Learning what to read: Focused machine reading

Enrique Noriega-Atala
Marco A. Valenzuela-Escárcega
Clayton T. Morrison
Mihai Surdeanu

University of Arizona
Tucson, Arizona, USA

{enoriega,marcov,claytonm,msurdeanu}@email.arizona.edu

Abstract

Recent efforts in bioinformatics have achieved tremendous progress in the machine reading of biomedical literature, and the assembly of the extracted biochemical interactions into large-scale models such as protein signaling pathways. However, batch machine reading of literature at today's scale (PubMed alone indexes over 1 million papers per year) is unfeasible due to both cost and processing overhead. In this work, we introduce a focused reading approach to guide the machine reading of biomedical literature towards *what* literature should be read to answer a biomedical query as efficiently as possible. We introduce a family of algorithms for focused reading, including an intuitive, strong baseline, and a second approach which uses a reinforcement learning (RL) framework that learns when to explore (widen the search) or exploit (narrow it). We demonstrate that the RL approach is capable of answering more queries than the baseline, while being more efficient, i.e., reading fewer documents.

1 Introduction

The millions of academic papers in the biomedical domain contain a vast amount of information that may lead to new hypotheses for disease treatment. However, scientists are faced with a problem of "undiscovered public knowledge," as they struggle to read and assimilate all of this information (Swanson, 1986). Furthermore, the literature is growing at an exponential rate (Pautasso, 2012); PubMed[1] has been adding more than a million papers per year since 2011. We have surpassed our ability to keep up with and integrate these findings through manual reading alone.

Large ongoing efforts, such as the BioNLP task community (Nédellec et al., 2013; Kim et al., 2012, 2009) and the DARPA Big Mechanism Program (Cohen, 2015), are making progress in advancing methods for machine reading and assembly of extracted biochemical interactions into large-scale models. However, to date, these methods rely either on the manual selection of relevant documents, or on the processing of large batches of documents that may or may not be relevant to the model being constructed.

Batch machine reading of literature at this scale poses a new, growing set of problems. First, access to some documents is costly. The PubMedCentral (PMC) Open Access Subset[2] (OA) is estimated[3] to comprise 20%[4] of the total literature; the remaining full-text documents are only available through paid access. Second, while there have been great advances in quality, machine reading is still not solved. Updates to our readers requires reprocessing the documents. For large document corpora, this quickly becomes the chief bottleneck in information extraction for model construction and analysis. Finally, even if we could cache all reading results, the search for connections between concepts within the extracted results should not be done blindly. At least in the biology domain, the many connections between biological entities and processes leads to a very high branching factor, making blind search for paths intractable.

To effectively read at this scale, we need to incorporate methods for *focused reading*: develop the ability to pose queries about concepts of interest and perform targeted, incremental search

[1] http://www.ncbi.nlm.nih.gov/pubmed

[2] https://www.ncbi.nlm.nih.gov/pmc/tools/openftlist/

[3] https://tinyurl.com/bachman-oa

[4] This includes 5% from PMC author manuscripts.

Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pages 2905–2910
Copenhagen, Denmark, September 7–11, 2017.

through the literature for connections between concepts while minimizing reading documents that are likely irrelevant.

In this paper we present what we believe is the first algorithm for focused reading. We make the following contributions:
(1) Present a general framework for a family of possible focused reading algorithms along with a baseline instance.
(2) Cast the design of focused reading algorithms in a reinforcement learning (RL) setting, where the machine decides if it should explore (i.e., cast a wider net) or exploit (i.e., focus reading on a specific topic).
(3) Evaluate our focused reading policies in terms of search efficiency and quality of information extracted. The evaluation demonstrates the effectiveness of the RL method: this approach found more information than the strong baseline we propose, while reading fewer documents.

2 Related Work

The past few years have seen a large body of work on information extraction (IE), particularly in the biomedical domain. This work is too vast to be comprehensively discussed here. We refer the interested reader to the BioNLP community (Nédellec et al., 2013; Kim et al., 2012, 2009, inter alia) for a starting point. However, most of this work focuses on *how* to read, not on *what* to read given a goal. To our knowledge, we are the first to focus on the latter task.

