TAC Streaming Multimedia KBP for AIDA

Phase 3 Evaluation Plan V1.4

Last Updated: December 15, 2022

Revision History

V1.0: October 26, 2021

- Initial release

V1.1: December 10, 2021

- Section 7.1: Updated Submission procedure to allow TA2 and TA3 dockers to be submitted subsequent to submission of system output that is evaluated for Task 2 and Task 3.
- Added a dry run period for all evaluation tasks (replacing the Task 3 quizlets), and delayed the evaluation period for all evaluation tasks.
- Modified representation of edge KEs so that the edge label does not include the type of the event or relation that is the subject of an AIF argument assertion, but consists of only the argument role (e.g., A0_pag_agent_creator).
- Section 2.4.1: Removed specification for how to represent relations that are true over multiple disjoint time intervals.
- Section 2.4.1: Replaced the M36 requirement that the members and prototype of the same `aida:SameAsCluster` must all have the same top-level class (EntityType, RelationType, or EventType), with the requirement that each member of a single `aida:SameAsCluster` must be an `aida:Entity` if the prototype is an `aida:Entity`, an `aida:Relation` if the prototype is an `aida:Relation`, and an `aida:Event` if the prototype is an `aida:Event`.
- Section 2.4.1: Specified restrictions for the number of values allowed in each field of the claim frame.
- Section 2.4.1: In the graph of Associated KEs for a claim frame, require that [the prototype of] each event or relation cluster must have exactly one type (one type assertion).
- Section 4.2: Redefined RoleSim(gr, sr), the similarity between argument roles.
- Section 4.5.1: Redefined the alignment of relation clusters for the Task 1 Temporal Metric
- Section 4.6: In a TRF triple, allow the event/relation to have more than one type (because the argument role label R no longer includes the event/relation type).
- Section 4.6: Removed the requirement that the event or relation type on the predicate of the argument assertion is similar to one of the types for the subject aida:SameAsCluster, because the predicate no longer includes the event or relation type.
- Added Claim Medium to claim frames.
- Don't require TA3 claim frames to include Claimer Provenance, Claim Location Provenance, or Claim Medium Provenance.
- Section 6.3: Require that in order for a claim frame field value to be Correct, it must have an informative descriptor (in addition to being justified in the associated document).

V1.2: June 23, 2022

- Section 2.4.1: Each aida:Claim may have at most 3 aida:claimerAffiliation.

- Section 2.4.1: Require that all justifications for the graph of Associated KEs for a claim frame must come from the same aida:sourceDocument as the claim frame.
- Section 4.5: Corrected typo for a situation that would indicate an inconsistent temporal annotation. The previous textual description was correct but the erroneous inequality "T4>T1" has been corrected:
    - T1 > T4 : aggregate "end before date" is earlier than aggregate "start after date"
- Section 5.2: Increased the number of TA2 clusters to pool to be C=3 clusters per query.
- Section 6.3: Fleshed out the description of TA3 evaluation, including adding weights to each field and defining the gain function for NDCG.
- Section 6.3.2: Added F1 metric to measure the number of unique combinations of the most important fields in each ranked list.

V1.3: June 25, 2022

- Section 5.2: Tightened up the description of TA2 evaluation by defining the notion of a pseudo-query.
- Section 5.2: Changed the overall score for TA2 to be mean AP over pseudo-queries, rather than the mean of the MAP for each query (where the MAP for a query is the mean AP over its pseudo-queries).

V1.4: December 15, 2022

- Section 2.1 Specified the version of the DWD overlay used for evaluation (xpo_v5.1a.json).
- Section 2.4.1: Noted that a single edge in a knowledge graph may have multiple different role labels.
- Section 4: Added more details about the Task 1 evaluation.
- Section 4.3: Added a second negation metric (NegationMetricV2), which computes P/R/F1 for retrieval of negated mentions from aligned mentions of aligned clusters.
- Section 4.5.1: Revised the alignment of relations to use the FMBM alignment for the Temporal evaluation.
- Section 4.6: Revised the definition of a TRF triple for the Event/Relation Argument Extraction evaluation.
- Section 4.7: Revised EdgeScore to use RolesPrecision and the normalized ClusterSim functions

1 Introduction

In scenarios such as natural disasters or international conflicts, analysts and the public are often confronted with a variety of information coming through multiple media sources. There is a need for technologies to analyze and extract knowledge from multilingual multimedia sources to develop and maintain an understanding of events, situations, and trends around the world, in order to respond to the situations.

The goal of DARPA's Active Interpretation of Disparate Alternatives (AIDA) Program is to develop a semantic engine that generates explicit alternative interpretations of events, situations, and trends from a variety of unstructured sources, for use in noisy, conflicting, and potentially deceptive information environments. This engine must be capable of automatically extracting knowledge elements (KE) from multiple languages and media sources to produce a knowledge graph (KG), aggregating information derived from those sources, and generating and exploring multiple hypotheses about the events, situations, and trends of interest. This engine must establish confidence measures for the derived knowledge and hypotheses, based on the accuracy of the analysis and the semantic coherence of each hypothesis. In addition, the engine must be able to communicate with its user to reveal the generated hypotheses and to allow the user to alter the hypotheses or to suggest new ones.

This document describes the specifications of the evaluation conducted by NIST to assess the performance of systems that have been developed in support of AIDA program goals. The streaming multimedia KBP track asks systems to extract knowledge elements from a stream of heterogeneous documents containing multilingual multimedia sources including text and image files; aggregate the knowledge elements from multiple documents (maintaining multiple interpretations and confidence values for KEs extracted or inferred from the documents); and develop claim frames or semantically coherent hypotheses, each of which represents an interpretation of the document stream.

Participation in NIST's Streaming Multimedia KBP (SM-KBP) evaluation is required for all DARPA AIDA performers responsible for the relevant technology areas in AIDA. Task 1 is also open to all researchers who find the evaluation tasks of interest. There is no cost to participate. Participants are encouraged to attend a post-evaluation workshop at their own expense to present and discuss their systems and results. Information and updates about the tasks and evaluation will be posted to the NIST SM-KBP website (TBA).

TAC is running the SM-KBP track for three evaluation cycles, corresponding to the three phases of the AIDA program:

- Phase 1 Evaluation
  - Evaluation windows June-August 2019
  - Workshop at TAC/TRECVID in Gaithersburg, MD in November 2019
- Phase 2 Evaluation
  - Evaluation windows September-October 2020
  - Workshop at TAC on February 22-23, 2021
- Phase 3 Evaluation

- Evaluation windows in 2022
- Workshop at TAC on February 3, 2023

1.1 Evaluation Tasks

Participants will be evaluated on a set of topics for a scenario that contains conflicting information. The scenario for the Phase 3 tasks is the COVID-19 Global Pandemic, with data sources in English, Spanish, and Russian.

Three tasks are evaluated:

- Task 1 (TA1): Extraction of entity, relation, and event (ERE) KEs, and arguments and temporal information for events and relations, from a stream of multilingual multimedia documents, to produce a document-level knowledge graph for each document. TA1 must extract and cluster ***all*** mentions of entities, relations, and events in the document and must include the negation status of mentions of events, relations, and event arguments.
- Task 2 (TA2): Construction of a knowledge graph by aggregating and linking document-level knowledge graphs produced by TA1; TA2 must also link an entity KE to a reference knowledge base (KB) if such an entity already exists in the reference KB.
- Task 3 (TA3): Generation of ranked lists of claim frames about particular topics from knowledge graphs produced by TA1 and TA2.

AIDA performers are required to participate in the tasks as outlined by their Statement of Work. Task 1 and Task 2 are standalone evaluations that measure general multilingual multimedia extraction (Task 1) and cross-document entity coreference (Task 2). Additionally, TA1 and TA2 must participate in Task 3 by performing extraction and cross-document aggregation specifically with the goal of helping TA3 generate claim frames about particular topics that the user has issued as queries to TA3.

Open participants (non-AIDA performers) may participate only in Task 1.

TA1, TA2, and TA3 are structured in a pipeline, such that all TAs are given a reference KB of entities, a stream of raw documents, and (in Task 3) some user-issued queries. TA1 must output a stream of document-level knowledge graphs (KGs); TA2 is given the output of TA1 and must output a corpus-level knowledge graph; and TA3 is given the output of TA1 and TA2 and must output a set of claim frames in response to the user queries. TA1 and TA2 may output different knowledge graphs depending on whether those knowledge graphs are intended for the standalone Task 1 and Task 2 evaluations, or whether they are intended for Task 3.

Unlike previous phases of AIDA, Phase 3 does not have any membrane restricting what information can be shared between TA1, TA2, and TA3. In Phase 3, raw source documents that had previously been given only to TA1 are now also accessible to TA2 and TA3 for all three tasks. For Task 3, TA1 and TA2 are also given access to the queries that the user issues to TA3, so that they can perform extraction and cross-document aggregation that are targeted towards each user query.

AIDA performers must submit their systems as well as system output. All knowledge graphs will be represented in the AIDA Interchange Format (AIF).

TA1 system output will be evaluated by comparison against gold standard annotations on a set of documents that have been exhaustively annotated for entities, relations, and events of selected types.

TA2 and TA3 will be evaluated by assessment of system output. Although all systems should be able to process data in streaming mode, TA2 and TA3 output will be evaluated at only one timepoint (at the end of the data stream). Each TA2 knowledge graph will be queried to determine how well it is able to corefer mentions of the same entity across different documents. TA3 claim frames will be evaluated in their entirety (subject to quality constraints) and will be evaluated for how well the ranked list of claim frames returned for each user query overlaps with the set of reference claim frames for the query without adding redundant, uninformative, or otherwise spurious claim frames.

Table 1: Input and output for TA1, TA2, and TA3

System	Input	Output
TA1	• Stream of multilingual multimedia documents; • Evaluation reference KB of entities; • User-issued queries (for Task 3)	A knowledge graph for each document, consisting of KEs found in the document
TA2	• Stream of multilingual multimedia documents; • Evaluation reference KB of entities; • Stream of document-level knowledge graphs from TA1; • User-issued queries (for Task 3)	A knowledge graph aggregating document-level knowledge graphs into a single corpus-level knowledge graph, with links to entities in the reference KB when applicable
TA3	• Stream of multilingual multimedia documents; • Evaluation reference KB of entities; • Stream of document-level knowledge graphs from TA1; • Knowledge graph from	Ranked list of claim frames for each query

	TA2; • User-issued queries (for Task 3)	

2 Knowledge Graphs

A **knowledge graph (KG**) represents all knowledge, whether it comes from the document stream or some shared background knowledge, or via insertion of knowledge by a human user. A **knowledge element (KE)** is a node or edge in a knowledge graph. A node in the knowledge graph represents an entity, relation, or event (ERE) and will sometimes be referred to as an entity KE, relation KE, or event KE. An edge in the knowledge graph connects an event KE or relation KE to the KE representing one of the arguments of the event or relation (the argument may be an entity, relation, or event); the edge is labeled by the role(s) that the argument has in the event or relation.

Each entity/relation/event KE in the knowledge graph has attributes associated with it, including the entity/relation/event type (e.g., "Q215627") and the mentions (a.k.a. justifications) of the entity, relation, or event in the source corpus. Similarly, each edge in the knowledge graph has attributes associated with it, including the edge label denoting the role of an event or relation argument, and justifications for the edge in the source corpus. Entity/relation/event node types (e.g., "Q215627") and edge labels (e.g., "A0_pag_agent_creator") are defined in the DARPA Wikidata (DWD).

When an entity makes a claim in a document, there are many ways that the claim can be expressed in the document, yielding many different document-level knowledge graphs with different types of entities, relations, and events that express essentially the same claim. A claim frame is a special kind of knowledge graph that not only has the ERE that express the claim, but also summarizes those ERE in a more human-readable format by

1) mapping the semantics of the KEs expressing *what* is being claimed, to a topic, subtopic, and populated claim template from a fixed inventory of topics, subtopics, and claim templates, and
2) filling in fields that explicitly identify additional information about the claimer and claiming event.

