The UTexas system for TAC 2019 SM-KBP Task 3: Hypothesis detection with graph convolutional networks

Pengxiang Cheng **Alexander Tomkovich** **Eric Holgate** **Su Wang** **Katrin Erk**
University of Texas at Austin
{pxcheng,alexander.tomkovich,holgate,shrekwang,katrin.erk}@utexas.edu

1 Introduction

Events of geopolitical interest can unfold quickly and are frequently surrounded by uncertainty. This can largely be attributed to the emergence of information across various modalities and media in different languages, and is frequently compounded by the presence of conflicting claims. The crash of flight MH-17 in Ukraine in 2014 serves as a good example. After the crash, a multitude of explanations arose, including Russia-affiliated aggression, Ukranian military aggression, an act of terror, and mechanical failure.

The Streaming Multimedia Knowledge Base Population (SM-KBP) track at TAC 2019 continues to address the challenges of this scenario in three tasks. Task 1 is the extraction of relevant information from multi-modal and multi-lingual sources. Task 2 is the aggregation of document-level knowledge into a multi-document knowledge graph, including cross-document coreference resolution. Task 3 is to probe the resulting knowledge graph for information relevant to a provided *statement of information need (SIN)*, entailing the separation of conflicting claims and the construction of internally consistent narratives (represented as subgraphs of the multi-document knowledge graph called *hypotheses*).

The UTexas team participated in Task 3, hypothesis generation. We view hypothesis generation as a one-class clustering task (Bekkerman and Crammer, 2008) of detecting further coherent entities and statements surrounding a *hypothesis seed*.

The UTexas system for TAC 2019 uses a hybrid system in which a neural model proposes statements for inclusion in the hypothesis cluster, and a rule-based filter that rejects statements based on either domain knowledge or domain-independent well-formedness criteria.

2 The Hypothesis Generation Task

SM-KBP Task 3 requires the generation of hypotheses from the multi-document knowledge graph output in Task 2. The graph adheres to AIDA Interchange Format (AIF) specifications (represented in RDF and saved in turtle format).

AIF is principally concerned with representing events, entities, and relations (EREs). EREs are characterized by AIF nodes. AIF statements describe the roles linking EREs, identify them within an ontology of types, and include a confidence score. Instances of EREs may also be coreferential, encoded in the knowledge graph via AIF SameAsCluster nodes. SameAsCluster membership is indicated by an AIF ClusterMembership statement. As ClusterMembership statements are AIF statements, they also include confidence metrics.

Hypotheses are generated from the multi-document knowledge graph based on *Statements of Information Need (SINs)*. A SIN describes the scenario of interest in XML by defining *entrypoints, temporal information*, and *frames*. Entrypoints identify AIF nodes which must be included in the hypothesis. Temporal information provides start and end times for events. Frames designate partial skeletons for hypotheses, defined by one or more *edges* comprising a predicate-argument triple. All of the arguments within an edge are defined with AIF ERE types (e.g., *Conflict.Attack.FirearmAttack_Attacker*).

Each frame can be thought of as a different take on the scenario (in the MH-17 example, there may be unique frames for Russia-affiliated or Ukranian military aggression, mechanical failure, etc.). While

each generated hypothesis might include (some of) the same EREs, the relationships defined in the frame edges situate them differently into distinct claims. In this way, hypotheses may be mutually exclusive due to incompatible statements. For example, it is impossible for both the Ukranian military and Russia-affiliated actors to have been *Agents* in a *Conflict.Attack* that brought flight MH-17 down. As stated above, we view hypothesis generation as a clustering task over AIF statements.

3 The UTexas system

3.1 Pipeline

The UTexas system is a hybrid neural-symbolic system. Given a SIN, we (1) identify entry points and compute *hypothesis seeds* (AIF subgraphs) that match the facets mentioned in the SIN. This component is symbolic. We then (2) expand the hypothesis seed by iteratively identifying adjacent statements that belong to the same hypothesis. This component is neural. We finally (3) use a filtering system to remove statements that violate logical constraints. This component is symbolic.

3.2 Neural model

We use a Graph Convolutional Network, GCN for short (Kipf and Welling, 2017; Marcheggiani and Titov, 2018). GCNs compute an embedding for each node in a graph, which is iteratively updated based on the embeddings of neighboring nodes. In our case, this neighbor-aware node embedding for an event or relation node will contain information about all possible arguments stated in the graph. For an entity node, the embedding will hold information about all events/relations that it may be involved in.

We convert the AIF graph into a bipartite graph in which both EREs and statements are nodes, as shown in **Figure 1**, where $A \ldots E$ are EREs, and $S1 \ldots S4$ are statements. Edge labels indicate which ERE is the subject and which is the object of a statement.

