RESEARCH

# Customizing the information extraction pipeline: An experiment in text mining for climate science

Elias Aamot[†] and X Y

Full list of author information is available at the end of the article
[†]Equal contributor

**Abstract**

Information extraction can assist researchers by providing structured access to key findings. In traditional information extraction systems, entities are extracted in a first step, and events in a subsequent step. In this paper, we explore information extraction to support research in climate change science. The structure of the information extraction task poses challenges to the canonical pipeline. We therefore explore adapting the extraction pipeline to the task at hand, and show that such customizations can yield improvements for certain tasks.

**Keywords:** information extraction; text mining; natural language processing; annotation

## 1 Introduction

Information extraction is the task of extracting structured information from natural language text. Historically, information extraction research focused on named entity recognition (NER), that is, figuring out which entities are mentioned in a piece of text. As research progressed, researchers started to turn their attention to gradually more complex tasks, such as relation extraction - the task of discovering the relations that hold between the extracted entities, and later event extraction, which aims to extract more information patterns, such as n-ary relations, or relations between relations, an example of the latter being discovering temporal ordering.

Machine learning based event extraction systems tend to use a pipeline of components, in which the first extracts entities, a subsequent component extracts triggers for events, and finally a component tries to determine participant relations between entities and events, and potenitally event-event relations. The pipeline architecture has two drawbacks: Error propagation, because each component makes hard choices that cannot be undone by downstream components, and the inability to model constraints from entities to events. Some recent research effort has therefore focused on joint extraction architecture that try to optimize globally accross all components. However, joint extraction architectures have not reached the level of maturity required for state-of-the-art performance, and have other drawbacks, such as the increased complexity of learning a joint probability distribution, that make them infeasible for certain circumstances, such as when training data is scarce. Off-the-shelf joint information extraction systems are also lacking.

Verify that th
are backed up
work section

The canonical information extraction pipeline architecture has worked well for many extraction tasks, but this paper argues that there are other ways of structuring the information extraction pipeline that can potentially yield better results for certain extraction tasks.

This paper presents an experiment made in the climate change domain showing the benefit from customizing the information extraction pipeline for the domain at hand.

The main contributions of this paper are:

- Showing that customizing the extraction pipeline can yield improved performance on certain tasks.
- Describing two types of changes that can be made to the pipeline: Reordering the components, and splitting a component into smaller components.
- Exploring heuristics that point out which extraction pipeline yields the best results for a certain extraction task.
- Introducing text mining in the climate change domain.

Section 2 will introduce text mining in the climate change domain, section 3 will introduce the information extraction systems used to conduct the experiment. The experiment itself will be described in section 4, and the results provided in the same section. Finally, section 5 will discuss the results and provide pointers for future work.

## 2 Text mining in the climate change domain

Research progress in the field of climate change is hampered by the wide range of relevant disciplines, which include, among others, climate science, earth science, oceanography, ecology and biochemistry. While a researcher may be able to keep up with the state-of-the-art within his/her field of specialization, it is impossible to keep track of all potentially useful findings in the related disciplines. Text mining can in this regards support the research effort by extracting structured and indexed representations of the findings in all relevant disciplines. This paper represents a line of work towards an information extraction systems to support scientific research in the domain of climate change science.

Climate change science aims to understand the causes and effects of climate change. The knowledge in the field can to a large extent be modelled as causal or correlative relations between variables, for instance *increase in atmospheric* $CO_2$ *causes decrease in oceanic p.H.* or *decrease in mineral nutrients correlates with decrease in phytoplankon abundance*. Our hypothesis is that this representional scheme is generic enough to represent fact accross all the relevant disciplines, yet specific enough to be useful to researchers in the domain. One observation that support this hypothesis is that this representational scheme is expressive enough to support inference of more complex structures, such as feedback loops —situations in which a change to a variable sets in motion a chain of events that, in turn, cause a new change to the original variable. Feedback loops are of particular interest to researchers in the field, due to the serious consequences of a potential feedback loop.