Reinforcement learning has been used to achieve state of the art performance in several natural language processing (NLP) and information retrieval (IR) tasks. For example, RL has been used to guide IR and filter irrelevant web content (Seo and Zhang, 2000; Zhang and Seo, 2001). More recently, RL has been combined with deep learning with great success, e.g., for improving coreference resolution (Clark and Manning, 2016). Finally, RL has been used to improve the efficiency of IE by learning how to incrementally reconcile new information and help choose what to look for next (Narasimhan et al., 2016), a task close to ours. This serves as an inspiration for the work we present here, but with a critical difference: Narasimhan et al. (2016) focus on slot filling using a pre-existing template. This makes both the information integration and stopping criteria well-defined. On the other hand, in our focused reading domain, we do not know ahead of time which new pieces of information are necessarily relevant and must be taken in context.

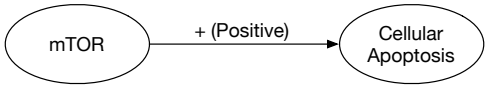

Figure 1: Example of a graph edge encoding the relation extracted from the text: *mTOR triggers cellular apoptosis*.

3 Focused Reading

Here we consider focused reading for the biomedical domain, and we focus on binary promotion/inhibition interactions between biochemical entities. In this setting, the machine reading (or IE) component constructs a directed graph, where vertices represent *participants* in an interaction (e.g., protein, gene, or a biological process), and edges represent directed activation interactions. Edge labels indicate whether the controller entity has a *positive* (promoting) or *negative* (inhibitory) influence on the controlled participant. Figure 1 shows an example edge in this graph.

We use REACH[5], an open source IE system (Valenzuela-Escárcega et al., 2015), to extract interactions from unstructured biomedical text and construct the graph above. We couple this IE system with a Lucene[6] index of biomedical publications to retrieve papers based on queries about participant mentions in the text (as discussed below).

Importantly, we essentially use IE as a black box (thus, our method could potentially work with any IE system), and focus on strategies that guide *what* the IE system reads for a complex information need. In particular, we consider the common scenario where a biologist (or other model-building process) queries the literature on:

> How does one participant (source) affect another (destination), where the connection is typically indirect?

This type of queries is common in biology, where such direct/indirect interactions are observed in experiments, but the explanation of why these dependencies exist is unclear.

Algorithm 1 outlines the general focused reading algorithm for this task. In the algorithm,

[5] https://github.com/clulab/reach
[6] https://lucene.apache.org

S, D, A, and B represent individual participants, where S and D are the **s**ource and **d**estination entities in the initial user query. G is the interaction graph that is iteratively constructed during the focused reading procedure, with V being the set of vertices (biochemical entities), and E the set of edges (promotion/inhibition interactions). Σ is the strategy that chooses which two entities/vertices to be used in the next information retrieval iteration. Q is a Lucene query automatically constructed in each iteration to retrieve new papers to read.

Algorithm 1 Focused reading framework

```
1:  procedure FocusedReading(S, D)
2:      G ← {{S, D}, ∅}
3:      repeat
4:          Σ ← EndpointStrategy(G)
5:          (A, B) ← ChooseEndPoints(Σ, G)
6:          Q ← ChooseQuery(A, B, G)
7:          (V, E) ← Lucene+Reach(Q)
8:          Expand(V, E, G)
9:      until IsConnected(S, D) OR StopConditionMet(G)
10: end procedure
```

The algorithm initializes the search graph as containing the two unconnected participants as vertices: $\{S, D\}$ (line 2). The algorithm then enters into its central loop (lines 3 through 9). The loop terminates when one or more directed paths connecting S to D are found, or when a stopping condition is met: either G has not changed since the previous run through the loop, or after exceeding some number of iterations through the loop (in this work, ten).

