Trigger Extraction in Biological Text for Question Answering

[CS224N]Fall 2012: 12-7-12

Jaclyn Chen jaclync@stanford.edu

Ashley Jin ashpjin@stanford.edu

Jessica Tai jmtai@stanford.edu

1 Introduction

When confronted with a large amount of text, it is difficult to manually extract important and relevant information from it. Having a way to extract that information is a highly interesting research problem that has the potential to improve question answering capabilities. Specifically, identifying sections of text that describe a term, phenomenon, or abstract idea and transforming them into structured formats is useful for returning answers based on knowledge and understanding of the text or document rather than answers based on similarity metrics like those used by search engines.

This Vulcan-associated project is an initial exploration to question answering in a large biological text corpus. Our corpus is the contents of a biology textbook. Our goal is to extract triggers from in biological processes manually extracted from the text corpus. Because a process can be described in many ways in natural language, we focus on identifying the active verb, trigger, of each step in the process as a method for understanding. In summary, given a process description, we will find the triggers while disregarding tangential or unrelated information.

The triggers are captured using features. These features are derived from the sentence structure, word features, and general linguistic context instead of similarity, keyword matching, or other non-linguistic characteristics. Then using a classifier, we find a subset of the linguistic features that will extract triggers with high recall and precision. We then analyze the subset of features returned by the decision tree to identify linguistic signals that characterize trigger words.

2 Related Work

Previous research in extracting events from of a biological process from text generally were more focused on specific types of triggers and events[1] rather than formalizing features to extract triggers from a broad range of unknown categories. Miwa et. al. explored the extraction of events, triggers and their respective arguments, using rich features. Their overall system dealt with two types of trigger and used a machine learning classifier per trigger. While their features inspire useful insight, the overall project was too specific to apply to extracting triggers without knowing the existing categorizations. Our project will

attempt to extract triggers using linguistic clues without specific knowledge about the corpus, e.g. prior knowledge about biology.

Other research projects have developed methods of extracting and relating triggers and arguments to the triggers [2] but again focus on specific types of biological triggers and events rather than general triggers and events. Riedel and McCallum jointly predicted triggers and arguments then used features along with arguments to filter out bad triggers. We will be using some similar features such as the part of speech tag but will not progress to finding arguments for the triggers.

3 Data

Because this is a new research project and there were not existing data sets available that fit our purposes, we had to annotate our own test and train data. Our group, along with some computer science students, biology students, and Vulcan employees manually extracted process descriptions from a textbook, Biology (8th edition) by Neil A. Campbell and Jane B. Reece.

A process description was defined as a continuous piece of text describing a sequence of events for some biological purpose. Ideally, each continuous piece of text would be between 2-6 sentences but some process descriptions were longer. These pieces of text did not include introductions to or generalized conclusions about the process described. Additionally, we tried to extract only processes that had at least 3 distinct steps.

For example, the following process description describes the process of a photosystem harvesting light:

> "*When a photon strikes a pigment molecule in a light-harvesting complex, the energy is passed from molecule to molecule until it reaches the reaction-center complex. Here, an excited electron from the special pair of chlorophyll a molecules is transferred to the primary electron acceptor.*"

Here the steps are (1) photon **strikes** a pigment molecule, (2) energy is **passed**, (3) electron is **transferred**.

We extracted 125 different process descriptions across numerous chapters of the biology textbook. Spanning multiple chapters ensured that the types of biological processes described were diverse and avoids overfitting on specific words when we applied our trigger extraction algorithms. For evaluation purpose, each process description was assigned a unique id that does not correlate to any features of the process description.

From these 125 process descriptions, we randomly chose 40 to be training data and 40 to be test data. We manually annotated the true triggers in the 40 training process descriptions. The 40 test process descriptions were annotated by external collaborators. Although those collaborators also reviewed part of the training data annotations, discrepancies in what constitutes a trigger still exist. We will discuss the effects of these discrepancies in the Analysis section.

The training and test data are stored in separate tab-delimited files with columns for process id, process description text, and a list of indexes of the true trigger words in the text.

4 Experiments

We used Stanford University's CoreNLP library to run linguistic analysis (e.g. parse trees, dependency trees, POS tagging) on our sources, process descriptions and book text. We used the weka java API for the decision tree classifier.

4.1 Baseline

In order to analyze the characteristics of the triggers, we implemented a baseline trigger extractor that chooses every single verb as a trigger. This guarantees that we will have high recall but very low precision. Now, we analyze the false positives and negatives for patterns.