Knowledge graphs in SM-KBP, including partially or fully filled claim frames, must be expressed as reified RDF triples, as defined by the **AIDA Interchange Format (AIF).** Standard AIF defines the format that should be use by participants to communicate between Task 1, Task 2, and Task 3 systems. Additionally, NIST defines further restrictions on AIF, called **NIST restricted AIF**, which participants should follow when producing KGs that will be submitted to NIST for evaluation. For each task, participants must output their KGs in NIST-restricted AIF (which NIST will query and evaluate); TA1 and TA2 may output an additional version of their KGs in standard AIF (to communicate with other TAs within the pipeline). It is the responsibility of participants within the same pipeline to communicate and agree among themselves about the semantics of the KG in standard AIF.

2.1 Ontology

Each entity, relation, and event node or edge in a knowledge graph must be labeled with one or more types from an ontology of entities, relations, and events. The ontology for Phase 3 is the **DARPA Wikidata (DWD)**, which is an enhanced version of Wikidata that defines entities, relations, events, and event/relation argument roles. Entity, relation, and event KEs in the submitted knowledge graphs will be limited to the types specified in DWD, and edge KEs in the submitted knowledge graph must be labeled with argument role labels defined in DWD. Wikidata is rich in Qnodes for entity classes and instances but has an impoverished representation of event and relation classes. Therefore, DWD adds an overlay of event and relation types to Wikidata, including argument roles and selectional preferences for those arguments. The argument roles have both a long name (consisting of a Propbank argument role, Propbank function tag, VerbNet argument role, and Description from Propbank, e.g., "A0_pag_agent_robber") and a short name (e.g., "A0_pag") consisting of just the first two parts of the long name.

DWD consists of:

1) A subset of a February 15, 2021 snapshot of Wikidata (including many Qnodes for entity classes that are not in the DWD overlay defined below).
2) A DWD overlay (xpo_v5.1a.json) released on July 20, 2022 by the DARPA Cross-Program Ontology (XPO) group consisting of:
    a. A manual overlay that is primarily a mapping from some Wikidata Qnodes and Pnodes to LDC ontology types. This overlay includes:
        i. A manual mapping from some Qnodes and Pnodes in DWD to the entity types, relation types, and event types in the LDC ontology. Each LDC type is mapped to at least one Qnode or Pnode, and each Qnode or Pnode can be mapped to zero or more LDC types.
        ii. Argument roles and selectional preferences for the event types and relation types in DWD that have been manually mapped to event types and relation types in the LDC ontology.
    b. An extensive overlay of semi-automatically defined event types and relation types that are represented as Wikidata Qnodes or Pnodes. This overlay includes:
        i. Argument roles and selectional preferences for the semi-automatically defined event types and relation types

DWD is expected to provide coverage of all the entity, relation, and event semantic categories needed for sharing information between TA1, TA2, and TA3.

The DWD overlay has mappings to Propbank, VerbNet, etc., to allow participating systems to leverage existing annotations and models trained on these resources. Additional information about the DWD overlay can be found at: https://github.com/e-spaulding/xpo .

DWD V2 (based on a February 15, 2021 snapshot of Wikidata) and xpo_v5.1a.json can both be downloaded from the SM-KBP 2022 data page: https://tac.nist.gov/2022/KBP/SM-KBP/data.html

2.2 Open Class Properties

In addition to the fixed set of types defined in DWD, nodes in the knowledge graph may have additional properties whose values are extracted or derived from the source documents. In particular, an entity node may have HasName, NumericValue, or TextValue properties, while an event or relation node may have an ldcTime property.

The HasName, NumericValue, and TextValue properties allow a name or other formulaic string to be associated with a particular entity in a knowledge graph in a way that's readily accessible to TA1/TA2/TA3. Each string is limited to 256 UTF-8 characters. Because the membrane has been removed in Phase 3, there is no restriction on which entity types are allowed to have these properties:

- **HasName**
  - Subject: An `aida:Entity`
  - Object: a string that must be a name for an entity. Each entity is allowed to have multiple HasName properties, one for each distinct name that is observed.

- **NumericValue**
  - Subject: (none)
  - Object: a numeric value

- **TextValue**
  - Subject: An `aida:Entity`
  - Object: a string

Temporal properties for events and relations should also be included in the knowledge graph when such information is available in the source data:

- **LDCTime**
  - Subject: An `aida:Event` or `aida:Relation`
  - Object: dates defining constraints on the start and end dates for the event or relation

These properties are intended to provide a way for TA1 to communicate limited string-valued properties to TA2 and TA3 to assist with coreference, etc. Additionally, evaluation queries will target some of the properties, such as event or relation dates and entity names.

2.3 Justifications

Each node or edge in a knowledge graph must contain one or more justifications. A **justification** for a KE is a single span that shows where the KE is asserted. A justification may come from the data stream, shared background knowledge, or human user. A justification for an entity node, relation node, or event node is a single span which will also be called a **mention** for that KE. A justification for an edge may contain multiple (one or two) spans from the same document,

possibly from different modalities, as needed to establish the connection between the relation or event and its argument.[1]

An ***informative mention*** (a.k.a. ***informative justification***) for an entity, relation, or event KE is defined to be a representative mention of the KE. If it is from text, the informative mention should be a name (preferred) or nominal mention rather than a pronominal mention; if it is from an image, the informative mention should (preferably) show a clear unobstructed image of the KE.

Given a source document ID and document element ID, a **span** is represented differently depending on the modality of the document element from which it came:

- TextJustification includes begin character offset and end character offset (defining a contiguous text span), and is used as justification for an entity, relation, event, or argument assertion
- ImageJustification includes one bounding box, and is used as justification for an entity, relation, event, or argument assertion

For image spans:

- bounding_box provides further localization of an entity inside an image file. A bounding box is needed for further localization of the justification if, for example, the KE is a single person but the frame contains multiple people. A bounding box for an image that includes the whole image is represented with <topleftx>0</topleftx>, <toplefty>0</toplefty>, <bottomrightx>image_width-1</bottomrightx>, <bottomrighty>image_height-1</bottomrighty>.

2.4 Representation in AIF

The **AIDA Interchange Format (AIF)** defines the format for a knowledge graph. Aside from external input that is available to all TAs (i.e., source documents, DWD ontology and reference KB, user-issued queries for Task 3), all inter-TA communication between TA1, TA2, and TA3 must be via knowledge graphs expressed as reified RDF triples that conform to the AIF.[2]

AIF allows a KE to contain **private data** that can be communicated to downstream TAs. Private data may include a variety of information that may be useful to downstream TAs and that are not standardized in AIF, including embeddings and similarity scores between mentions.

---

[1] Because the TA1 gold standard annotation is providing only a single span for each argument assertion justification (namely, a mention of the event or relation, in some local context that justifies the argument assertion), TA1 performers should also provide an argument assertion justification that includes an event or relation mention that is in some local context that justifies the argument assertion, in order to support evaluation by comparison against the gold standard.

[2] Note that this document uses the term "node" to refer to an entity KE, relation KE, or event KE in a knowledge graph (as described above) or to a node in LDC's linking KB (i.e., an entity, relation, or event that has a kb_id entry in the kb_linking.tab table of LDC's annotation packages). The term "node" in this document is ***never*** used to refer to an IRI, a literal, or a blank node in a general RDF graph; this document talks about nodes and edges at a higher level of abstraction, where a single node KE (for an entity, relation, or event KE) or an edge KE in a knowledge graph is representing by *multiple* RDF statements that together define the KE and its attributes, including type and justifications.

2.4.1 Restricted AIF

For purposes of evaluation, NIST defines further restrictions on AIF, called **NIST restricted AIF,** which imposes additional syntactic restrictions as well as semantics to AIF.

A single real-world entity, event or relation (an entity KE, relation KE or event KE in a knowledge graph) is represented in NIST-restricted AIF by an `aida:SameAsCluster` that can contain multiple members, where each member is an `aida:Entity, aida:Relation` or `aida:Event`. An AIF cluster for an entity has members (`aida:Entity`) that collectively contain the mentions referring to the same real-world entity; the same is true with events and relation clusters, though the definition of coreference may be fuzzier than for entities. For entities, relations, and events, NIST follows LDC's determination of coreference. Mentions that are linked to the same global kb_id in the kb_linking.tab file of LDC's gold standard annotations provide some examples of within-document coreference. For Task 1 evaluation, TA1 should produce AIF clusters that can be mapped one-to-one with LDC's qnode_kb_id_identity.

Each `aida:SameAsCluster` must have exactly one prototype `aida:Entity`, `aida:Relation`, or `aida:Event`, which should aggregate important properties of the real-world entity, relation, or event, including arguments. The prototype may be a member of the `aida:SameAsCluster` but need not be; the prototype can be a composite object aggregating information from multiple mentions of the entity, relation, or event. The prototype encapsulates information for each KE that an end user might want to see, and is the starting point for NIST evaluation queries that probe the AIF graph. With the exception of all mentions and mention types for each node KE, the HasName property for entities, and the Negated attribute for event and relation mentions, all information needed for evaluation should be asserted directly on the prototypes in NIST-restricted AIF:

- The set of type(s) for the entity KE (or relation KE or event KE) in a TA1 or TA2 graph is the union of the set of types asserted for the *members* of the `aida:SameAsCluster` representing the KE.
- The set of type(s) for the entity KE (or relation KE or event KE) in a TA3 graph is the union of the set of types asserted for the *prototype* of the `aida:SameAsCluster` representing the KE.
- The set of mentions of the entity KE (or relation KE or event KE) is the union of the set of justifications for the type assertions for each `aida:Entity` (or `aida:Relation` or `aida:Event`) that is a *member* of the `aida:SameAsCluster` representing the KE.[3]
    - The set of negated mentions of a relation KE or event KE is the union of the set of justifications for the type assertions for each `aida:Relation` or `aida:Event` that has the Negated attribute and that is a *member* of the `aida:SameAsCluster` representing the KE.
- The HasName properties for an entity KE are the properties asserted on the *members* of the `aida:SameAsCluster` representing the KE

[3] Although it is possible to require that all mentions for the entity, relation, or event must be collected and made justifications of the prototype, such a representation would be inefficient, as the mentions can be accessed readily from the members, without loss of information.

- The NumericValue, and TextValue properties for an entity KE are the properties asserted on the prototype
- The handle of an entity KE (required for TA3) is the handle on the prototype
- The temporal property of an event or relation KE is the temporal property asserted on the prototype
- The informative justifications for the entity, relation, or event KE are the informative justifications asserted on the prototype
- An edge between an event (or relation) KE and an argument KE exists if and only if its representation in NIST-restricted AIF has an argument assertion between the *prototype* of the `aida:SameAsCluster` representing the event (or relation) KE and the *prototype* of the `aida:SameAsCluster` representing the argument KE.
    - The set of label(s) for the edge is the set of `rdf:predicate` (e.g., `"A0_pag_causer_doctor"^^xsd:string`) in these AIF argument assertions.
    - The set of justifications for the edge is the union of the set of justifications for these AIF argument assertions.
    - A labeled edge between an event (or relation) KE and an argument KE is negated if and only if the argument assertion between the *prototype* of the `aida:SameAsCluster` representing the event (or relation) KE and the *prototype* of the `aida:SameAsCluster` representing the argument KE has the Negated attribute.
    - The number of edges in the graph is the number of unique (subject, object) pairs such that there is an argument assertion between the subject and the object (i.e., multiple different role labels may be asserted for the same edge).

