Advances in Cognitive Systems (2019) 1–6 Submitted 05/2019; published 08/2019

# Predicting State Changes in Procedural Text using Analogical Question Answering

**Danilo Ribeiro** DNRIBEIRO@U.NORTHWESTERN.EDU
**Thomas Hinrichs** T-HINRICHS@NORTHWESTERN.EDU
**Maxwell Crouse** MVCROUSE@U.NORTHWESTERN.EDU
**Kenneth Forbus** FORBUS@NORTHWESTERN.EDU
Qualitative Reasoning Group, Northwestern University, 2233 Tech Drive, Evanston, IL, 60208

**Maria Chang** MARIA.CHANG@IBM.COM
IBM TJ Watson Research Center, 1101 Kitchawan Road, Yorktown Heights, NY 10598

**Michael Witbrock** M.WITBROCK@AUCKLAND.AC.NZ
School of Computer Science, The University of Auckland, Auckland, 1010, New Zealand

## Abstract

Many of the changes in the world that happen over time are characterized by processes. Creating programs that comprehend procedural text (e.g. the stages of photosynthesis) is a crucial task in natural language understanding. In this paper we present a novel approach that uses analogical question answering to predict what state changes affect entities in a paragraph describing a process. We start from the hypothesis that human level QA requires multiple layers of rich, relational representations. For this reason, our model is built on the Companion Cognitive Architecture, which has a large knowledge base and a general-purpose semantic parser. During training, the system uses the output of the semantic parser to automatically construct *query cases*, which link annotated answers to semantic interpretations (i.e. logical statements). When faced with unseen questions, the system retrieves relevant query cases by analogy and uses them to predict sentence level state changes. To obtain a globally consistent sequence of events, we apply common sense constraints over the whole paragraph via dynamic programming. We test our system on AI2's ProPara dataset where our approach achieves results comparable to top-performing models.

## 1. Introduction

Answering questions about paragraphs that describe processes is still a challenging task for machine reading comprehension (MRC) systems. This genre of text is pervasive (e.g. manuals, recipes, road safety rules, scientific protocols, etc.) and understanding them often requires keeping track of how the world's state evolves over time. For instance, consider the paragraph describing photosynthesis in Figure 1. To answer the posed question, an agent would need to infer not only the state changes of each entity in the paragraph, but also the (often implicit) causality between such change events.

"Chloroplasts in the leaf of the plant traps light from the sun. The roots absorb water and minerals from the soil. This combination of water and minerals flows from the stem into the leaf. Carbon dioxide enters the **leaf**. Light, water and minerals, and the carbon dioxide all mix together. This mixture forms **sugar** (glucose) which is what the plant eats. Oxygen goes out of the leaf through the stomata."

**Question**: Where is sugar produced?
**Answer**: In the leaf.

*Figure 1*. Example of a paragraph from ProPara describing the stages of photosynthesis and a follow up question about entities involved in this process.

Most of the recent work on question answering (QA) tasks involving procedural text uses artificial neural networks (Tandon et al., 2018; Das et al., 2018). However, we believe there are two fundamental problems with such models: (1) they have limited semantic understanding and (2) they lack general world knowledge that is easily inspectable. Many times, these neural models rely on surface cues alone to make inferences (Clark, Dalvi, & Tandon, 2018). Such approaches also lead to domain-specific language systems that do not gracefully extend to other tasks.

Our system approaches this task differently. We leverage a large knowledge base and use a general-purpose semantic parser that generates rich semantic interpretations. During training, the semantic parser generates explicit relational representations from the input sentence in the form of logical statements. The system automatically learns a mapping between these logical statements and the annotated labels from the underlying supervised learning task. These learned mappings are stored in the form of *query cases*, which are then analogically retrieved to answer downstream questions. We call this method *Analogical Question Answering* (AQA), which has been previously used to answer questions from Geoquery (Crouse, McFate, & Forbus, 2018a) and to recognize physical processes in science paragraphs (Crouse, McFate, & Forbus, 2018b).

We extend AQA to answer questions from ProPara (Dalvi et al., 2018). In the following sections, we show how our model produces strong results with interpretable predictions. The main contributions of this paper are (1) use AQA to answer questions about procedural text (2) extend the previous AQA ontological mapping to automatically compute ontological similarity weights (3) integrate learned cases with probability scores to help the model handle noise on the training data (4) merge local predictions with common-sense constraints using dynamic programming to generate a consistent sequence of events.

## 2. Related Work

This work is not the first instance of analogical reasoning being used to interpret and answer natural language questions. The work of Barbella & Forbus (2011) introduced analogical dialogue acts (ADAs) as a means of characterizing the intended purpose of discourse contents in setting up an explicit analogy. The system was able to identify ADAs in text and use this information to construct predicate calculus representations of the base and target, leading to knowledge that could be used subsequently in answering questions about novel scenarios.

Prior work by Chang (2016) explored a combination of modalities (language and sketching) to interpret and understand instructional analogies. Given an instructional analogy from a set of analogies used by middle-school teachers, expressed via a combination of simplified English and hand-drawn sketches, her system was able to learn the contents of the domains well enough to answer questions drawn from elementary school science tests about the concepts in the analogies. The sketches were drawn with CogSketch (Forbus et al., 2011), a sketch understanding system that automatically produces visual and conceptual relational representations from digital ink.