This paper deals with automatic annotation rather than full information extraction. Annotation is the process of marking relevant text spans with category labels, and determining which relations hold between the marked text spans. For each task, the annotation guidelines specifies the set of category labels and relations which are used, and how they are used. The goal of automatic annotation is to automatically annotate unseen text with human-like annotation quality. To bridge the gap

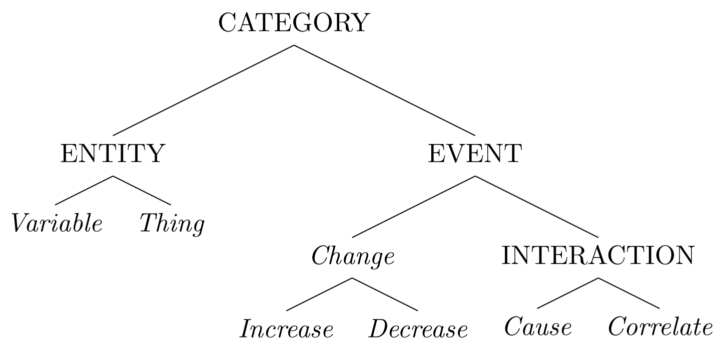

**Figure 1** Type Hierarchy of triggers in the Annotation Scheme

between automatic annotation and full information extraction, a post-processing procedure extracts the annotations and stores them as structured data.

The annotation scheme used in the climate change science domain is described in detail in []. The remainder of this section will present the most important aspects of the annotation scheme. As an entry point, consider the following text fragment, taken from [1] (some parentheses removed for clarity):

> Increasing concentrations of CO2 cause a strong decline in growth, which decreases by up to 53% over the investigated CO2 range. Although the total carbon quota (TPC) is not affected by CO2, the organic carbon quota (POC) gradually increases while the inorganic carbon quota (PIC) shows a substantial decrease.

The desired annotation for the sentences under the annotation scheme is presented in Figure [TODO: Erwin, I need your help for this]. The general annotation framework consists of two primitives: *Triggers* and *arguments*. Triggers are used to mark text spans that express interesting information of a specific type. Each trigger belongs to a category, taken from the categories presented in Table 1. The usage of each category will be explained below. Arguments are directed links that mark relations between trigger spans. Arguments are typed, and certain types of trigger categories are required to take certain argument types, as explained below.

*Variable* is in our annotation scheme defined as a *quantitative variable*, meaning an entity than can be measured and assigned some value along some ordered axis, either as a numerical value or a value from a totally ordered set of discrete states. This includes among other counts, frequencies and ratios. Examples of variables from the corpus include $CO_2$, *organic carbon quota*, *surface ocean pH* and *mean light intensities*. Scientific articles normally contain a wide range of variables, but our annotation scheme limits itself to quantitative variables that occur in an event of interest, as only these can be used for making inferences by the downstream components.

*Thing* is used to annotate any span of text that functions as the argument of an event, but does not fulfil the requirements for being annotated as a variable, as described above. This occurs in phrases such as "*down-regulation of the gene*", where *down-regulation* clearly signals a quantitative change, and the only possible explicit argument of the change is *the gene*. However, *the gene* is not a variable. The variable that changes is the implicit *activity level of the gene*, under the view that *down-regulate* can be paraphrased as "*decrease the activity level of*". The explicit argument is therefore annotated as a *thing* rather than a *variable*.

The underlying motivation for maintaining *thing* and *variable* as separate categories, is that the category *thing* signals to the post-processing procedure that additional semantic interpretation should be factored in from the event trigger in order to specify the variable in question, as in the *down-regulate* example.

The change event categories are used to mark text spans that describe a directional, quantitative change in the value of a quantiative variable. *Increase* is used to annotate a change in the positive direction, *decrease* annotates changes in the negative direction and *change* is used when the direction of the change is underspecified in the text. The change event takes the entity that undergoes the change as a *theme* argument. A change event is not annotated unless an explicit entity that undergoes the signalled change can be inferred from the text.

