill must be correct for the entirety to be judged correct. To receive credit for identifying Maggie Simpson as Patty Bouvier's niece, the knowledge base must not only include Maggie as the slot fill, but must also represent Maggie as Marge's child, and Marge as Patty's sibling:[16]

Evaluation query:	Nieces and nephews of Patty Bouvier (`per:siblings`, `per:children`)
Ground Truth:	`:PattyBouvier per:siblings :MargeSimpson`
	`:MargeSimpson per:children :MaggieSimpson`
Submission:	`:PattyBouvier per:siblings :MargeSimpson`
	`:MargeSimpson per:children :MaggieSimpson` ⇒ **correct**

A KB that indicated that Maggie was Patty's niece because she was Patty's sister Selma's child would be scored as incorrect:

Evaluation query:	Nieces and nephews of Patty Bouvier (`per:siblings`, `per:children`)
Ground Truth:	`:PattyBouvier per:siblings :MargeSimpson`
	`:MargeSimpson per:children :MaggieSimpson`
Submission:	`:PattyBouvier per:siblings :SelmaBouvier`
	`:SelmaBouvier per:children :MaggieSimpson` ⇒ **incorrect**

[15] A pseudo-slot is similar to the concept of a *role chain,* which is supported by some knowledge representation systems based on description logic, including OWL 2.

[16] In each of these examples, only the subject, predicate and object are shown, and only a subset of the relevant knowledge base is presented. Each entity is named after the mention that gave rise to it.

In addition, the object of the *final* relation in a pseudo-slot may be rated as redundant if it is equivalent to another fill for the pseudo-slot. Redundant answers are counted as Wrong for the purposes of scoring:

Evaluation query:	Nieces and nephews of Patty Bouvier (`per:siblings`, `per:children`)
Ground Truth:	`:PattyBouvier per:siblings :MargeSimpson`
	`:MargeSimpson per:children :MaggieSimpson`
	`:MaggieSimpson per:alternate_names "Margaret Simpson"`
Submission:	`:PattyBouvier per:siblings :MargeSimpson`
	`:MargeSimpson per:children :MaggieSimpson` ⇒ **correct**
	`:MargeSimpson per:children :MargaretSimpson` ⇒ **redundant**

However, objects of relations other than the final relation will never be rated as redundant:

Evaluation query:	Nieces and nephews of Patty Bouvier (`per:siblings`, `per:children`)
Ground Truth:	`:PattyBouvier per:siblings :MargeSimpson`
	`:MargeSimpson per:children :LisaSimpson`
	`:MargeSimpson per:children :BartSimpson`
	`:MargeSimpson per:alternate_names "Marjorie Simpson"`
Submission:	`:PattyBouvier per:siblings :MargeSimpson`
	`:PattyBouvier per:siblings :MarjorieSimpson`
	`:MargeSimpson per:children :LisaSimpson` ⇒ **correct**
	`:MarjorieSimpson per:children :BartSimpson` ⇒ **correct**

Here, Marge Simpson and Marjorie Simpson represent the same person in the ground truth, but two distinct entities in the KB. However, because the query is about Marge's children and not about Marge herself, both responses to the evaluation query are assessed as correct.

Since in Cold Start the facts being evaluated come from sequences of triples, confidence scores would need to be combined if we wanted to generate confidence scores for a derived pseudo-relation. Three general score combination functions are min, max and product. The proper way to combine scores of course depends on the meaning of those scores; for now, Cold Start will select the product function as a reasonable confidence combination function.

Composite Evaluation Scoring

Given the above approach to assessment, basic scoring for a given system proceeds as follows:

- Each response assessed as Wrong or ineXact, is counted as *Spurious*
- Each response for Round 2 whose Round 1 parent filler is assessed as Wrong or ineXact, is counted as *Spurious.* This scoring policy assumes that the Cold Start system output is intended for fully automatic downstream analytics; hence, a hop1 response is Wrong if the hop0 parent response is Wrong.
- Responses assessed as Correct or Inexact are grouped into equivalence classes. For queries with an entity as the predicate object, if the system has a NAM entity mention in the equivalence class, or if the system has only NOM entity mentions and the equivalence class is annotated as NOM, then the responses in the equivalence class are counted as *Right*; otherwise, if the system has only NOM entity mentions in the equivalence class and the equivalence class is annotated as NAM, then the responses in the equivalence class are counted as *Ignore* (i.e., treated as if it was never returned by the system) and removed from the equivalence class. Thus, named entity mentions are preferred and a named mention must be returned if one exists.