The alignment method used during the training portion of AQA is intended to find a mapping from the outputs of a semantic parser to some task-specific logical form. This is similar to the work of Fan & Porter (2004), which introduced Loose-speak, an algorithm that would take in a (possibly malformed) novice user's query and map it to a query more likely to return good results. As part of the mapping process, their algorithm employed a range of heuristics, some of which are analogous to the ontological alignment heuristics of our approach (e.g. type hierarchy similarities).

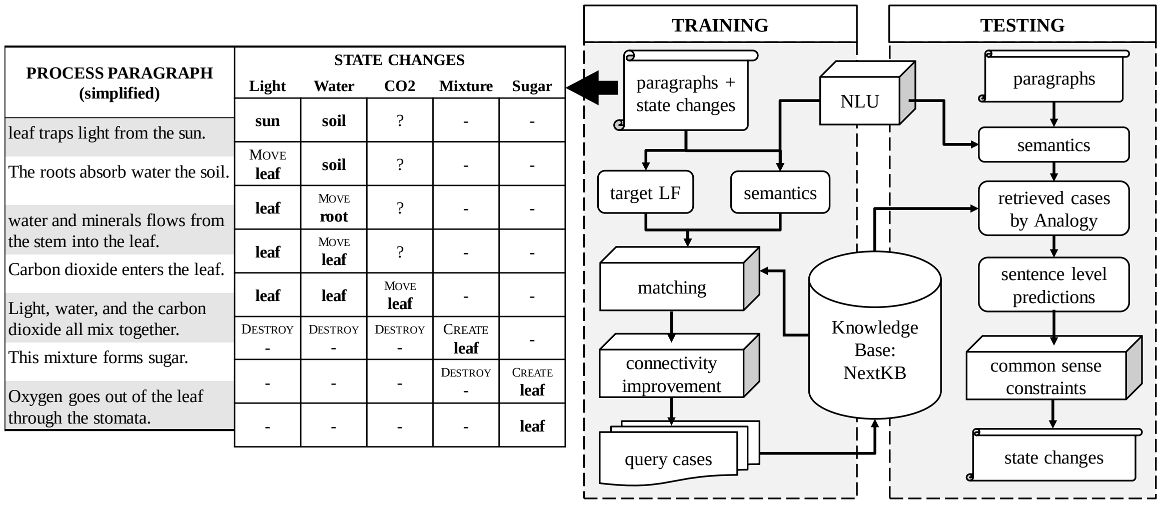
The method of case retrieval is commonly used by case-based reasoning (CBR) systems, however AQA distinguishes from general CBR work since it uses analogy, i.e. structural similarity (Forbus et al., 2017), on the parsed semantics to retrieve cases instead of shallow pattern matching or standard information retrieval methods (Dufour-Lussier et al., 2012; Burke et al., 1997). Furthermore, many CBR systems use domain specific ontologies (Asiimwe et al., 2007) as opposed to general-purpose knowledge bases that integrates varied sources of knowledge.

The related work involving MRC tasks on procedural texts often rely on artificial neural networks. These models encode the state of each entity in a paragraph as hidden vectors that are updated from sentence to sentence (Henaff et al., 2017; Seo et al., 2017). To predict the entity's state value (e.g. location of an entity), models select a span from the paragraph using attention mechanisms (Dalvi et al., 2018). These previously mentioned methods were able to predict state changes on a local level with fair accuracy but suffered from inconsistent global predictions. To mitigate this problem, more recent models attempted to apply common-sense constraints using neural structured prediction (Tandon et al., 2018, Gupta & Durret, 2019) or learn such constraints by explicitly constructing dynamic knowledge graphs (Das et al., 2018). Another artificial neural network model (Du et al., 2019) explored the fact that different paragraphs describing the same process (e.g. photosynthesis) will usually contain consistent labels for the same entity, improving their model by adding such consistency bias.

The work of Clark et al. (2018) used a hybrid approach, not completely relying on neural networks. Their model integrated off-the-shelf semantic role labeling with VerbNet derived rules to map each sentence to its effects on the world state, instead of relying on hidden vectors. They also compiled a list of common-sense rules that were applied as a post-processing step to ensure global consistency.

## 3. Problem Definition

We are addressing the task of procedural text comprehension and we use the ProPara benchmark (Dalvi et al., 2018) to evaluate our system. ProPara contains 488 crowdsourced paragraphs and



| **PROCESS PARAGRAPH (simplified)** | **STATE CHANGES** | | | | |
|---|---|---|---|---|---|
| | **Light** | **Water** | **CO2** | **Mixture** | **Sugar** |
| leaf traps light from the sun. | **sun** | **soil** | ? | - | - |
| The roots absorb water the soil. | MOVE **leaf** | **soil** | ? | - | - |
| water and minerals flows from the stem into the leaf. | **leaf** | MOVE **root** | ? | - | - |
| Carbon dioxide enters the leaf. | **leaf** | MOVE **leaf** | ? | - | - |
| Light, water, and the carbon dioxide all mix together. | **leaf** | **leaf** | MOVE **leaf** | - | - |
| This mixture forms sugar. | DESTROY - | DESTROY - | DESTROY - | CREATE **leaf** | - |
| Oxygen goes out of the leaf through the stomata. | - | - | - | DESTROY - | CREATE **leaf** |
| | - | - | - | - | **leaf** |

*Figure 2*. Overview of system components and data flow during the training and testing phase. The input data is comprised of paragraphs describing processes (e.g. photosynthesis), together with a detailed annotation of how the state of each participant (e.g. Light, and Water) change throughout the paragraph.