At each pass through the loop the algorithm grows the search graph as follows:

1. The graph G is initialized with two nodes, the source and destination in the user's information need, and no edges (because we have not read any papers yet).
2. Given the current graph, choose a strategy, Σ, for selecting which entities to query next: *exploration* or *exploitation* (line 4). In general, exploration aims to widen the search space by adding many more nodes to the graph, whereas exploitation aims to narrow the search by focusing on entities in a specific region of the graph.
3. Using strategy Σ, choose the next entities to attempt to link: (A, B) (line 5).
4. Choose a query, Q: again, using *exploration* or *exploitation*, following the same intuition as with the entity choice strategy (line 6). Here exploration queries retrieve a wider range of documents, while exploitation queries are more restrictive.
5. Run the Lucene query to retrieve papers and process the papers using the IE system. The result of this call is a set of interactions, similar to that in Figure 1 (line 7).
6. Add the new interaction participant entities (vertices V) and directed influences (edges E) to the search graph (line 8).
7. If the source and destination entities are connected in G, stop: the user's information need has been addressed. Otherwise, continue from step 2.

The central loop performs a bidirectional search in which each iteration expands the search horizon outward from S and D. Algorithm 1 represents a family of possible focused reading algorithms, differentiated by how each of the functions in the main loop are implemented. In this work, IsConnected stops after a single path is found, but a variant could consider finding multiple paths, paths of some length, or incorporate other criteria about the properties of the path. We next consider particular choices for the inner loop functions.

4 Baseline Algorithm and Evaluation

The main functions that affect the search behavior of Algorithm 1 are EndpointStrategy and ChooseQuery. Here we describe a *baseline* focused reading implementation in which EndpointStrategy and ChooseQuery aim to find any path between S and D as quickly as possible.

For EndpointStrategy, we follow the intuition that some participants in a biological graph tend to be connected to more participants than others, and therefore more likely to yield interactions providing paths between participants in general. Our heuristic is therefore to choose new participants to query that currently have the most inward and outgoing edges (i.e., highest vertex degree) in the current state of G (disallowing choosing an entity pair used in a previous query).

Now that we have our candidate participants (A, B), our next step is to formulate how we will use these participants to retrieve new papers. Here we consider two classes of query: (1) we restrict our query to only retrieve papers that simultaneously mention both A and B, therefore more likely retrieving a paper with a direct link between A and B (*exploit*), or (2) we retrieve papers that mention

	Baseline	RL Query Policy	
# IR queries	573	433	25% decrease
Unique papers read	26,197	19,883	24% decrease
# Paths recovered (out of 289)	189 (65%)	198 (68%)	3% increase

Table 1: Results of the baseline and RL Query Policy for the focused reading of biomedical literature.

either A or B, therefore generally retrieving more papers that will introduce more new participants (*explore*). For our baseline, where we are trying to find a path between S and D as quickly as possible, we implement a greedy CHOOSEQUERY: first try the conjunctive exploitation query; if no documents are retrieved, then "relax" the search to the disjunctive exploration query.

To evaluate the baseline, we constructed a data set based on a collection of papers seeded by a set of 132 entities that come from the University of Pittsburgh DyCE[7] model, a biomolecular model of pancreatic cancer (Telmer et al., 2017). Using these entities, we retrieved 70,719 papers that mention them. We processed all papers using REACH, extracting all of the interactions mentioned, and converted them into a single graph. The resulting graph consisted of approximately 80,000 vertices, 115,000 edges, and had an average (undirected) vertex degree of 24. We will refer to this graph as the *REACH graph*, as it represents what *can* be retrieved by REACH from the set of 70K papers. Next, we identified which pairs of the original 132 entities are connected by directed paths in DyCE. A total of 789 pairs were found. We used 289 of these entity pairs as testing queries (i.e., generating queries that aim to explain how a given pair is connected according to the literature). The other 500 pairs were held out to train the RL method described below.

We ran this baseline focused reading algorithm on each of the 289 pairs of participants, in each case attempting to recover a directed path from one to the other. The results are summarized in the middle column of Table 1. By issuing 573 queries, the baseline read 26,197 papers out of the total 70,719 papers (37% of the corpus), in order to recover 189 of the 289 paths (65%).

