NTNU – Trondheim
Norwegian University of
Science and Technology

Literature-based knowledge discovery in climate science

Elias Aamot

Master of Science in Informatics
Submission date: July 2014
Supervisor: Pinar Öztürk, IDI
Co-supervisor: Erwin Marsi, IDI

Norwegian University of Science and Technology
Department of Computer and Information Science

Abstract

Climate change caused by anthropogenic activity is one of the biggest challenges of our time. Researchers are striving to understand the effects of global warming on the ecological systems of the oceans, and how these ecological systems influence the global climate, a line of research that is crucial in order to counteract or adapt to the effects of global warming. A major challenge that researchers in this area are facing, is the huge amount of potentially relevant literature, as insights from widely different fields such as biology, chemistry, climatology and oceanography can prove crucial in understanding the effects of global warming on the oceans. To alleviate some of the work load from researchers, information extraction tools can be used to extract relevant information from the scientific literature automatically, and discovery support tools can be developed to assist researchers in their efforts. This master thesis conducts fundamental research into the development of discovery support tools for oceanographic climate science, focusing primarily on the information extraction component.

Acknowledgements

This thesis has been written under the supervision of Pinar Öztürk and Erwin Marsi. I am extremely grateful for the guidance and support I have been given by my supervisors.

This thesis is carried out in the context of the EU Framework 7 project OCEAN-CERTAIN[1]. OCEAN-CERTAIN is a cross-disciplinary research project involving researchers from various disciplines, that aims to discover the impact of multiple stressors on the oceanic food web and the biological pump (see 2.3 for details). The project started in the fall of 2013, and is expected to run over four years. As a part of this project, researchers at IDI are developing a literature-based discovery tool for the oceanographic climate science domain, and this thesis represents my contribution to the project.

Some sections of this thesis have been used in an OCEAN-CERTAIN deliverable [Marsi et al., 2014a] and appeared as a workshop paper [Marsi et al., 2014b], but all material presented in this thesis has been originally written by me.

[1]http://oceancertain.eu/

Contents

Chapter 1

Introduction

The scientific literature is growing so rapidly that scientists are forced to specialize in ever smaller disciplines to keep up with the state-of-the-art. This inevitably leads to the fragmentation of science into scientific communities with very little inter-communication. In this situation, it is easy to imagine that one scientific community may hold some piece of knowledge that is useful for another scientific community, but even when this knowledge is published, the other scientific community does not become aware of the desired knowledge, because it is too far from their specialization for any member of that community to read the paper in question.

A historical example can illustrate this kind of situation (based on an example given in [Gordon et al., 2001]): Darwin postulated the theory of evolution, but he did not have any scientific explanation of how properties are inherited. In the same period, Mendel formulated the first theory of genetics and heredity, but the scientific community failed to show any interest in his theories. These theories were complimentary in the sense that the theory of heredity could strengthen the theory of evolution by explaining how traits are inherited, making itself relevant to the scientific community in the process. However, it seems that no single researcher was aware of both these theories, leading to a lack of interest that caused Mendel's theories to be more or less ignored by the scientific community for three decades.

Don R. Swanson, a librarian, argued that situations such as the one explained above are abundant in modern science due to its fragmentation, and investigated how Information Retrieval techniques could be exploited to help researchers discover complementary knowledge from publicly available scientific papers[Swanson, 1986]. This field of research, which applies Information Retrieval and Artificial Intelligence techniques to assist researchers in discovering knowledge hidden in the scientific literature, has been dubbed *Literature-Based Discovery* (LBD)[1].

[1]Some researchers also refer to the field using the more informative term *Literature-Based Knowledge Discovery* (LBKD), however this thesis follows the convention that is followed by most researchers in the field.

Climate change caused by anthropogenic activity is one of the biggest challenges of our time, prompting increased research into the causes and effects of climate change in order to counteract or adapt to the effects of global warming. NTNU is currently participating in an EU Framework 7 project, OCEAN-CERTAIN[2], that strives do uncover how climate change is affecting the biological systems in ocean, and the oceanic ecosystem's ability to mitigate climate change by absorbing atmospheric CO_2 through photosynthesis (see chapter 2.3 for details). This field of research, oceanographic climate science (OCS) is highly cross-disciplinary, and involves researchers from widely different disciplines, such as biology, chemistry, oceanography and social science. Due to its cross-disciplinary nature, OCS is likely to benefit from a literature-based discovery system to assist in the discovery process, and one of NTNU's tasks in the project is the development of such a system.