- **Reference** = number of single-valued pseudo-slots with a correct response + number of equivalence classes[17] for all list-valued pseudo-slots

After each pooled response has been assessed and (if assessed as Correct or Inexact) put into an equivalence class, NIST will score a submitted KB by computing a variant of mean average precision (MAP) for the KB. Average precision (AP) for a given query and submitted run, is computed in the following way:

0. Let k be the maximum number of justifications assessed per relation asserted in the KB. For TAC 2017, it is expected that k=3 for KB submissions. (SF submissions will have k=1.) We will also require that for a given filler, at most one justification is returned per document (ResolveQueries will consider only the highest confidence justification for that document).

1. For each candidate filler (node) returned for the query, let the node confidence be the aggregate of the confidences of its justifications -- up to k justifications j1, j2,...jk per node, having confidence c1, c2,..ck. If the node has a parent (i.e., the filler is a hop_1 filler), its node confidence is the product of the node confidence of its parent entity, and the aggregate of the confidences of its justifications. There are many possible ways of aggregating justification-level confidence values to produce a node-level confidence value, and which aggregation function will depend largely on how confidence values are used in each use case. For TAC 2017, the confidence of a node with justifications having confidence (c1, c2,…,ck), assuming the confidences are sorting in decreasing order, will be a normalized weighted sum of the confidence values, weighted based on the rank of the justification:

(c1/1 + c2/2 +…. + ck/k)/(1 + ½ + … + 1/k).

2. Rank the candidate fillers by their node confidence.

3. Go down the list of nodes, and assign a value v to each node, 0 <= v <= 1; The value v is how correct this node is, and is found by matching this node with at most one equivalence class, and finding the number of justifications associated with this matched equivalence class. To compute the value v for the node:

The node f contains a set of triples (j, c, e), where j is a justification for this filler, e is the equivalence class assigned by LDC to this justification (0 if the justification is Wrong), and c is the confidence that the system associated with the justification. For each equivalence class, compute the value of the filler f, given the equivalence class. There are various alternatives for computing the value of f, depending on the use case. For 2017, the value of a node f is the fraction of known justification documents (for the matched equivalence class) that are found in the correct justifications for f.

Value(f | e) = (number of justification documents in f that have equivalence class e) / min(k , number of different known justification documents for equivalence class e)

Match the filler f to the equivalence class e with highest Value(f | e) , under the constraint that the equivalence class has not already been matched earlier (by a higher ranked filler). The value v of

[17] See *TAC KBP 2015 Slot Descriptions* and *TAC KBP 2015 Assessment Guidelines* for further information on how and when two slot fills are treated as equivalent.

the filler is zero if the filler cannot be matched to an equivalence class, either because all justifications in the filler are Wrong, or because all equivalence classes in the filler have already been matched to some other higher ranked filler(s). The value v measures the number of correct justifications documents that the KB provides for the matched equivalence class; v is a recall measure over justification documents.

4. We then compute AP as usual, except that a "Correct" item in the ranked list does not always get counted as "1", but has some value $0 <= v <= 1$. As is customary for AP, we sum over all retrieved nodes for the query and divide by N = the number of known equivalence classes ("relevant items") for the query. If the list has fewer than N item, we pad the list to make it of length N and treat the additional item as having value 0.

The above protocol assumes hop_0 and hop_1 fillers are evaluated together for a given query, but we will also compute AP for only hop_0 queries or only hop_1 queries. Note also that Step 3 penalizes a system for merging two nodes (since purity decreases, and hence the value decreases), or for returning some incorrect justifications along with the correct justifications. It also penalizes a system for splitting a correct node into multiple nodes, since each equivalence class can be matched to (and contribute value to) at most one candidate filler node.