Interaction categories mark text spans that explicitly signal interactions between two change events. The types of interactions that are used in the corpus are *cause* and *correlate*. Cause events take two arguments: An *agent*, which marks the change event that causes the other change event to come about. The *theme* argument is used for the caused change event. Correlation events also take two arguments, the *theme* and *co-theme*. It was noticed during corpus annotation that there is a tendency in natural language to focus on one of the changes as initiating the other, giving such correlations a directional interpretation. For correlations with an directional interpretation, the initiating event is marked as the theme, and the other event is marked as the co-theme. If the correlation has no directional interpretation, the event that is the syntactic object of the correlation trigger is marked as co-theme, and the other event as theme by default.

Sometimes an interaction holds between a change event and an entity or between two entities, rather than between two change events. An example of this is shown in the annotation of "*Although the total carbon quota (TPC) is not affected by CO2 ...*". Such entities are interpreted as the arguments of an implicit *change*.

It is not uncommon that the trigger word for a change event is used causatively, as in the sentence "*Heightened atmospheric $CO_2$ levels increase global temperatures*", where *increase* both signals an increase in the variable *global temperatures* and signals a causal relationship between "*heightened atmospheric $CO_2$ levels*" and "*increased global temperatures*". In such cases, both change events are annotated normally, but in addition, the causative change event takes the other change event as an *agent* argument, as if it were an interaction event.

One aspect in which our annotation scheme differs from the types of annotation schemes traditionally used in biomedical corpora, such as GENIA[2], is that biomedical corpora normally annotate entities with types from a rich domain ontology, whereas our annotation scheme only subtypes entities into the broad categories *variable* and *thing*. The main reason for this is the lack of onotologies that cover all the disciplines related to climate change science.

## 3 Systems

Initial experiments showed that the traditional information extraction pipeline performed poorly on the extraction task defined above. We hypothesised that the performance of the traditional pipeline on our extraction task was degreaded by the fact that trigger words are not as indicative of entities in our annotation scheme as they are in most biomedical annotation schemes. This is mainly because whether a phrase corresponds to a variable of interest strongly depends on the context. As an example, compare the usage of $CO_2$ n the following sentences:

1 "$CO_2$ is a colourless, odourless gas."

2 "Increasing concentrations of $CO_2$ cause a strong decline in growth."

To test the hypothesis, we conducted an experiment which evaluated a system using an alternative pipeline against a basine system using the canonical pipeline. This section presents both systems.

### 3.1 Baseline system: TEES

To act as a benchmark for the traditional information extraction pipeline, the Turku Event Extraction System (TEES) [3] was selected. TEES was selected for the evaluation because it is has shown state-of-the-art performance over several years of development, can extract events with arbitrary structures and can be trained with custom data sets. TEES has been developed for BioNLP, and has attained high scores on a number of shared tasks, including BioNLP 2009 (1st place) [4], BioNLP 2011 (1st place in 4/8 tasks) [5] and Drug-Drug Interactions 2011 Challenge (4th place)[3]. The source code is publicly available[1].

The TEES pipeline consists of the following steps: Preprocessing, Named Entity Recognition, Trigger Detection, Edge Detection and Unmerging.

Preprocessing consists of sentence splitting, constituency parsing and conversion to Stanford dependency representation[6]. When TEES is applied in the biomedical domain, sentence splitting is conducted using the GENIA sentence splitter[2], and parsing using the BLLIP-parser [7] with the McClosky model for biomedical text [8]. Conversion to dependency format is performed by the Stanford dependency converter [6]. Because the textual material in our corpus was not from the biomedical domain, the domain independent Stanford parser [9] was instead used for preprocessing in the experiment presented in this paper.

The Named Entity Recognition, Trigger Detection, Edge Detection and Unmerging components are all implemented as machine learning-based multiclass classifiers that use the $SVM^{MULTICLASS}$ implementation of SVM for classification.