NIST will probe TA1/TA2/TA3 output knowledge graphs using SPARQL queries that assume the following NIST restricted AIF for knowledge graphs; all teams should ensure that their AIF knowledge graphs conform to these requirements, which adds constraints beyond what is generally applicable in AIF.[4]

1. An entity, event, or relation KE node must be represented by an AIF cluster (`aida:SameAsCluster`), even if there is only one member of the cluster.
2. Each entity/relation/event AIF cluster must have an IRI, which NIST will interpret to be a unique ID for the entity, relation, or event KE.
3. An entity/relation/event AIF cluster must **not** be hierarchical; rather, the KE node must be represented as a single-level AIF cluster whose members are not clusters themselves.
4. Each entity/relation/event AIF cluster must have exactly one prototype, which must be the prototype of only one `aida:SameAsCluster`.
5. Each KE node has one metatype (entity, or relation, or event), which is determined by the prototype of the `aida:SameAsCluster`. The KE node is interpreted to be an entity if the

[4] In general, the AIF graph that is produce by TA1/TA2 is allowed to contain many things that will never be accessed by any of NIST's SPARQL queries; in the specification of restricted AIF, NIST is pointing out how information should be represented in the submitted AIF graphs so as to allow NIST's SPARQL queries to access the information that is needed and expected for the evaluation.

prototype is an `aida:Entity`; an event if the prototype is an `aida:Event`; and a relation if the prototype is an `aida:Relation`.[5]

6. Each member of a single `aida:SameAsCluster` must be an `aida:Entity` if the prototype is an `aida:Entity`, an `aida:Relation` if the prototype is an `aida:Relation`, and an `aida:Event` if the prototype is an `aida:Event`.
7. In order to represent a richer set of mention-level properties, NIST recommends that each mention of an entity KE (or relation KE or event KE) be represented with exactly one `aida:Entity` (or `aida:Relation` or `aida:Event`) that is a *member* of the `aida:SameAsCluster` representing the KE.
8. For purposes of evaluation, the mentions of an entity, relation, or event shall be the union of the set of justifications for the type assertions for *members* of the `aida:SameAsCluster` cluster representing the entity, relation, or event.
9. For purposes of TA1 evaluation, the set of negated mentions of a relation or event shall be the union of the set of justifications for the type assertions for each `aida:Relation` or `aida:Event` that has the Negated attribute and that is a *member* of the `aida:SameAsCluster` representing the relation or event.
10. For purposes of TA3 evaluation, an event or relation is negated if and only if the *prototype* of the `aida:SameAsCluster` representing the event (or relation) has the Negated attribute.
11. A type assertion is a statement whose `rdf:predicate` is `rdf:type`.
    a. For purposes of TA1 and TA2 evaluation, the type(s) of an entity, relation, or event is the union of the set of types asserted on the *members* of the `aida:SameAsCluster` representing the entity, relation, or event.
    b. For purposes of TA3 evaluation, the type(s) of an entity, relation, or event is the union of the set of types asserted on the *prototype* of the `aida:SameAsCluster` representing the entity, relation, or event.
12. A KE edge must be represented by one or more argument assertions. An argument assertion has an `rdf:predicate` that is an event argument role or relation argument role (e.g., `"A0_pag_causer_doctor"^^xsd:string`); the `rdf:subject` of an argument assertion should be an `aida:Event` or `aida:Relation`, while the `rdf:object` should be an `aida:Entity`, `aida:Event`, or `aida:Relation`. The KE edge is a pair consisting of the `rdf:subject` and `rdf:object` of the argument assertion. The labeled edge is a triple consisting of the `rdf:subject`, `rdf:predicate`, and `rdf:object` of the argument assertion.
    a. For purposes of evaluation, NIST will ignore an argument assertion unless both the `rdf:subject` and `rdf:object` is a prototype of an `aida:SameAsCluster`.
    b. For purposes of evaluation, a labeled edge between an event (or relation) and an argument is negated if and only if the argument assertion between the *prototype* of the `aida:SameAsCluster` representing the event (or relation) and the *prototype* of the `aida:SameAsCluster` representing the argument has the Negated attribute.
13. A justification for an AIF argument assertion must have either one or two justification spans and must be represented by `aida:CompoundJustification`, even if only one span

[5] This definition for the metatype for the KE node ignores the DWD types that are asserted for the KE. Therefore, the KE is interpreted to be an entity if the prototype is an `aida:Entity`, even if there is a type assertion saying that the prototype (or one of the members) of the `aida:SameAsCluster` has a DWD event type.

is provided. (If an argument assertion in a TA1 or TA2 AIF graph does not have any justification spans, it will be ignored by NIST for the purposes of the TA1 and TA2 evaluation.)
    a. No more than two spans are allowed for a single `aida:CompoundJustification`
    b. The spans in an `aida:CompoundJustification` must come from the same (parent) document, and `aida:CompoundJustification` must contain at least one span.
    c. The `aida:CompoundJustification` must be used only for justifications of argument assertions, and not for justifications for entities, events, or relation KEs.
14. Each image justification must include an explicit bounding box.
15. Each confidence value must be greater than 0 and less than or equal to 1.
16. TA2 must link entities to the evaluation reference KB; `aida:link` with confidence must be used to assert that an `aida:Entity` can be coreferenced with an entity in the evaluation reference KB.
    a. `aida:link` must have one or more `aida:linkAssertion`; each `aida:linkAssertion` must have exactly one `aida:linkTarget` and exactly one `aida:confidence`
    b. TA2 queries will look at the `aida:link` on the prototype of the `aida:SameAsCluster` to determine which cluster (entity KE) is asserted to be the same as the query entry point.
17. AIF will allow each `aida:Entity`, `aida:Event`, or `aida:Relation` to specify up to one informative mention per document.
    a. For TA1 and TA2, the prototype of each entity KE (cluster) must have exactly one informative mention per document if and only if the entity is mentioned in the document.
    b. TA1, TA2, and TA3 must have exactly one informative mention for each `aida:Entity`, `aida:Event`, and `aida:Relation` prototype that is an object of an AIF argument assertion, for each document that provides a justification for the argument assertion. For a given justification for an AIF argument assertion in a particular document, assessors will look at both the justification and the informative mention of the object in that document, to determine whether the AIF argument assertion is correct. If no informative mention is provided for the object for the document then, for the purposes of evaluation, it will be as if the justification for the argument assertion from that document did not exist in the knowledge graph.
18. Each justification span must have exactly one `aida:source` (containing the document element ID) and one `aida:sourceDocument` (containing the document ID).
19. Each `aida:hasName`, `aida:textValue`, and `aida:numericValue` string is limited to 256 UTF-8 characters
20. Each `aida:Event`, and `aida:Relation` prototype can have any number of `aida:LDCTime` structures.[6]

[6] It is expected that each prototype will have at most one temporal 4-tuple which aggregates temporal information for that event or relation, across all mentions. However, if there is informational conflict about the time of the event/relation in the source corpus, multiple temporal 4-tuples may be needed to represent the conflicting information.

21. Requirements for `aida:LDCTime`:
    a. Each `aida:LDCTime` must have exactly one `aida:start` with "AFTER" as the `aida:timeType` (representing the temporal constraint T1)
    b. Each `aida:LDCTime` must have exactly one `aida:start` with "BEFORE" as the `aida:timeType` (representing the temporal constraint T2)
    c. Each `aida:LDCTime` must have exactly one `aida:end` with "AFTER" as the `aida:timeType` (representing the temporal constraint T3)
    d. Each `aida:LDCTime` must have exactly one `aida:end` with "BEFORE" as the `aida:timeType` (representing the temporal constraint T4)
22. Each justification for an entity, event, or relation that will be examined by NIST SPARQL queries must be part of a type statement; if a mention is not a justification for a type statement, NIST will ignore that mention.

Additional TA3 requirements:

1. Each claim frame must be submitted as a single, standalone file to NIST for evaluation, containing exactly one `aida:Claim` and the entity, relation, and event KEs and edges that comprise the graph of Associated KEs for that claim frame.
    a. Entity, relation, and event KEs and edges that are extraneous to the graph of Associated KEs for the claim frame should ***not*** be included in the claim frame file. TA3 evaluation will assume that all KEs that are in the claim frame file are Associated KEs for that claim. It is expected that there will be only a small number of event and relation KEs in each claim frame file.
2. Each `aida:Claim` must have exactly one of each of the following (otherwise, the entire `aida:Claim` is ignored):
    a. `aida:sourceDocument`
    b. `aida:claimID`
    c. `aida:topic`
    d. `aida:subtopic`
    e. `aida:claimTemplate` (except for Condition 7, which has no `aida:claimTemplate`)
    f. `aida:xVariable` (except for Condition 7, which has no `aida:xVariable`)
    g. `aida:claimer`
    h. `aida:epistemic`
    i. `aida:sentiment`
3. Each `aida:Claim` must have one or more `aida:claimSemantics`.
4. Each `aida:Claim` must have one or more `aida:associatedKEs`.
5. Each `aida:associatedKEs` or `aida:claimSemantics` is the IRI of an `aida:SameAsCluster` representing the event (or relation) KE
6. Each `aida:Claim` may have at most 3 `aida:claimerAffiliation`.
7. Each `aida:Claim` must have at most one `aida:naturalLanguageDescription`
8. Each `aida:Claim` must have at most one `aida:claimDateTime`.
9. Each `aida:Claim` must have at most one `aida:claimLocation`.
10. Each `aida:Claim` must have at most one `aida:claimMedium`.
11. Each `aida:claimComponent` must have exactly one `aida:componentName`.

12. Each `aida:claimComponent` must have exactly one `aida:componentIdentity`.
13. Each `aida:ClaimComponent` must have at least one and at most 5 `aida:componentType`.
14. Justifications must be included for all entity, relation, and event KEs and edges that comprise the graph of Associated KEs.
    a. Each edge KE should have no more than 2 justifications (only two will be assessed; if more than two are provided, NIST will arbitrarily select two to assess and ignore the other justifications).
    b. Extraneous mentions should not be included for node KEs; ideally, include only informative mentions.
    c. All justifications for the graph of Associated KEs for a claim frame must come from the same `aida:sourceDocument` as the claim frame.
15. The graph of Associated KEs must have no disjunctions or alternative interpretations. In particular, [the prototype of] each event or relation cluster must have exactly one type assertion.
16. The graph of Associated KEs must have at least one event cluster or relation cluster with at least one edge.
17. The prototype of each entity (cluster) in the graph of Associated KEs must have exactly one text description (called a "handle"), which will be displayed to the assessor to represent the entity when the claim frame is being assessed.
    a. The handle may be a name string or a string specifying offsets of a particular text mention in the corpus.
        i. If the handle is a string in the format <`aida:sourceDocument`>:<`aida:source`>:(<`aida:startOffset`>,0)-(<`aida:endOffsetInclusive`>,0) (e.g., "IC0015L91:HC00002ZO:(4551,0)-(4570,0)"), NIST will extract the string in that location in that source document and source document element (assuming that the modality of the document element is text) and display it as the handle for assessment.
        ii. All other handle strings will be displayed as-is for assessment.
    b. The handle can be a generated string rather than a text span that is extracted from the corpus, and can be used to represent, for example, an entity that only appears in images.
    c. TA3 must coordinate with TA2 and TA1 to be able to get whatever information TA3 needs to provide NIST with a cluster-level textual "handle" for each entity for the TA3 evaluation.

3 Data

Training and evaluation data that is specific to the SM-KBP 2022 task will be listed on the SM-KBP data page (TBA). Additional training/development data, as listed in the 2022 TAC Evaluation License Agreement, is also available by request from the LDC. To obtain any training/development/evaluation data that is distributed by the LDC, all evaluation participants must register for the TAC Streaming Multimedia KBP track, submit the Agreement Concerning

Dissemination of TAC Results, and submit the 2022 TAC Evaluation License Agreement found on the NIST SM-KBP website.[7]

Details about data resources for the evaluation are available in a separate data plan from the Linguistic Data Consortium.

3.1 Training Data

Data for some practice topics in the COVID-19 Global Pandemic Scenario will be distributed by the LDC, including source documents, exhaustive TA1 annotation for some selected documents and DWD types, and TA3 annotation for some selected documents and topics.

3.2 Source Corpus

AIDA has several file formats (for various media) that are detailed in the documentation in the source corpus packages released by LDC. For Phase 3, multimedia documents will include only text and images.

The evaluation source corpus consists of approximately 2,000 multimedia documents that will be in the same format as the source corpus for the practice topics.

3.3 Reference KB

The DWD will serve as the reference KB for entities, in addition to serving as the ontology of entity, relation, and event types (and argument roles) for AIDA. This reference KB will be used for annotation as well as evaluation.

Task 2 participants must link specific entities in their KG to the reference KB when the entity is in the reference KB. Task 1 participants may also link specific entities to the reference KB but are not required to.

4 Task 1

4.1 Task 1 Definition

TA1 will process one document at a time and output a document-level knowledge graph for each input document from the document stream. A document may contain multiple document elements in multiple modalities; therefore, cross-lingual and cross-modal entity and event coreference are required. TA1 may also perform some cross-document entity coreference by linking entity KEs to entities in DWD (this may be useful for TA2 and TA3 downstream) but is not required for the purposes of evaluation of Task 1.

[7] AIDA program performers do not need to submit the 2022 TAC Evaluation License Agreement if they are participating only in SM-KBP.

For each document, TA1 will produce a document-level knowledge graph, including the DWD type(s) for each entity, relation, or event. Each event and relation must include all arguments, specified using DWD argument roles, in addition to including temporal properties attested in the document. For each entity, relation, or event, TA1 must extract and cluster ***all*** mentions in the document and assign a DWD type for the mention following guidelines for assigning most precise types to mentions. The negation status of event mentions, relation mentions, and event arguments must also be included.