3100 sentences total. It is different from other procedural text datasets such as bAbI (Weston et al., 2015) and SCoNE (Long et al., 2016) because it contains non-synthetic natural language paragraphs describing various kinds of real-world processes. The topics are diverse, ranging from mundane tasks (e.g. "how to run a dishwasher") to scientific explanations (e.g. "how are stars formed?"). The ProPara benchmark is especially challenging because it contains many events affecting entities that are not explicitly mentioned in the text. In some cases, the MRC system is required to have common-sense knowledge to correctly infer the state changes (Clark et al., 2018).

ProPara paragraphs were labeled by human annotators. These annotators identified relevant entities in the process, which they called participants, and how these entities' properties (specifically their existence and location) change over time. More formally, the data consists of a set of paragraphs about processes, each containing a list of $N$ sentences $s_1, \dots, s_N$ and $M$ participants $p_1, \dots, p_M$. The annotations that describe the location and existence of each participant before and after each sentence are given in the form of a grid with $N + 1$ rows and $M$ columns, represented as $G_{n,m} : n \in \{0, \dots, N\}$ and $m \in \{0, \dots, M - 1\}$. The grid cells contain three types of states, either the participant does not exist (labeled as "-") or it exists but the location cannot be inferred from the text (labeled as "?") or it exists, and the participant's location is known (labeled with the participant's location). Figure 2 shows an example of a process paragraph together with the annotated state changes.

## 4. Model

Our model uses Step Semantics (Forbus et al. 2019) and AQA to bridge between natural language semantics and task semantics. Step Semantics combines QP theory (Forbus, 1984) and FrameNet

```
(isa participant1 Participant)
(isa event1 CreationEvent)
(isa tolocation1 Location)
(outputsCreated event1 participant1)
(outputsCreatedLocation event1 tolocation1)

(isa participant2 Participant)
(isa fromlocation2 Location)
(isa event2 DestructionEvent)
(inputsDestroyed event2 participant2)
(inputsDestroyedLocation event2 fromlocation2)

(isa participant3 Participant)
(isa event3 MovementEvent)
(isa fromlocation3 Location)
(isa tolocation3 Location)
(objectMoving event3 participant3)
(fromLocation event3 fromlocation3)
(toLocation event3 tolocation3)
```

*Figure 3*. Target logical forms. Relations and collections from OpenCyc ontology in bold. Remaining tokens in the logical form represent open variables.

(Fillmore et al. 2001) to represent the changes expressed in language describing the steps of a process. For ProPara, the relevant step types are changes in existence (i.e. Creation/Destruction events) and changes in property (i.e. Movement events), as previously mentioned. Mappings from lexemes to FrameNet descriptions, which in turn are mapped to the OpenCyc ontology, provide a stable foundation for building representational bridges to the task-specific representations. As shown below, the training process provides automatic disambiguation of language, using top-down constraints of constructing step representations from the text. Figure 2 has an overview of our system components and how the data flows through the system.

## 4.1 Background

We build on the Companion cognitive architecture, which incorporates analogical learning and reasoning capabilities, along with a natural language understanding system and visual processing capabilities. It uses the NextKB[1] knowledge base, which integrates material from OpenCyc, FrameNet, WordNet, VerbNet, a large-scale lexicon (McFate & Forbus, 2011) and support for analogical and qualitative reasoning. The general-purpose nature of these knowledge resources simplifies building task models. The analogical capabilities used to retrieve query case is matching via the Structure Mapping Engine SME (Forbus et al., 2017). SME compares two structured, relational representations, and produces one or more mappings. Mappings consist of (1) a set of correspondences, specifying what entities and statements in one description go with entities and statements in the other, (2) a score indicating how similar the descriptions are, and (3) candidate inferences that project information from one description to the other. The method discussed here uses analogical matching to apply knowledge learned during training to new sentences.

[1] NextKB is available via a CC-Attribution license, from http://www.qrg.northwestern.edu/nextkb/index.html