TEES has participated mostly in event extraction-focused shared tasks, in which named entities were given as input to the system, so the NER component is normally not used, but can be included into the pipeline if required by the task, which was the case in the experiment presented in this paper. The NER component makes a linear pass over all the tokens in the sentence, classifying every token as belonging to either one of the entity categories or as not an entity. Normally, each individual token is treated as an independently extracted entity, but if two or more subsequent tokens are classified as belonging to the same class, they are merged to form a multi-token entity if a multi-token entity with the exact same text string can be found in the training data. Whereas this strategy works well for the biomedical domain where the length of entity text spans is normally three words or less, it fails to

[1]https://github.com/jbjorne/TEES

[2]http://www.nactem.ac.uk/y-matsu/geniass/

produce entities of unbounded length as required by our annotation scheme for climate change science. However, as described below, we define the scoring metrics in such a way that this should not negatively influence the performance of TEES, based on the simplifying assumption that identifying the correct entity boundaries is a task of a post-processing component.

Trigger Detection also makes a linear pass over the tokens in the sentence, detecting trigger words for event categories. This is also cast as a multiclass classification problem, where every token is either classified as belonging to one of the event categories, or as *none*, i.e. not a trigger for any event category. In the same way as with NER, multiple subsequent tokens can be merged to a single trigger if given the same class, and a trigger with the exact same text string can be found in the training data.

Edge Detection detects arguments of the triggered events. First, the system generates all the argument relation candidates, which are all pairs of events and entities or events and events. For every candidate, determining whether an argument relation holds between the two triggers or not is again cast as a multiclass classification problem, where the possible classes consists of all posssible argument relations (theme, co-theme, agent) as well as the none-relation. Because this scheme tries to detect argument relations between not only events and entities, but also between pairs of events, this can create complex events with any arbitrary structure. Note here that detection only operates with pairs of *triggers*, so that it can detect for instance that an agent relation holds between a cause event trigger and a change event trigger, without regards to whether the argument structure of the change event has been inferred yet.

In case of overlapping events, i.e. events that share some trigger, some cleaning up is required to separate the events. This is performed by the Unmerging step, which uses a classifier to determine whether an argument branch should be unmerged into a separate event.

## 3.2 Alternative pipeline

The main observation that motivated our change to the canonical pipeline was that entities in our annotation scheme appeared hard to extract directly from text without any information about the change events in the sentence. To ammend this, we split event extraction into to subtasks - change event extraction and interaction event extraction - and reorganized the order of the components, so that change event extraction comes first, before entity extraction. To the best of our knowledge, no out-of-the-box information extraction exist that allows such customization to the pipepline. TEES could not straightforwardly be modified to support this modification, because the features used for entity extraction are not designed to exploit the knowledge provided by the change event extraction step.

A deterministic pattern matching system was developed as a prototype system using our alternative pipeline. This prototype system uses a two-step pipeline. The first step detects change event triggers and entity arguments. The second step tries to detect interaction events in sentences with multiple change events. In addition, a preprocessing step handles sentence splitting, lemmatisation and parsing. The same preprocessing machinery is used for both systems in the experiment.

| | |
|---|---|
| TRIGGER ”increase” | TRIGGER ”decline” |
| TYPE increase | TYPE decrease |
| VAR S amod T | VAR T prep ”in” pobj S |