As in 2016, each Cold Start evaluation query in 2017 may have more than one entry point, and the number of entry points may differ arbitrarily between Cold Start evaluation queries. The primary metric for the composite evaluation of 2017 Cold Start KB systems will be a macro-average score, Mean MAP (MMAP):

- MMAP: For a given query, compute AP for each entry point as outlined above. The MAP score for a query is the mean of the AP scores of each of its constituent entry points. The MMAP score for the system is the mean of its query-level MAP scores. The MMAP metric gives equal weight to each query, and (within each query) equal weight to each of its entry points.

Submissions

A four-week window from Thursday June 29 to Thursday July 27 will be available for downloading the TAC KBP 2017 Evaluation Source Corpus, producing Cold Start KB and SF system output, and submitting results. Systems should not be modified once the corpus has been downloaded. Starting Thursday, July 13, participants in the SF task may email NIST to request the SF evaluation queries, but teams participating in both the SF and CSKB tasks must submit all CSKB runs before requesting the SF evaluation queries from NIST.

For each task (SF and CSKB), a team may submit up to 5 runs for each of the following 4 language conditions:

1. Monolingual English: entity mentions, slot fills and provenances are extracted only from English documents. Evaluation queries will contain entry points only from English documents.
2. Monolingual Spanish: entity mentions, slot fills and provenances are extracted only from Spanish documents. Evaluation queries will contain entry points only from Spanish documents.

3. Monolingual Chinese: entity mentions, slot fills and provenances are extracted only from Chinese documents. Evaluation queries will contain entry points only from Chinese documents.
4. Cross-lingual: entity mentions, slot fills and provenances are extracted from any combination of English, Spanish, and Chinese documents. Evaluation queries will contain entry points from any of these languages, and the slot filler and justifications can come from a language different from the entry point. Because justification spans for a single justification must come from the same document, the cross-lingual nature of this evaluation condition is due to the exercise of cross-lingual EDL.

If a team submits a run involving more than one language under the Cross-lingual condition, it must also submit at least one run under the monolingual condition for each language involved (with a description of which monolingual run configurations were used for each cross-lingual run).

Submitted runs must be ranked (1-5). The run ID included in each team's submission file must be a concatenation of the team's TAC KBP 2017 team ID, the task (KB or SF), the language condition (ENG, CMN, SPA, or XLING), and a rank (1-5); thus "Acme_KB_XLING_1" would be the top-ranked run for the Acme team for the CSKB task under the cross-lingual condition.

The top-ranked submission must be made as a 'closed' system; in particular, it must not access the Web during the evaluation period. All submissions must obey the following external resource restrictions:

- Structured knowledge bases (e.g., Wikipedia infoboxes, DBPedia, Freebase) may not be used to directly fill slots or directly validate candidate slot fillers.
- Structured knowledge base entries for target entities may not be edited, either during, or after the evaluation.

In addition, because Cold Start focuses on the condition where the knowledge base is initially empty, we ask that each participating site submit at least one run that consults external entity knowledge bases only after entities and relations have been extracted from the document collection. Details about submission procedures will be communicated to the track mailing list. Tools to validate formats will be available on the TAC Web site (http://www.nist.gov/tac/2016/KBP/ColdStart/tools.html).

Appendix

Relation	Inverse(s)
per:children	per:parents
per:other_family	per:other_family
per:parents	per:children
per:siblings	per:siblings
per:spouse	per:spouse
per:employee_or_member_of	{org,gpe}:employees_or_members*
per:schools_attended	org:students*
per:city_of_birth	gpe:births_in_city*
per:stateorprovince_of_birth	gpe:births_in_stateorprovince*
per:country_of_birth	gpe:births_in_country*
per:cities_of_residence	gpe:residents_of_city*
per:statesorprovinces_of_residence	gpe:residents_of_stateorprovince
per:countries_of_residence	gpe:residents_of_country*
per:city_of_death	gpe:deaths_in_city*
per:stateorprovince_of_death	gpe:deaths_in_stateorprovince*
per:country_of_death	gpe:deaths_in_country*
org:shareholders	{per,org,gpe}:holds_shares_in*
org:founded_by	{per,org,gpe}:organizations_founded*
org:top_members_employees	per:top_member_employee_of*
{org,gpe}:member_of	org:members
org:members	{org,gpe}:member_of
org:parents	{org,gpe}:subsidiaries
org:subsidiaries	org:parents
org:city_of_headquarters	gpe:headquarters_in_city*
org:stateorprovince_of_headquarters	gpe:headquarters_in_stateorprovince*
org:country_of_headquarters	gpe:headquarters_in_country*