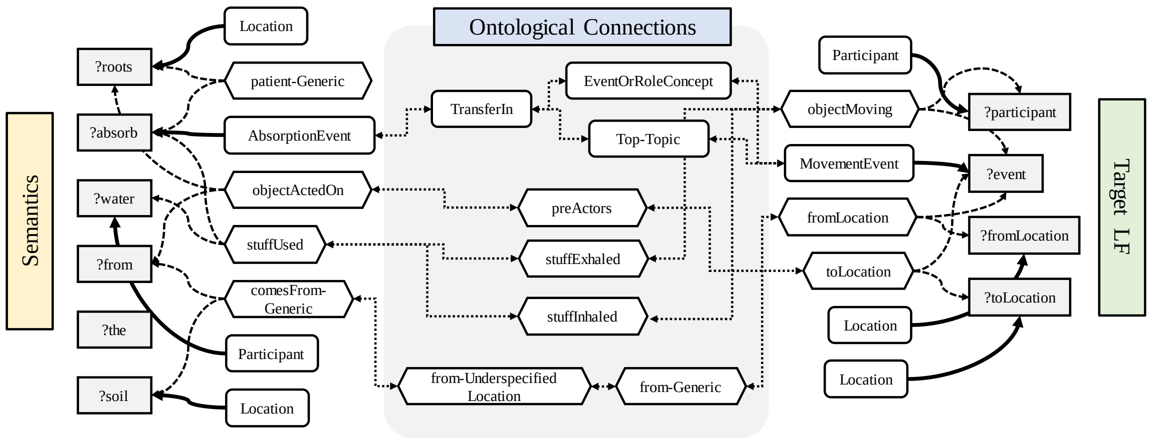


*Figure 4*. Overview of the matching between semantics and target logical forms (Target LF). Trivial connections (e.g. equality) and full semantics are left out for simplicity. The semantics were generated for the sentence "Roots absorb water from the soil". Gray squares contain logical variables. The full, dashed, and dotted arrows represent `isa` statements, role relations, and ontological structural relations, respectively.

The natural language system (Tomai & Forbus 2009) called EA NLU combines Allen's (1994) syntactic parser with ideas from Discourse Representation Theory (Kamp & Reyle, 1993) to produce semantic representations from sentences. Specifically, EA NLU produces multiple choice sets of logical statements in the Cyc representational language. Each choice set is distinct and internally consistent. Making an interpretation choice amounts to choosing one set of logical statements as the correct interpretation and ruling out the other sets. By explicitly representing syntactic and semantic choice sets, it is possible to make interpretation choices in a task-specific way that can be learned during training as described below.

## 4.2 Query Cases Construction

Training proceeds in a similar way to that described in (Crouse *et al*. 2018b). Given a target logical form and some natural-language text, a query case is generated that pairs the most relevant semantic forms (from a semantic parse of the text) to the target logical form (shown in Figure 3). The pairing occurs in two stages. First an initial one-to-one matching pairs each expression of the target logical form to the single most relevant semantic form. Then, the semantics that are used in the matching are connected using a conflict-aware minimal Steiner Tree algorithm.

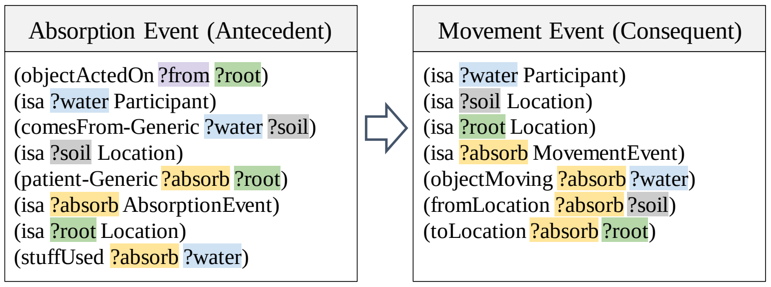
Figure 4 illustrates the one-to-one matching step during the creation of a query case for the simplified ProPara sentence *"The roots absorb water from the soil"*. The training data has this sentence labeled as a movement event for participant *"water"*. In this process the tokens associated with participant $p_i$ and its locations in a sentence $s_i$ are identified through partial string matching. The semantic forms are then augmented with `isa` statements linking the corresponding discourse variables to the collections `Participant` and `Location`. The objective of the graph matching stage is to select a minimal set of semantic forms that justify the observed target logical forms. To do this, a bipartite matching algorithm finds the best one-to-one assignment of expressions from the target logical form to expressions from the semantic parse with respect to ontological similarity, structural overlap, and conflict avoidance. It also ensures each semantic form in the matching is non-conflicting (i.e. does not contain semantic forms that correspond to conflicting word senses or syntactic parse choices).

To determine the ontological similarity of two expressions, first the concepts (i.e. predicates, functions, collections, and entities) of both expressions are extracted into sets $A$ and $B$. Then, a connection graph search (Faloutsos, McCurley, & Tomkins, 2004) is performed between each pair in $A \times B$ through the structural facts of the knowledge base. Large knowledge bases include structural relations which are used to define structural relationships between entities, collections, predicates, and functions in the knowledge base. Examples from the OpenCyc ontology in NextKB include type argument constraints for predicates and functions (e.g. `argIsa`), instance relations (e.g. `isa`), and type hierarchies (e.g. `genls` / `specs`). However, NextKB maintains a much larger and richer set of structural relations (also obtained from OpenCyc), with relations like `resultIsa` denoting that the result of a given function is an instance of the specified collection.