TA1 must process each document independently of other documents. NIST will evaluate output for only selected documents in the data stream.

For the AIDA Phase 3 Evaluation, selected documents will be exhaustively annotated for entity, relation, and event mentions having selected types.[8] TA1 will be evaluated against the gold standard annotations in these documents.

The event/relation frame metric will be the primary metric used to rank TA1 runs, but additional diagnostic metrics will be reported.

4.2 Task 1 Filtering

NIST will filter for evaluable ERE (i.e., AIF clusters) and their evaluable mentions. Task 1 evaluation will ignore all system ERE (and their mentions) that are not evaluable.

We translate the LDC annotation types (e.g., "PER.Combatant.Mercenary") into DWD types (e.g., "Q178197") using the DWD overlay, and we define the "taggable LDC ontology" to be the DWD translation of the subset of the LDC annotation ontology that is actually selected for exhaustive TA1 annotation. We define an "evaluable DWD type" to be a type that is very similar to one of the types that is in the taggable LDC ontology, according to a similarity function DWDTypeSim(p,q) that returns the similarity between two type nodes in DWD, using the DWD overlay and similarity metrics from the ISI KGTK similarity service.[9]

We define DWDTypeSim(p,q) to be the average of the two ontology-based KGTK similarity metrics (class and jc):

- class: an ontology-based measure based on Jaccard Similarity of the respective super class sets of two nodes, inversely weighted by the instance counts of the classes. For this measure, classes high up in the ontology with very high transitive instance counts are weighted lower than more specific classes with lower counts
- jc: an ontology-based measure using an interpretation of the Jiang Conrath ontological distance. Instance counts are used to compute probabilities and normalize to the distance through the entity node (Q351201) to get a similarity. If a node pair has multiple most-specific subsumers, the maximum similarity based on those will be used.
- DWDTypeSim(p,q) = mean (class(p,q), jc(p,q))

[8] The Linguistic Data Consortium is releasing a separate data plan that describes the data resources and annotations developed to support the evaluation.
[9] https://github.com/usc-isi-i2/kgtk-similarity

An ERE is said to have an evaluable DWD type if it has some DWD type p such that DWDTypeSim(p,q) >= alpha, for some DWD type q that is in the taggable LDC ontology. The default value for alpha is 0.9.

Additionally, a DWD type is evaluable if it is similar (SS or NN) to a type in the taggable LDC ontology, according to the DWD overlay.

- We set DWDTypeSim(p,q)=1.0 if one type is an SS (synonym) of the other type, according to the DWD overlay.
- We set DWDTypeSim(p,q)=0.9 if one type is an NN (near neighbor) of the other type, according to the DWD overlay.

A system ERE is evaluable if and only if it passes the following filter:

1. The system ERE has some evaluable DWD type , or
2. The system ERE is aligned with a gold ERE in the filtering maximal bipartite matching between system clusters and gold clusters, described below.

Filtering Maximal Bipartite Matching (FMBM) between system clusters and gold clusters:

Let

- G be a gold cluster, g be a mention in G, |G| be the number of mentions in G
- S be a system cluster, s be a mention in S, |S| be the number of mentions in S
- type(m) be a type for a mention m
- MetatypeSim(G,S) = 1 if G and S have the same metatype (one of Entity, Relation, or Event); otherwise, MetatypeSim(G,S)=0
- TypeSim(G,S) be the confidence-weighted similarity between types in G and types in S
    - Let raw_conf(type(s)) be the product of the confidence that s has type type(s), and the confidence that s is a member of the cluster S
    - Let max_raw_conf(S) be the maximum raw_conf(type(s)) for all s in S
    - Let conf(type(s)) = raw_conf(type(s)) / max_raw_conf(S) be the scaled confidence value of type(s)
    - TypeSim(G,S) = conf(type(s)) * conf(type(g)) * DWDTypeSim(type(g),type(s)), for a gold mention g and system mention s that maximizes TypeSim(G,S)
- RoleSim(gr, sr) be the similarity between gold argument role gr, and system argument role sr
    - RoleSim(gr, sr) is computed as below:
        - RoleSim(gr,sr) = 1 if gr and sr have the same short name (e.g., “A0_pag” for an event argument, or “A0” for a relation argument), 0 otherwise
- mentionSim(g,s) be the similarity between gold mention g and system mention s,
    - mentionSim(g,s) = IOU(g,s) if IOU(g,s) ≥ iota; otherwise, mentionSim(g,s)=0
        - The default is iota = 0.1
- MentionSim(G,S) be the similarity between mentions in G and mentions in S,
    - MentionSim(G,S) is computed as below:
        - For each gold mention g in G and system mention s in S, compute mentionSim (g,s)

        - Find a maximal bipartite matching between mentions in G and mentions in S, to maximize total mentionSim (g,s) for aligned mentions
        - MentionSim(G,S) = number of aligned mention pairs (g,s) in the maximal bipartite matching of mentions
- Sim(G,S) be the unnormalized similarity between gold cluster G and system cluster S.
    - Sim(G,S) = MetatypeSim(G,S) * TypeSim(G,S) * MentionSim(G,S) if TypeSim(G,S) > minTypeSim; otherwise, Sim(G,S)=0
        - NIST will provide scores using various values of minTypeSim (0.0, 0.1, ..., 0.8, 0.9) as the threshold for filtering.
- ClusterSim(G,S) be the normalized similarity between gold cluster G and system cluster S
    - P = Sim(G,S) / |S|
    - R = Sim(G,S) / |G|
    - ClusterSim(G,S) = 2*P*R/(P+R)

We define the Filtering Maximum Bipartite Matching (FMBM) to be a maximal bipartite matching between system clusters and gold clusters, that maximize the sum of Sim(G,S) for aligned clusters G and S.

If an ERE passes the filter, then all of the mentions of the ERE are considered to be evaluable.[10]

4.3 Negation Detection

Evaluation of negation detection measures how well TA1 is able to detect negated mentions of relations and events. TA1 is evaluated only on system events and relations that have been aligned with gold events and relations in the FMBM. We report two versions of the Negation Metric, but NegationMetricV2 is the preferred version because it focuses on finding negated mentions and doesn't penalize TA1 for poor coreference.

NegationMetricV1: For each aligned system cluster S, we compute negation status correctness Neg(s,S) for each mention s in system cluster S. (This is evaluated as a negation status classification task on mentions in aligned clusters, where systems get penalized for unaligned mentions and incorrect negation status of aligned mentions.)
Let

- G be the gold cluster that is aligned to S in the FMBM.
- Neg(s,S) = 1 if and only if s is aligned to some gold mention g in the Maximal bipartite matching between mentions in G and mentions in S, and g and s have the same value for the Negated attribute.
- Negation Status Precision = P = (total sum of Neg(s,S) over all mentions s of aligned system clusters S) / (total number of mentions s of aligned system clusters S)
- Negation Status Recall = R = (total sum of Neg(s,S) over all mentions s of aligned system clusters S) / (total number of mentions g of aligned gold clusters G)

[10] It is possible that type X (e.g., weapon) is selected for exhaustive annotation, but not type Y (e.g., commodity), resulting in an entity (e.g., Jason's baseball bat) having some of its mentions tagged (when used as a weapon) but not others. However, it is expected to happen extremely rarely.

- Negation Status F1 = 2 * P*R / (P+R)

For NegationMetricV1, NIST will report Negation Status F1 as the score for Negation Detection.

NegationMetricV2: TA1 is evaluated only on aligned system and gold mentions, and only for system events and relations that have been aligned with gold events and relations in the FMBM. We compute precision, recall, and F1 only for negated mentions, only over aligned mentions of aligned clusters. No credit is giving for correctly predicting "not negated" because this is the most common negation status. We compute NegationMetricV2 as follows:

- TP=0
- FN=0
- FP=0
- Foreach aligned pair of gold and system clusters (G,S):
    - Foreach aligned pair of gold and system mentions (g,s), where g is a mention in G and s is a mention in S
        - If g is negated and s is negated then TP++
        - Else if g is negated and s is not negated then FN++
        - Else if g is not negated and s is negated then FP++
- Negation Precision = TP/(TP + FP); Negation Precision is 0 if TP+FP is 0
- Negation Recall = TP/(TP + FN); Negation Recall is 0 if TP+FN is 0
- Negation F1 = 2 * Precision * Recall / (Precision + Recall); Negation F1 is 0 if Precision+Recall is 0

For NegationMetricV2, NIST will report Negation F1 as the score for Negation Detection.

4.4 Entity/Event Detection and Linking Evaluation (EEDL)

AIDA entity/event detection and linking (EEDL) evaluation requires systems to find all mentions of all entities and events, although NIST only evaluates the entities and events that pass the initial filtering step (Section 4.2). The task requires systems to 1) find all mention spans of entities and events, 2) label each mention with one or more DWD types, and 3) coreference mentions of the same entity or event by making them justifications for type assertions on members of the same `aida:SameAsCluster`.

The primary metric for EEDL will be a clustering metric that takes into account the types assigned to the clusters, in addition to mention similarity; a secondary metric will measure how well the set of types associated with the system entity or event cluster is similar to the set of types in the aligned gold cluster, where similarity between the set of types is computed using TypeSim (defined in Section 4.2) which in turn uses DWD similarity functions.

4.4.1 Coreference Metric

For each document, NIST will evaluate coreference chains (`aida:SameAsCluster`) of evaluable entities (or events). Any entity (or event) `aida:SameAsCluster` will be ignored in the evaluation if it does not pass the filtering step (Section 4.2). A coreference chain is evaluable if and only if the corresponding cluster is evaluable.

For the final score for the Coreference Metric, NIST will report mention-level CEAF using the alignment from the FMBM.

4.4.2 Type Metric

Each system cluster has a set of system entity/event types associated with it, and each gold cluster has a set of gold entity/event types associated with it.

In order to create the FMBM (Section 4.2), NIST computes TypeSim(G,S) for each pair of aligned system cluster and gold cluster. Unaligned system clusters and gold clusters each have TypeSim=0. For the final score for the EEDL Type Metric, NIST will report mean TypeSim, where the mean is computed over the unaligned clusters and the pairs of aligned clusters.

4.5 Temporal Evaluation

Associated with each prototype event or relation will be a 4-tuple of dates, [T1 T2 T3 T4], indicating that the event occurred (or the relation is true) for a period beginning at some time between T1 and T2 and ending at some time between T3 and T4 (inclusive). A missing date for any of the 4 constraints implies a lack of a constraint. Thus [- 20110101 20110101 -] implies that the relation was true starting on or before January 1, 2011 and ending on or after January 1, 2011; i.e., that it was true on January 1, 2011 and no further information is available about when it started or ended. Similarly, [20100101 20101231 - -] implies that the relation was true starting at some time in 2010.

The representation of aggregate temporal information using the tuple [T1 T2 T3 T4] follows the definition for the Temporal Slot Filling task for TAC KBP 2013 and, as such, has the same limitations; some types of temporal information cannot be captured by a 4-tuple, including:

- Durations where neither endpoint is known ("he worked for IBM for 7 years")
- Relations between slots ("she married Fred two years after moving to Seattle", where the date of the moving event is not specified in the document)
- Regularly recurring events ("each Friday")
- Fuzzy relations ("lately", "recently")
- Relations which are true over multiple disjoint intervals ("Cleveland was President from 1885 to 1889 and from 1893 to 1897").[11]

TA1 must explicitly provide aggregate temporal information [T1 T2 T3 T4] in the prototype for each event or relation `aida:SameAsCluster` for which any temporal information is known. Each constraint in the 4-tuple should have no more than one (possibly unknown or underspecified)

[11] For inter-TA communication, performers may choose to represent such relations as two different relation clusters (one for each time interval). However, LDC annotation would represent this relation as a single relation cluster, and the aggregate temporal information for this relation would be the earliest start date (1885) and the latest end date (1897). Therefore, for purposes of the standalone TA1 temporal evaluation, TA1 should represent such relations in the same way.

date. An unknown date should leave year, month, and day unspecified (to indicate lack of constraint). An underspecified date may leave any of year, month, and day unspecified.