Table 2. Entity-valued SF slots. Slots with asterisks represent inverse relations that will need to be added by participants from previous years Slot Filling task (2014 and earlier). The type qualifier of each relation (*per*, *org* or *gpe*) is the type of its subject, while the type qualifier for its inverse is the type of its object. A set of types means that any of those types is acceptable for that slot. All submitted slot names must use only a single type specification.

per:alternate_names	org:alternate_names
per:date_of_birth	org:political_religious_affiliation
per:age	org:number_of_employees_members
per:origin	org:date_founded
per:date_of_death	org:date_dissolved
per:cause_of_death	org:website
per:title	
per:religion	
per:charges	

Table 3. String-valued SF slots.

Sentiment Predicates			
Subject	Predicate	Object	Inverse Predicate
PER,ORG,GPE, LOC,FAC	{per,org,gpe,loc,fac}:dislikes	PER,ORG,GPE, LOC,FAC	{per,org,gpe,loc,fac}:is_disliked_by
PER,ORG,GPE, LOC,FAC	{per,org,gpe,loc,fac}:is_disliked_by	PER,ORG,GPE, LOC,FAC	{per,org,gpe,loc,fac}:dislikes
PER,ORG,GPE, LOC,FAC	{per,org,gpe,loc,fac}:is_liked_by	PER,ORG,GPE, LOC,FAC	{per,org,gpe,loc,fac}:likes
PER,ORG,GPE, LOC,FAC	{per,org,gpe,loc,fac}:likes	PER,ORG,GPE, LOC,FAC	{per,org,gpe,loc,fac}:is_liked_by

Subject	Predicate	Object	Inverse Predicate
CONFLICT.ATTACK	conflict.attack:attacker	PER,ORG,GPE,STRING	{per,gpe,org}:conflict.attack_attacker
CONFLICT.ATTACK	conflict.attack:instrument	STRING	none
CONFLICT.ATTACK	conflict.attack:target	PER,ORG,GPE,FAC,STRING	{per,gpe,org,fac}:conflict.attack_target
CONFLICT.ATTACK	conflict.attack:time	STRING	none
CONFLICT.ATTACK	conflict.attack:place	GPE,LOC,FAC,STRING	{gpe,loc,fac}:conflict.attack_place
CONFLICT.DEMONSTRATE	conflict.demonstrate:entity	PER,ORG,STRING	{per,org}:conflict.demonstrate_entity
CONFLICT.DEMONSTRATE	conflict.demonstrate:time	STRING	none
CONFLICT.DEMONSTRATE	conflict.demonstrate:place	GPE,LOC,FAC,STRING	{gpe,loc,fac}:conflict.demonstrate_place
CONTACT.BROADCAST	contact.broadcast:audience	PER,ORG,GPE,STRING	{per,org,gpe}:contact.broadcast_audience
CONTACT.BROADCAST	contact.broadcast:entity	PER,ORG,GPE,STRING	{per,org,gpe}:contact.broadcast_entity
CONTACT.BROADCAST	contact.broadcast:time	STRING	none
CONTACT.BROADCAST	contact.broadcast:place	GPE,LOC,FAC,STRING	{gpe,loc,fac}:contact.broadcast_place
CONTACT.CONTACT	contact.contact:entity	PER,ORG,GPE,STRING	{per,org,gpe}:contact.contact_entity
CONTACT.CONTACT	contact.contact:time	STRING	none
CONTACT.CONTACT	contact.contact:place	GPE,LOC,FAC,STRING	{gpe,loc,fac}:contact.contact_place
CONTACT.CORRESPONDENCE	contact.correspondence:entity	PER,ORG,GPE,STRING	{per,org,gpe}:contact.correspondence_entity
CONTACT.CORRESPONDENCE	contact.correspondence:time	STRING	none
CONTACT.CORRESPONDENCE	contact.correspondence:place	GPE,LOC,FAC,STRING	{gpe,loc,fac}:contact.correspondence_place
CONTACT.MEET	contact.meet:entity	PER,ORG,GPE,STRING	{per,org,gpe}:contact.meet_entity
CONTACT.MEET	contact.meet:time	STRING	none
CONTACT.MEET	contact.meet:place	GPE,LOC,FAC,STRING	{gpe,loc,fac}:contact.meet_place
JUSTICE.ARREST-JAIL	justice.arrest-jail:agent	PER,ORG,GPE,STRING	{per,org,gpe}:justice.arrest-jail_agent