The set of all structural facts (those facts whose predicate is a structural relation) can be thought of as a graph with each edge being a structural fact and each vertex being an entity within a structural fact. A connection subgraph between two entities is a set of paths connecting the two entities together through the graph formed by structural facts. This weighting scheme gives higher weights to those pairs of entities that can be connected through sparser regions of the knowledge base. The final ontological similarity between a pair of expressions is then the sum of weights between each of their underlying concepts. More formally, let $\Psi = \{P_1, \dots, P_U\}$ be the set of paths between two nodes $N_1$ and $N_2$. $P_u = (x_{u,1}, \dots, x_{u,V})$ be the vertices in each of these paths. Let $\deg(x_{u,v})$ be the out-degree of a vertex $x_{u,v}$, the ontological similarity weight ($osw$) between nodes $N_1$ and $N_2$ is given by:

$$osw(N_1, N_2) = \sum_{P_u \in \Psi} \prod_{x_{u,v} \in P_u} \frac{1}{deg(x_{u,v})}$$

Following Crouse et al. (2018b), we consider the set of expressions in the target logical form and the set of expressions in the semantic parse to be graphs. The structural overlap score of a matching is given by the number of times two expressions in the matching are neighbors in their respective graphs. More concretely, consider the semantic forms of two words in the same sentence versus the semantic forms of two words separated by several sentences. The semantic forms for words in the same sentence will be much closer in their respective graphs than those separated by several sentences, which will yield higher structural overlap scores. The conflict avoidance score of an edge is a function of the number of expressions in the semantic parse of the text that conflict with the semantic form in the edge (e.g. due to alternative word senses, parse choices, etc.).



Absorption Event (Antecedent)

```
(objectActedOn ?from ?root)
(isa ?water Participant)
(comesFrom-Generic ?water ?soil)
(isa ?soil Location)
(patient-Generic ?absorb ?root)
(isa ?absorb AbsorptionEvent)
(isa ?root Location)
(stuffUsed ?absorb ?water)
```

Movement Event (Consequent)

```
(isa ?water Participant)
(isa ?soil Location)
(isa ?root Location)
(isa ?absorb MovementEvent)
(objectMoving ?absorb ?water)
(fromLocation ?absorb ?soil)
(toLocation ?absorb ?root)
```

*Figure 5*. Query case example. Corresponding variables highlighted with same colors. Both antecedent and consequent logical forms are stored in a single query case.

The matching procedure is a hill-climbing algorithm that starts by greedily finding the bipartite matching that optimizes for only ontological similarity and conflict avoidance. Then, it searches the conflict-free 2-exchange neighborhood of the matching (i.e. the set of all matchings obtainable by exchanging at most two edges of the matching for two other edges, where the new matching contains no conflicting semantic forms) until it can no longer find a matching more optimal with respect to each of the three criteria listed above.

The result of the matching procedure is a conflict-free assignment of semantic forms to logical expressions that is potentially disconnected (i.e. it is possible for expressions to share no entities in common, even if they are drawn from the same sentence). This is both a theoretical (i.e. there is a lack of a coherent justification and context that ties all the semantics together to explain a given target logical form) and a practical issue (SME performs more effectively when there is more interconnected structure between entities). To solve this, the second stage finds a minimum conflict-free Steiner tree in the graph formed by the semantic parse of the text that connects each of the entities from the semantic forms of the matching together.

For example, in Figure 4 the role relation `patient-Generic` cannot be connected by the initial one-to-one matching but will be added to the query case after the Steiner Tree optimization algorithm. This can be thought of as providing the minimum context connecting each of the most relevant semantic forms to the input logical expressions. The final generated query case is shown in Figure 5. Query cases that do not cover enough of the target logical form are discarded (e.g. when event or participant are not matched). All the remaining cases are stored in the knowledge base for subsequent analogical retrieval. Both the antecedent from the semantics and the consequent from the target logical form will be part of such query cases.

## 4.3 State Change Predictions

During the testing phase, our system predicts the entire state change grid given a paragraph and its participants (which are known *a priori*). The first step is to evaluate each sentence locally. Sentences are converted into a set of semantic forms. Again, the semantic forms are extended with `isa` statements linking the corresponding discourse variables to the collections `Participant` and `Location`. The participants are identified by partial string matching and all remaining nouns in the sentence are considered as potential locations. The set of semantic forms are then used by SME to retrieve all the query cases that could potentially match the test sentence.

In previous papers, AQA was used for answering factoid questions. It would combine the cases to generate a query that could retrieve facts stored in the KB. Our goal does not require probing the KB since the answers are contained within the paragraphs themselves. Instead, we use the consequent of individual cases to infer the state change and the location of a certain participant. This new problem can now be viewed as a ranking problem: which retrieved case will contain the consequent that would correctly predict the state change? The previous AQA systems would rank cases by considering how much coverage, overlap and conflict could be found between the antecedents and the semantic choice sets. Those scores proved to be not as useful for our task since only one case is chosen.

To design an appropriate ranking score, we considered the fact that retrieval of cases by analogy can suffer from noisy data. If the training data contains an example of state change that was incorrectly labeled, the system could retrieve that query case and possibly make an incorrect prediction. In order to mitigate this problem, we look at the statistics over all stored cases. We compute the antecedent correlation score that captures how well the query case antecedents can predict the event in its consequent logical form. Let $\mathrm{Ev_c}$ and $\mathrm{Ev_a}$ be the retrieved case's event ontological collection from its consequent and antecedents, respectively. Let the set $\Omega = \{R_1, \dots, R_\omega\}$ be the relations in the antecedents. Let $\lambda$ be a hyper-parameter that influences how the probabilities will affect the correlation score compared to the size of $\Omega$. Then the correlation score is given by:

$$score = |\Omega| + \lambda\left(\sum_{R_\mathrm{i} \in \Omega} \mathrm{P}(\mathrm{Ev_c} \mid \mathrm{Ev_a}, R_\mathrm{i})\right)$$

Instead of choosing the highest scoring case as our final prediction and discarding the retrieved case with lower scores, our algorithm feeds all retrieved cases with distinct consequents, together with their scores, to the next prediction step.