The TA1 temporal evaluation requires that any dates that are evaluated must be fully specified (including year, month, and day). NIST will attempt to translate each underspecified date (from the AIF graph) into a fully specified date for use in the evaluation:

- If year is specified but month and day are left unspecified for a particular constraint, then
  - the fully specified date is interpreted to be the first day of the year if the constraint is T1 or T3
  - the fully specified date is interpreted to be the last day of the year if the constraint is T2 or T4
- If year and month are specified but day is left unspecified for a particular constraint, then
  - the day is interpreted to be the first day of the month if the constraint is T1 or T3
  - the day is interpreted to be the last day of the month if the constraint is T2 or T4
- All other cases of underspecified dates will be interpreted as unspecified (missing) dates.

To produce the aggregate temporal information for a real-world event or relation from LDC's annotations, NIST will aggregate the temporal information associated with all members of the SameAsCluster representing the event/relation:

- Aggregate T1: earliest date for `aida:start AFTER`
- Aggregate T2: earliest date for `aida:start BEFORE`
- Aggregate T3: latest date for `aida:end AFTER`
- Aggregate T4: latest date for `aida:end BEFORE`

If any of the following conditions is true, then there's an inconsistency in LDC's temporal annotations, and the entire LDC temporal 4-tuple will be excluded from the evaluation.

- T1 > T2 : aggregate "start before date" is earlier than aggregate "start after date"
- T3 > T4 : aggregate "end before date" is earlier than aggregate "end after date"
- T1 > T4 : aggregate "end before date" is earlier than aggregate "start after date"

4.5.1 Temporal Metric

NIST will evaluate TA1 temporal information per real-world event or relation (i.e., a cluster) rather than per event/relation mention.

For the Temporal Evaluation:

- A system *event* (`aida:SameAsCluster`) is evaluable if and only if it passes the initial filtering step (Section 4.2)
- A system *relation* (`aida:SameAsCluster`) is evaluable if and only if it passes the initial filtering step (Section 4.2) AND it has *exactly* two arguments that are evaluable for the EEDL evaluation.

Evaluation will proceed as follows:

1. NIST will align the system event/relation prototypes with the gold event/relation prototypes:

- A system event prototype will be aligned with a gold event prototype if and only if the two events are aligned in the FMBM.
- A system relation prototype will be aligned with a gold relation prototype if and only if the two relations are aligned in the FMBM.

2. For each pair of aligned system and gold prototypes such that the gold prototype has a valid date for any of its aggregate temporal constraints, NIST will compute *S(slot)*, which measures the similarity between the temporal constraints in the system prototype versus the temporal constraints in the aligned gold prototype.
   - If more than one temporal tuple is in the system prototype,[12] NIST will compute the mean *S(slot)* over all of the temporal tuples in the system prototype.
3. NIST will report the temporal score to be the sum of *S(slot)* for pairs of aligned system and gold prototypes, divided by the number of gold prototypes that have some aggregate temporal information.

The simplest scoring scheme would mark each valid temporal constraint as correct or incorrect. However, because the time information provided by documents may be only approximate, such all-or-nothing scoring is likely to lead to problems. Instead, we use a score measuring the similarity of each constraint in the gold and system temporal tuple. Let the date in the gold temporal tuple be k_i and the date in the system temporal tuple be r_i; let $d_i = |k_i - r_i|$, measured in years. Then the score for the temporal constraints on an event or relation is

$$S(slot) = \tfrac{1}{4} \bullet \sum_{i=1}^{4} \frac{c}{c + d_i}$$

$$c = \begin{cases} c_{overconstraining}, if\ (i \in \{1,3\} \wedge r_i > k_i) \vee (i \in \{2,4\} \wedge r_i < k_i) \\ c_{vagueness}, otherwise \end{cases}$$

where $c_{overconstraining}$ and $c_{vagueness}$ are two constants (tentatively both set to 1/12 year) such that errors of that amount get 50% credit. This yields a score between 0 and 1.
If any of the 4 constraints in the gold temporal tuple is absent, NIST will divide by the number of non-empty gold constraints (rather than 4) when computing *S(slot)*. The absence of a system constraint in T1 or T3 is treated as a value of -∞; the absence of a system constraint in T2 or T4 is treated as a value of +∞. [13]

4.6 Event/Relation Argument Extraction Evaluation

For the Argument Extraction Evaluation, NIST will evaluate systems' ability to extract edges between events (or relations) and their arguments, and to label the edges with correct

[12] Multiple aggregate temporal tuples are allowed in order to enable TA1 and TA2 to indicate alternative interpretations about the time for an event or relation. However, the gold event or relation in the TA1 temporal evaluation is guaranteed to have no more than one aggregate temporal 4-tuple (because it is expected that there will rarely be any informational conflict about the time for an event/relation within a single document).
[13] In Phase 2, annotators did not always have access to the document creation date, so some temporal information that can be inferred from the document creation date will be absent from the gold annotation. Therefore, TA1 will not be penalized for asserting a date for a temporal constraint if that constraint is absent from the gold annotation.

event/relation types and roles.[14] A ***TRF triple*** is defined to be a KE edge in the knowledge graph, consisting of (*SubjectCluster, Roles, FillerCluster*), where *SubjectCluster* is the ID of an event or relation cluster prototype, *FillerCluster* is the ID of some ERE cluster prototype, and *Roles* is the set of short names for roles that the *FillerCluster* has in the event or relation, e.g., (Event1, {A0_pag}, Entity2}.[15] The number of unique TRF triples in a knowledge graph is the number of KE edges, which is the number of unique (*SubjectCluster, FillerCluster*) pairs, such that some argument assertion is made between *SubjectCluster* and *FillerCluster*.

For the Argument Extraction Evaluation, the set of evaluable TRF triples is the set of TRF triples generated by evaluable argument assertions. A system argument assertion in AIF is evaluable if and only if

1. the argument assertion has a subject that is the prototype of an ERE that passes the initial filtering step (Section 4.2) AND
2. the argument assertion has an object this is a prototype of an ERE that passes the initial filtering step (Section 4.2).

NIST will identify unique TRF triples in the system output and gold annotation, and align gold edges to system edges to maximize total TRFscore(goldTRF, systemTRF) for aligned pairs. We define TRFscore as follows;

TRFscore(goldTRF, systemTRF)= TypeSim(goldSubjectCluster, systemSubjectCluster) * RolesPrecision(goldRoles, systemRoles) * ClusterSim(goldFillerCluster, systemFillerCluster).[16]

RolesPrecision(goldRoles, systemRoles) is defined to be the fraction of roles in systemRoles that is also in the goldRoles.[17]

[14] Evaluation of event/relation argument extraction by comparison against a gold standard, is loosely based on the methodology in TAC KBP EAL 2014-2017, where a TRF triple represents a specified filler entity (F) having a specified role (R) for some event with a specified type (T).

[15] It's possible (though rare) for a filler to have more than one role in the same event; for example, an event may have two types (Infect and DiseaseOutbreak), possibly from two mentions of the same event, and the Victim in the DiseaseOutbreak could have role A2_gol, whereas the same Victim in the Infect event could have role A1_gol. It may be that the function tag part of the role label (i.e. the "_gol" part) generally aligns better across individual event mentions than the numeric part, but that may not always be the case.

[16] The subject clusters of aligned TRF triples don't have to be consistent across aligned TRF; instead, subject clusters only have to have the same metatype – either event or relation – and similar event or relation types.

[17] Multiple gold role label short names for an edge may arise because LDC asserts different roles for the same edge, or because a single LDC role label could be mapped to more than one DWD role label short name. In order to allow multiple gold role label short names (because of multiple ways of mapping an LDC role label to a DWD role label), and still discourage systems from returning spurious role label short names, we use the asymmetric function RolesPrecision(goldRoles, systemRoles) to require precise system labeling of argument roles, but not require the system to get all of the alternative gold role labels.
RolesPrecision(goldRoles,systemRoles) = (number of roles in common between goldRoles and systemRoles)/(number of roles in systemRoles).

For the final Event/Relation Argument Extraction score, we report meanTRFscore, where mean TRFscore is computed over aligned TRF triples, and unaligned system and gold TRF triples (an unaligned TRF triple has TRFscore=0).

NIST will report Argument Extraction scores for 3 sets of TRF triples:

A. All: TRF triples from all evaluable system argument assertions, compared against TRF triples from all gold argument assertions
B. Negated: TRF triples from evaluable system argument assertions that are negated, compared against TRF triples from gold argument assertions that are negated.
C. Non-negated: TRF triples from evaluable system argument assertions that are not negated, compared against TRF triples from gold argument assertions that are not negated.

NIST will report two variants of Argument Extraction scores:

1. Justified Argument Extraction: In order for a system TFR triple to align with a gold TRF triple, at least one of the system's top two highest confidence argument assertion must have a correct justification.
    - A justification for a system argument assertion is defined to be correct if it overlaps with a justification in the gold standard with IOU $\geq \alpha$ (α may vary with modality). For Phases 2 and 3 evaluations, we set IOU > 0.
2. Asserted Argument Extraction: ignores argument justification.

4.7 Event/Relation Frame Evaluation

The primary metric for Task 1 evaluation will be the Event/Relation Frame score, measuring the overlap between event/relation frames in the system versus the gold standard.

For the Event/Relation Frame Evaluation:

- A system *event* cluster and its argument assertions are evaluable if and only if they are evaluable for the Argument Extraction Evaluation.
- A system *relation* cluster and its argument assertions are evaluable if and only if they are evaluable for the Argument Extraction Evaluation AND the relation (`aida:SameAsCluster`) has ***exactly*** two arguments that are evaluable for the EEDL Evaluation.

The TA1 event/relation frame scorer will:

1. Align the system event and relation clusters with the gold event and relation clusters, using the alignment from the FMBM (Section 4.2)
2. For each aligned pair of system and gold cluster:
    a. Compute EdgeScore for edges in each pair of aligned system and gold clusters where EdgeScore(goldEdge, systemEdge) = ClusterSim(goldSubjectCluster, systemSubjectCluster) * RolesPrecision(goldRoles, systemRoles) * ClusterSim(goldFillerCluster, systemFillerCluster)

b. Align edges in gold cluster to those in system cluster using maximal bipartite matching to maximize total EdgeScore for aligned pairs
c. Compute average EdgeScore, where the average is taken over unaligned edges, and aligned pairs of system and gold edges.

3. Report mean average EdgeScore, where the mean is computed over unaligned clusters, and aligned pairs of system and gold clusters.

5 Task 2

5.1 Task 2 Definition

TA2 system is given 1) a stream of source documents, 2) a reference KB and 3) a stream of document-level knowledge graphs from TA1, and must construct a knowledge graph from the KE stream, aggregating information for the same real-world entity, relation, or event. TA2 must also link an entity in their KG to the reference KB if the entity is already in the reference KB.

The document-level KB will indicate which TA1 system produced each KE, and each TA2 system may choose to use KEs from one or more TA1 systems. (It is expected that participants will establish TA1-TA2 pairings well in advance of the evaluation window in order to be able to communicate using mutually agreed upon conventions for representing information and uncertaintly.)

TA2 Entity Evaluation will evaluate how well TA2 is able to corefer mentions of the same named entity across documents.

5.2 TA2 Entity Evaluation

TA2 Entity Evaluation will evaluate how well TA2 is able to corefer mentions of the same entity across documents. NIST will develop a set of TA2 entity queries, where each entity query will find all entities that "match" some descriptor. A descriptor is one of the following:

1. A KB ID from the Reference KB (describing no more than one entity)
2. A name string, e.g., "John Smith", or "Sergei" (which can describe more than one entity)

Queries will include some entities that have more than one name variant in the corpus, and some names that are ambiguous and can describe more than one entity in the corpus.

TA2 is expected to link an entity prototype to the reference KB if applicable, and include all name strings for the entity in the hasName properties of the entity. TA2 is required to include at least one informative justification per entity prototype per document.