1.1 Architecture of a literature-based discovery system

As will be discussed in further detail in the literature survey of literature-based discovery approaches in chapter 2, LBD approaches have varying degrees of sophistication, from simple methods based on lexical statistics to approaches exploiting structured information extracted from the literature by use of information extraction tools. There is a strong correlation between reliance on domain specific resources, such as NLP tools and concept ontologies, and precision of system output.

It is believed that the four year duration of the OCEAN-CERTAIN is sufficient to develop a sophisticated literature-based discovery system for oceanographic climate science, including the required information extraction tools for the domain. The proposed architecture for the information extraction-based LBD system can be conceptualised as a pipeline with the following steps:

Document retrieval: Documents that are potentially relevant to the topic are identified and downloaded. In our case, *document* is taken to mean scientific articles, and documents are relevant if they are about oceanographic climate science or any related domain. Document retrieval involves legal and practical issues, as scientific articles are published by different publishers with widely varying access APIs and policies for text mining. The practicalities of document retrieval are not considered in this thesis.

Information Extraction: Structured information is derived from the literature in the form of *domain rules*, which represent facts about the domain. The present thesis focuses on exploring this step.

Discovery support: The user interacts with the system to explore the extracted information in order to look for new potential hypotheses. Human-

[2]http://oceancertain.eu/what-is-ocean-certain/

computer interaction design is a large part of designing the discovery support component, which is outside the scope of this thesis.

The information extraction step, considered as a black box, takes natural language text as input and produces a corresponding structured representation of the targeted types of information. This task is relatively hard owing to the large distance between the source text and the domain rules extracted, as illustrated by the sentence in (1) and the corresponding desired domain rule in figure (2).

(1) Under eutrophic conditions where autotrophs dominate, nitrogen recycling is less efficient which again favours export of the accumulated organic matter.

(2) $DECREASE(nitrogen\ recycling\ efficiency) \Rightarrow$
$INCREASE(organic\ matter\ export)$

To reduce the difficulty of information extraction, we introduce an intermediary representation, annotated text, which is a mark-up of the concepts and relations of interest that occur in the text. The concepts and relations of interest vary widely between domains and applications, resulting in different *annotation schemes*. Annotation schemes will be described further in chapter 3.1. A possible annotation for the previous example sentence is shown in 1.1 (the first part of the sentence does not contain any annotations, and is therefore omitted).

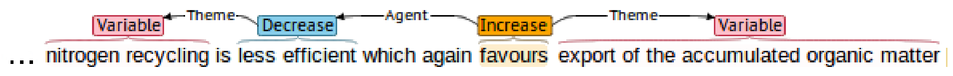
Variable
Theme
Decrease
Agent
Increase
Theme
Variable
... nitrogen recycling is less efficient which again favours export of the accumulated organic matter

Figure 1.1: Example of annotated text

When including annotated text as an intermediary layer, information extraction can be divided into two sub-tasks: Automatic annotation and extraction of domain rules from the annotated text. The present thesis focuses mostly on the task of automatic annotation, but domain rule extraction is also considered.

1.2 Goals and Research Questions

This thesis focuses on the development of the information extraction component for a future literature-based discovery system in oceanographic climate science. Due to the relatively short duration of the thesis compared to the OCEAN-CERTAIN project, the thesis does not focus on development of complete tools, but rather on exploring some principal design decision. Research goals addressed by this thesis are:

1. Identification of which discovery types are targeted in the domain.
2. Development of an annotation scheme which allows the extraction of knowledge that can support the desired discoveries.

3. Creation of an annotated corpus.

4. Creation of tools for automatic annotation.

5. Extraction of domain rules from annotated data.

Research questions that spring out of the goals are:

1. What kinds of information should be extracted from the scientific literature in order to support discovery in the domain?

2. What challenges are present for corpus annotation?

3. Can intelligent tools be developed that improve annotation speed and quality?

4. How do pattern-matching and machine learning approaches differ in the their performance for automatic annotation?

5. What are the expected challenges for reasoning with the domain rules?

1.3 Structure of the thesis

The flow of the work presented in this thesis is shown in figure 1.3. First, an annotation scheme was developed to encompass the type of information that supports discovery in the domain, fulfilling research goals 1 and 2, and answering research question 1 in the process. Then, a corpus was manually annotated according to the developed annotation scheme, completing research goal 3 and discovering the answers to research question 2. The annotated corpus was then used for development/adaptation of two prototype systems for automatic annotation, one based on pattern matching, and one based on machine learning, partially fulfilling research goal 4. The performance of the systems was then compared, yielding insights into research question 4. Research question 3 was answered by evaluating the output from one of the systems from the perspective of an annotator. Finally, domain rules were extracted from the annotated text (both manually and automatically annotated) to provide answers for research question 5, and complying with research goal 5.