## 4.4 Common Sense Constraints

At this point, we have all the conflicting sentence level predictions for each participant. In order to obtain an output that is globally consistent we apply common sense constraints using dynamic programming. Below is the list of both hard and soft common-sense constraints:

- **Inertia**: participants will not change their state or location until an event occurs.
- **Collocation**: when a participant A is destroyed and another participant B is created in a given sentence, we will assume A was converted to B. Therefore, if the location of A was known prior to this conversion event, and the location of B is unknown, then we assume that B was created at the same location.
- **Existence**: if a certain participant already exists it cannot be created.
- **Absence**: if a certain participant does not exist it cannot be moved or destroyed.

**Algorithm 1** COMMON-SENSE-OPTIMIZATION

```
Input: Table P. Each entry P[n][m] is a list of predicted state changes for
sentence $s_n$ and participant $p_m$
Parameters: P-penalty, R-penalty
Output: State change grid
 1: Initialize table D with size (N+1, M, 2).
 2: for n from 1 to N do
 3:     for m from 0 to M-1 do
 4:         Let predictions = P[n][m+1] and let w be 0
 5:         for p in predictions do
 6:             if event(p) = CreationEvent then
 7:                 Let w be 1 if R-penalty criteria is met
 8:                 D[n][m][0] ← D[n-1][m][1] + score(p) - (w * R-penalty)
 9:             if event(p) = DestructionEvent then
10:                 D[n][m][1] ← D[n-1][m][0] + score(p)
11:             if event(p) = MovementEvent then
12:                 D[n][m][0] ← D[n-1][m][0] + score(p)
13:             if event(p) = Participant-Found then
14:                 Let w be 1 if P-penalty criteria is met
15:                 D[n][m][1] ← D[n][m][1] - (w * P-penalty)
16:             end if
17:             D[n][m][0] ← D[n][m][0]
18:             D[n][m][1] ← D[n][m][1]
19:         end for
20:     end for
21: end for
22: return RECONSTRUCT-OUTPUT-GRID(D)
```

- **Presence**: if a participant is referenced in a sentence, it is likely to exist during that step. For this reason, we penalize global predictions that assume a participant does not exist even when it is mentioned. This presence penalty value is called *P-penalty*.
- **Re-existence**: for the most part, objects do not pop in and out of existence very often. We introduced another penalty value called *R-penalty* that is applied when a participant is destroyed in one sentence and created in the following sentence.

Algorithm 1 shows in detail how we enforce the common-sense constraints. The constraints *Existence* and *Absence* are hard constraints while *Presence* and *Re-existence* can be skipped with a cost (i.e. *P-penalty* and *R-penalty*, respectively). Note that the dynamic programming table updates that use the "←" operator are only applied if the new value is greater than the DP table value or if the DP cell was not set before. The RECONSTRUCT-OUTPUT-GRID function is not shown in detail, but it takes the output of the dynamic programming algorithm and reconstructs the final state change grid, applying *Inertia* and *Collocation* constraints when needed. Tandon et al. (2018) and Clark et al. (2018) also applied common sense constrains to steer their model away from inconsistent predictions, but they used a different set of constraints and hard rules / back-tracking instead of dynamic programming.

*Table 1*. Accuracy and F1 results for ProPara dataset questions on both sentence-level evaluation and paragraph level evaluation.

| Technique | Model | Sentence-level | | | | | Document-level | | |
|---|---|---|---|---|---|---|---|---|---|
| | | Cat-1 | Cat-2 | Cat-3 | Micro-avg | Macro-avg | Preci. | Recall | F1 |
| Hybrid | PROCOMP | 57.14 | 20.33 | 2.40 | 26.24 | 26.62 | - | - | - |
| Artificial NN | QRN | 52.37 | 15.51 | 10.92 | 26.49 | 26.26 | 55.5 | 31.3 | 40.0 |
| | EntNet | 51.62 | 18.83 | 7.77 | 25.96 | 26.07 | 50.2 | 33.5 | 40.2 |
| | PROLOCAL | 62.65 | 30.50 | 10.35 | 33.96 | 34.50 | **77.4** | 22.9 | 35.3 |
| | PROGLOBAL | 62.95 | 36.39 | 35.90 | 45.37 | 45.08 | 46.7 | 52.4 | 49.4 |
| | PROSTRUCT | - | - | - | - | - | 74.2 | 42.1 | 53.7 |
| | LACE | - | - | - | - | - | 75.3 | 45.4 | 56.6 |
| | KG-MRC | 62.86 | 40.00 | 38.23 | 46.62 | 47.03 | 64.5 | 50.7 | 56.8 |
| | NCET | **73.68** | **47.09** | **41.03** | **53.93** | **53.97** | 67.1 | **58.5** | **62.5** |
| Analogy | AQA (Ours) | 61.58 | 40.14 | 18.59 | 39.38 | 40.10 | 62.0 | 45.1 | 52.3 |

## 5. Experiments

In this section we report how our model compares with published results from previous procedural text MRC systems on ProPara. The only model that uses separate sources of knowledge and does not fully rely on neural networks to predict state changes is PROCOMP from Clark et al. (2018). All remaining models are mostly built on neural networks, namely Query Reduction Networks (QRN) by Seo et al. (2017), Recurrent Entity Networks (EntNet) by Henaff et al. (2017), PROLOCAL and PROGLOBAL by Dalvi et al. (2018), PROSTRUCT by Tandon et al. (2018), KG-MRC by Das et al. (2018), LACE by Du et al. (2019), and NCET by Gupta and Durrett (2019).