NIST will apply the TA2 entity queries to the TA2 KB and assess the responses to compute Mean Average Precision (MAP) for the TA2 KB. Evaluation will proceed as follows:

For each entity query q:

1. Rank system entity clusters

    a. For a descriptor that is a KB ID, rank the cluster by the confidence of linking the cluster prototype to the KB ID (as in M36).
    b. For a descriptor that is a name string, rank the cluster based on the highest cluster membership confidence of the `aida:Entity` having hasName property that matches the query name string.[18] NIST will require an exact match between the name string in the query and the name string in the hasName property
2. Select up to C clusters to pool (C will vary per descriptor and should ideally be at least the number of different entities that are expected to match the descriptor)
    a. For KB ID descriptors: C=1
    b. For name string descriptors: C=3
3. For each cluster to pool, select the highest confidence informative justifications (from the entity prototype) from up to K=100 documents (one informative justification per document) and add to the pool for LDC to assess.
4. LDC assesses correctness of pooled informative justifications for each query (Does the informative justification include an entity that matches the query descriptor?)
5. LDC clusters the correct mentions into equivalence classes, one equivalence class per real world entity that matches the descriptor.
    a. If the descriptor in the query is a KB ID from the Evaluation Reference KB, then there should be only one equivalence class.
    b. If the descriptor in the query is a name (e.g., "Sergei"), it may describe more than one entity, in which case, LDC should create one equivalence class for each entity.
6. NIST creates a pseudo-query (q, gold_j) for each of the G entities in the corpus that is described by the query's descriptor.
7. NIST computes AP for each cluster sys_i (i=1..C) and each equivalence class gold_j (j=1..G), pretending that cluster sys_i was returned for the pseudo-query (q, gold_j) having unambiguous descriptor:
    a. Rank all informative justifications in cluster sys_i by aggregate confidence of the informative justification
    b. The Value of an informative justification is 0 if it is not assessed to have equivalence class gold_j; otherwise, the Value is 1.
    c. GT for AP is the number of unique documents (across all queries) that have equivalence class gold_j in the assessments.
    d. We compute AP over the K=100 items that were pooled/assessed by dividing the sum of the precision values in the ranked list by min(K,GT)
        i. N.B.: Standard AP would divide the sum of the precision values by GT, but that would unfairly penalize systems if K<<GT.
8. NIST computes a maximal bipartite matching between the clusters and the equivalence classes so as to maximize sum of AP of aligned pairs.
9. For each pseudo-query (q, gold_j), the final AP for the pseudo-query is the AP for the cluster sys_i that is aligned with the equivalence class gold_j in the maximal bipartite matching.
10. For purposes of computing MAP, the number of counted pseudo-queries for query q is min(C,G).

[18] Because no confidence is associated with the hasName property, NIST will look for the hasName property on the members of the aida:SameAsCluster rather than the prototype

NIST reports Mean AP over all pseudo-queries:

MAP = sum of AP over all pseudo-queries / sum of counted pseudo-queries over all queries

The final MAP gives more weight to entities that have more name variants, and more weight to queries that are ambiguous and can describe more than one entity.

6 Task 3

6.1 Overview

NIST proposes to evaluate automatic TA1-TA2-TA3 pipelines in 3 TA3 evaluation conditions. Conditions 5-6 comprise the official Phase 3 evaluation tasks for TA3, while Condition 7 is a stretch goal and may be evaluated only after the end of Phase 3. Prior to the official Phase 3 evaluations, there will be a dry run for Conditions 5-7, which should give performers an opportunity to exercise their entire TA1-TA2-TA3 pipelines in preparation for the official evaluations.

- Dry Run (Conditions 5-7)
  - A dry run for Conditions 5-7 will be conducted after performers have had a chance to manually annotate and familiarize themselves with the requirements for the claim frame fields during the hackathon.
  - The goal of the dry run is to give performers the opportunity to exercise and modify their entire TA1-TA2-TA3 pipeline in advance of the official Phase 3 evaluations. During the dry run, performers are allowed to manually inspect their input and output to make any needed modifications to their system.
  - In the dry run, performers will be given a corpus of practice source documents (LDC2021E11) and queries in the same format as for the final evaluations. Performers should process the entire corpus of source documents and output the requested list of claim frames for each evaluation condition in the same format as required for the final evaluations.
- Final Phase 3 evaluations (Conditions 5-7)
  - TA3 must return a ranked list of fully populated claim frames in response to each query
  - For Condition 5, a query is a claim frame.
  - For Condition 6, a query is a topic, subtopic and associated claim template.
  - For Condition 7, a query is a topic without any claim template.
  - Conditions 5-6 comprise the official Phase 3 evaluations for TA3
  - Condition 7 is a stretch goal and might not be evaluated within AIDA.

6.2 Task Definitions

For each evaluation condition, TA3 must return a ranked list of fully populated claim frames in response to each query. The query may be a claim frame (Condition 5); or a topic, subtopic and associated claim template (Condition 6); or a topic without any claim template (Condition 7).

At the beginning of the TA3 evaluation period, all TA1, TA2, and TA3 performers are given a corpus of ~2000 evaluation documents and the queries for each of the three TA3 evaluation conditions, so that they can generate output locally to pass on to downstream TAs. Performers' algorithms must process the input evaluation queries and documents fully automatically. TA1, TA2, and TA3 must each submit their system output by the designated deadlines for the evaluation.

The evaluation queries must be kept blind. Performers are ***not*** allowed to tune their systems by manually inspecting the evaluation queries in any way – either by examining the queries or by examining the output that are generated in response to the queries.

6.2.1 Condition 5

In Condition 5, systems are given an evaluation source corpus of ~2000 documents, a set of ~3 evaluation topics, and ~5 partially filled "query" claim frames for each topic. For each query claim frame, TA3 must return a ranked list of fully-filled claim frames on the same topic, filled with information found in the evaluation source corpus. TA3 must also designate each claim frame in the list as either "supporting", "refuting", or "related" to though not supporting or refuting the query claim frame. The ranked list should not contain identical claim frames, nor should it contain the query claim frame itself; the claim frames should be ranked so that more dissimilar claim frames are at the top of the ranked list.

For a given query claim frame, systems may use the information in the query claim frame (e.g., supporting document, X Variable Type Qnodes, etc.) to help retrieve additional claim frames that are on the same topic. However, ***systems must process each query claim frame independently of all other query claim frames***.

Input:

1) A ***topics file*** in .tsv format and
2) a set of ***query claim frame files*** in AIF, with one .ttl file per query claim frame.

The input topics file is a .tsv file with the following tab-separated columns defining each topic:

1. Topic ID: ID of topic in Column 2.
2. Topic: one topic, uniquely identified by the Topic ID in Column 1; e.g., "Contracting the virus"
3. Subtopic: one subtopic for the topic in Column 2; e.g., "Who contracts the virus"
4. Claim Template: one claim template for the subtopic in Column 3; e.g., "X can catch COVID-19"
5. Claim Template Qnodes (provided only for the dry run, and not for the official evaluation): a set of Qnodes for the Claim Template in Column 4. Multiple Qnodes are separated by a semicolon ";". Alternative Qnodes for a single semantic element are separated by a pipe "|".

Each topic may have multiple subtopics, which appear as multiple rows having the same values in Column 1 (Topic ID) and Column 2 (Topic).

Each input query claim frame file is named <QueryClaimFrameID>.ttl and has values for the following fields (if available in the supporting document):

- Document ID
- Claim ID
- Topic, Subtopic, Claim Template
- X Variable, X Variable Identity Qnode or NIL ID, X Variable Type Qnode(s)
- Claimer, Claimer Identity Qnode or NIL ID, Claimer Type Qnode(s)
- Claimer Provenance
- Claimer Affiliation, Claimer Affiliation Identity Qnode or NIL ID, Claimer Affiliation Type Qnode(s)
- Epistemic Status
- Sentiment Status
- Claim Date/Time
- Claim Location, Claim Location Identity Qnode or NIL ID, Claim Location Type Qnode(s)
- Claim Location Provenance
- Claim Medium, Claim Medium Identity Qnode or NIL ID, Claim Medium Type Qnode(s)
- Claim Medium Provenance

Output: One directory for each query claim frame, named with the <QueryClaimFrameID>. Each directory should have:

1) A ***Condition 5 ranking file*** for the query claim frame, named <QueryClaimFrameID>. ranking.tsv, containing the ranked list of claim frames that are on the same topic as the query claim frame and
2) a set of ***claim frame files*** in AIF, with exactly one <ClaimFrameID>.ttl file for each <ClaimFrameID> that is listed in Column 2 of the ranking file

Each Condition 5 ranking file is named <QueryClaimFrameID>.ranking.tsv and should be a tab-delimited file with the following fields:

1. Query Claim Frame ID: Claim Frame ID of query claim frame
2. Claim Frame ID: ID of claim frame that is on the same topic as the query claim frame in Column 1
3. Rank: A numeric rank for the claim frame in Column 2, in response to the query claim frame in Column 1
4. Claim Relation: Exactly one of {supporting, refuting, related}.
    a. Claim Relation should be "supporting" if and only if the claim (having Claim Frame ID in Column 2) supports the query claim frame (having Query Claim Frame ID in Column 1).
    b. Claim Relation should be "refuting" if and only if the claim (having Claim Frame ID in Column 2) refutes the query claim frame (having Query Claim Frame ID in Column 1).
    c. Claim Relation should be "related" if and only if the claim (having Claim Frame ID in Column 2) is related to the query claim frame (having Query Claim Frame ID in Column 1) but neither supports nor refutes it.

There must be only one line for each unique combination of values in Columns 1 and 2 (i.e., each Claim Frame ID that is returned in response to a query claim frame must have exactly one rank and exactly one claim relation tag). There must be only one line for each unique combination of values in Columns 1 and 3 (i.e., there can be no ties in the ranked list of Claim Frame IDs returned in response to the query claim frame).

Each output claim frame file is named <ClaimFrameID>.ttl and must include:

1. Document ID
2. Claim ID
3. Topic, Subtopic, Claim Template
4. X Variable, X Variable Identity Qnode or NIL ID, X Variable Type Qnode(s)
5. Claim Semantics
6. Claimer, Claimer Identity Qnode or NIL ID, Claimer Type Qnode(s)
7. Claimer Affiliation, Claimer Affiliation Identity Qnode or NIL ID, Claimer Affiliation Type Qnode(s)
8. Epistemic Status
9. Sentiment Status
10. Claim Date/Time
11. Claim Location, Claim Location Identity Qnode or NIL ID, Claim Location Type Qnode(s)
12. Claim Medium, Claim Medium Identity Qnode or NIL ID, Claim Medium Type Qnode(s)
13. Associated KEs

6.2.2 Condition 6

In Condition 6, systems are given an evaluation source corpus of ~2000 documents and a set of ~3 query topics. For each query topic, TA3 must return a ranked list of claim frames for that topic. The ranked list should not contain identical claim frames; the claim frames should be ranked so that more dissimilar claim frames are at the top of the ranked list.

Input:

1) A ***topics file*** in .tsv format (same format as for Condition 5).

The input topics file has the following tab-separated columns:

1. Topic ID: ID of topic in Column 2.
2. Topic: one topic, uniquely identified by the Topic ID in Column 1; e.g., "Contracting the virus"
3. Subtopic: one subtopic for the topic in Column 2; e.g., "Who contracts the virus"
4. Claim Template: one claim template for the subtopic in Column 3; e.g., "X can catch COVID-19"
5. Claim Template Qnodes (provided only for the dry run, and not for the official evaluation): a set of Qnodes for the Claim Template in Column 4. Multiple Qnodes are separated by a semicolon ";". Alternative Qnodes for a single semantic element are separated by a pipe "|".

Each topic may have multiple subtopics, which appear as multiple rows having the same values in Column 1 (Topic ID) and Column 2 (Topic). The query topics for Condition 6 are the set of unique values in Column 1 of the topics file.

Output: One directory for each query topic, named with the <TopicID> from Column 1 of the topics file. Each directory should have:

1) a ***Condition 6 ranking file*** for the query topic, named <TopicID>. ranking.tsv, containing the ranked list of claim frames that are on the same topic as the query topic and
2) a set of ***claim frame files*** in AIF, with exactly one <ClaimFrameID>.ttl file for each <ClaimFrameID> that is listed in Column 2 of the ranking file

Each Condition 6 ranking file is named <TopicID>.ranking.tsv and should be a tab-delimited file with the following fields:

1. Topic ID: ID of the topic (Column 1 of the topics tsv file).
2. Claim Frame ID: ID of claim frame that is on the same topic as the query topic in Column 1
3. Rank: A numeric rank for the claim frame in Column 2, in response to the query topic in Column 1

There must be only one line for each unique combination of values in Columns 1 and 2 (i.e., each Claim Frame ID that is returned in response to a query topic must have exactly one rank). There must be only one line for each unique combination of values in Columns 1 and 3 (i.e., there can be no ties in the ranked list of Claim Frame IDs returned in response to the query topic).