**Figure 2** Example of two patterns for change events

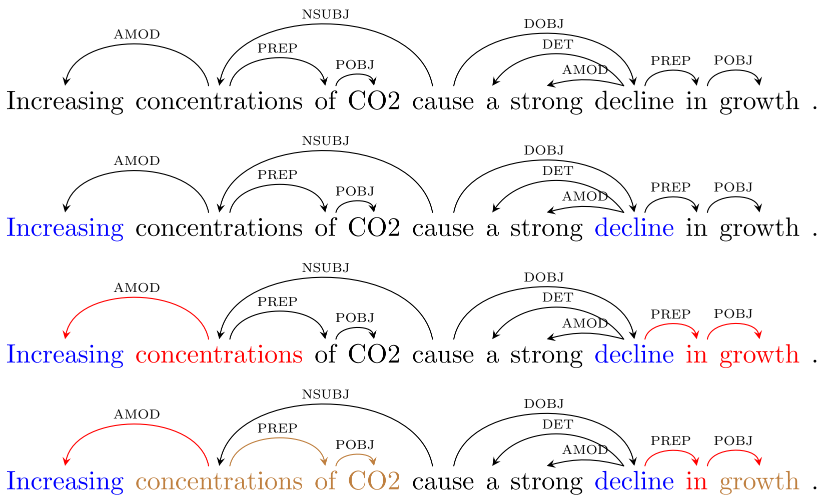
AMOD
NSUBJ
PREP
POBJ
DOBJ
DET
AMOD
PREP
POBJ
Increasing concentrations of CO2 cause a strong decline in growth .

**Figure 3** Example of matching process for change events

Figure 2 gives two example of patterns used in the first step. Each pattern consists of a trigger lemma, an event type and an argument type with a dependency path. The extraction process is illustrated in Figure 3. During extraction, the system first tries to match the trigger lemma against the lemmas used in the sentence. Trigger lemma matches for the two patterns provided in Figure 2 are shown in blue. For each trigger lemma match, the system then tries to match the argument dependency path of the pattern to the dependency representation of the sentence. In the argument dependency path, the letter T represent the node of the trigger word, and the letter S, which will be explained shortly, can match any node. The matching paths are coloured in red in the example.

After matching the pattern, the trigger node becomes annotated as a change event of the type specified in the pattern. The subtree rooted at the node labelled as S will become the theme argument of the change event. As shown in the example, sometimes the trigger node lies in the subtree of the S node, in which case that branch is pruned away, extracting ”*concentrations of CO2*” rather than ”*Increasing concentrations of CO2*” as the argument. The argument becomes annotated as an entity with the entity category specified in the pattern, which can be either VAR (variable) or THN (thing).

For the second step, where interaction events are extracted, simpler patterns are used, with examples of this kind of pattern shown in Figure 4. In addition to the type parameter, *cause* patterns have an *agent* parameter, which specifies which of the two change events gets marked as the agent in case of a match. Finally, all interaction event patterns have a list of sub-patterns. Each sub-pattern consists of a location

| | |
|---|---|
| TYPE cause | TYPE correlate |
| AGENT left | AFTER correlated |
| BETWEEN cause | BETWEEN and |

**Figure 4** Example of two patterns for interaction events

**[Increasing concentrations of CO2] cause a strong [decline in growth].**

| | |
|---|---|
| BEFORE | |
| BETWEEN | cause a strong |
| AFTER | . |

**Figure 5** Splitting a sentence for matching of interaction event patterns

and a string. During matching, only sentences with multiple extracted change events are considered. For every pair of adjacent change events, the sentence is divided into three parts: The words before the first change event (BEFORE), the words between the change events (BETWEEN) and the words after the second change event (AFTER). This is illustrated in Figure 5. If the strings of all the sub-patterns match in the specified location, then an interaction event of the corresponding type is annotated at the text span matched by the first sub-pattern.

As the observant reader may have noticed, correlations do not have any parameter that specifies which argument event takes which argument role. This is because it was observed that the pattern itself is not sufficient to determine which argument takes which role. As a default, the leftmost argument is chosen as the theme and the rightmost as co-theme.

As our corpus consisted of 10 annotated papers, the patterns used in the experiment were manually developed from eight of the papers in the corpus, with the remaining two held out as test data. For every annotated event in the development corpus, a pattern was written that would extract that event. Events that were impossible to extract with the types of patterns explained above, such as interactions between non-adjacent change events were ignored. To improve performance, each pattern was manually evaluated against the development data, and patterns that yielded a high ratio of false positives on the development data were excluded.