### 5.1 Evaluation and Results

Previous work on ProPara evaluated their system using two different metrics. The first evaluation proposed by Dalvi et al. (2018) focused mostly on sentence level predictions. The following three categories of questions were asked for each participant p in a paragraph:

Cat-1: Is p created (destroyed, moved) in the process?
Cat-2: When is p created (destroyed, moved)?
Cat-3: Where is p created (destroyed, moved from/to)?

On the other hand, the evaluation proposed by Tandon et al. (2018) tests the system in a coarser paragraph level. The following types of questions are asked:

Q1: What are the inputs to the process?
Q2: What are the outputs of the process?
Q3: What conversions occur, when and where?

Q4: What movements occur, when and where?

Such questions are templated and can be deterministically answered from the output state changes grid. For the first metric (sentence-level evaluation) the results are shown for each category separately, as well as their average. For the second metric (paragraph-level evaluation) the answers are combined into a F1 score. Table 1 shows our results along with previously mentioned models. In the sentence-level evaluation we outperform all other models in Cat-2 except CNET, while having strong results for Cat-1. Our model has worse performance on Cat-3 compared to PROGLOBAL, KG-MRC and CNET, which brings our average accuracy down. We believe that such problems were mostly caused by incomplete semantic parses and by the simplifying assumption that locations should be within the same sentence that a state change event occurs.

The previous published results of document-level evaluation are not broken down into sub-categories, but we believe our results are suffering from the same problem as in the sentence-level evaluation: the system cannot find the location of the participants as reliably as other models. However, we believe these are strong results, especially considering the novelty of our approach.

## 5.2 Error Analysis

To further evaluate AQA we highlight some common mistakes made by the system. We randomly selected five paragraphs from the ProPara dev set and manually categorized the errors as follows:

- **Implicit state changes:** 33% of the errors were due to state changes that happen even when they are not explicitly mentioned in the text. For example, the sentence *"The trash truck travels to the landfill"* has an implicit movement of the *trash bags* (that were previously placed inside the truck) to the *landfill*. Such cases are especially challenging to handle since they often require common-sense reasoning and implicit references among words.
- **Parse errors or missing semantics:** 28% of the errors were caused by incorrect or incomplete semantic parses. Such errors are caused by various reasons including malformed input sentences (e.g. typos); missing lexical, syntactical or semantic information in the knowledge base; challenging sentences where the participant and the location are very far apart; and errors during pronoun co-reference mapping.
- **Incorrectly retrieved query cases:** 21% of the errors happened when a query case was missing or incorrectly retrieved during the test phase. This could happen either when the training data does not contain certain events (e.g. the event *bag up* in the sentence *"Trash is bagged up"* is not part of the training data) or when the sentence seems to imply a state change (e.g. the sentence *"Large amounts of sediment gradually pile on top of the original sediment"* wrongly retrieves cases that infer that *sediment* moved) even though no change happened when you consider the context.
- **Incorrect ranking of scores:** 14% of the errors were caused by incorrect ranking of retrieved cases or order of events generated by the common-sense optimization phase. The challenge here is that some events may retrieve cases with conflicting consequences.

- **Noisy data:** 4% of the errors occurred due to data that was incorrectly labeled by the human annotators.

## 6. Conclusion

Natural language understanding is a difficult problem. It takes years for people to attain basic fluency and they continue to improve over decades. Human learning is cumulative: People do not relearn everything from scratch when tackling a new task or domain. Our approach exploits this insight: By reusing the semantic parser and knowledge base across the entirety of language-using tasks that an architecture faces, improvements are shared, and learning is simplified.

In this paper we show how AQA can be applied to answer questions about procedural text, testing our system on the ProPara benchmark. Our model seems to be the first that does not rely on artificial neural networks to make inferences, and yet we obtain results that are comparable to the best performing models. The use of symbolic language and ontologies makes it possible to inspect the stored and retrieved query cases and understand how the model is making inferences. To a certain degree, our system has the advantage of being more interpretable. Another advantage is that our architecture is tied to a large knowledge base. Error analysis by Dalvi et al. (2018) indicates that 37% of their errors were also due to implicit creation/destruction events (e.g. if water cools down enough, it will become ice or snow, even if this is not explicitly stated in the sentence). We plan to use NextKB in future work to better handle questions that require the use of common-sense knowledge.

## Acknowledgements

We are grateful to Joseph A. Blass, Samuel Hill, Felicity M. Lu-Hill and Constantine Nakos for their comments and edits. This research was supported in part by the Machine Learning, Reasoning, and Intelligence Program of the Office of Naval Research.

## References

Allen, J. (1994) *Natural Language Understanding. (2nd ed)*. Redwood City, CA.: Benjamin/Cummings.