The next chapter will provide a survey of the literature-based discovery literature, and provide some required background on information extraction and oceanographic climate science. Chapter 3 will present the annotation scheme developed for oceanographic climate science and describe the annotation process. Chapters 4 and 5 will present the two different approaches to automatic annotation. Chapter 6 presents the investigation into extraction of domain knowledge from annotated text. Finally, chapter 7 summarises the research findings and provides pointers for future research.

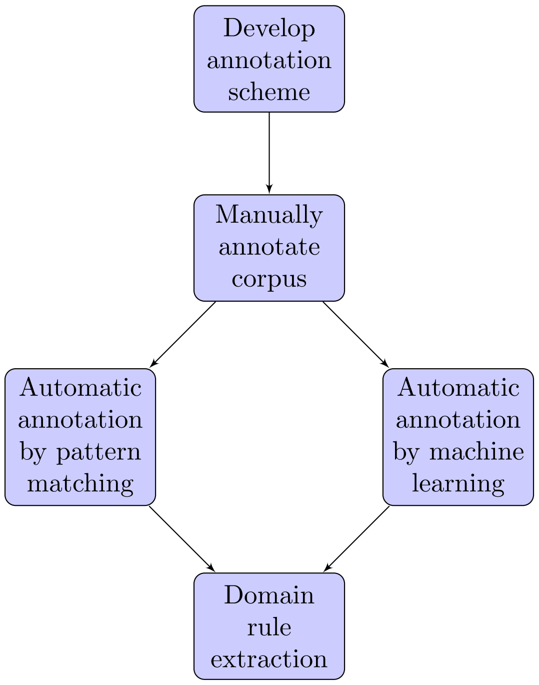
Develop annotation scheme
Manually annotate corpus
Automatic annotation by pattern matching
Automatic annotation by machine learning
Domain rule extraction

Figure 1.2: Workflow diagram of the work presented in this thesis.

Chapter 2

Background

This chapter presents the background knowledge required to understand the experiments conducted in this thesis and their motivations. The first and largest section represents a literature survey of the state-of-the-art and historical developments of literature-based discovery. The chapter closes with two sections introducing information extraction and oceanographic climate science, to equip the reader with the required background to understand the rest of the thesis.

2.1 Literature-Based Discovery

The field of Literature-Based Discovery (LBD) arose when Swanson [1986] observed that if a literature L_1 asserted the fact $a \rightarrow b$, and a disjoint literature L_2 asserted the fact $b \rightarrow c$, then the concept denoted by b could function as a bridge between L_1 and L_2, leading to the discovery of the hypothesis $a \rightarrow c$[1]. As a proof that the method words, he showed that one biomedical paper had concluded that eating fish oils reduces blood viscosity ($fish$ *oil* $\rightarrow$ *blood viscosity*), and that another paper showed that patients with Raynaud's disease tend to exhibit high blood viscosity (*blood viscosity* $\rightarrow$ *Raynaud*). These two facts led Swanson to form the hypothesis that fish oils can be used in the treatment of Raynaud's disease ($fish$ *oil* $\rightarrow$ *Raynaud*). This hypothesis was subsequently confirmed experimentally [Digiacomo et al., 1989].

The steps in this process are not logically sound, as the two relations are not necessarily transitive. Nevertheless, Swanson's example clearly demonstrated that the procedure is able to produce interesting results. The general approach of deriving potential facts by combining facts from disjoint literatures by means

[1]A note on terminology: In the LBD literature, capital letters are normally used for the A, B and C concepts. In this thesis, minuscules will be used to represent individual concepts, while capital letters represent sets of concepts.

Also, some authors use A to denote the goal concept, and C for the starting concept. This report uses the most commonly used terminology, in which a denotes the starting concept, and c denotes the goal concept.

of a common concept has been dubbed *Swanson linking*, and is also referred to as the *ABC model*.

Swanson and Smalheiser [1997] explain that the discovery of the ABC structure and the fish oil-Raynaud's disease connection happened accidentally. This discovery led Swanson to conduct literature searches aided by existing information retrieval tools to search for more undiscovered public knowledge using the ABC model, resulting in the discovery of eleven connections between migraine and magnesium [Swanson, 1988]. As the discovery process was extremely time consuming, requiring the researcher to read hundreds of papers, Swanson later developed a computational tool, Arrowsmith, to streamline the discovery process.