After running our first baseline where every verb in the verb phrase identified as a trigger, we analyzed the results on the 40 annotated training processes. In our evaluation codes, we listed the false positives and false negatives for each process. We examined most of the errors and categorized them into the following with estimated frequencies. The most common errors we observed were:

- Auxiliary verbs (~20%)

Verbs like "is, are, have" and its derivative forms are auxiliary verbs that lead to properties of the subject or a main verb, and thus are never triggers. In our second version of baseline, we remove these auxiliary verbs by checking the verb with a list of all auxiliary verbs and its derivatives.

- POS tag errors (~20%)

Surprisingly, we noticed a fair amount of errors stemming from incorrect POS tags. While a verb can be mistagged as an adjective or a noun and oftentimes cause false negatives, a noun can also be mistagged as a verb and result in false positives. These incorrect POS tags further propagate down to the parsing process, causing incorrect parsing structures.

- Relative clauses (~20%)

It is fairly often that a verb is describing a noun, instead of participating in the process. For example, a verb can be describing the function that its subject does, while the subject is involved in the process through another trigger. In these cases, we utilize the parse tree structure that a noun phrase tree includes all description sub-phrases as relative clauses.

- Stative verbs (~15%)

According to the semantics of verbs, certain verbs do not express a state change to its subject. These stative verbs behave like triggers in parsing structure, and thus we compiled a list of stative verbs with online resources and observations. Ideally, a comprehensive and easy-to-use dictionary for verbs should be available that categorize each verb also depending on its context.

- Verbs among verbs/nominalizations (~12%)

There are situations where an expression is carried out by a series of verbs, possibly connected by prepositions, adverbs, or "to." All of the verbs could contribute to the whole expression, yet most often only one verb is considered as a trigger. This trigger is the verb that has the most clear and correct meaning of action. However, in our project, we do not have resources to analyze the verb semantics and deal with such cases.

- Parsing errors (~5%)

Incorrect parsing structures can lead to false negatives/positives when a verb is not identified or misplaced. In particular, in biology text, certain symbols or numbers, which are part of the chemical formula, can be incorrectly POS tagged and result in parsing errors. These errors can be avoided by importing knowledge in biology field to group a series of nouns, symbols and numbers instead of treating them separately.

- Verbs with weak subjects (~4%)

Certain verbs are not triggers because their subjects are “weak”, i.e. not supposed to be a subject that acts on or is acted on during a biological process. For example, if the subject is “process”, the main verb might be a legitimate trigger word. However, the sentence is a result of the process and thus the verb does not participate in the process itself. In such cases, we would need to extract and classify the subject or manually remove these summary sentences from the process paragraph.

- Nominalizations (~3%)

An event or action can be described in the form of a verb or a noun. Most of the verbs have their noun forms, called nominalizations. Especially in literature like a textbook, nominalizations are used often for more fluent and sophisticated writing. Mappings between verbs to its nominalization forms, such as NomLex and WordNet derivation links, allow us to map a noun to its verb form. These errors were false negatives in our current verb-only baseline.

- Author speaking (~1%)

Occasionally in the textbook, we saw authors speaking while describing processes. Some of these sentences do contain trigger-like verbs, however these sentences are authors’ suggestions instead of being part of a biological process. In such cases, we can extract the subject of a verb and discard it if the subject is a first-person pronoun.

Based on these categories of errors, we compiled a list of signals that might indicate if a verb/noun is a true trigger. Our signal-based system implements these features.

Originally, we also chose all nominalizations, words with noun part of speech tags derived from action verbs, as potential triggers as well. Nominalizations were detected by two sources: 1) NomLex, and 2) WordNet derivation links. Both sources map nominalization nouns to verbs. However, after analysis, we realized that most nominalizations were not triggers. Therefore, we removed nominalization extraction from our baseline. We will discuss signals that might indicate good nominalization triggers (as opposed to filtering out bad nominalizations triggers) in the Nominalization Signals section.

4.2 Signal-based Algorithms

As our first improvements to the algorithm, we implemented features (signals) for determining if a verb is or is not a trigger (Remember, we decided to not include nominalizations by default.). Now our algorithm considers every word in the linguistic tree tagged as a verb or under “Verb Phrase (VP)” tree a trigger candidate. The candidate will be added to our list of triggers if it is not filtered out by the activated features.

Due to our high confidence in auxiliary verbs and words inside parentheses almost never corresponding to triggers, we added those to the baseline and used the resulting precision and recall scores as the new baseline for deciding the usefulness of the remaining features. Briefly, we will describe the features and their implementations.

