Modeling Biological Processes for Reading Comprehension

Jonathan Berant[*] **Vivek Srikumar**[*] **Pei-Chun Chen, Brad Huang and Christopher D. Manning**
Stanford University, Stanford

Abby Vander Linden and Brittany Harding
University of Washington, Seattle

Peter Clark
Allen Institute for Artificial Intelligence, Seattle

Abstract

Machine reading calls for programs that read and understand text, but most current work only attempts to extract facts from redundant web-scale corpora. In this paper, we focus on a new reading comprehension task that requires complex reasoning over a single document. The input is a paragraph describing a biological process, and the goal is to answer questions that require an understanding of the relations between entities and events in the process. To answer the questions, we first predict a rich structure representing the process in the paragraph. Then, we map the question to a formal query, which is executed against the predicted structure. We demonstrate that answering questions via predicted structures substantially improves accuracy over baselines that use shallower representations.

1 Introduction

The goal of machine reading is to develop programs that read text to learn about the world and make decisions based on accumulated knowledge. Work in this field has focused mostly on macro-reading, i.e., processing large text collections and extracting knowledge bases of facts (Etzioni et al., 2006; Carlson et al., 2010; Fader et al., 2011). Such methods rely on redundancy, and are thus suitable for answering common factoid questions which have ample evidence in text (Fader et al., 2013). However, reading a single document (micro-reading) to answer comprehension questions that require deep reasoning is currently beyond the scope of state-of-the-art systems.

In this paper, we introduce a task where given a paragraph describing a process, the goal is to answer reading comprehension questions that test understanding of the underlying structure. In particular, we consider processes in biology textbooks such as this excerpt and the question that follows:

> "... **Water is split**, providing a source of electrons and protons (hydrogen ions, H^+) and giving off O_2 as a by-product. **Light absorbed** by chlorophyll drives a **transfer of the electrons and hydrogen ions** from water to an acceptor called $NADP^+$ ..."

Q What can the splitting of water lead to?

a Light absorption

b Transfer of ions

This excerpt describes a process in which a complex set of events and entities are related to one another. A system trying to answer this question must extract a rich structure spanning multiple sentences and reason that *water splitting* combined with *light absorption* leads to *transfer of ions*. Note that shallow methods, which rely on lexical overlap or text proximity, will fail. Indeed, both answers are covered by the paragraph and the wrong answer is closer in the text to the question.

We propose a novel method that tackles this challenging problem (see Figure 1). First, we train a supervised structure predictor that learns to extract entities, events and their relations describing the biological process. This is a difficult problem because events have complex interactions that span multiple sentences. Then, treating this structure as a small knowledge-base, we map questions to formal queries that are executed against the structure to provide the answer.

Micro-reading is an important aspect of natural language understanding (Richardson et al., 2013; Kushman et al., 2014). In this work, we focus specifically on modeling processes, where events and entities relate to one another through complex interactions. While we work in the biology

[*] Both authors equally contributed to the paper.

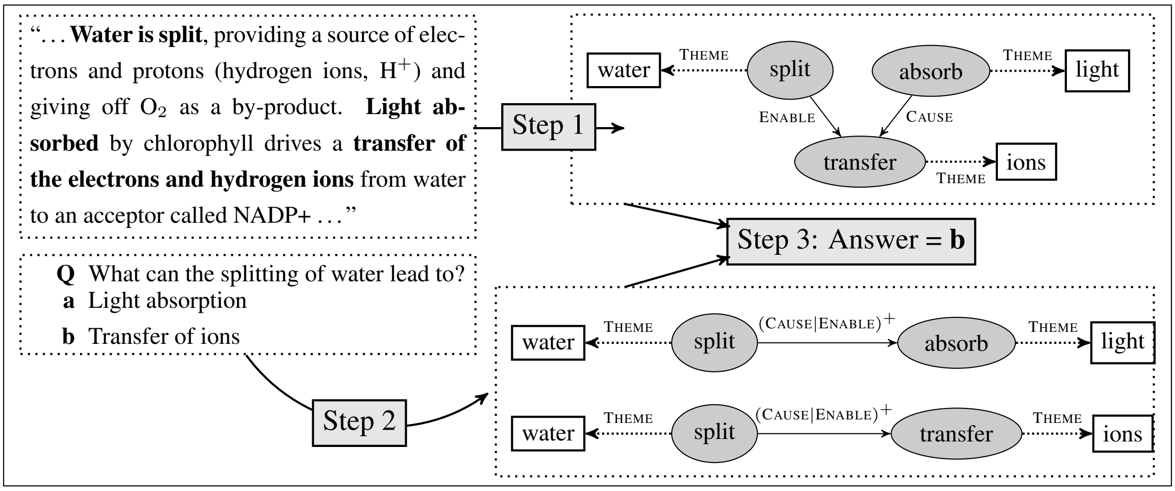

"... Water is split, providing a source of electrons and protons (hydrogen ions, H+) and giving off O2 as a by-product. Light absorbed by chlorophyll drives a transfer of the electrons and hydrogen ions from water to an acceptor called NADP+ ..."
Q What can the splitting of water lead to?
a Light absorption
b Transfer of ions
Step 1
Step 2
Step 3: Answer = b
water
THEME
split
absorb
THEME
light
ENABLE
CAUSE
transfer
THEME
ions
water
THEME
split
(CAUSE|ENABLE)+
absorb
THEME
light
water
THEME
split
(CAUSE|ENABLE)+
transfer
THEME
ions

Figure 1: An overview of our reading comprehension system. First, we predict a structure from the input paragraph (the top right portion shows a partial structure skipping some arguments for brevity). Circles denote events, squares denote arguments, solid arrows represent event-event relations, and dashed arrows represent event-argument relations. Second, we map the question paired with each answer into a query that will be answered using the structure. The bottom right shows the query representation. Last, the two queries are executed against the structure, and a final answer is returned.

domain, processes are abundant in domains such as chemistry, economics, manufacturing, and even everyday events like shopping or cooking, and our model can be applied to these domains as well.