5 Reinforcement Learning for Focussed Reading

We analyzed the baseline's behavior in the evaluation to identify the conditions under which it failed to find paths. From this, we found that some of the failures could be avoided had we used a different strategy for CHOOSEQUERY, i.e., the baseline chose to exploit when it should have explored more. The conditions for making different choices depend on the current state of G, and earlier query behavior can affect later query opportunities, making this an iterative decision making problem and a natural fit for a RL formulation.

Inspired by this observation, we consider RL for finding a better policy for CHOOSEQUERY. We'll refer to an instance of the focused reading algorithm with a learned CHOOSEQUERY policy as the *RL Query Policy*. All other focus reading functionality is the same as in the baseline. For actions, we consider a simple binary action choice: exploit (conjunctive query) or explore (disjunctive query). We represent the state of the search using a set of features that include: (f1) the current iteration of the search; (f2) the number of times a participant has been used in previous queries; (f3) whether the participants are chosen from the same connected component in G; (f4) the vertex degree of participants; and (f5) the search iteration in which a participant was introduced. With the goal of recovering paths as quickly as possible, we provide a reward of $+1$ if the algorithm successfully finds a path, a reward of -1 if the search fails to find a path, and assess a "living reward" of -0.05 for each step during the search, to encourage trying to finish the search as quickly as possible.

We trained the RL Query Policy using the SARSA (Sutton and Barto, 1998) RL algorithm. As the number of unique states is large, we used a linear approximation of the q-function. Once the policy converged during training, we then fixed the linear estimate of the q-function and used this as a fixed policy for selecting queries. We trained the RL Query Policy on the separate set of 500 entity pairs, and evaluated it on the same data set of 289 participant pairs used to evaluate the baseline. Table 1 summaries the results of both the baseline and the RL Query Policy. The Query Policy resulted in a 25% decrease in the number of queries that were run, leading to a 24% drop in the number of papers that were read, while at the same time *increasing* the number of paths recovered by 3%. We tested the statistical significance of the

[7] **Dy**namic **C**ell **E**nvironment model of pancreatic cancer.

	All features	− Iteration number (f1)	− Query counts (f2)	− Same component (f3)	− Ranks (f4)	− Particip. intro. (f5)
Paths found	198	199	200	201	**202**	196
Papers read	19,883	20,918	20,531	20,463	27,708	**17,936**
Queries made	433	484	484	467	469	**403**

Table 2: Ablation test on the features used to represent the RL state.

	Empty query results	Ungrounded participant(s)	Low yield from IE
Error cause	12	4	2

Table 3: Error analysis on 18 queries that failed under the RL algorithm.

difference in results between the baseline and RL policy by performing a bootstrap resampling test. Our hypotheses were that the policy reads fewer papers, makes fewer queries and finds more paths. The resulting estimated p-values for fewer papers and fewer queries was found to be near 0, and < 0.003 for finding more paths. An ablation study of the state features found that features (f2) and (f5) had the largest impact on number of papers read; both model the history of the reading task (see the next section for details). This highlights that the RL model is indeed learning to model the entire iterative process.

6 Analysis

Feature Ablation Test: We performed an ablation test on the features that encode the RL state. The results are summarized in Table 2. Similar to Section 5, we grouped the features into five different groups, and we measured the impact of removing one feature group at a time. Overall, the amount of paths found doesn't have a significant amount of variance, but the efficiency of the search (amount of papers read and number of queries made) depends on several feature groups. For example, features (f1), (f2), and (f4) have a large effect on both the number of papers read and the number of queries made. Removing the feature (f5) actually reduces the number of papers read by approximately 2K with a minimal reduction in the number of paths found, which suggests that this task could benefit from feature selection.