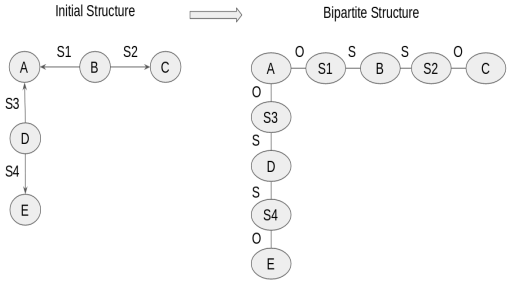

Figure 1: Bipartite Conversion: Turning statements into nodes. In the original graph, statements are edge labels. In the converted graph, statements are nodes, and edge labels indicate subjects and objects.

We implement an iterative update procedure in which EREs and statements are updated separately in order to allow information to propagate at the same rate at which it would in a normal GCN setting. (See Monti et al. (2018) for a similar approach, which they coin "dual-primal" graph convolutional networks.) We call each update step a *layer*. One update of the representation h_e of an ERE *e* is as follows (notation modeled after Marcheggiani and Titov (2018)):

$$\mathbf{h}_e^{l+1} = ReLU\left(\mathbf{W}_{ere}^l \mathbf{h}_e^l + b_{ere}^l\right) + \sum_{s \in N_{stmt}(e)} ReLU\left(\mathbf{W}_{D(e,s)}^l \mathbf{h}_s^l + b_{D(e,s)}^l\right) \quad (1)$$

where l is the current layer of the GCN network, $\mathbf{W}^l{}_{ere}$ is a self-transformation at layer l for the ERE being updated, $N_{stmt}(e)$ is the set of all statements adjacent to the ERE *e*, and $\mathbf{W}^l{}_{D(e,s)}$ is a linear layer for processing adjacent statements, conditioned on whether the ERE is the subject or the object of each statement (hence, *D* for "direction"), or whether the statement is a typing statement.

Similarly, we update the embedding h_s of a statement *s* via

$$\mathbf{h}_s^{l+1} = ReLU\left(\mathbf{W}_{stmt}^l \mathbf{h}_s^l + b_{stmt}^l\right) + \sum_{e \in N_{ere}(s)} ReLU\left(\mathbf{W}_{D(s,e)}^l \mathbf{h}_e^{l+1} + b_{D(s,e)}^l\right) \quad (2)$$

We perform the above procedure twice for a two-layer GCN.

The node embeddings in the GCN are used to iteratively expand a hypothesis seed into a hypothesis. In each expansion step, one statement is added to the hypothesis. All statements that are not in the hypothesis but are adjacent to a hypothesis ERE are *candidate statements*. All EREs adjacent to any included statement are considered included in the hypothesis.

The model scores the relevance of each candidate statement with respect to the current hypothesis through an attention layer.

We experimented with two forms of attention: (a) bilinear attention and (b) concatenative attention (following Luong et al. (2015)), defined as

$$score(\mathbf{h}_{s_c}, \mathbf{h}_{s_i}) = \begin{cases} \mathbf{h}_{s_c}^T \mathbf{W}_{sts} \mathbf{h}_{s_i} & \text{Bilinear} \\ \mathbf{v}_{sts}^T tanh(\mathbf{W}_{sts}[\mathbf{h}_{s_c}; \mathbf{h}_{s_i}]) & \text{Concat} \end{cases} \quad (3)$$

where $\mathbf{h}_{s_c}$ and $\mathbf{h}_{s_i}$ are the final GCN embeddings for candidate statement s_c and hypothesis statement s_i, respectively; $\mathbf{W}_{sts}$ is a linear transformation used when calculating statement-to-statement attention scores; and $\mathbf{v}_{sts}$ is an additional linear layer. For calculating scores between candidate statements and hypothesis EREs, we have a separate set of equations which use a different set of linear layers $\mathbf{W}_{ste}$ and $\mathbf{v}_{ste}$ (for "statement-to-ERE").

We tested different regimes of hypothesis expansion at training time. In a *teacher forcing* setting, the admitted statement at each expansion step is the highest-rated statement that comes from the gold hypothesis. In a *non-forced* setting, the statement rated the highest by the model is added to the hypothesis, even if it is not in the target graph. We experimented with both of those, as well as *annealed teacher forcing*, where the probability of teacher forcing reduces over the course of training.

Our submitted system is trained with concatenative attention (b) with an *annealed teacher forcing* training regime. We discuss some other variants in Section 5.