The contributions of this paper are:

1. We propose a reading comprehension task which requires deep reasoning over structures that represent complex relations between multiple events and entities.
2. We present PROCESSBANK, a new dataset consisting of descriptions of biological processes, fully-annotated with rich process structures, and accompanied by multiple-choice questions.
3. We present a novel method for answering questions, by predicting process structures and mapping questions to queries. We demonstrate that by predicting structures we can improve reading comprehension accuracy over baselines that do not exploit the underlying structure.

The data and code for this paper are available at `http://www-nlp.stanford.edu/software/bioprocess`.

2 Task Definition and Setup

This section describes the reading comprehension task we address and the accompanying dataset. We will use the example in Figure 1 as our running example throughout the paper.

Our goal is to tackle a complex reading comprehension setting that centers on understanding the underlying meaning of a process description. We target a multiple-choice setting in which each input consists of a paragraph of text describing a biological process, a question, and two possible answers. The goal is to identify the correct answer using the text (Figure 1, left). We used the 148 paragraphs from the textbook *Biology* (Campbell and Reece, 2005) that were manually identified by Scaria et al. (2013). We extended this set to 200 paragraphs by including additional paragraphs that describe biological processes. Each paragraph in the collection represents a single biological process and describes a set of events, their participants and their interactions.

Because we target understanding of paragraph meaning, we use the following desiderata for building the corpus of questions and answers:

1. The questions should focus on the events and entities participating in the process described in the paragraph, and answering the questions should require reasoning about the relations between those events and entities.
2. Both answers should have similar lexical overlap with the paragraph. Moreover, names of entities and events in the question and answers should appear as in the paragraph and not using synonyms. This is to ensure that the task revolves around reading comprehension rather than lexical variability.[1]

A biologist created the question-answer part of

[1] Lexical variability is an important problem in NLP, but is not the focus of this task.

the corpus comprising of 585 questions spread over the 200 paragraphs. A second annotator validated 326 randomly chosen questions and agreed on the correct answer with the first annotator in 98.1% of cases. We provide the annotation guidelines in the supplementary material.

Figure 1 (left) shows an excerpt of a paragraph describing a process and an example of a question based on it. In general, questions test an understanding of the interactions between multiple events (such as causality, inhibition, temporal ordering), or between events and entities (i.e., roles of entities in events), and require complex reasoning about chains of event-event and event-entity relations.

3 The Structure of Processes

A natural first step for answering reading comprehension questions is to identify a structured representation of the text. In this section, we define this structure. We broadly follow the definition of Scaria et al. (2013), but modify important aspects, highlighted at the end of this section.

A paragraph describing a process is a sequence of tokens that describes events, entities and their relations (see Figure 1, top right). A *process* is a directed graph $(\mathcal{T}, \mathcal{A}, \mathcal{E}_{tt}, \mathcal{E}_{ta})$, where the nodes $\mathcal{T}$ are labeled event triggers, the nodes $\mathcal{A}$ are arguments, $\mathcal{E}_{tt}$ are labeled edges describing event-event relations, and $\mathcal{E}_{ta}$ are labeled edges from triggers to arguments denoting semantic roles (see Figure 1 top right for a partial structure of the running example). The goal of process extraction is to generate the process graph given the input paragraph.

Triggers and arguments A trigger is a token span denoting the occurrence of an event. In Figure 1, *split*, *absorbed* and *transfer* are event triggers. In rare cases, a trigger denotes the *non-occurrence* of an event. For example, in *"sympatric speciation can occur when gene flow is blocked"*, *sympatric speciation* occurs if *gene flow* does *not* happen. Thus, nodes in $\mathcal{T}$ are labeled as either a T-YES or T-NO to distinguish triggers of events that occur from triggers of events that do not occur. Arguments are token spans denoting entities that participate in the process (such as *water*, *light* and *ions* in Figure 1).

Semantic roles The edges $\mathcal{E}_{ta}$ from triggers to arguments are labeled by the semantic roles AGENT, THEME, SOURCE, DESTINATION, LOCATION, RESULT, and OTHER for all other roles. Our running example shows three THEME semantic roles for the three triggers. For brevity, the figure does not show the RESULT of the event *split*, namely, both *source of electrons and protons (hydrogen ions, H^+)* and O_2.

Event-event relations The directed edges $\mathcal{E}_{tt}$ between triggers are labeled by one of eight possible event-event relations. These relations are central to answering reading comprehension questions, which test understanding of the dependencies and causal relations between the process events. We first define three relations that express a dependency between two event triggers u and v.

1. CAUSE denotes that u starts before v, and if u happens then v happens (Figure 1).
2. ENABLE denotes that u creates conditions necessary for the occurrence of v. This means that u starts before v and v can only happen if u happens (Figure 1).[2]
3. PREVENT denotes that u starts before v and if u happens, then v does not happen.