There are two modes of discovery in literature-based discovery: *Open discovery* and *closed discovery*. In open discovery, the researcher only knows the starting concept a, and is interested in uncovering undiscovered public knowledge related to a. A researcher who looks for consequences of ocean acidification might conduct an open discovery search with $a = ocean\ acidification$. In closed-discovery, the researcher knows both the starting concept a and the goal concept c, and is interested in finding concepts B that yield an explanation of the relationship between the two terms. A researcher who hypothesizes that ocean acidification might cause a reduction in phytoplankton population may try to discover the causality chain by conducting a closed discovery search with $a = ocean\ acidification, c =$ *phytoplankton population*. The standard discovery process in LBD is to first conduct an open discovery search for interesting related concepts, and subsequently explore the most interesting concepts more closely with a closed discovery search.

2.1.1 Swanson and Smalheiser: Arrowsmith

The Arrowsmith system[Swanson and Smalheiser, 1997] resulted from Swanson's efforts to create a literature search tool to assist researchers in conducting literature-based discovery searches. The approach taken by the Arrowsmith system is based on the observation that related concepts tend to exhibit higher degrees of co-occurrence in the literature than unrelated concepts, and that degree of co-occurrence therefore can be used as a statistical heuristic for discovering relations. This idea has been adopted by most LBD systems.

The literature for a term q, written $L(q)$, is defined as the set of titles of the papers that is returned by conducting a Medline search for all papers that contain q. Medline is a commonly used indexed literature database for biomedical and life sciences[2]. Arrowsmith uses paper titles only, because the authors believe them to contain the most compact knowledge.

The system conducts an open discovery search as follows: $L(a)$ is retrieved from Medline. The potential B concepts are extracted as the list of unique words in $L(a)$, after a stop list of approximately 5000 words has been applied. The B-term set is further pruned by removing all the words that have lesser relative

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

frequency in $L(a)$ than in Medline, thus keeping only the candidates that exhibit a higher degree of co-occurrence with a than expected. The potential B terms are subsequently presented to the user, who can then remove words that are thought to be unsuitable. For each $b_i \in B$, $L(b_i)$ is retrieved and a set C_i is generated, subject to the same stopword and frequency restrictions as before. The union of the C_i sets, C, is called the *discovery candidates*. The remaining C terms are finally ranked according to the number of b terms that connect them to the a term, and presented to the user. The ranking metric is based on the assumption that c terms that are connected to a through many b terms are more likely to be biologically relevant for a.

As evaluation, the paper showed that Arrowsmith was able to reproduce Swanson's earlier discoveries. Arrowsmith has been public available since 2001, and has had on average approximately unique 1200 users per month[Smalheiser et al., 2008]. However, no major discoveries have been attributed to Arrowsmith, to the best of my knowledge.

2.1.2 Gordon and Lindsey: Information retrieval-based methods

Gordon and Lindsay [1996] [Lindsay and Gordon, 1999] developed their own LBD system at the same time as Arrowsmith was being developed. Their system differed from Arrowsmith in several ways:

- While Arrowsmith was word-based, their system used n-grams (n=1,2,3) as the unit of analysis. A stop list was applied by removing all n-grams in which at least one word occurred in the stop list. N-grams let the system discover relations between multi-word terms, such as *fish oil* and *Raynaud's Disesase*, which are extremely common. On the other hand, there are many more n-grams than single words, leading to increased computational load and more difficult task of filtering out irrelevant discovery candidates.

- Their system used entire Medline records, comprising of keywords, abstracts and titles, whereas Arrowsmith only used paper titles, thus utilizing more of the available textual data.

- Their system employed a collection of common information retrieval metrics such as *term frequency*, *document frequency* and *tf*idf* to discover relations between terms, where Arrowsmith only used relative frequencies. In their system, a human interacts closely with the system and selects terms from the lists suggested by the various metrics.

Evaluation of the system was conducted by using the system to successfully replicate Swanson's discoveries, with significant human guidance.

The lexical statistical approach is so generic that it lends itself directly to application in any domain. In a later paper, Gordon et al. [2001] employ this approach to conduct LBD searches directly on the World Wide Web, searching for application areas for genetic algorithms, yielding several potential results, but no structured evaluation was conducted.

2.1.3 Gordon and Dumais: Latent semantic indexing

Gordon and Dumais [1998] propose exploiting the ability of certain vector-based semantic models such as Latent semantic indexing (LSI) to discover implicit relationships between terms for LBD. LSI is a statistical semantic model that uses mathematical techniques to uncover similarities between terms based on a collection of textual data.

The authors first train an LSI semantic model on $L(a)$, and let the user choose as b one of the terms most similar to a according to the semantic model. A new semantic model is then built from $L(b)$, and discovery candidates are ranked according to their similarity to a in the $L(b)$ model. Their experiments showed that the resulting B and C term candidate sets closely resemble the lists produced by the information retrieval inspired lexical statistics.