3.3 Training

We created synthetic data for training, as no in-domain training data was available. To obtain data similar to the evaluation data, we artificially merged AIF graphs of English documents into graph salads.

To approximate the AIDA domain, we extracted 80k documents from Wikipedia that are about wars and conflicts, identified through heuristics over the Wikipedia category structure. A Wikipedia dump from 2013 was used to avoid including documents about the Ukraine crisis (per the "time machine principle"). The 80k Wikipedia documents were then parsed with the ISI docker tool from the GAIA TA1 team (Li et al., 2019). Parsing was successful for 72,167 of the 80k documents. We additionally used 1,893 English documents from the M09 unsequestered document collection of the AIDA program, also parsed with the ISI docker tool. As the parsing tool we used created labels from an outdated version of the AIDA type ontology, we created a rule-based mapping from the old labels to the new and applied it to all data.

Each "graph salad" was created from three source documents. We randomly selected 3 triples of entity or event nodes from the 3 component document graphs such that the nodes in each triple had the same types, and merged each triple into a single node to simulate conflicting information about the node. Nodes with high neighborhood density in the graph were preferred to achieve more challenging hypotheses. One of the original three graphs was chosen as the target graph. The hypothesis seed was created to consist of one of the merged nodes, along with at most two adjacent statements from the target graph. To make the task sufficiently challenging, we required all merged nodes to be accessible from the hypothesis node.

We created "graph salads" from Wikipedia documents alone ("**Wiki-Wiki**"), from a mixture of Wikipedia and AIDA documents ("**Wiki-AIDA**"), and from AIDA documents alone ("**AIDA-AIDA**"). Initial experiments showed that performance deteriorated when AIDA-AIDA salads were included in training, so they were subsequently excluded. Training was via curriculum learning, first on Wiki-Wiki salads and then on Wiki-AIDA. The training/validation/test set totals were 80k/10k/10k mixtures for

the Wiki-Wiki set, 16k/2k/2k mixtures for the Wiki-AIDA set, and 16k/2k/2k mixtures for the AIDA-AIDA set. We use the Adam optimizer (Kingma and Ba, 2015) and reset the optimizer state in between curricula.

4 Results

There are two sub-tasks in SM-KBP Task 3: Task 3a, in which the input is a knowledge graph from a TA2 system; and Task 3b, where the input is a knowledge graph annotated by LDC. We submitted three runs to Task 3a, and one run to Task 3b. We summarize the official evaluation results of our four runs in Table 1.

	GAIA_1-OPERA_3. Colorado_1. UTexas_2	OPERA_4. Colorado_1. UTexas_2	OPERA_TA1a_aditi_V5. OPERA_TA2_aditi_V5. UTexas_2	LDC_2. LDC_2. UTexas_3
edge correctness	0.253	0.3139	0.3655	0.8017
edge coherence	0.3607	0.3691	0.4475	0.8612
KE coherence	0.6108	0.6012	0.5851	0.8979
KE relevance (strict)	0.1566	0.2763	0.3295	0.8312
KE relevance (lenient)	0.6552	0.4543	0.4836	0.9034
argument coverage	0	0.0031	0.0032	0.01

Table 1: Official evaluation results on our 4 runs.

In the table, *LDC_2.LDC_2.UTexas_3* is for Task 3b, while the other three runs are for Task 3a. For the evaluation metrics: *edge correctness* is the percentage of edges in our hypotheses that are correct, *edge coherence* is the percentage of coherent edges, and *KE coherence* is the percentage of coherent KEs (event or relation clusters). *KE relevance (strict)* is the percentage of knowledge elements that are fully relevant, and *KE relevance (lenient)* is the percentage of KEs that are fully or partially relevant. *Argument coverage* is the percentage of edges in all prevailing theories that can be matched in our hypotheses.

From Table 1, we can see that the *LDC_2.LDC_2.UTexas_3* run in Task 3b does significantly better than all Task 3a runs across all but the *argument coverage* metrics. This is not surprising, given that Task 3b uses LDC annotated knowledge graph as input, which should be less noisy than system generated KBs used in Task 3a. The three system KBs perform reasonably well on *KE coherence* and *KE relevance (lenient)*, but are less satisfactory on other metrics. All of our runs, including *LDC_2.LDC_2.UTexas_3*, perform very poorly on *argument coverage*. This could be due to the fact that we do not have any in-domain training data, and the synthetic training data that we constructed has a large statistical discrepancy compared to the actual evaluation data, which we will discuss more in Section 5.