In processes, events sometimes depend on more than one other event. For example, in Figure 1 (right top) *transfer of ions* depends on both *water splitting* as well as *light absorption*. Conversely, in Figure 2, the *shifting* event results in either one of two events but not both. To express both conjunctions and disjunctions of related events we add the relations CAUSE-OR, ENABLE-OR and PREVENT-OR, which express disjunctions, while the default CAUSE, ENABLE, and PREVENT express conjunction (Compare the CAUSE-OR relations in Figure 2 with the relations in Figure 1).

We define the SUPER relation to denote that event u is part of event v. (In Figure 2, *slippage* is a sub-event of *replication*.) Last, we use the event coreference relation SAME to denote two event mentions referring to the same event.

Notice that the assignments of relation labels interact across different pairs of events. As an example, if event u causes event v, then v can not cause u. Our inference algorithm uses such structural constraints when predicting process structure (Section 4).

[2] In this work, we do not distinguish causation from facilitation, where u can help v but is not absolutely required. We instructed the annotators to ignore the inherent uncertainty in these cases and use CAUSE.

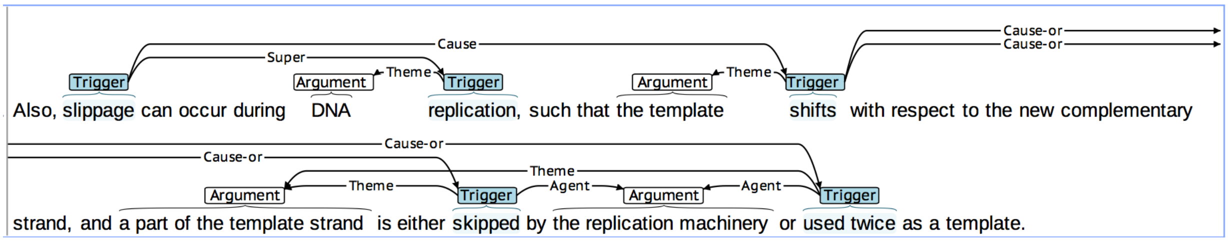

Cause-or
Cause-or
Cause
Super
Theme
Theme
Trigger
Argument
Trigger
Argument
Trigger
Also, slippage can occur during DNA replication, such that the template shifts with respect to the new complementary
Cause-or
Cause-or
Theme
Theme
Agent
Agent
Argument
Trigger
Argument
Trigger
strand, and a part of the template strand is either skipped by the replication machinery or used twice as a template.

Figure 2: Partial example of a process, as annotated in our dataset.

	Avg	Min	Max
# of triggers	7.0	2	18
# of arguments	11.3	1	36
# of relation	7.9	1	37

Table 1: Statistics of triggers, arguments and relations over the 200 annotated paragraphs.

Three biologists annotated the same 200 paragraphs described in Section 2 using the brat annotation tool (Stenetorp et al., 2012). For each paragraph, one annotator annotated the process, and a second validated its correctness. Importantly, the questions and answers were authored separately by a different annotator, thus ensuring that the questions and answers are independent from the annotated structures. Table 1 gives statistics over the dataset. The annotation guidelines are included in the supplementary material.

Relation to Scaria et al. (2013) Scaria et al. (2013) also defined processes as graphs where nodes are events and edges describe event-event relations. Our definition differs in a few important aspects.

First, the set of event-event relations in that work included temporal relations in addition to causal ones. In this work, we posit that because events in a process are inter-related, causal dependencies are sufficient to capture the relevant temporal ordering between them. Figure 1 illustrates this phenomenon, where the temporal ordering between the events of *water splitting* and *light absorption* is unspecified. It does not matter whether one happens before, during, or after the other. Furthermore, the incoming causal links to *transfer* imply that the event should happen after *splitting* and *absorption*.

A second difference is that Scaria et al. (2013) do not include disjunctions and conjunctions of events in their formulation. Last, Scaria et al. (2013) predict only relations given input triggers, while we predict a full process structure.

4 Predicting Process Structures

We now describe the first step of our algorithm. Given an input paragraph we predict events, their arguments and event-event relations (Figure 1, top). We decompose this into three sub-problems:

1. Labeling trigger candidates using a multi-class classifier (Section 4.1).
2. For each trigger, identifying an over-complete set of possible arguments, using a classifier tuned for high recall (Section 4.2).
3. Jointly assigning argument labels and relation labels for all trigger pairs (Section 4.3).

The event-event relations CAUSE, ENABLE, CAUSE-OR and ENABLE-OR, form a semantic cluster: If (u, v) is labeled by one of these, then the occurrence of v depends on the occurrence of u. Since our dataset is small, we share statistics by collapsing all four labels to a single ENABLE label. Similarly, we collapse the PREVENT and PREVENT-OR labels, overall reducing the number of relations to four.

For brevity, in what follows we only provide a flavor of the features we extract, and refer the reader to the supplementary material for details.

4.1 Predicting Event Triggers

The first step is to identify the events in the process. We model the trigger detector as a multi-class classifier that labels all content words in the paragraph as one of T-YES, T-NO or NOT-TRIGGER (Recall that a word can trigger an event that occurred, an event that did not occur, or not be a trigger at all). For simplicity, we model triggers as single words, but in the gold annotation about 14% are phrases (such as *gene flow*). Thus, we evaluate trigger prediction by taking heads of gold phrases. To train the classifier, we extract the lemma and POS tag of the word and adjacent words, dependency path to the root, POS tag of children and parent in the dependency tree, and clustering features from WordNet (Fellbaum, 1998), Nomlex (Macleod et al., 1998), Levin verb classes (Levin, 1993), and a list of biological processes compiled from Wikipedia.