In another experiment, they built a semantic model from a random sample over all Medline papers, and looked directly for c terms in the semantic model by considering the terms most similar to a. This "zoomed-out" approach produced different results than the previous, Swanson linking-inspired, approach, which the authors claimed meant that the two methods are complementary and could therefore be used in parallel, but no in-depth evaluation was conducted on the quality of the results.

2.1.4 Cory: LBD in the humanities

While almost all LBD research has been conducted in the biomedical sciences, Cory [1997] applied LBD techniques to the humanities by using a manual search procedure. Using this procedure, he discovered an analogy between the American poet Robert Frost and the ancient Greek philosopher Carneades, by means of Swanson linking through the American philosopher William James.

The author initially expressed doubts whether the experiment would give results, doubting that the more creative, and therefore less precise, word usage in the humanities would lend itself to literature-based discovery as easily as the specialized terminology of the biomedical sciences. However, this did not cause any problems, as proper names exhibited the desired precision, and the author chose to use only proper names as terms.

2.1.5 Wren et al.: Ranking metrics

Wren et al. [2004] point out that the structure of term co-occurrence relationships in scientific literature is such that most terms are connected to any other term within few steps. This *small world phenomenon* has two implications for LBD: Firstly, using only one intermediary (B) step is sufficient to find almost all interesting c terms, so there is no need to perform Swanson linking through additional intermediary concepts (e.g. ABCD). Secondly, research focus should be shifted away from discovering discovery candidates to ranking them, because discovering potential candidates is trivial.

The paper proposes ranking implicit relationships by comparing the number of observed indirect connections between a and c (i.e. the number of $b$terms connceting $a$to c) to the number of expected connections in a random network model, given the relative promiscuity of the intermediary terms. In other words, the discovery candidate $a \rightarrow c$ will be ranked higher if connected by relatively rare terms, such as the names of specific genes, than if connected by common terms such as "experiment" or "disease". The system developed uncovered an implicit connection between the drug Chlorpromazine (CPZ) and the disease Cardiac hypertrophy, and a lab experiment was conducted, confirming the claim.

In another paper, Wren [2004] emphasizes the importance of using a statistically sound ranking mechanism, such as "chi-square tests, log-likelihood ratios, z-scores or t-scores", because co-occurrence measures that do not take the promiscuity of the terms into account, i.e. by simply counting the number of co-occurrences, bias towards more general, and therefore less interesting, relationships. The paper further proposes an extension to the mutual information measure (MIM) as a ranking measure. No real evaluation of the method was performed. The author tried to replicate Swanson's discoveries, but the system was not able to do so. However, that does not mean that the results produced by the system are wrong, they just differ from Swanson's discovery.

2.1.6 Concept-based approaches

Several researchers advocate using domain specific concepts taken from an ontology or controlled vocabularies instead of n-gram tokens. Standardized concepts provides three benefits over n-grams: Firstly, synonyms and spelling variants are mapped to the same semantic concept. Secondly, using concepts allows for ranking and filtering according to semantic categories, for instance restricting potential c terms to drugs or treatments when looking for a cure for a disease. Finally, it becomes easier to constrain the search space by removing spurious or irrelevant n-grams at an early stage, as they don't map to any concept in the domain. On the other hand, concept extraction from raw text is a non-trivial operation, which requires manual effort or automatic entity recognition tools (see chapter 2.2.1 for details). Also, this requires an ontology or controlled vocabulary resource for the domain to be applicable.

Weeber et al. [2001] show in an experiment that the number of concepts is significantly lower than the number of n-grams, even after stop lists are applied (8,362 n-grams vs. 5,998 concepts). Their system, DAD (Disease-Adverse reaction-Drug), uses MetaMap, a commonly used Entity Recognition tool for the biomedical domain that maps natural language text to entities in the Unified Medical Language System (UMLS), a standardized vocabulary for biomedical literature (see 2.2.3for a description of MetaMap). DAD allows the user to specify which semantic categories to consider, by for instance only allowing concepts of the type *pharmacological substance* as c concepts, further reducing the number of search paths.

Their approach was able to replicate both Swanson's *Raynaud's-fish oil* and *migraine-magnesium* discoveries, but it was discovered that MetaMap maps both *mg* (milligram) and *Mg* (magnesium) to the concept *magnesium*, giving optimistic results for the migraine-magnesium experiment. This is but one example showing that a major concern with employing NLP tools in an LBD system is that system performance becomes closely tied to the performance of the tools it employs.