Each output claim frame file is named <ClaimFrameID>.ttl and must include:

1. Document ID
2. Claim ID
3. Topic, Subtopic, Claim Template
4. X Variable, X Variable Identity Qnode or NIL ID, X Variable Type Qnode(s)
5. Claim Semantics
6. Claimer, Claimer Identity Qnode or NIL ID, Claimer Type Qnode(s)
7. Claimer Affiliation, Claimer Affiliation Identity Qnode or NIL ID, Claimer Affiliation Type Qnode(s)
8. Epistemic Status
9. Sentiment Status
10. Claim Date/Time
11. Claim Location, Claim Location Identity Qnode or NIL ID, Claim Location Type Qnode(s)
12. Claim Medium, Claim Medium Identity Qnode or NIL ID, Claim Medium Type Qnode(s)
13. Associated KEs
14. Supporting Claims: Claim IDs of related claims (from among the output list of claim frames for this topic) that support this claim
15. Refuting Claims: Claim IDs of related claims (from among the output list of claim frames for this topic) that refute/contradict this claim

16. Related Claims: Claim IDs of related claims (from among the output list of claim frames for this topic) that neither support nor refute this claim

The categorization of supporting/refuting/related claims is not expected to be evaluated for the official evaluation of Condition 6, but systems should do this categorization in case such an evaluation is warranted.

6.2.3 Condition 7

In Condition 7, systems are given an evaluation source corpus of ~2000 documents, and a set of ~8 query topics. Each topic will be provided as English text (e.g., "Where are masks recommended?") but will ***not*** be accompanied by any claim templates. For each topic, TA3 must return a ranked list of claim frames for that topic. The ranked list should not contain identical claim frames; the claim frames should be ranked so that more dissimilar claim frames are at the top of the ranked list.

Input:

1) A topics file in tsv format, containing a list of query topics

The input topics file is a .tsv file with the following tab-separated columns defining each query topic:

1. Topic ID: ID of topic in Column 2.
2. Topic: one topic, uniquely identified by the Topic ID in Column 1; e.g., "Contracting the virus"

Output: One directory for each query topic, named with the <TopicID> from Column 1 of the topics file. Each directory should have:

1) a ***Condition 7 ranking file*** for the query topic, named <TopicID>. ranking.tsv, containing the ranked list of claim frames that are on the same topic as the query topic and
2) a set of ***claim frame files*** in AIF, with exactly one <ClaimFrameID>.ttl file for each <ClaimFrameID> that is listed in Column 2 of the ranking file

Each Condition 7 ranking file is named <TopicID>.ranking.tsv and should be a tab-delimited file with the following fields:

1. Topic ID: ID of the topic (Column 1 of the topics tsv file).
2. Claim Frame ID: ID of claim frame that is on the same topic as the query topic in Column 1
3. Rank: A numeric rank for the claim frame in Column 2, in response to the query topic in Column 1

There must be only one line for each unique combination of values in Columns 1 and 2 (i.e., each Claim Frame ID that is returned in response to a query topic must have exactly one rank). There must be only one line for each unique combination of values in Columns 1 and 3 (i.e., there can be no ties in the ranked list of Claim Frame IDs returned in response to the query topic).

Each output claim frame file is named <ClaimFrameID>.ttl and must include:

1. Document ID
2. Claim ID
3. Topic
4. Claim Semantics
5. Claimer, Claimer Identity Qnode or NIL ID, Claimer Type Qnode(s)
6. Claimer Affiliation, Claimer Affiliation Identity Qnode or NIL ID, Claimer Affiliation Type Qnode(s)
7. Epistemic Status
8. Sentiment Status
9. Claim Date/Time
10. Claim Location, Claim Location Identity Qnode or NIL ID, Claim Location Type Qnode(s)
11. Claim Medium, Claim Medium Identity Qnode or NIL ID, Claim Medium Type Qnode(s)
12. Associated KEs
13. Supporting Claims: Claim IDs of related claims (from among the output list of claim frames for this topic) that support this claim
14. Refuting Claims: Claim IDs of related claims (from among the output list of claim frames for this topic) that refute/contradict this claim
15. Related Claims: Claim IDs of related claims (from among the output list of claim frames for this topic) that neither support nor refute this claim

Because Condition 7 output will not have claim templates and X variables, evaluation will not be exactly the same as for Conditions 5 and 6, but is TBD.

6.3 Evaluation (Condition 5 and 6)

For each query topic in Condition 6, NIST will pool and LDC will assess the top-ranked claim frames in the output ranked list for the topic.

For each query claim frame in Condition 5, NIST will pool and LDC will assess the top-ranked claim frames in 6 ranked lists extracted from the full output ranked list for the query claim frame, where the claim in each extracted list must have a particular target claim relation:

1. On-topic: Claim frames in the rankings file (i.e., claim frames that the system has identified as being on the same topic as the query claim frame, ignoring whether the claim frame is designated as supporting versus refuting versus merely related to the query claim frame).
2. Supporting: Claim frames in the rankings file that the system tags as supporting the query claim frame
3. Refuting: Claim frames in the rankings file that the system tags as refuting the query claim frame
4. Related: Claim frames in the rankings file that the system tags as related to the query claim frame but are neither supporting nor refuting the query claim frame.
5. Non-refuting: Claim frames in the rankings file that the system tags as ***not*** refuting the query claim frame (i.e., claims that the system tags as supporting or related).

6. Non-supporting: Claim frames in the rankings file that the system tags as ***not*** supporting the query claim frame (i.e., claims that the system tags as refuting or related).

Claim frames will be assessed for correctness of claim relation, informativeness of descriptors returned for claim frame field values, and correctness of justifications. In order to have correct claim relation, the claim frame must always be on topic. Furthermore, when the ranked list is restricted to one of {Supporting, Refuting, Related} in Condition 5, the relation tagged by the system must match the relation manually assigned by the assessor. When the ranked list is restricted to one of {Non-refuting, Non-supporting}, the relation tagged by the system must be compatible with the relation manually assigned by the assessor; however, compatible but non-matching relation tags result in partial credit (in particular, the claim frame gets only half credit if the system tag is "related" but the gold tag is "supporting" in List 5 or "refuting" in List 6).

For purposes of evaluation, we assume that the user has ranked the fields in the claim frame by importance or centrality, and that each field is given a weight. Phase 3 evaluation assumes the following weights and ranking of fields by importance:

1) Claim Template w1 = 32
2) Epistemic Status w2 = 32
3) X Variable w3 = 32
4) Claimer w4 = 16
5) Claimer Affiliation w5 = 4
6) Claim Location w6 = 1
7) Claim Medium w7 = 1
8) Claim Date/Time w8 = 1
9) Sentiment Status w9 = 0.5

TA3 should return claim frames that have diverse values for more important fields, before returning claim frames that fill in less important fields but don't provide any new information for the more important fields. The top most important fields are considered ***required*** fields, while remaining fields are considered ***desirable*** fields. For Phase 3 evaluation, the required fields are Fields 1-4.

Evaluation will give higher scores to TA3 claim frames that exhibit:

1) Correctness
2) Completeness
3) Diversity across the list of returned claim frames

Correctness: A field value in the claim frame is Correct if it has at least one informative descriptor that is attested in the accompanying document. A descriptor for a field value can be a handle string or a QNode; a descriptor is informative if it is (preferably) an identity QNode from the reference KB, or else a handle string that is a name, or else a fine-grained type QNode; but not if it is a handle string that's a pronoun or foreign language phrase.
In order for a claim frame to be Correct in any degree, it must minimally be on topic and have Correct values for all *required* fields.

Completeness: The claim frame should have Correct values for as many fields as possible. Values for *desirable* fields should be provided whenever they are attested in the source documents, with a preference for more important fields.

Diversity: The ranked list of claim frames should include a large number of conflicting claims, and diverse values for the more important fields. The claim frames should be ranked so that more dissimilar claim frames are at the top of the ranked list.

For each ranked list of claim frames, NIST will report:

1. A variant of **Normalized Discounted Cumulative Gain** (NDCG).
2. F1 over unique combinations of the most important fields in the returned claim frames, calculated at the cutoff that yields maximal F1 for the ranked list.

There will be exhaustive manual annotation of claim frames for a limited number of documents, and the set of correct claim frames for a query and target claim relation will consist of the combined set of manual claim frames (including query claim frames) and correct system claim frames.

6.3.1 NDCG

Scoring a ranked list of claim frames will be done by a variant of NDCG**.** While the *gain* in standard NDCG can be computed for an item independently of other items in the ranked list, the gain of a claim frame in a ranked list will measure the amount of information in the claim frame that is new, compared to all previously seen claim frames in the list (including the query claim frame, for Condition 5). To compute discounted cumulative gain (DCG) for the ranked list, the gain is accumulated from the top of the list to the bottom, with the gain of each claim frame discounted at lower ranks. The maximal value of DCG may be vastly different, depending on the number of different claims that exist for a given query and claim relation. Therefore, in order to compare results across different queries and claim relations, we create an ideal ranking for each query and claim relation (from the set of all correct claim frames for the query and claim relation) and compute the ideal DCG (IDCG) for the ideal ranking. NDCG for a ranked list is defined to be DCG/IDCG.

Let Novelty(C1,C2) be the degree of novelty of claim frame C1 compared to claim frame C2. Novelty(C1,C2) is an asymmetric function that measures the total amount of Correct new information that is contributed by C1 (that's not in C2). We define Novelty(C1,C2) to be the weighted sum of the fields in C1 that contribute Correct new information, weighted by the importance of the field.

We assume the following model for how a user determines whether a particular field f_x in claim frame C1 contributes new information when compared to a previously seen claim frame C2: For each field f_x, let Context(f_x) be the set of fields defining the context in which f_x should be interpreted when determining whether f_x contributes new information; if any of the fields in Context(f_x) in C1 contributes new information when compared to C2, then f_x in C1 is considered to contribute new information compared to C2 (regardless of whether the value of f_x is identical in both C1 and C2).

Phase 3 evaluation assumes that the user has specified the following definition of Context(f_x) for each field in the claim frame:

1) Context(Claim Template) = {X Variable, Epistemic Status}
2) Context(Epistemic Status) = {Claim Template, X Variable}
3) Context(X Variable) = {Claim Template, Epistemic Status}
4) Context(Claimer) = {Claim Template, X Variable, Epistemic Status}
5) Context(Claimer Affiliation) = {Claim Template, X Variable, Epistemic Status, Claimer}
6) Context(Claim Location) = {Claim Template, X Variable, Epistemic Status, Claimer}
7) Context(Claim Medium) = {Claim Template, X Variable, Epistemic Status, Claimer}
8) Context(Claim Date/Time) = {Claim Template, X Variable, Epistemic Status, Claimer}
9) Context(Sentiment Status) = {Claim Template, X Variable, Epistemic Status, Claimer}

A value for a field f_x in claim C1 is novel with respect to C2 if the value of f_x in C1 is Correct *and*:

1. the field f_x in C2 has no Correct value, *or*
2. the value of f_x in C1 is *different* from every value of f_x in C2 (i.e., the value of f_x in C1 is not identical or nearly identical to any value of f_x in C2), *or*
3. there is some field f_y in Context(f_x), such that a value of f_y in C1 is novel with respect to C2.

Near-identity is measured using a coarse similarity function that is defined by human judgment. In Phase 3, two values for a field f_x are considered to be nearly identical if they have the same similarity equivalence class (*ec_similarity*) in LDC's assessment. For Claim Date/Time, two temporal 4-tuples are considered to be nearly identical if the dates in the corresponding constraint field are within one month of each other.

Let MarginalNovelty(C1) be the degree of novelty of claim frame C1 at a particular rank in a ranked list, compared to all previously seen claim frames in the ranked list. We define MarginalNovelty(C1) to be the minimal value for Novelty(C1,C2), calculated over all previously seen claim frames C2 in the ranked list.

The score for a ranked list of claim frames will be NDCG, where the gain for a claim frame C1 in the ranked list is defined to be the MarginalNovelty(C1), scaled by the degree of correctness of the claim relation.

The scaling factor for correctness of claim relation is defined as follows:

- When the ranked list is restricted to {On-topic}, the scaling factor is 1.0.
- When the ranked list is restricted to one of {Supporting, Refuting, Related}, the scaling factor is 1.0 if the relation tagged by the system exactly matches the relation manually assigned by the assessor, and is 0 otherwise.
- When the ranked list is restricted to {Non-refuting}, the scaling factor is 1.0 if the relation tagged by the system exactly matches the relation manually assigned by the assessor, 0.5 if the system tag is "related" but the gold tag is "supporting", and 0 otherwise.