4.2 Filtering Argument Candidates

Labeling trigger-argument edges is similar to semantic role labeling. Following the standard approach (Punyakanok et al., 2008), for each trigger we collect all constituents in the same sentence to build an over-complete set of plausible candidate arguments. This set is pruned with a binary classifier that is tuned for high recall (akin to the argument identifier in SRL systems). On the development set we filter more than half of the argument candidates, while achieving more than 99% recall. This classifier is trained using argument identification features from Punyakanok et al. (2008).

At the end of this step, each trigger has a set of candidate arguments which will be labeled during joint inference. In further discussion, the argument candidates for trigger t are denoted by $\mathcal{A}_t$.

4.3 Predicting Arguments and Relations

Given the output of the trigger classifier, our goal is to jointly predict event-argument and event-event relations. We model this as an integer linear program (ILP) instance described below. We first describe the inference setup assuming a model that scores inference decisions and defer description of learning to Section 4.4. The ILP has two types of decision variables: arguments and relations.

Argument variables These variables capture the decision that a candidate argument a, belonging to the set $\mathcal{A}_t$ of argument candidates, takes a label A (from Section 3). We denote the Boolean variables by $y_{t,a,A}$, which are assigned a score $b_{t,a,A}$ by the model. We include an additional label NULL-ARG, indicating that the candidate is not an argument for the trigger.

Event-event relation variables These variables capture the decision that a pair of triggers t_1 and t_2 are connected by a directed edge (t_1, t_2) labeled by the relation R. We denote these variables by $z_{t_1,t_2,R}$, which are associated with a score $c_{t_1,t_2,R}$. Again, we introduce a label NULL-REL to indicate triggers that are not connected by an edge.

Name	Description
Unique labels	Every argument candidate and trigger pair has exactly one label.
Argument overlap	Two arguments of the same trigger cannot overlap.
Relation symmetry	The SAME relation is symmetric. All other relations are anti-symmetric, i.e., for any relation label other than SAME, at most one of (t_i, t_j) or (t_j, t_i) can take that label and the other is assigned the label NULL-REL.
Max arguments per trigger	Every trigger can have no more than two arguments with the same label.
Max triggers per argument	The same span of text can not be an argument for more than two triggers.
Connectivity	The triggers must form a connected graph, framed as flow constraints as in Magnanti and Wolsey (1995) and Martins et al. (2009).
Shared arguments	If the same span of text is an argument of two triggers, then the triggers must be connected by a relation that is not NULL-REL. This ensures that triggers that share arguments are related.
Unique parent	For any trigger, at most one outgoing edge can be labeled SUPER.

Table 2: Constraints for joint inference.

Formulation Given the two sets of variables, the objective of inference is to find a global assignment that maximizes the score. That is, the objective can be stated as follows:

$$\max_{\mathbf{y},\mathbf{z}} \sum_{t,a\in\mathcal{A}_t,A} b_{t,a,A}\cdot y_{t,a,A} + \sum_{t_1,t_2,R} c_{t_1,t_2,R}\cdot z_{t_1,t_2,R}$$

Here, $\mathbf{y}$ and $\mathbf{z}$ refer to all the argument and relation variables respectively.

Clearly, all possible assignments to the inference variables are not feasible and there are both structural as well as prior knowledge constraints over the output space. Table 2 states the constraints we include, which are expressed as linear inequalities over output variables using standard techniques (e.g., (Roth and Yih, 2004)).

4.4 Learning in the Joint Model

We train both the trigger classifier and the argument identifier using L$_2$-regularized logistic regression. For the joint model, we use a linear model for the scoring functions, and train jointly using the structured averaged perceptron algorithm (Collins, 2002).

Since argument labeling is similar to semantic role labeling (SRL), we extract standard SRL features given the trigger and argument from the syntactic tree for the corresponding sentence. In addition, we add features extracted from an off-the-shelf SRL system. We also include all feature conjunctions. For event relations, we include the features described in Scaria et al. (2013), as well as context features for both triggers, and the dependency path between them, if one exists.

5 Question Answering via Structures

This section describes our question answering system that, given a process structure, a question and two answers, chooses the correct answer (steps 2 and 3 in Figure 1).

Our strategy is to treat the process structure as a small knowledge-base. We map each answer along with the question into a *structured query* that we compare against the structure. The query can prove either the correctness or incorrectness of the answer being considered. That is, either we get a valid match for an answer (proving that the corresponding answer is correct), or we get a refutation in the form of a contradicted causal chain (thus proving that the *other answer* is correct). This is similar to theorem proving approaches suggested in the past for factoid question answering (Moldovan et al., 2003).

The rest of this section is divided into three parts: Section 5.1 defines the queries we use, Section 5.2 describes a rule-based algorithm for converting a question and an answer into a query and finally, 5.3 describes the overall algorithm.

5.1 Queries over Processes

We model a query as a directed graph path with regular expressions over edge labels. The bottom right portion of Figure 1 shows examples of queries for our running example. In general, given a question and one of the answer candidates, one end of the path is populated by a trigger/argument found in the question and the other is populated with a trigger/ argument from the answer.

We define a query to consist of three parts:

1. A regular expression over relation labels, describing permissible paths,
2. A source trigger/argument node, and
3. A target trigger/argument node.

For example, the bottom query in Figure 1 looks for paths labeled with CAUSE or ENABLE edges from the event *split* to the event *transfer*.