Pratt and Yetisgen-Yildiz [2003] developed a system, LitLinker, which originally also used MetaMap, but they later found it too computationally expensive for practical use [Yetisgen-Yildiz and Pratt, 2006]. Instead, they decided to use MeSH terms. MeSH is a controlled vocabulary used for indexing biomedical papers, used to index all Pubmed papers manually. MeSH terms can be queried directly from Pubmed with the standard API.

In a preprocessing step, LitLinker counts the number of occurrences of every MeSH term in the literature of every other MeSH term. From these counts, the mean and standard deviation of the number of occurrences of a MeSH term across the literatures is computed. In the discovery process, a MeSH term a is considered to be related to another MeSH term b, if $a$occurs more frequently in $L(b)$ than statistically expected, measured as using the z-score.

Yetisgen-Yildiz and Pratt identified three classes of uninteresting links and terms that should be pruned automatically by system: (1) too broad terms (giving the examples *medicine*, *disease* and *human*), (2) too closely related terms (giving the example *migraine* and *headache*), and (3) semantically nonsensical connections. The first class is handled by removing any concept if it is strictly more specific in the MeSH ontology hierarchy than any included term. The second class is handled by pruning all links between terms that are closely related (grandparents, parents, siblings and children) in the ontology. The third class is handled by letting the user specify which semantic classes of concepts are allowed to link.

2.1.7 Lee et al.: Context Similarity

Lee et al. [2011] notes that LBD systems tend generate a lot of spurious discovery candidates, creating a lot of manual work for the user. To amend this situation, they propose a different kind of candidate discovery mechanism based on similarity of contexts, in which two facts $a \rightarrow b$ and $b \rightarrow c$ can only be combined if their contexts are sufficiently similar.

Their system first extract concepts taken from UMLS, the Comparative Toxicogenomics Database (CTD) and Pharmacogenetics Knowledge Base (ParmGKB) by means of a Hidden Markov Model-based entity recognition system (see 2.2.2 for more details). In their system, a liberal heuristic postulates a relation between two concepts if they co-occur at least once in any sentence.

The system then proceeds to build context vectors for every relation type (a relation type being defined as a pair (a, b) of concepts). A context vector is a vector of integers, where every element corresponds to one concept type. The context vector is filled out by, for every relation type, counting the number of occurrences of each concept in the papers where the relation occurs. Context similarity can then be measured by a standard measure of vector similarity, such as the Cosine Similarity. Discovery candidates are ranked according to the number of b concepts connecting them to a where the connecting relations have a context similarity measure over a predefined threshold.

Evaluation was conducted by seeing whether the 10 highest ranked $a \to c$ relations from the system were meaningful, which was interpreted to mean "were known in the literature". 7 of 10 relation were found in the literature, as opposed to 6 of 10 that were extracted using a benchmark co-occurrence frequency model. This evaluation procedure can at best be said to be strange, seeing as how the point of LBD is to discover relations *not* found directly in the literature.

2.1.8 Yetisgen-Yildiz and Pratt: Evaluation Efforts

LBD papers tend to evaluate performance by replicating Swanson's discoveries. Yetisgen-Yildiz and Pratt [2009] identify four aspects in which this evaluation methodology is insufficient:

- The evaluation should consider the entire C term set. An evaluation based solely on the existence or lack thereof of a single term in the C term set is necessarily incomplete. A realistic evaluation should consider all discovery terms, spurious as well as correct.

- The evaluation methodology should be based on multiple experiments. A majority of researchers only tried to recreate one or both of Swanson's discoveries, leading to statistically unreliable results. For an evaluation to be representative of system performance, it must be repeatable for different starting terms.

- The evaluation methodology should be independent of any prior knowledge. When replicating a known discovery, it is easy to tune the system in a way that gives the desired results for the given problem, without improving the overall performance of the system. In other words, there is an overlap between the training data and test data, leading to a biased evaluation. This is especially critical in systems where the human user is heavily involved in the discovery process.

- The evaluation methodology should enable comparison of different systems. A methodology that only evaluates systems according to a binary classification (able or unable to replicate a discovery) cannot be used to quantitatively compare different approaches.

The authors proceed to propose an evaluation methodology that satisfies the aforementioned desiderata: For a test keyword t, they divide $L(t)$ into two sets, the *pre-cut-off set* which only consists of papers published before a given date, and the *post-cut-off set* which consists of all the papers published after the cut-off date. They then conduct the discovery process on the pre-cut-off set, and calculate standard Information Retrieval (IR) evaluation metrics by comparing the results to a gold standard based on the post-cut-off set. The gold standard is created as all terms that occur in $L(t)$ in the post-cut-off set, but do not occur in the pre-cut-off set, based on the assumption that if a term only co-occurs with $t$after the cut-off date, then it represents a discovery that was made after the cut-off date.