JUSTICE.ARREST-JAIL	justice.arrest-jail:crime	STRING	none
JUSTICE.ARREST-JAIL	justice.arrest-jail:person	PER,STRING	per:justice.arrest-jail_person
JUSTICE.ARREST-JAIL	justice.arrest-jail:time	STRING	none
JUSTICE.ARREST-JAIL	justice.arrest-jail:place	GPE,LOC,FAC,STRING	{gpe,loc,fac}:justice.arrest-jail_place
LIFE.DIE	life.die:agent	PER,ORG,GPE,STRING	{per,org,gpe}:life.die_agent
LIFE.DIE	life.die:instrument	STRING	none
LIFE.DIE	life.die:victim	PER,STRING	per:life.die_victim
LIFE.DIE	life.die:time	STRING	none
LIFE.DIE	life.die:place	GPE,LOC,FAC,STRING	{gpe,loc,fac}:life.die_place
LIFE.INJURE	life.injure:agent	PER,ORG,GPE,STRING	{per,org,gpe}:life.injure_agent
LIFE.INJURE	life.injure:instrument	STRING	none
LIFE.INJURE	life.injure:victim	PER,STRING	per:life.injure_victim
LIFE.INJURE	life.injure:time	STRING	none
LIFE.INJURE	life.injure:place	GPE,LOC,FAC,STRING	{gpe,loc,fac}:life.injure_place
MANUFACTURE.ARTIFACT	manufacture.artifact:agent	PER,ORG,GPE,STRING	{per,org,gpe}:manufacture.artifact_agent
MANUFACTURE.ARTIFACT	manufacture.artifact:artifact	FAC,STRING	fac:manufacture.artifact_artifact
MANUFACTURE.ARTIFACT	manufacture.artifact:instrument	STRING	none
MANUFACTURE.ARTIFACT	manufacture.artifact:time	STRING	none
MANUFACTURE.ARTIFACT	manufacture.artifact:place	GPE,LOC,FAC,STRING	gpe,loc,fac}:manufacture.artifact_place
MOVEMENT.TRANSPORT-ARTIFACT	movement.transport-artifact:agent	PER,ORG,GPE,STRING	{per,org,gpe}:movement.transport-artifact_agent
MOVEMENT.TRANSPORT-ARTIFACT	movement.transport-artifact:artifact	FAC,STRING	fac:movement.transport-artifact_artifact
MOVEMENT.TRANSPORT-ARTIFACT	movement.transport-artifact:destination	GPE,LOC,FAC,STRING	{gpe,loc,fac}:movement.transport-artifact_destination
MOVEMENT.TRANSPORT-ARTIFACT	movement.transport-artifact:instrument	STRING	none
MOVEMENT.TRANSPORT-ARTIFACT	movement.transport-artifact:origin	GPE,LOC,FAC,STRING	{gpe,loc,fac}:movement.transport-artifact_origin
MOVEMENT.TRANSPORT-ARTIFACT	movement.transport-artifact:time	STRING	none
MOVEMENT.TRANSPOR	movement.transport-person:agent	PER,ORG,GPE,STRING	{per,org,gpe}:movement.transport-