Note that the representation of questions as directed paths is a modeling choice and did not influence the authoring of the questions. Indeed, while most questions do fit this model, there are rare cases that require a more complex query structure.

5.2 Query Generation

Mapping a question and an answer into a query involves identifying the components of the query listed above. We do this in two phases: (1) In the *alignment phase*, we align triggers and arguments in the question and answer to the process structure to give us candidate source and target nodes. (2) In the *query construction phase*, we identify the regular expression and the direction of the query using the question, the answer and the alignment.

We identify three broad categories of QA pairs (see Table 3) that can be identified using simple lexical rules: (a) *Dependency questions* ask which event or argument depends on another event or argument, (b) *Temporal questions* ask about temporal ordering of events, and (c) *True-false questions* ask whether some fact is true. Below, we describe the two phases of query generation primarily in the context of dependency questions with a brief discussion about temporal and true-false questions at the end of the section.

Alignment Phase We align triggers in the structure to the question and the answer by matching lemmas or nominalizations. In case of multiple matches, we use the context to disambiguate and resolve ties using the highest matching candidate in the syntactic dependency tree.

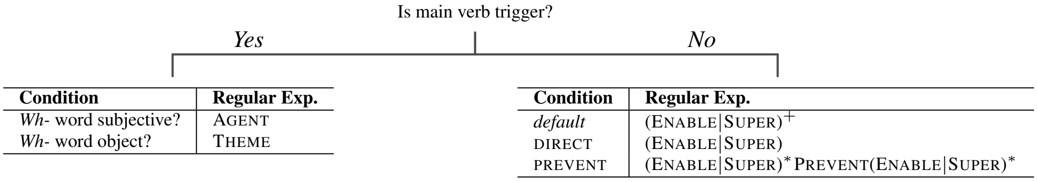
We align arguments in the question and the answer in a similar manner. Since arguments are typically several words long, we prefer maximal spans. Additionally, if a question (or an answer) contains an aligned trigger, we prefer to align words to its arguments.

Query Construction Phase We construct a query using the aligned question and answer triggers/arguments. We will explain query construction using our running example (reproduced as the dependency question in Table 3).

First, we identify the source and the target of the query. We select either the source or the target to be a question node and populate the other end of the query path with an answer node. To make the choice between source or target for the question node, we use the main verb in the question, its voice and relative position of the question word with respect to the main verb. In our example, the main verb *lead to* is in active voice and the question word *what* is not in subject position. This places the trigger from the question as the source of the query path (see both queries in the bottom right portion of the running example). In contrast, had the verb been *require*, the trigger would be the target of the query. We construct two verb clusters that indicate query direction using a small seed set

Type	Example	# (%)
Dependency	**Q**: What can the splitting of water lead to? **a**: Light absorption **b**: Transfer of ions	407 (69.57%)
Temporal	**Q**: What is the correct order of events? **a**: PDGF binds to tyrosine kinases, then cells divide, then wound healing **b**: Cells divide, then PDGF binds to tyrosine kinases, then wound healing	57 (9.74%)
True-False	**Q**: Cdk associates with MPF to become cyclin **a**: True **b**: False	121 (20.68%)

Table 3: Examples and statistics for each of the three coarse types of questions.



Condition	Regular Exp.
Wh- word subjective?	AGENT
Wh- word object?	THEME

Condition	Regular Exp.
default	(ENABLE|SUPER)$^+$
DIRECT	(ENABLE|SUPER)
PREVENT	(ENABLE|SUPER)*PREVENT(ENABLE|SUPER)*

Figure 3: Rules for determining the regular expressions for queries concerning two triggers. In each table, the **condition** column decides the regular expression to be chosen. In the left table, we make the choice based on the path from the root to the *Wh-* word in the question. In the right table, if the word *directly* modifies the main trigger, the DIRECT regular expression is chosen. If the main verb in the question is in the synset of *prevent*, *inhibit*, *stop* or *prohibit*, we select the PREVENT regular expression. Otherwise, the default one is chosen. We omit the relation label SAME from the expressions, but allow going through any number of edges labeled by SAME when matching expressions to the structure.

that we expand using WordNet.

The final step in constructing the query is to identify the regular expression for the path connecting the source and the target. Due to paucity of data, we do not map a question and an answer to arbitrary regular expressions. Instead, we construct a small set of regular expressions, and build a rule-based system that selects one. We used the training set to construct the regular expressions and we found that they answer most questions (see Section 6.4). We determine the regular expression based on whether the main verb in the sentence is a trigger and whether the source and target of the path are triggers or arguments. Figure 3 shows the possible regular expressions and the procedure for choosing one when both the source and target are triggers. If either of them are argument nodes, we append the appropriate semantic role to the regular expression, based on whether the argument is the source or the target of the path (or both).

True-false questions are treated similarly, except that both source and target are chosen from the question. For temporal questions, we seek to identify the ordering of events in the answers. We use the keywords *first*, *then*, or *simultaneously* to identify the implied order in the answer. We use the regular expression SUPER$^+$ for questions asking about simultaneous events and ENABLE$^+$ for those asking about sequential events.

5.3 Answering Questions

We match the query of an answer to the process structure to identify the answer. In case of a match, the corresponding answer is chosen. The matching path can be thought of as a *proof* for the answer.