One particular group of patterns that created exceptionally many false positives was correlations triggered by single prepositions, such as ”in”, ”with” or ”under”. This group represented a significant proportion of all correlations in the development data, but was nevertheless removed from the final set of patterns, due to the high rate of false negatives. While conducting the experiment described here, it was observed that the problem of false positives can be avoided by using patterns that match against the dependency structure, akin to the patterns used for change events, for this type of correlations.

## 4 Experiments

This section presents the experiments.

| | TEES | | | PMS | | |
|---|---|---|---|---|---|---|
| | Precision | Recall | F-score | Precision | Recall | F-Score |
| Variable | 0.41 | 0.40 | 0.40 | 0.71 | 0.51 | 0.59 |
| Thing | 0.33 | 0.14 | 0.20 | 0.5 | 0.22 | 0.31 |
| Increase | 0.74 | 0.46 | 0.57 | 0.97 | 0.78 | 0.86 |
| Decrease | 0.76 | 0.39 | 0.52 | 0.88 | 0.71 | 0.79 |
| Change | 0.47 | 0.16 | 0.24 | 0.72 | 0.64 | 0.68 |
| Cause | 0.11 | 0.01 | 0.02 | 0.72 | 0.33 | 0.45 |
| Correlate | 0.68 | 0.13 | 0.22 | 1.00 | 0.01 | 0.02 |
| Macro | 0.50 | 0.24 | 0.31 | 0.79 | 0.46 | 0.53 |
| Micro | 0.50 | 0.33 | 0.38 | 0.81 | 0.51 | 0.58 |

**Table 1** Evaluation metrics, trigger categories.

### 4.1 Test data

The annotated corpus consists of 10 full length scientific papers that were selected by a domain expert as represenative for the domain. The methods and materials section of the papers was not annotated, because this section consistently failed to yield any relevant information. All papers were taken from the open access journal PLOS ONE[3], in order to avoid copyright issues when sharing the annotated corpus and any extracted material.

The annotation itself was conducted using the brat rapid annotation tool**??**. All 10 papers were annotated by author EAA, and subsequently reviewed by author EM. All disagreements were solved through discussion. Additional papers have been added to the corpus since the conclusion of the experiment presented here. The corpus is publicly available in its entirety through GitHub[4].

### 4.2 Evaluation methods

TEES was evaluated by five-fold cross-validation, each fold consisting of two papers. As eight papers had been used during development of the pattern matching system, the pattern matching system was evaluated once only, using the two remaining papers as test data.

In the evaluation, an element is matched to an element in the gold standard if the text spans exhibit any degree of overlap, thus abstracting away from exact scoping of text spans. This was done to not penalize TEES for its biomedical-centric approach to extracting multiword expressions. A predicted trigger was counted as a true positive if it matched an element of the same annotation category in the gold standard, and a false positive if it didn't. A trigger in the gold standard was counted as a false negative if it could not be matched to a predicted trigger of the same category.

## 5 Results

Precision, recall and f-score for each category for both systems were calculated and presented in Tables 1 and Table 2.

It should be kept in mind that particularly the results produced provide a low estimate of the performance of a mature information extraction system on the task. The pattern matching system used to provide the benchmark estimates is extremely primitive, and has a large potential for improvement, whereas TEES, being a machine learning based system, likely suffers from lack of training data.

[3] www.plosone.org

[4] https://github.com/OC-NTNU/OCC

| | TEES | | | PMS | | |
|---|---|---|---|---|---|---|
| | Precision | Recall | F-score | Precision | Recall | F-Score |
| Agent | 0.26 | 0.02 | 0.04 | 0.22 | 0.06 | 0.09 |
| Theme | 0.39 | 0.19 | 0.23 | 0.43 | 0.32 | 0.37 |
| Co-theme | 0.32 | 0.06 | 0.10 | 0.00 | 0.00 | 0.00 |
| Macro | 0.32 | 0.09 | 0.12 | 0.22 | 0.13 | 0.15 |
| Micro | 0.36 | 0.14 | 0.18 | 0.36 | 0.26 | 0.30 |

**Table 2** Evaluation metrics, argument categories.

It can be seen from Table 1 that the PMS outperforms TEES on the entity categories (*variable*, *thing*). This seems to confirm that the additional information provided by having extracted change events yields benefits that far outweight the error propagted from the earlier step.