The basic IR evaluation metrics used are precision and recall, defined as

$$P_i = \frac{|T_i \cap G_i|}{|T_i|}$$

$$R_i = \frac{|T_i \cap G_i|}{|G_i|}$$

where T_i is the set of target terms generated by the system given starting term i, and G_i is the set of terms in the gold standard created for starting term i. Several evaluation metrics can be derived from precision and recall, such as *11-point average interpolated precision curves*, where the precision values are calculated for each of 11 recall levels (0 to 1 with increments of 0.1).

Using their evaluation methodology, Yetisgen-Yildiz and Pratt performed the first quantitative evaluation of candidate ranking and relation generation algorithms in LBD. The candidate ranking metrics that were evaluated were Linking Term Count (LTC), i.e. the number of b terms connecting a and c, Average Minimum Weight (AMW), i.e. the average over the minimum of the weights of $a \to b_i$ and $b_i \to c$ for every b_i, and Literature Cohesiveness (COH), a measure developed by Swanson but not widely adopted by other researchers. Experiments showed that LTC gave better precision at all levels of recall. The relation generation techniques that were considered were association rules, tf-idf, z-score and MIM. The experiment showed that association rules give the best precision score (8.8%) but the worst recall score (53.76%), while tf-idf gave the best recall (88.0%) but a rather low precision (2.29%).

The evaluation effort was an important contribution to the LBD field, but the field could benefit from more quantitative evaluation. First of all, all candidate ranking/generation techniques and ranking metrics were tested with only one value of the parameters (for instance the cut-off thresholds for tf-idf and z-score). Comparing the performance of different settings for the parameters would yield a better understanding of each of the metrics, and could lead to results completely different than those reported. Secondly, only a small subset of possible relation generation/ranking techniques and discovery candidate ranking metrics were tested.

The authors identified two challenges with the proposed evaluation procedure:

Selecting the cut-off date Because the gold standard should contain all the possible discoveries from a starting term, it is important that the post-cut-off period spans a sufficient amount of time for actual discoveries to have time to emerge. On the other hand, it is important to place the cut-off date late enough for the pre-cut-off set to contain enough material for realistic knowledge discovery.

Creating the gold standard Determining the gold standard discoveries from the post-cut-off set is non-trivial. Yetisgen-Yildiz and Pratt use co-occurrence, and define a new discovery as any term that occurs in the post-cut-off set, and not in the pre-cut-off set, but this is clearly just an approximation, as a term can occur in the post-cut-off set without corresponding to a discovery.

A critique that can be raised against the evaluation procedure is that the validity of a gold standard based on the post-cut-off set is questionable for several reasons: Kostoff [2007] pointed out that it is very difficult to verify that a discovery has not been made before the cut-off date. Also, as the mechanism for automatically creating the gold standard closely resembles the discovery mechanism, the sample is biased, and the evaluation is therefore bound to give optimistic estimates. Another question is whether quantitative measures reflect the usefulness of the LBD system: When at all is said and done, the usefulness of a LBD system equates to its ability to support users in discovering knowledge.

2.1.9 Hristovski et al.: Bitola

Hristovski et al. [2001] developed a system called Bitola[3] that discovered *association rules* between MeSH terms. Association rules mining is a common data mining method for discovering relations between variables in a database. Association rules are traditionally used for market basket analysis, in which rules of the type $\{pizza, steak\} \rightarrow \{coca\ cola\}$ are inferred, stating that if somebody buys pizza and steak, he/she is likely to buy coca cola as well. In Bitola's discovery step, basic associations are first mined from the co-occurrence patterns of MeSH terms. Subsequently, indirect associations $a \rightarrow c$ are inferred by combining association rules on the form $a \rightarrow b_i$ and $b_i \rightarrow c$, and ranked according to the sum of strengths of the connecting association rules.

In later papers [Hristovski et al., 2006, 2008] the authors point out some problems with the co-occurrence based LBD paradigm: Firstly, no explicit explanation of the relation between the a and c terms is given. Secondly, a large number of spurious relations are discovered, as demonstrated by the low precision values witnessed in the evaluation paper (chapter 2.1.8). Both aspects increase the time needed to examine the output of the system by the human user. The authors suggest that employing information extraction techniques to extract explicit relations from the papers can improve performance on both points.

The resulting, augmented, version of Bitola mainly uses the relation extraction tool *SemRep* to extract relations from the downloaded Pubmed abstracts (see 2.2.4 for details about SemRep). They also use another tool, *BioMedLee*, because the two tools extract different kinds of relations that can be useful to the system.