If neither query matches the graph (or both do), we check if either answer contradicts the structure. To do so, we find an undirected path from the source to the target. In the event of a match, if the matching path traverses any ENABLE edge in the incorrect direction, we treat this as a *refutation* for the corresponding answer and select the other one. In our running example, in addition to the valid path for the second query, for the first query we see that there is an undirected path from *split* to *absorb* through *transfer* that matches the first query. This tells us that *light absorption* cannot be the answer because it is not along a causal path from *split*.

Finally, if none of the queries results in a match, we look for any unlabeled path between the source and the target, before backing off to a dependency-based proximity baseline described in Section 6. When there are multiple aligning nodes in the question and answer, we look for any proof or refutation before backing off to the baselines.

6 Empirical Evaluation

In this section we aim to empirically evaluate whether we can improve reading comprehension accuracy by predicting process structures. We first provide details of the experimental setup.

6.1 Experimental setup

We used 150 processes (435 questions) for training and 50 processes (150 questions) as the test set. For development, we randomly split the training set 10 times (80%/20%), and tuned hyperparameters by maximizing average accuracy on question answering. We preprocessed the paragraphs with the Stanford CoreNLP pipeline version 3.4 (Manning et al., 2014) and Illinois SRL (Punyakanok et al., 2008; Clarke et al., 2012). We used the Gurobi optimization package[3] for inference.

We compare our system PROREAD to baselines that do not have access to the process structure:

1. BOW: For each answer, we compute the proportion of content word lemmas covered by the paragraph and choose the one with higher coverage. For true-false questions, we compute the coverage of the question statement, and answer "True" if it is higher than a threshold tuned on the development set.
2. TEXTPROX: For dependency questions, we align content word lemmas in both the question and answer against the text and select the answer whose aligned tokens are closer to the aligned tokens of the question. For temporal questions, we return the answer for which the order of events is identical to their order in the paragraph. For true-false questions, we return "True" if the number of bigrams from the question covered in the text is higher than a threshold tuned on the development set.
3. SYNTPROX: For dependency questions, we use proximity as in TEXTPROX, except that distance is measured using dependency tree edges. To support multiple sentences we connect roots of adjacent sentences with bidirectional edges. For temporal questions this baseline is identical to TEXTPROX. For true-false questions, we compute the number of dependency tree edges in the question statement covered by edges in the paragraph (an edge has a source lemma, relation, and target lemma), and answer "True" if the coverage is higher than a threshold tuned on the training set.

Method	Depen.	Temp.	True-false	All
PROREAD	**68.1**	**80.0**	**55.6**	**66.7**
SYNTPROX	61.9	70.0	48.1	60.0
TEXTPROX	58.4	70.0	33.3	54.7
BOW	47.8	40.0	44.4	46.7
GOLD	77.9	80.0	70.4	76.7

Table 4: Reading comprehension test set accuracy. The **All** column shows overall accuracy across all questions. The first three columns show accuracy for each coarse type.

To separate the contribution of process structures from the performance of our structure predictor, we also run our QA system given manually annotated gold standard structures (GOLD).[4]

6.2 Reading Comprehension Task

We evaluate our system using accuracy, i.e., the proportion of questions answered correctly. Table 4 presents test set results, where we break down questions by their coarse-type.

PROREAD improves accuracy compared to the best baseline by 6.7 absolute points (last column). Most of the gain is due to improvement on dependency questions, which are the most common question type. The performance of BOW indicates that lexical coverage alone does not distinguish the correct answer from the wrong answer. In fact, guessing the answer with higher lexical overlap results in performance that is slightly lower than random. Text proximity and syntactic proximity provide a stronger cue, but exploiting predicted process structures substantially outperforms these baselines.

Examining results using gold information highlights the importance of process structures independently of the structure predictor. Results of GOLD demonstrate that given gold structures we can obtain a dramatic improvement of almost 17 points compared to the baselines, using our simple deterministic QA system.

Results on true-false questions are low for PROREAD and all the baselines. True-false questions are harder for two main reasons. First, in dependency and temporal questions, we create a query for both answers, and can find a proof or a refutation for either one of them. In true-false

[3] http://www.gurobi.com/

[4] We also ran an experiment where gold triggers are given and arguments and relations are predicted. We found that this results in slightly higher performance compared to PROREAD.

	Precision	Recall	F_1
Triggers	75.4	73.9	74.6
Arguments	43.4	34.4	38.3
Relations	27.0	22.5	24.6

Table 5: Structured prediction test set results.

questions we must determine given a single statement whether it holds. Second, an analysis of true-false questions reveals that they focus less on relations between events and entities in the process, and require modeling lexical variability.[5]

6.3 Structure Prediction Task

Our evaluation demonstrates that gold structures improve accuracy substantially more than predicted structures. To examine this, we now directly evaluate the structure predictor by comparing micro-average precision, recall and F_1 between predicted and gold structures (Table 5).

While performance for trigger identification is reasonable, performance on argument and relation prediction is low. This explains the higher performance obtained in reading comprehension given gold structures. Note that errors in trigger prediction propagate to argument and relation prediction – a relation cannot be predicted correctly if either one of the related triggers is not previously identified. One reason for low performance is the small size of the dataset. Thus, training process predictors with less supervision is an important direction for future work. Furthermore, the task of process prediction is inherently difficult, because often relations are expressed only indirectly in text. For example, in Figure 1 the relation between *water splitting* and *transfer of ions* is only recoverable by understanding that water provides the ions that need to be transferred.