1. Auxiliary Verb removal: A list of auxiliary verbs is compiled based on linguistics definitions. For each potential trigger, we find the lemma of candidate and discard it if the lemma exists in the list of auxiliary verbs.
2. Ignoring Parentheses: Discard all potential triggers that appear inside parentheses.
3. Past Tense Detection: If the verb POS tag is “VBN”, check if any of the child dependencies have a “VBD” label. If so, the currently considered trigger is in the past tense.
4. Relative Clause Detection: If the verb is in a VP that is has an NP as an ancestor, this verb is in a relative clause describing a noun.

5. Stativeness Ratio: To measure stativeness, we ran a script over the entire book text that parsed the sentence, found every verb, and recorded the number of times that verb was used as a present participle or gerund (part of speech tag VBG) versus the number of times it was not. The ratio of number of times used as VBG versus number of times used overall is the our measure for the stative-ness of the verb. A verb with a higher score is more likely to be a trigger than a verb with a low score. The current implementation is very basic and finding features to describe the stativeness of a verb is a separate, interesting problem.
6. Negation Detection: If the verb has a parent dependent that is ‘not’, ‘n’t’ or ‘without’, the current verb is being negated in the sentence.
7. Labelling Verb Detection: Checked each potential trigger’s lemma against a list of “labelling verbs” such as ‘call’ and ‘name’
8. Inter-Event Verb Detection: Checked each potential trigger’s lemma against a list of ‘inter-event verbs’ such as ‘result’ and ‘cause’
9. First Person Subject Detection: Removed verb triggers if their subject was a first person pronoun (“we”, “I”) as such sentences usually represent the opinion or viewpoint of the book authors. We used the dependency tree to find the subject of the verb and PA3’s Pronoun.java file to determine the speaker of a given pronoun
10. Sequence Word Detection: If the currently considered trigger has a parent dependent word that is an ordinal or a sequence identification word, e.g. After, then, before, then the current trigger is more likely to be a trigger. The current implementation could be improved to better capture locality.

We will display and analyze the results of these features in the Results section.

4.3 Nominalization Signals

Although our project focus was on filtering out irrelevant verbs from our triggers, we experimented with a few signals that might determine if a noun should be added to our trigger set. Here we briefly describe our initial observations and the signals we implemented. However, much more experimentation and analysis can be done for nominalizations.

Especially in biology, processes can be referred to by nouns, e.g. phosphorylation or activation. To determine if a noun has a nominalized verb form, we extracted all the nouns and their verb forms present in NomLex and WordNet’s derivation links datasets. After constructing gold standards, we had two key observations: 1) there were some property similarities about noun nominalization triggers and 2) most noun nominalizations were not used as triggers. We then attempted to construct signals to indicate that a nominalization was a trigger (opposed to signals to remove verbs from the trigger set).

We noted that if the head of the most general NP of a sentence was a nominalizatoin, it was more likely to be a trigger. However, finding the head of the most general NP proved to be a complex problem so we leave its implementation and testing to Future Work.

Another strong observation for nominalized triggers is that they often appeared as either the subject or the object of an inter-event verb (our finite, self-defined list of verbs such as “occur”, “happen”, “cause”, “result”) when they were a trigger. Since these “inter-event” verbs are used to connect different steps of the process, both the subject and the object of such verbs could be good trigger candidates. Below is the pseudocode for these features:

```
for each sentence, obtain its dependency tree
      for each mention head noun that can be nominalized into a verb
      include as trigger if
         ■ subject of inter-event verb
         ■ object of inter-event verb
         ■ contains prepositional arguments
```

We will briefly discuss the results of these nominalization features in the Results section.

4.4 Decision Tree Algorithms

After implementing features, we decided to further analyze the impact of different features using a decision tree classifier. We use the 40 annotated training processes to generate train files, one for verbs and one for nominalization, that describe potential triggers using the active features and the true classes. We use the annotated testing processes to generate similar test data files. The weka java api reads in both data files and performs a decision tree classification. The output is displayed in the Results section.

5 Results

As mentioned in Data, we had 40 training examples, process descriptions with annotated gold triggers. To test the accuracy of our feature combinations, the program extracted trigger indexes and compared the extracted trigger indexes with the gold trigger indexes. Because the baseline already had very high recall, we focused on increasing precision.

We first implemented all the features and tested them individually against the new baseline, extracting all verbs and filtering out auxiliary verbs and verbs inside parentheses. In our evaluation, macro precision/ recall is calculated by averaging over all processes. While the macro measures treat every process equally, micro precision/recall weighs on the number of extracted triggers and gold triggers, respectively, of each process. Thus, macro measures capture the overall performance and avoid favoring longer processes; micro measures show more process-level performance and are fair on variable lengths of processes.