For both systems, performance was higher on the change event categories (*increase*, *decrease* and *change*) than on the entity categories, or in fact, on any other trigger cateogry group. This indicates that the change event categories is the group of categories that is easiest to recognize correctly in this extraction task. The PMS outperformed TEES on all change event categories. It is likely tha the performance of TEES is affected negatively by errors in the earlier step, as the features used for event trigger detection include the number of entities in the sentence and entities within a linear window of the word in question.

For the interaction events, the PMS clearly outperforms the TEES system on the *cause* category. However, the opposite holds true for the *correlation* category. This is most likely due to the fact that the most productive patterns for correlations were removed to avoid false positives.

Analysis of performance on argument categories does not bring any new insights: Performance on the *theme* category is higher for the PMS, due to the increased number of correct change events detected. The PMS scores 0.00 for the *co-theme* category, due to fact that the system at the current stage of development is unable to properly distinguish themes and co-themes in correlations. It is surprising that the performance for the *agent* category is rather similar for the two systems, especially given that the PMS by far outperforms TEES on the cause category. A significant reason for this is likely the fact that the PMS cannot handle causative change events at current stage of development.

Overall, the PMS system generally outperforms TTES system. The results are not perfect due to data limitation for TEES and system primitivity for the PMS, and due to the fact that the two systems take different approaches to information extraction, but the results still show that there could be some merit to altering the traditional information extraction pipeline.

## 6 Discussion and conclusion

It is likely that much of the success of the alternative pipeline is due to the fact that it started by extracting from change events, which appear to be the easiest categories. Evidence found during this step facilitates extraction of the more difficult categories. This can be contrasted with the traditional pipeline system, which started from the quite difficult entity categories, and the error-prone evidence found the that step could not improve performance on the already quite easy events categories. This result may possibly be generalized to the hypothesis that a pipeline information extraction system will attain the best results by starting extraction with the easiest category group.

The hypothesis is supported by the following probability theoretic argument: Information extraction can be formulated finding $argmax_{\mathbf{X}}\ P(X_0, X_1, ..., X_n|Y)$, where $X_0...X_n$ are the extraction item classes (entities, events and relations). Because it is ofen infeasable to finding argmax over the joint probablity distibution, the pipeline approach approximates the global optimum by decomposing the joint distribution using the chain rule of probability $P(X_0, X_1...X_n|Y) = P(X_0|Y)P(X_1|X_0, Y)P(X_2|X_1, X_0, Y)\dots P(X_n|X_{n-1}, ..., X_0, Y)$ and greedily optimizing each step in the chain, thus computing $\hat{X_0} = argmax_{X_0}\ P(X_0|Y)$, $\hat{X_1} = argmax_{X_1}\ P(X_1|\hat{X_0}, Y)$ ... . In this setup, any error in $X_0$ will aversely affect all subsequent steps, as they then will try to optimize based on a suboptimal variable assignment. On the other hand, an error in $X_n$ will only affect the step itself. To minimize the approximation error, one should put the item classes that are least error-prone and least dependent on information from the other item classes first.

Verifying the hypothesis empirically, and exploring other heuristics for determining the optimal pipeline architecture for an extraction task remain tasks for future research. The development of an information extraction system that can lets the user make arbitrary adjustments to the pipeline would make this line of research immediately applicable. A longer term goal is the development of a system that is able to automatically select the optimal pipeline based directly on the data.