T-PERSON			person_agent
MOVEMENT.TRANSPORT-PERSON	movement.transport-person:destination	GPE,LOC,FAC,STRING	{gpe,loc,fac}:movement.transport-person_destination
MOVEMENT.TRANSPORT-PERSON	movement.transport-person:instrument	STRING	none
MOVEMENT.TRANSPORT-PERSON	movement.transport-person:origin	GPE,LOC,FAC,STRING	{gpe,loc,fac}:movement.transport-person_origin
MOVEMENT.TRANSPORT-PERSON	movement.transport-person:person	PER,STRING	per:movement.transport-person_person
MOVEMENT.TRANSPORT-PERSON	movement.transport-person:time	STRING	none
PERSONNEL.ELECT	personnel.elect:agent	PER,ORG,GPE,STRING	{per,org,gpe}:personnel.elect_agent
PERSONNEL.ELECT	personnel.elect:person	PER,STRING	per:personnel.elect_person
PERSONNEL.ELECT	personnel.elect:position	STRING	none
PERSONNEL.ELECT	personnel.elect:time	STRING	none
PERSONNEL.ELECT	personnel.elect:place	GPE,LOC,FAC,STRING	{gpe,loc,fac}:personnel.elect_place
PERSONNEL.END-POSITION	personnel.end-position:entity	ORG,GPE,STRING	{org,gpe}:personnel.end-position_entity
PERSONNEL.END-POSITION	personnel.end-position:person	PER,STRING	per:personnel.end-position_person
PERSONNEL.END-POSITION	personnel.end-position:position	STRING	none
PERSONNEL.END-POSITION	personnel.end-position:time	STRING	none
PERSONNEL.END-POSITION	personnel.end-position:place	GPE,LOC,FAC,STRING	{gpe,loc,fac}:personnel.end-position_place
PERSONNEL.START-POSITION	personnel.start-position:entity	ORG,GPE,STRING	{org,gpe}:personnel.start-position_entity
PERSONNEL.START-POSITION	personnel.start-position:person	PER,STRING	per:personnel.start-position_person
PERSONNEL.START-POSITION	personnel.start-position:position	STRING	none
PERSONNEL.START-POSITION	personnel.start-position:time	STRING	none
PERSONNEL.START-POSITION	personnel.start-position:place	GPE,LOC,FAC,STRING	{gpe,loc,fac}:personnel.start-position_place
TRANSACTION.TRANSACTION	transaction.transaction:beneficiary	PER,ORG,GPE,STRING	{per,org,gpe}:transaction.transaction_beneficiary
TRANSACTION.TRANSACTION	transaction.transaction:giver	PER,ORG,GPE,STRING	{per,org,gpe}:transaction.transaction_giver
TRANSACTION.TRANSACTION	transaction.transaction:recipient	PER,ORG,GPE,STRING	{per,org,gpe}:transaction.transaction_recipient
TRANSACTION.TRANSACTION	transaction.transaction:time	STRING	none

TRANSACTION.TRANSACTION	transaction.transaction:place	GPE,LOC,FAC,STRING	{gpe,loc,fac}:transaction.transaction_place
TRANSACTION.TRANSFER-MONEY	transaction.transfer-money:beneficiary	PER,ORG,GPE,STRING	{per,org,gpe}:transaction.transfer-money_beneficiary
TRANSACTION.TRANSFER-MONEY	transaction.transfer-money:giver	PER,ORG,GPE,STRING	{per,org,gpe}:transaction.transfer-money_giver
TRANSACTION.TRANSFER-MONEY	transaction.transfer-money:money	STRING	none
TRANSACTION.TRANSFER-MONEY	transaction.transfer-money:recipient	PER,ORG,GPE,STRING	{per,org,gpe}:transaction.transfer-money_recipient
TRANSACTION.TRANSFER-MONEY	transaction.transfer-money:time	STRING	none
TRANSACTION.TRANSFER-MONEY	transaction.transfer-money:place	GPE,LOC,FAC,STRING	{gpe,loc,fac}:transaction.transfer-money_place
TRANSACTION.TRANSFER-OWNERSHIP	transaction.transfer-ownership:beneficiary	PER,ORG,GPE,STRING	{per,org,gpe}:transaction.transfer-ownership_beneficiary
TRANSACTION.TRANSFER-OWNERSHIP	transaction.transfer-ownership:giver	PER,ORG,GPE,STRING	{per,org,gpe}:transaction.transfer-ownership_giver
TRANSACTION.TRANSFER-OWNERSHIP	transaction.transfer-ownership:recipient	PER,ORG,GPE,STRING	{per,org,gpe}:transaction.transfer-ownership_recipient
TRANSACTION.TRANSFER-OWNERSHIP	transaction.transfer-ownership:thing	FAC,ORG,STRING	{fac,org}:transaction.transfer-ownership_thing
TRANSACTION.TRANSFER-OWNERSHIP	transaction.transfer-ownership:time	STRING	none
TRANSACTION.TRANSFER-OWNERSHIP	transaction.transfer-ownership:place	GPE,LOC,FAC,STRING	{gpe,loc,fac}:transaction.transfer-ownership_place