Asiimwe, S., Craw, S., Taylor, B., & Wiratunga, N. (2007). Case authoring: from textual reports to knowledge-rich cases. In International Conference on Case-Based Reasoning (pp. 179-193). Springer, Berlin, Heidelberg.

Barbella, D. M., & Forbus, K. D. (2011). Analogical Dialogue Acts: Supporting Learning by Reading Analogies in Instructional Texts. In *Proceedings of AAAI-11*, San Francisco, CA.

Burke, R. D., Hammond, K. J., Kulyukin, V., Lytinen, S. L., Tomuro, N., & Schoenberg, S. (1997). Question answering from frequently asked question files: Experiences with the faq finder system. *AI magazine*, 18(2), 57-57.

Chang M. (2016). Capturing Qualitative Science Knowledge with Multimodal Instructional Analogies (Doctoral dissertation, Northwestern University).

Clark, P., Dalvi, B., & Tandon, N. (2018). What happened? Leveraging VerbNet to predict the effects of actions in procedural text. *arXiv preprint arXiv:1804.05435.*

Crouse, M., McFate, C., & Forbus, K. (2018a). Learning from Unannotated QA Pairs to Analogically Disambiguate and Answer Questions. In *Proceedings of AAAI*, 2018.

Crouse, M., McFate, C., & Forbus, K. (2018b). Learning to Build Qualitative Scenario Models from Natural Language. In *Proceedings of the 31st Workshop on Qualitative Reasoning*, 2018.

Dalvi, B., Huang, L., Tandon, N., Yih, W., & Clark, P. (2018). Tracking state changes in procedural text: a challenge dataset and models for process paragraph comprehension. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1595–1604, New Orleans, Louisiana. Association for Computational Linguistics.

Das, R., Munkhdalai, T., Yuan, X., Trischler, A., & McCallum, A. (2018). Building Dynamic Knowledge Graphs from Text using Machine Reading Comprehension. In *Proceedings of the International Conference on Learning Representations (ICLR)*, New Orleans, Louisiana.

Du, X., Dalvi, B., Tandon, N., Bosselut, A., Yih, W., Clark, P., Cardie, C. (2019). Be Consistent! Improving Procedural Text Comprehension using Label Consistency. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Minneapolis, Minnesota. Association for Computational Linguistics.

Dufour-Lussier, V., Ber, F. L., Lieber, J., Meilender, T., & Nauer, E. (2012). Semi-automatic annotation process for procedural texts: An application on cooking recipes. *arXiv preprint arXiv:1209.5663*.

Faloutsos, C., McCurley, K., & Tomkins, A. (2004). Fast discovery of connection subgraphs. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, Chicago, Illinois.

Fan, J., & Porter, B. (2014). Interpreting loosely encoded questions. In *Proceedings of AAAI-14* (pp. 399-405).

Fillmore, C. J., Wooters, C., & Baker, C. F. (2001). Building a large lexical databank which provides deep semantics. In *PACLIC-15* (pp. 3-26).

Forbus, K. (1984). Qualitative process theory. *Artificial Intelligence*, 24, 85-168.

Forbus, K., Chang, M., Ribeiro, D., Hinrichs, T., Crouse M., & Witbrock M. (2019). Step Semantics: Representation for State Changes in Natural Language. In *Proceedings of the AAAI 2019 Workshop on Complex Question-Answering*.

Forbus, K., Ferguson, R. W., & Gentner, D. (2017). Extending SME to Handle Large-Scale Cognitive Modeling. *Cognitive Science*, 41(5), 1152-1201.

Forbus, K., Usher, J., Lovett, A., Lockwood, K., & Wetzel, J. (2011). CogSketch: Sketch understanding for cognitive science research and for education. Topics in Cognitive Science, 3(4), 648-666.

Gupta, A. & Durrett, G. (2019). Tracking Discrete and Continuous Entity State for Process Understanding. *arXiv preprint arXiv:1904.03518.*

Henaff, M., Weston, J., Szlam, A., Bordes, A., & LeCun Y. (2017). Tracking the World State with Recurrent Entity Networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

Hans Kamp and Uwe Reyle (1993). *From Discourse to Logic: Introduction to Model Theoretic Semantics of Natural Language. Kluwer Academic Dordrecht*; Boston.

Long, R., Pasupat, P., & Liang, P. (2016). Simpler context-dependent logical forms via model projections. In *ACL-16*.

McFate, C. J., & Forbus, K. D. (2011). NULEX: an open-license broad coverage lexicon. In *Proceedings NACL-HLT-11*

Seo, M., Kembhavi, A., Farhadi, A., & Hajishirzi H. (2017). Bidirectional attention flow for machine comprehension. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

Tandon, N., Dalvi, B., Grus, J., Yih, W., Bosselut, A., & Clark P. (2018). Reasoning about actions and state changes by injecting commonsense knowledge. In *EMNLP*, 2018.

Tomai, E., & Forbus, K. D. (2009). EA NLU: Practical Language Understanding for Cognitive Modeling. In *FLAIRS Conference* 2009.

Weston, J., Bordes, A., Chopra, S., Rush, A. M., van Merriënboer, B., Joulin, A., & Mikolov, T. (2015). Towards ai-complete question answering: A set of prerequisite toy tasks. arXiv preprint arXiv:1502.05698.