- When the ranked list is restricted to {Non-supporting}, the scaling factor is 1.0 if the relation tagged by the system exactly matches the relation manually assigned by the assessor, 0.5 if the system tag is "related" but the gold tag is "refuting", and 0 otherwise.

6.3.2 F1

For each ranked list of claim frames, NIST will report F1 for unique combinations of the most important fields in the claim frames, for claim frames that are Correct and whose assessed claim relation is compatible with the target claim relation. The reported F1 is calculated at the cutoff point that yields maximal F1 for the ranked list. F1 will be computed over different sets of fields, yielding different variants of F1 (A, B, or C):

A. Required F1 fields: Claim Template, X Variable, Claim Epistemic, Claimer
B. Required F1 fields: Claim Template, X Variable, Claim Epistemic
C. Required F1 fields: Claim Template, X Variable

Values for Epistemic status are considered to be the same if they have the same polarity. Values for X Variable or Claimer are considered to be the same if they have the same similarity equivalence class (similarity_ec).

6.4 Assessment (Conditions 5 and 6)

For each run and query topic in Condition 6, NIST will pool the top N1=30 claim frames returned for the topic.

For each run and query claim frame in Condition 5, NIST will pool the top-ranked claim frames from each list:

1. N1=30 claim frames tagged as On-topic
2. N2=30 claim frames tagged as Supporting
3. N3=30 claim frames tagged as Refuting
4. N4=30 claim frames tagged as Related
5. N5=30 claim frames tagged as Non-refuting
6. N6=30 claim frames tagged as Non-supporting

NIST will pool with a cap on the number of unique documents that must be assessed.
LDC will assess the pooled claim frames for Correctness of field values, Similarity of values for the same claim frame field, and Cross-claim relations.

6.4.1 Correctness

The assessor will judge the correctness of each claim frame that was pooled for a document. The assessor will first assess if the claim frame as a whole is fatally flawed (e.g., off-topic, vacuous, or not understandable). For claim frames that are not fatally flawed, the assessor will then assess Correctness of each field value in the claim frame to ensure that it has an informative descriptor and is justified in the accompanying document.

When determining if a system-provided document justifies some field value of a claim frame, the assessor will look at regions that are in the vicinity of the system-provided provenance/justification spans but might also look at the entire document.[19]

The official scoring metric based on manual assessment assumes that the user will not rely on the KEs to determine whether a field value of a claim frame (CF) is supported in a document; rather, it's sufficient for the field values of the CF to be supported somewhere in the document, regardless of how the system has represented the CF as KEs.

However, even though the KEs might not be examined by a human user, they comprise the machine-readable version of the CF and must also be correct and suitable for automatic consumption by, for example:

- automatic scorers that are trained on the KEs and documents that have been judged as being "Correct" for the claim
- downstream analytics that further automatically process the claim frames.

Therefore, the assessor will assess both the correctness of the human readable version of the claim (i.e., does the document support the information in the CF?), as well as the machine readable version of the claim (i.e., do the Associated KEs express the meaning of the CF, and are those Associated KEs supported in the document?).

Let F be the field or combination of fields that is being assessed for the claim frame. The assessor will judge: To what extent do the ERE and their justification in the Associated KEs express the meaning of F in the claim frame? This is broken down into judgments along 3 dimensions:

1. To what extent does the document support the meaning of F?
2. To what extent do the KEs express the meaning of F?
3. To what extent do the justification support the KEs?

In order for the human-readable version of the CF to be correct, only dimension 1 needs to be correct. In order for the machine-readable version of the CF to be correct, dimensions 2 and 3 must be correct.

Correctness of Required fields: Judgments 1-4 assess the required fields in the claim frame. Judgment i for a required field must be made only if all previous judgments are some form of Correct (i.e., "Correct" or "Inexact"); otherwise, the value for judgment i is N/A.

Correctness of Desirable fields: Judgments 5-9 assess the desirable fields in the claim frame. Judgment j for a desirable field must be made only if judgments for all required fields are some form of Correct; otherwise, the value for judgment j is N/A.

Judgements 1 and 2 ask: To what extent do the ERE and their justification in the Associated Knowledge Elements express the meaning of the populated Claim Template (Claim Template + X Variable):

[19] It is intractable to review the entire document for each claim frame if we want to assess large numbers of claims.

- F = Claim Template:
  - Judgment 1: Claim Template: {Correct, Incorrect, Inexact}
- F = Claim Template + X Variable:
  - Judgment 2: Supported and informative X Variable: {Correct, Incorrect, Inexact, N/A}

Judgements 3 and 4 ask: To what extent do the ERE and their justification in the Associated Knowledge Elements show that Claimer holds that populated Claim Template has Epistemic Status (S):

- F = Claimer:
  - Judgment 3: Supported and informative Claimer: {Correct, Incorrect, Inexact, N/A}
- F = Claimer + Epistemic Status:
  - Judgment 4a: Supported Epistemic Status polarity: {Correct, Incorrect, Inexact, N/A}
  - Judgment 4b: Supported Epistemic Status strength: {Correct, Incorrect, Inexact, N/A}

Judgments 5-9 ask: To what extent do the ERE and their justification in the Associated Knowledge Elements support the values in the desirable fields of the claim frame:

- F = Claimer Affiliation:
  - Judgment 5: Supported and informative Claimer Affiliation: {Correct, Incorrect, Inexact, N/A}
- F = Claim Location:
  - Judgment 6: Supported and informative Claim Location: {Correct, Incorrect, Inexact, N/A}
- F = Claim Medium:
  - Judgment 7: Supported and informative Claim Medium: {Correct, Incorrect, Inexact, N/A}
- F = Claim Date/Time:
  - Judgment 8: Supported and informative Claim Date/Time: {Correct, Incorrect, Inexact, N/A}
- F = Sentiment:
  - Judgment 9: Supported Sentiment Status: {Correct, Incorrect, Inexact, N/A}

6.4.2 Similarity

For all Correct claim frames and their Correct field values (pooled from query claim frames, LDC annotations, and all TA3 claim frames that have been assessed as having Correct values for all required fields), assessors will:

1) Create global identity equivalence classes (identity_ec) for values in each claim frame field.
2) Create coarse similarity equivalence classes (similarity_ec) for values in each claim frame field.
   a. Claim Template (automatic string match)
   b. Epistemic Status (automatic string match of polarity)

c. X Variable (with Correct informative descriptors: Handle string, Identity Qnode, and Type Qnodes)
d. Claimer (with Correct informative descriptors: Handle string, Identity Qnode, and Type Qnodes)
e. Claimer Affiliation (with Correct informative descriptors: Handle string, Identity Qnode, and Type Qnodes)
f. Claim Location (with Correct informative descriptors: Handle string, Identity Qnode, and Type Qnodes)
g. Claim Medium (with Correct informative descriptors: Handle string, Identity Qnode, and Type Qnodes)
h. Sentiment Status (automatic string match)

These assessments will be used to approximate a coarse-grained similarity function between claim frames, used for scoring.

6.4.3 Cross-Claim Relation

LDC will assess cross-claim relations for Condition 5. For each query claim frame Q in Condition 5 and each correct, on-topic claim frame C for that topic (pooled from query claim frames, LDC annotations, and all TA3 claim frames that have been assessed as having Correct values for all required fields), LDC will determine the Claim Relation between Q and C: Exactly one of {identical, supporting, refuting, related}.

a. "identical" if and only if claim C is identical to the query claim frame Q.
b. "supporting" if and only if claim frame C supports the query claim frame Q.
c. "refuting" if and only if claim frame C refutes the query claim frame Q.
d. "related" if and only if claim frame C is related to the query claim frame Q but neither supports nor refutes it.

7 Submissions

7.1 Submission procedure

AIDA performers must submit their systems as well as system output. For Task 2 and Task 3, all TAs must submit their system output by designated deadlines during the evaluation period, for use as input to downstream TAs or for NIST to evaluate (see AIDA Phase 3 Evaluation Schedule for TA1/TA2/TA3 milestones); the dockers that produced these output must subsequently be submitted to CACI to facilitate technology transfer. For Task 1, AIDA TA1 performers must submit at least one docker that's optimized for the Task 1 evaluation by June 6, 2022; other TA1 systems that produced output targeted for Task 2 or Task 3 must also be submitted before the end of the Task 1 evaluation window (before the end of the AIDA program).

The TA1 dockers that are optimized for the Task 1 evaluation do not need to produce the same KBs that are provided as input to TA2 and TA3 (and that are optimized for Task 2 and Task 3). Task 1 evaluates general information extraction, but different applications may require systems to be optimized in different ways. Similarly, the TA2 KBs that are submitted for the TA2 cross-

document entity coreference evaluation do not need to be the same KBs that are provided as input to TA3.

7.2 Number of submissions

For Task 1, each team is allowed to submit up to 10 runs over the course of the Task 1 evaluation window.

For Task 2, each team must take input from at least two TA1 teams and must submit at least two and no more than N runs for evaluation, where N is the number of end-to-end pipelines that the TA2 team is in. At least 2 TA2 KBs from each team will be assessed; additional TA2 KBs will be assessed only if resources allow. TA2 teams must rank their submissions in order of priority of assessment.

For Task 3, each team must take input from at least two TA2 teams and must submit two runs for evaluation.

NIST will evaluate only those knowledge graphs that are in restricted AIF format. The only time that NIST will allow a team to submit a run for evaluation and not have it count towards their submission limit, will be if TA1 scores are all 0, or if SPARQL queries return no response.

8 Participation

8.1 Rules of Participation

The evaluation is an open evaluation where test data may be sent to the participants who will process and submit the output to NIST. As such, the participants must agree to the following rules:

- Teams that would like to participate in any of the SM-KBP 2022 tasks must register for the SM-KBP track of TAC 2022.
- Teams are generally allowed to use external resources that have not been distributed by NIST/LDC for SM-KBP; however, Task 3 participants may not use Wikipedia or scenario-relevant resources released after the start date of the scenario. Because of the rich information about events and relations in Wikidata, all participants (Task 1, Task 2, and Task 3) should only use the authorized version of Wikidata (i.e., DWD).
- The participant agrees not to investigate the evaluation data. Both human/manual and automatic probing of the evaluation data is prohibited, to ensure that all participating systems have the same amount of information on the evaluation data.
- The only time replacing an existing submission is allowed is when it is determined the submission has a systematic bug, at which time, teams will need to contact NIST to enable resubmission and withdrawal of the old submission. Submissions that are withdrawn will not count toward the submission limit.
- At the conclusion of the evaluation, each team is required to submit a system description that covers their submissions for all tasks the team is participating in.
- The participant agrees to the rules governing the publication of the results.

8.2 Publication of Evaluation Results

This evaluation follows an open model to promote interchange with the outside world. At the conclusion of the evaluation cycle, NIST will create a report that documents the evaluation. The report will be posted on the NIST web space and will identify the participants and the scores from various metrics achieved for tasks. The report that NIST creates should not be construed or represented as endorsements for any participant's system or commercial product, or as official findings on the part of NIST or the U.S. Government.

The rules governing the publication of the TAC evaluation results are similar to those used in other NIST evaluations.

- Participants are free to publish results for their own system, but participants must not publicly compare their results with other participants (ranking, score differences, etc.) without explicit written consent from the other participants.
- While participants may report their own results, participants may not make advertising claims about winning the evaluation or claim NIST endorsement of their system(s). Per U.S. Code of Federal Regulations (15 C.F.R. § 200.113): *NIST does not approve, recommend, or endorse any proprietary product or proprietary material. No reference shall be made to NIST, or to reports or results furnished by NIST in any advertising or sales promotion which would indicate or imply that NIST approves, recommends, or endorses any proprietary product or proprietary material, or which has as its purpose an intent to cause directly or indirectly the advertised product to be used or purchased because of NIST test reports or results.*
- All publications must contain the following NIST disclaimer:

  NIST serves to coordinate the evaluations in order to support research and to help advance the state- of-the-art. NIST evaluations are not viewed as a competition, and such results reported by NIST are not to be construed, or represented, as endorsements of any participant's system, or as official findings on the part of NIST or the U.S. Government.