Event Predicates (event as object)			
Subject	Predicate	Object	Inverse Predicate
PER,ORG,GPE	{per,org,gpe}:conflict.attack_attacker	CONFLICT.ATTACK	conflict.attack:attacker
GPE,LOC,FAC	{gpe,loc,fac}:conflict.attack_place	CONFLICT.ATTACK	conflict.attack:place
PER,ORG,GPE,FAC	{per,org,gpe,fac}:conflict.attack_target	CONFLICT.ATTACK	conflict.attack:target
PER,ORG	per,org:conflict.demonstrate_entity	CONFLICT.DEMONSTRATE	conflict.demonstrate:entity

GPE,LOC,FAC	{gpe,loc,fac}:conflict.demonstrate_place	CONFLICT.DEMONSTRATE	conflict.demonstrate:place
PER,ORG,GPE	{per,org,gpe}:contact.broadcast_audience	CONTACT.BROADCAST	contact.broadcast:audience
PER,ORG,GPE	{per,org,gpe}:contact.broadcast_entity	CONTACT.BROADCAST	contact.broadcast:entity
GPE,LOC,FAC	{gpe,loc,fac}:contact.broadcast_place	CONTACT.BROADCAST	contact.broadcast:place
PER,ORG,GPE	{per,org,gpe}:contact.contact_entity	CONTACT.CONTACT	contact.contact:entity
GPE,LOC,FAC	{gpe,loc,fac}:contact.contact_place	CONTACT.CONTACT	contact.contact:place
PER,ORG,GPE	{per,org,gpe}:contact.correspondence_entity	CONTACT.CORRESPONDENCE	contact.correspondence:entity
GPE,LOC,FAC	{gpe,loc,fac}:contact.correspondence_place	CONTACT.CORRESPONDENCE	contact.correspondence:place
PER,ORG,GPE	{per,org,gpe}:contact.meet_entity	CONTACT.MEET	contact.meet:entity
GPE,LOC,FAC	{gpe,loc,fac}:contact.meet_place	CONTACT.MEET	contact.meet:place
PER,ORG,GPE	{per,org,gpe}:justice.arrest-jail_agent	JUSTICE.ARREST-JAIL	justice.arrest-jail:agent
PER	per:justice.arrest-jail_person	JUSTICE.ARREST-JAIL	justice.arrest-jail:person
GPE,LOC,FAC	{gpe,loc,fac}:justice.arrest-jail_place	JUSTICE.ARREST-JAIL	justice.arrest-jail:place
PER,ORG,GPE	{per,org,gpe}:life.die_agent	LIFE.DIE	life.die:agent
GPE,LOC,FAC	{gpe,loc,fac}:life.die_place	LIFE.DIE	life.die:place
PER	per:life.die_victim	LIFE.DIE	life.die:victim
PER,ORG,GPE	{per,org,gpe}:life.injure_agent	LIFE.INJURE	life.injure:agent
GPE,LOC,FAC	{gpe,loc,fac}:life.injure_place	LIFE.INJURE	life.injure:place
PER	per:life.injure_victim	LIFE.INJURE	life.injure:victim
PER,ORG,GPE	{per,org,gpe}:manufacture.artifact_agent	MANUFACTURE.ARTIFACT	manufacture.artifact:agent
FAC	fac:manufacture.artifact_artifact	MANUFACTURE.ARTIFACT	manufacture.artifact:artifact
GPE,LOC,FAC	{gpe,loc,fac}:manufacture.artifact_place	MANUFACTURE.ARTIFACT	manufacture.artifact:place
PER,ORG,GPE	{per,org,gpe}:movement.transport-artifact_agent	MOVEMENT.TRANSPORT-ARTIFACT	movement.transport-artifact:agent
FAC	fac:movement.transport-artifact_artifact	MOVEMENT.TRANSPORT-ARTIFACT	movement.transport-artifact:artifact
GPE,LOC,FAC	{gpe,loc,fac}:movement.transport-artifact_destination	MOVEMENT.TRANSPORT-ARTIFACT	movement.transport-artifact:destination
GPE,LOC,FAC	{gpe,loc,fac}:movement.transport-artifact_origin	MOVEMENT.TRANSPORT-ARTIFACT	movement.transport-artifact:origin
PER,ORG,GPE	{per,org,gpe}:movement.transport-person_agent	MOVEMENT.TRANSPORT-PERSON	movement.transport-person:agent
GPE,LOC,FAC	{gpe,loc,fac}:movement.transport-		movement.transport-