Nevertheless, we find that questions can often be answered correctly even if the structure contains some errors. For example, the gold structure for the sentence *"Some ... radioisotopes have long half-lives, allowing ... "*, contains the trigger *long half-lives*, while we predict *have* as a trigger and *long half-lives* as an argument. This is good enough to answer questions related to this part of the structure correctly, and overall, to improve performance using predicted structures.

Reason	GOLD	PROREAD
Alignment	35%	15%
Missing from annotation	25%	10%
Entity coreference	20%	10%
Missing regular expression	10%	
Lexical variability	5%	10%
Error in predicted structure		55%
Other	5%	

Table 6: Error analysis results. An explanation of the various categories are in the body of the paper.

6.4 Error Analysis

This section presents the results of an analysis of 20 sampled errors of GOLD (gold structures), and 20 errors of PROREAD (predicted structures). We have categorized the primary reason for error in Table 6.

As expected, the main problem when using predicted structures, is structure errors which account for more than half of the errors.

Errors in GOLD are distributed across various categories, which we briefly describe. *Alignment* errors occur due to multiple words aligning to multiple triggers and arguments. For example, in the question *"What is the result of gases being produced in the lysosome?"*, the answer *"engulfed pathogens are poisoned"* is incorrectly aligned to the trigger *engulfed* rather than to *poisoned*.

Another reason for errors are cases where questions are asked about parts of the paragraph that are *missing from annotation*. This is possible since questions were authored independently of structure annotation. Two other causes for errors are *entity coreference* errors, where a referent for an entity is missing from the structure, and *lexical variability*, where the author of questions uses names for triggers or arguments that are missing from the paragraph, and so alignment fails.

Last, in 10% of the cases in GOLD we found that the answer could not be retrieved using the set of regular expressions that are currently used by our QA system.

7 Discussion

This work touches on several strands of work in NLP including information extraction, semantic role labeling, semantic parsing and reading comprehension.

Event and relation extraction have been studied via the ACE data (Doddington et al., 2004) and related work. The BioNLP shared tasks (Kim et al., 2009; Kim et al., 2011; Riedel and McCal-

[5] The low performance of TEXTPROX and SYNTPROX on true-false questions can also be attributed to the fact that we tuned a threshold parameter on the training set, and this did not generalize well to the test set.

lum, 2011) focused on biomedical data to extract events and their arguments. Event-event relations have been mostly studied from the perspective of temporal ordering; e.g., (Chambers and Jurafsky, 2008; Yoshikawa et al., 2009; Do et al., 2012; McCloskyand Manning, 2012). The process structure predicted in this work differs from these lines of work in two important ways: First, we predict events, arguments *and* their interactions from multiple sentences, while most earlier work focused on one or two of these components. Second, we model processes, and thus target causal relations between events, rather than temporal order only.

Our semantic role annotation is similar to existing SRL schemes such as PropBank (Palmer et al., 2005), FrameNet (Ruppenhofer et al., 2006) and BioProp (Chou et al., 2006). However, in contrast to PropBank and FrameNet, we do not allow all verbs to trigger events and instead let the annotators decide on biologically important triggers, which are not restricted to verbs (unlike BioProp, where 30 pre-specified verbs were selected for annotation). Like PropBank and BioProp, the argument labels are not trigger specific.

Mapping questions to queries is effectively a semantic parsing task. In recent years, several lines of work addressed semantic parsing using various formalisms and levels of supervision (Zettlemoyer and Collins, 2005; Wong and Mooney, 2006; Clarke et al., 2010; Berant et al., 2013). In particular, Krishnamurthy and Kollar (2013) learned to map natural language utterances to referents in an image by constructing a KB from the image and then mapping the utterance to a query over the KB. This is analogous to our process of constructing a process structure and performing QA by querying that structure. In our work, we parse questions into graph-based queries, suitable for modeling processes, using a rule-based heuristic. Training a statistical semantic parser that will replace the QA system is an interesting direction for future research.

Multiple choice reading comprehension tests are a natural choice for evaluating machine reading. Hirschman et al. (1999) presented a bag-of-words approach to retrieving sentences for reading comprehension. Richardson et al. (2013) recently released the MCTest reading comprehension dataset that examines understanding of fictional stories. Their work shares our goal of advancing micro-reading, but they do not focus on process understanding.

Developing programs that perform deep reasoning over complex descriptions of processes is an important step on the road to fulfilling the higher goals of machine reading. In this paper, we present an end-to-end system for reading comprehension of paragraphs which describe biological processes. This is, to the best of our knowledge, the first system to both predict a rich structured representation that includes entities, events and their relations, *and* utilize this structure for answering reading comprehension questions. We also created a new dataset, PROCESSBANK, which contains 200 paragraphs that are both fully-annotated with process structure, as well as accompanied by questions. We empirically demonstrated that modeling biological processes can substantially improve reading comprehension accuracy in this domain.

Acknowledgments

The authors would like to thank Luke Amuchastegui for authoring the multiple-choice questions, and also the anonymous reviewers for their constructive feedback. We thank the Allen Institute for Artificial Intelligence for assistance in funding this work.

References

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of EMNLP*.

Neil Campbell and Jane Reece. 2005. *Biology*. Benjamin Cummings.

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka Jr., and Tom M. Mitchell. 2010. Toward an architecture for never-ending language learning. In *Proceedings of AAAI*.

Nathanael Chambers and Daniel Jurafsky. 2008. Jointly combining implicit constraints improves temporal ordering. In *Proceedings of EMNLP*.