Table 1: Testing Data Precision, Recall, and F1 scores

TRAIN	**nom?**	**macro precision**	**macro recall**	**macro F1**	**micro precision**	**micro recall**	**micro F1**
original baseline	no	0.502654	0.860972	0.634735	0.447090	0.857868	0.587826
new baseline	no	0.605876	0.860972	0.711243	0.550489	0.857868	0.670635
+past tense	no	0.620132	0.845556	0.715509	0.560811	0.842640	0.673428
+relative clause	no	0.635441	0.849722	0.727123	0.573883	0.847716	0.684426

+negation	no	0.614348	0.860972	0.717047	0.555921	0.857868	0.674651
+term labelling	no	0.618279	0.860972	0.718717	0.561462	0.857868	0.678715
+inter-event labeling	no	0.620816	0.860972	0.721433	0.56714	0.857868	0.682828
+first person	no	0.601280	0.844306	0.702365	0.52980	0.847716	0.669339
+sequence word	no	0.605876	0.860972	0.711243	0.550489	0.857868	0.670635
+all*	no	0.674115	0.817639	0.738973	0.615970	0.822335	0.704348
original baseline	yes	0.281719	0.962917	0.435906	0.244845	0.964467	0.390545
new baseline	yes	0.320593	0.962917	0.481032	0.281065	0.964467	0.435281

* we did not test for stativeness since the VBG ratio was not a strong signal and generally resulted in NaN precision given any threshold value.
** these results were calculated over the training set

After implementing the features, we compared the new baseline performance with the greatest active feature set on a different dataset, test data. The results are shown below.

Table 2: Testing Data Precision, Recall, and F1 scores

TEST	**nom?**	**macro precision**	**macro recall**	**macro F1**	**micro precision**	**micro recall**	**micro F1**
new baseline	no	0.635055	0.761974	0.692749	0.603636	0.761468	0.673428
+all*	no	0.730039	0.705713	0.717670	0.700000	0.706422	0.703196
new baseline	yes	0.413110	0.949177	0.575671	0.344426	0.949541	0.505495
+ all*	yes	0.584067	0.795039	0.673416	0.540881	0.788991	0.641791

Finally, the weka decision tree output after running on test data (for greatest active feature set) is shown below. Weka created one decision tree for verb triggers and another for nominalization triggers.

Table 3: Decision Tree Results (Weka)

Verb Decision Tree	Nom Decision Tree
is-inter-event = 0 | stativeness <= 0.016949: no (105.0/16.0) | stativeness > 0.016949 | | relative-clause = 0 | | | past-tense = 0 | | | | negated = 0: yes (238.0/89.0) | | | | negated = 1: no (2.0) | | | past-tense = 1: no (3.0) | | relative-clause = 1: no (12.0/1.0) is-inter-event = 1: no (9.0) Number of Leaves : 6 Size of the tree : 11	: no (248.0/8.0) Number of Leaves : 1 Size of the tree : 1
Correctly Classified Instances 231 71.2963 % Incorrectly Classified Instances 93 28.7037 % Kappa statistic 0.4259 Mean absolute error 0.3934 Root mean squared error 0.4584 Relative absolute error 78.6715 % Root relative squared error 91.2372 % Total Number of Instances 324	Correctly Classified Instances 208 87.395 % Incorrectly Classified Instances 30 12.605 % Kappa statistic 0 Mean absolute error 0.1502 Root mean squared error 0.3449 Relative absolute error 98.1706 % Root relative squared error 100.2904 % Total Number of Instances 238

6 Analysis

6.1 Training data tests

From table 1, we can see that most of the features we implemented raised precision while lowering recall. This is expected behavior for good features since we do not expect individual features (except for auxiliary removal and parenthesis ignoring) to detect triggers with complete accuracy. In general, the

features filter out some bad trigger candidates while losing a couple good triggers.

We are not concerned about the lower recall since decisions about trigger candidates should be made as a combination of all the features. Our implementation that uses “all” features is still a naive implementation where if the trigger candidate fails to satisfy a feature constraint, it gets thrown away. In future implementations, the trigger should be judged based on a weighted combination of its features or using a smarter classifier, e.g. linear SVM.

When running tests on the nominalization features, we noticed that the features capturing nouns that appeared as subject and objects of inter-event verbs were the more effective signals that we implemented. Additionally, the feature that chose nominalizations that were arguments of inter-event verbs was too widely applicable and lowered our precision by generating many new false positives.