	person_destination	MOVEMENT.TRANSPORT-PERSON	person:destination
GPE,LOC,FAC	{gpe,loc,fac}:movement.transport-person_origin	MOVEMENT.TRANSPORT-PERSON	movement.transport-person:origin
PER	per:movement.transport-person_person	MOVEMENT.TRANSPORT-PERSON	movement.transport-person:person
PER,ORG,GPE	{per,org,gpe}:personnel.elect_agent	PERSONNEL.ELECT	personnel.elect:agent
PER	per:personnel.elect_person	PERSONNEL.ELECT	personnel.elect:person
GPE,LOC,FAC	{gpe,loc,fac}:personnel.elect_place	PERSONNEL.ELECT	personnel.elect:place
ORG,GPE	{org,gpe}:personnel.end-position_entity	PERSONNEL.END-POSITION	personnel.end-position:entity
PER	per:personnel.end-position_person	PERSONNEL.END-POSITION	personnel.end-position:person
GPE,LOC,FAC	{gpe,loc,fac}:personnel.end-position_place	PERSONNEL.END-POSITION	personnel.end-position:place
ORG,GPE	{org,gpe}:personnel.start-position_entity	PERSONNEL.START-POSITION	personnel.start-position:entity
PER	per:personnel.start-position_person	PERSONNEL.START-POSITION	personnel.start-position:person
GPE,LOC,FAC	{gpe,loc,fac}:personnel.start-position_place	PERSONNEL.START-POSITION	personnel.start-position:place
PER,ORG,GPE	{per,org,gpe}:transaction.transaction_beneficiary	TRANSACTION.TRANSACTION	transaction.transaction:beneficiary
PER,ORG,GPE	{per,org,gpe}:transaction.transaction_giver	TRANSACTION.TRANSACTION	transaction.transaction:giver
GPE,LOC,FAC	{gpe,loc,fac}:transaction.transaction_place	TRANSACTION.TRANSACTION	transaction.transaction:place
PER,ORG,GPE	{per,org,gpe}:transaction.transaction_recipient	TRANSACTION.TRANSACTION	transaction.transaction:recipient
PER,ORG,GPE	{per,org,gpe}:transaction.transfer-money_beneficiary	TRANSACTION.TRANSFER-MONEY	transaction.transfer-money:beneficiary
PER,ORG,GPE	{per,org,gpe}:transaction.transfer-money_giver	TRANSACTION.TRANSFER-MONEY	transaction.transfer-money:giver
GPE,LOC,FAC	{gpe,loc,fac}:transaction.transfer-money_place	TRANSACTION.TRANSFER-MONEY	transaction.transfer-money:place
PER,ORG,GPE	{per,org,gpe}:transaction.transfer-money_recipient	TRANSACTION.TRANSFER-MONEY	transaction.transfer-money:recipient
PER,ORG,GPE	{per,org,gpe}:transaction.transfer-ownership_beneficiary	TRANSACTION.TRANSFER-OWNERSHIP	transaction.transfer-ownership:beneficiary
PER,ORG,GPE	{per,org,gpe}:transaction.transfer-ownership_giver	TRANSACTION.TRANSFER-OWNERSHIP	transaction.transfer-ownership:giver
GPE,LOC,FAC	{gpe,loc,fac}:transaction.transfer-		transaction.transfer-

	ownership_place	TRANSACTION.TRANSFER-OWNERSHIP	ownership:place
PER,ORG,GPE	{per,org,gpe}:transaction.transfer-ownership_recipient	TRANSACTION.TRANSFER-OWNERSHIP	transaction.transfer-ownership:recipient
FAC,ORG	{fac,org}:transaction.transfer-ownership_thing	TRANSACTION.TRANSFER-OWNERSHIP	transaction.transfer-ownership:thing

Change History

- Version 1.0
    - Original draft version, based on the 2016 specification