Wen-Chi Chou, Richard Tzong-Han Tsai, Ying-Shan Su, Wei Ku, Ting-Yi Sung, and Wen-Lian Hsu. 2006. A semi-automatic method for annotating a biomedical proposition bank. In *Proceedings of the Workshop on Frontiers in Linguistically Annotated Corpora 2006*, July.

James Clarke, Dan Goldwasser, Ming-Wei Chang, and Dan Roth. 2010. Driving semantic parsing from the world's response. In *Proceedings of CoNLL*.

James Clarke, Vivek Srikumar, Mark Sammons, and Dan Roth. 2012. An NLP Curator (or: How I

Learned to Stop Worrying and Love NLP Pipelines). In *Proceedings of LREC*.

Michael Collins. 2002. Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. In *Proceedings of ACL*.

Quang Do, Wei Lu, and Dan Roth. 2012. Joint inference for event timeline construction. In *Proceedings of EMNLP-CoNLL*.

George R. Doddington, Alexis Mitchell, Mark A. Przybocki, Lance A. Ramshaw, Stephanie Strassel, and Ralph M. Weischedel. 2004. The Automatic Content Extraction (ACE) Program-Tasks, Data, and Evaluation. In *Proceedings of LREC*.

Oren Etzioni, Michele Banko, and Michael J. Cafarella. 2006. Machine reading. In *Proceedings of AAAI*.

Anthony Fader, Stephen Soderland, and Oren Etzioni. 2011. Identifying relations for open information extraction. In *Proceedings of EMNLP*.

Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. 2013. Paraphrase-Driven Learning for Open Question Answering. In *Proceedings of ACL*.

Christiane Fellbaum, editor. 1998. *WordNet: An electronic lexical database*. MIT Press.

Lynette Hirschman, Marc Light, Eric Breck, and John D. Burger. 1999. Deep read: A reading comprehension system. In *Proceedings of ACL*.

Jin-Dong Kim, Tomoko Ohta, Sampo Pyysalo, Yoshinobu Kano, and Junichi Tsujii. 2009. Overview of BioNLP 09 shared task on event extraction. In *Proceedings of BioNLP*.

Jin-Dong Kim, Sampo Pyysalo, Tomoko Ohta, Robert Bossy, and Junichi Tsujii. 2011. Overview of BioNLP shared task 2011. In *Proceedings of BioNLP*.

Jayant Krishnamurthy and Thomas Kollar. 2013. Jointly learning to parse and perceive: Connecting natural language to the physical world. *TACL*, 1:193–206.

Nate Kushman, Yoav Artzi, Luke Zettlemoyer, and Regina Barzilay. 2014. Learning to automatically solve algebra word problems. In *Proceedings of ACL*.

Beth Levin. 1993. *English verb classes and alternations: A preliminary investigation*. University of Chicago Press.

Catherine Macleod, Ralph Grishman, Adam Meyers, Leslie Barrett, and Ruth Reeves. 1998. Nomlex: A lexicon of nominalizations. In *Proceedings of EURALEX*.

Thomas L. Magnanti and Laurence A. Wolsey. 1995. Optimal trees. *Handbooks in operations research and management science*, 7:503–615.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of ACL: System Demonstrations*.

André L. Martins, Noah A. Smith, and Eric P. Xing. 2009. Concise integer linear programming formulations for dependency parsing. In *Proceedings of ACL/IJCNLP*.

David McClosky and Christopher D. Manning. 2012. Learning constraints for consistent timeline extraction. In *Proceedings of EMNLP-CoNLL*.

Dan Moldovan, Christine Clark, Sanda Harabagiu, and Steve Maiorano. 2003. Cogex: A logic prover for question answering. In *Proceedings of NAACL-HLT*.

Martha Palmer, Paul Kingsbury, and Daniel Gildea. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106.

Vasin Punyakanok, Dan Roth, and Wen-tau Yih. 2008. The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, 34(2).

Matthew Richardson, Christopher JC Burges, and Erin Renshaw. 2013. MCTest: A challenge dataset for the open-domain machine comprehension of text. In *Proceedings of EMNLP*.

Sebastian Riedel and Andrew McCallum. 2011. Fast and robust joint models for biomedical event extraction. In *Proceedings of EMNLP*.

Dan Roth and Wen-tau Yih. 2004. A linear programming formulation for global inference in natural language tasks. In *Proceedings of CoNLL*.

Josef Ruppenhofer, Michael Ellsworth, Miriam RL Petruck, Christopher R. Johnson, and Jan Scheffczyk. 2006. FrameNet II: Extended theory and practice. *Berkeley FrameNet Release*, 1.

Aju Thalappillil Scaria, Jonathan Berant, Mengqiu Wang, Peter Clark, Justin Lewis, Brittany Harding, and Christopher D. Manning. 2013. Learning biological processes with global constraints. In *Proceedings of EMNLP*.

Pontus Stenetorp, Sampo Pyysalo, Goran Topić, Tomoko Ohta, Sophia Ananiadou, and Jun'ichi Tsujii. 2012. Brat: a web-based tool for NLP-assisted text annotation. In *Proceedings of the demonstrations at EACL*.

Yuk Wah Wong and Raymond J. Mooney. 2006. Learning for semantic parsing with statistical machine translation. In *Proceedings of HLT-NAACL*.

Katsumasa Yoshikawa, Sebastian Riedel, Masayuki Asahara, and Yuji Matsumoto. 2009. Jointly identifying temporal relations with Markov logic. In *Proceedings of ACL/IJCNLP*.

Luke S. Zettlemoyer and Michael Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *Proceedings of UAI*.