RL Policy Error Analysis: Lastly, we analyzed the execution trace of eighteen (20% of the errors) of the searches that failed to find a path under RL. The results are summarized in Table 3. The table shows that the main source of failures is receiving *no results* from the information retrieval query, i.e., when the IR system returns zero documents for the chosen query. This is typically caused by over-constrained queries. The second most common source of failures was *ungrounded participants*, i.e., when at least one of the selected participants that form the query could not be linked to our protein knowledge base. This is generally caused by mistakes in our NER sequence model, and also tends to yield no results from the IR component. Finally, the *low yield from IE* situation appears when the the information produced through machine reading in one iteration is scarce and adds no new components to the interaction graph, again resulting in a stop condition.

7 Discussion and future work

We introduced a framework for the focused reading of biomedical literature, which is necessary to handle the data overload that plagues even machine reading approaches. We have presented a generic focused reading algorithm, an intuitive, strong baseline algorithm that instantiates it, and formulated an RL approach that learns how to efficiently query the paper repository that feeds the machine reading component. We showed that the RL-based focused reading is more efficient than the baseline (i.e., it reads 24% fewer papers), while answering 7% more queries.

There are many exciting directions in which to take this work. First, more of the focused reading algorithm can be subject to RL, with the CHOOSEENDPOINTS policy being the clear next candidate. Second, we can expand focused reading to efficiently search for multiple paths between S and D. Finally, we will incorporate additional biological constraints (e.g., focus on pathways that exist in specific species) into the search itself.

Acknowledgments

This work was partially funded by the DARPA Big Mechanism program under ARO contract W911NF-14-1-0395.

Dr. Mihai Surdeanu discloses a financial interest in Lum.ai. This interest has been disclosed to the University of Arizona Institutional Review Committee and is being managed in accordance with its conflict of interest policies.

References

Kevin Clark and Christopher D Manning. 2016. Deep reinforcement learning for mention-ranking coreference models. *arXiv preprint arXiv:1609.08667* .

Paul R. Cohen. 2015. DARPA's Big Mechanism program. *Physical Biology* 12(4):045008.

Jin-Dong Kim, Ngan Nguyen, Yue Wang, Jun'ichi Tsujii, Toshihisa Takagi, and Akinori Yonezawa. 2012. The genia event and protein coreference tasks of the bionlp shared task 2011. *BMC bioinformatics* 13(11):1.

Jin-Dong Kim, Tomoko Ohta, Sampo Pyysalo, Yoshinobu Kano, and Jun'ichi Tsujii. 2009. Overview of bionlp'09 shared task on event extraction. In *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing: Shared Task*. Association for Computational Linguistics, pages 1–9.

Karthik Narasimhan, Adam Yala, and Regina Barzilay. 2016. Improving information extraction by acquiring external evidence with reinforcement learning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP 2016)*.

Claire Nédellec, Robert Bossy, Jin-Dong Kim, Jung-Jae Kim, Tomoko Ohta, Sampo Pyysalo, and Pierre Zweigenbaum. 2013. Overview of bionlp shared task 2013. In *Proceedings of the BioNLP Shared Task 2013 Workshop*. pages 1–7.

Marco Pautasso. 2012. Publication growth in biological sub-fields: patterns, predictability and sustainability. *Sustainability* 4(12):3234–3247.

Young-Woo Seo and Byoung-Tak Zhang. 2000. A reinforcement learning agent for personalized information filtering. In *Proceedings of the 5th international conference on Intelligent user interfaces*. ACM, pages 248–251.

Richard S Sutton and Andrew G Barto. 1998. *Reinforcement learning: An introduction*. MIT press Cambridge.

Don R Swanson. 1986. Undiscovered public knowledge. *The Library Quarterly* 56(2):103–118.

C. A. Telmer, K. Sayed, A. A. Butchy, Kaltenmeir, Michael Lotze, and N. Miskov-Zivanov. 2017. Manuscript in preparation.

Marco A. Valenzuela-Escárcega, Gustave Hahn-Powell, Thomas Hicks, and Mihai Surdeanu. 2015. A domain-independent rule-based framework for event extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing: Software Demonstrations (ACL-IJCNLP)*.

Byoung-Tak Zhang and Young-Woo Seo. 2001. Personalized web-document filtering using reinforcement learning. *Applied Artificial Intelligence* 15(7):665–685.