Running the extaction experiments has also given insight into the strengths and weaknesses of our annotation scheme, and refining the annotation scheme to sort out unnecessary complexities also remains an important research task. For instance, experiments in reasoning have uncovered that it does not make sense to make a binary distinction between entities that require no semantic interpretation (*variables*) and entities that require full semantic interpretation (*things*), because entities always require semantic interpretation to a certain degree. As an example, consider *growth* in the example sentence "Increasing concentrations of CO2 cause a strong decline in growth...". *Growth* is arguably a quantitative variable, but for this to be useful during reasoning, a semantic interpretation component must disambiguate further what thing in the real world that the growth pertains to. In future versions of the annotation scheme, the category *thing* will therefore be deprecated, and the category *variable* used in all cases. All entities will then be passed to the semantic interpretation module.

THERE IS PLENTY OF FUTURE WORK THAT WE CAN MENTION HERE, AMONG OTHERS ONTOLOGIES, ADDING CO-REF TREATMENT, VARIABLE DISAMBIGUATION, REASONING, LDB ETC. IS IT WORTH MENTIONING, OR JUST NOISE FOR THE READER?

**Competing interests**
The authors declare that they have no competing interests.

**Author's contributions**
Text for this section ...

**Acknowledgements**
Much of the work presented in this article was conducted as a part of the Master's project of author Elias Aamot. Elias Aamot is currently affiliated with Google, Inc., but the work presented here was substantially conducted while affiliated with the Norwegian University of Science and Technology.

**References**

1. Van de Waal, D.B., John, U., Ziveri, P., Reichart, G.-J., Hoins, M., Sluijs, A., Rost, B.: Ocean acidification reduces growth and calcification in a marine dinoflagellate. PLoS ONE **8**(6), 65987 (2013). doi:10.1371/journal.pone.0065987
2. Kim, J.-D., Ohta, T., Tsujii, J.: Corpus annotation for mining biomedical events from literature. BMC Bioinformatics **9**(1), 10 (2008). doi:10.1186/1471-2105-9-10
3. Björne, J., Airola, A., Pahikkala, T., Salakoski, T.: Drug-Drug Interaction Extraction from Biomedical Texts with SVM and RLS Classifiers. In: Proceedings of the 1st Challenge Task on Drug-Drug Interaction Extraction, Huelva, Spain, pp. 35–42 (2011). http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.232.2832
4. Björne, J., Heimonen, J., Ginter, F., Airola, A., Pahikkala, T., Salakoski, T.: Extracting Complex Biological Events with Rich Graph-Based Feature Sets. In: Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task, pp. 10–18. Association for Computational Linguistics, Boulder, Colorado (2009). http://www.aclweb.org/anthology-new//W/W09/W09-1402.bib
5. Björne, J., Salakoski, T.: Generalizing Biomedical Event Extraction. In: Proceedings of BioNLP Shared Task 2011 Workshop, pp. 183–191. Association for Computational Linguistics, Portland, Oregon, USA (2011). http://www.aclweb.org/anthology-new//W/W11/W11-1828.bib
6. de Marneffe, M.-C., Manning, C.D.: The stanford typed dependencies representation. In: Coling 2008: Proceedings of the Workshop on Cross-Framework and Cross-Domain Parser Evaluation. CrossParser '08, pp. 1–8. Association for Computational Linguistics, Stroudsburg, PA, USA (2008). http://dl.acm.org/citation.cfm?id=1608858.1608859
7. Charniak, E., Johnson, M.: Coarse-to-fine n-best parsing and maxent discriminative reranking. In: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics. ACL '05, pp. 173–180. Association for Computational Linguistics, Stroudsburg, PA, USA (2005). doi:10.3115/1219840.1219862. http://dx.doi.org/10.3115/1219840.1219862
8. McClosky, D., Charniak, E.: Self-training for biomedical parsing. In: Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics on Human Language Technologies: Short Papers. HLT-Short '08, pp. 101–104. Association for Computational Linguistics, Stroudsburg, PA, USA (2008). http://dl.acm.org/citation.cfm?id=1557690.1557717
9. Klein, D., Manning, C.D.: Accurate unlexicalized parsing. In: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1. ACL '03, pp. 423–430. Association for Computational Linguistics, Stroudsburg, PA, USA (2003). doi:10.3115/1075096.1075150. http://dx.doi.org/10.3115/1075096.1075150