6.2 Testing data tests

From table 2, we see that the baseline recall on the test data without using nominalizations was lower overall while the precision was higher. This is likely due to overfitting on the training data. Given that we only had 40 training examples, many features do not have numerous examples. For example, a feature like negation may only be represented once or twice over the whole training or test data but may be much more prevalent in reality. For this particular test set, the number of nominalization triggers was significantly higher than in the training data.

However, the pattern of increased precision but lowered recall when using the greatest active verb feature set was evident on the testing data as well as the training data. This helps confirm that our selection of verb features is generally good for filtering out bad triggers.

As expected, the new baseline with all noun nominalizations increased recall but lowered precision. We can consider this recall the maximum attainable recall due to parse and dependency tree errors that are intrinsic to the corenlp library and beyond our control. The recall gets smaller when we run “all” features because we are not taking all nominalization candidates by default but instead using the features to decide if we should include the candidate in the trigger set. (Again, this is the opposite of verb triggers that we mark as triggers by default and filter out based on features.) We notice that the precision when using all the features, verb and nominalizations, is higher than the nominalization baseline but lower than the scores for just using verb features. This is due to the numerous false positives introduced by the nominalization features. Therefore, our nominalization features are not rich enough to capture the characteristics of nominalized triggers.

6.3 Classifier

From Table 3, the decision tree output for verbs, we can identify the strongest triggers. Keeping in mind that overfitting due to the size of the available data is a factor in our experiments, we see that identifying and discarding inter-event verbs is very helpful. Surprisingly, the decision tree then identifies stativeness ratio of the verb as another strong trigger. Remember that in generating results for individual features, the stativeness ratio was unhelpful. The decision tree was able to find an appropriate threshold for the ratio that helped classification significantly. Relative clause and past tense are not identified as strong feature individually but when used to filter down triggers to a question of negation, become part of a strong combination of features for classification. Overall, the decision tree is able to correctly identify over 70% of our test data based off our small feature set and training data.

The nominalization decision tree is very simple. The decision tree simply classifies all nominalization trigger candidates as “not trigger”. Despite implementing a few basic features, the overwhelming number of negative examples in the nominalization data forces the decision tree into making a naive classification that results in 87% accuracy! In order to improve classification in terms of identifying helpful features (not necessarily focusing on improving accuracy), we will need to find more complex, accurate features for nominalizations and find more positive nominalization trigger examples. Using a smarter classifier would not be helpful in this particular context.

7 Summary and Future Work

Our experiments in implementing features that identify good triggers in process descriptions allowed us to find good features for verb triggers but not for nominalization triggers. We were able to implement features for verb trigger classifications that did not require any knowledge about the context of the text excerpts (i.e. that they were related to biology) but relied solely on linguistic context and structure. We were able to classify verbs into “action/event” related verbs and “not action/event” verbs based solely on the features we chose and using a relatively naive classifier with greater than 70% accuracy. Thus, our experiments in finding features that would help a program begin to understand the information given by text were helpful for future work to build upon.

Future work in trigger extraction from these process descriptions should focus on (1) data collection, (2) nominalization triggers, and (3) smarter classifiers. One constraint on our experiments was simply the lack of annotated data for training and testing. The small size forced us to be very cautious about overfitting the features to the data and it will be interesting to see if the features we identified are still useful over a much larger dataset. Additionally, our foray into nominalization triggers did not identify useful features due to the dataset skew so future work should try specifically to construct features for identifying the nominalization triggers. Finally, we used a decision tree that chooses based on mostly binary features. Being able to construct and incorporate more complex, discrete or continuous valued features, and use them in a smarter classifier could lead to more interesting results and insights into process understanding.

Related expansions of this project include ordering triggers chronologically, identifying arguments for triggers, and continuing to answering questions using the current understanding of the process triggers.

8 Acknowledgements

We would like to acknowledge and thank Jonathan Berant, our project advisor, for his guidance and input. Additionally, we would like to thank our external collaborators including Vulcan employees who helped annotate and extract data for our train and test sets.

9 References

[1]Makoto, M., Saetre, R., Kim, J.D., and Tsujii, J. "Event Extraction with Complex Event Classification Using Rich Features." *Journal of Bioinformatics and Computational Biology 8.1* 2010: 131-146.

[2]Riedal, S. and McCallum, A. “Fast and Robust Joint Models for Biomedical Event Extraction.” *EMNLP '11 Proceedings of the Conference on Empirical Methods in Natural Language Processing* 2011: 1-12.