5 Discussion

	Wiki-Wiki	Wiki-AIDA	LDC	KB1	KB2	KB3	KB4
Stmts. available	3.3	3.0	32.9	8.6	554.6	3.1	127.8
Stmts. chosen in hypotheses	-	-	1.7	1.6	1.9	1.8	1.9

Table 2: Graph density measured as average number of adjacent event or relation statements per entity, and average number of event or relation statements for each entity chosen in a hypothesis

Graph density. We had low numbers on argument coverage, the percentage of edges in all prevailing theories being matched in the hypotheses. Overall, our neural model had a tendency towards creating relatively sparse subgraphs, possibly due to the fact that our synthetic training data consisted of graphs that were far less dense than most knowledge bases encountered during the evaluation. Table 2 shows the average number of adjacent event and relation statements per entity in the synthetic training data, in the LDC knowledge base, and in four system knowledge bases that our model was applied to.[1] It is striking

[1]We only submitted results for three out of the four system KBs, as the fourth yielded hypotheses that in postprocessing grew too large to be submitted.

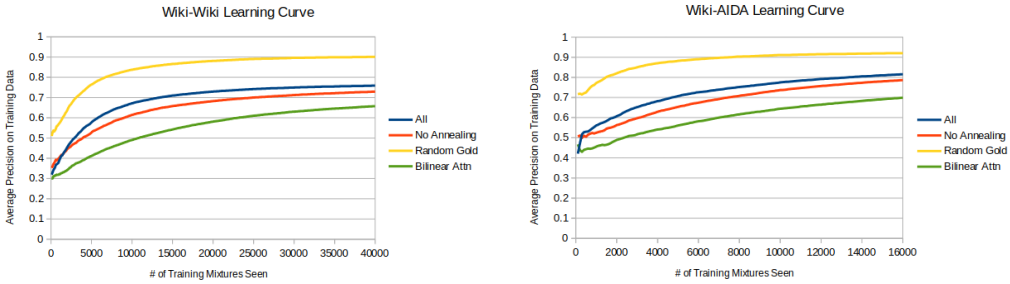

Figure 2: Learning curves for training on Wiki-Wiki and Wiki-AIDA graph salads

	E101	E102	E103
LDC	5%	50%	1%
TA2 KBs	24%	55%	39%

Table 3: Percentage of statements removed during symbolic filtering

that the number of chosen statements for an entity (row 2) remains relatively uniform across KBs, even though the available statements for each entity (row 1) vary dramatically across KBs.

Ablation studies. Learning curves track performance on the training set during learning. Shown in Figure 2 are learning curves while training on mixtures consisting only of Wikipedia articles, and subsequent training on mixtures comprising both Wikipedia and AIDA documents. "All" is the version used in the evaluation. "No Annealing" has a fixed teacher forcing rate; as the learning curves show, it is beneficial to have the teacher forcing rate change during training. "Random Gold" does teacher forcing with a random statement from the target graph rather than the target graph statement ranked highest by the model; this is a setting that was not used in the evaluation but which we tested lately, and it shows promising performance. "Bilinear Attn" is bilinear attention as described in Equation 3, which performs poorly.

Symbolic filtering. Symbolic filtering contained both general filters (for example, relation nodes need two adjacent statements) and domain-specific filters. As an example of the latter, we filtered away a sniper attacking himself. The filters were highly utilized in the system, as can be seen in Table 3, which shows percentages of statements removed during filtering.

Acknowledgements

This research was supported by the DARPA AIDA program under AFRL grant FA8750-18-2-0017. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the view of DARPA, DoD or the US government. We acknowledge the Texas Advanced Computing Center for providing grid resources that contributed to these results. Results presented in this paper were obtained using the Chameleon testbed supported by the National Science Foundation.

References

Ron Bekkerman and Koby Crammer. 2008. One-class clustering in the text domain. In *Proceedings of EMNLP*.

Diederik P. Kingma and Jimmy Lei Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR*.

Thomas N. Kipf and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *Proceedings of ICLR*.

Manling Li, Ying Lin, Joseph Hoover, Spencer Whitehead, Clare R. Voss, Morteza Dehghani, and Heng Ji. 2019. Multilingual entity, relation, event and human value extraction. In *Proceedings of NAACL-HLT*.

Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of EMNLP*.

Diego Marcheggiani and Ivan Titov. 2018. Encoding sentences with graph convolutional networks for semantic role labeling. In *Proceedings of EMNLP*, Copenhagen, Denmark.

Federico Monti, Oleksandr Shchur, Aleksandar Bojchevski, Or Litany, Stephan Gnnemann, and Michael M. Bronstein. 2018. Dual-primal graph convolutional networks. arXiv preprint arXiv:1806.00770.