The authors furthermore introduce the notion of a *Discovery Pattern*, which is a collection of relations that is believed to give rise to an interesting discovery

[3]`http://ibmi3.mf.uni-lj.si/bitola/`

in the domain. One discovery pattern, *maybe_treats* can informally be stated as: If a disease leads to a biological change, and a drug leads to the opposite change, then the drug may be able to treat the disease.

The integration between Bitola and the information extraction components presented in the system is rather crude; for a given query term, Bitola uses the co-occurrence based model to output a set of related terms and the set of papers connecting each related term to the query term. The connecting papers must then be manually input into the information extraction components to extract the relation between the query term and the related terms. Following a discovery pattern requires extracting relations between several concepts until a chain of the correct relations has been found. The possibility to integrate Bitola and the information extraction tools more tightly was raised as possible future work, but the authors state that they believe that it is important to have a co-occurrence component to constrain which papers are sent to the Information Extraction components, as the computational work associated with information extraction is significant.

Ahlers et al. [2007] and Hristovski et al. [2010] continue this line of research, introducing new Discovery Patterns. Both papers simplify the use of tools, by relying on only SemRep to produce the relations. Also, these approaches do not rely on an underlying co-occurrence based component, but rather extract relations from all papers on a given subject in Pubmed.

Evaluation of Bitola was limited to a replication of Swanson's first discovery, and a proof-of-concept search that discovered a hypothetical connection between Huntington's Disease and Insulin. No clinical trials have been conducted to support the claim, however.

2.1.10 Wilkowski et al.: Discovery Browsing

Wilkowski et al. [2011] introduce the notion *Discovery Browsing*, where the user interactively and incrementally builds a graph over the concepts of interest. In the type of graph used in their system, concepts are represented as nodes, and relations as edges. The concepts and relations are queried from a pre-compiled database of relations that have been extracted by SemRep from a large sample of Pubmed (7 million papers).

The initial graph is created by querying the database for all relations involving the *a* concept. The user can then select which nodes to expand further, gradually growing the graph in the most interesting areas. The system presents the potential concepts for expansion ranked by *degree centrality*, i.e. the number of other nodes it is connected to. This graph growing process can be regarded as corresponding to the open discovery task. After graph construction, the user selects the start and end concepts for a closed discovery task, and the system presents the various paths connecting the concepts ranked by their summed degree centrality.

The system was applied tentatively to discover concepts related to depression. The authors claimed that the output of the system provided both well-researched and under-researched connections, and that the system therefore could be used to highlight connections that could benefit from further research. However, no quantitative evidence backed up the claim. Other authors have criticised the approach because it can eliminate outliers, as there is a positive correlation between how well-known the connection is, and the degree centrality score[Cameron et al., 2013]. The authors themselves pointed out that inverse degree centrality can be used if the focus is to extract unexpected connections.

Cairelli et al. [2013] applied Discovery Browsing to discover an undiscovered explanation for why obesity seems beneficial for patients in critical care, even though it has detrimental effect on the health of ordinary patients. Even though the literature supports the explanation, no clinical study has been undertaken to support the claim.

2.1.11 Cohen et al.: Predication-based semantic indexing

Cohen et al. [2012a] propose a hyperdimensional computing technique they call *predication-based semantic indexing* (PSI) for efficient representation and reasoning in the concept-relation space. In PSI, concepts and relations are represented as high-dimensional vectors, where the content of a concept's vector is determined by combining the vectors of all the relations it occurs in, and all the concepts it is related to, weighted by the frequency of the relation. The system uses SemRep to extract relations from a sample of 8,182,882 Medline records as input to the training process. Inference in this hyperdimensional space can be performed by ordinary vector operations. The paper shows how PSI enables analogical reasoning along the lines of "x is to what as y is to z?" without explicitly traversing the intermediary relation paths between y and z, leading to efficient inference.

The system could originally only infer analogies along a single one of the pathways connecting two concepts x and y. In a later paper Cohen et al. [2012b] expanded the PSI to allow for analogies along multiple pathways, by introducing a vector operation simulating quantum superposition, efficiently reasoning over the entire subgraph connecting x and y. The paper claims that because real world concepts tend to interact through several pathways, literature-based discovery should strive to be able to reason following a similar pattern.

2.1.12 Ijaz et al.: MKEM

MKEM (Multi-level Knowledge Emergence Model for Mining Undiscovered Public Knowledge) [Ijaz et al., 2009] differs from the other information extraction-based LBD systems described in this section by not employing external information extraction tools for extraction of semantic predictions. Instead, the developers included a training step, where the system is presented with a collection of sentences ($\sim$150 were used in the paper) where the user annotates the concept and relation words of interest manually. From this training data, a set of extraction rules are derived by taking the paths in the parse trees connecting the concepts and relation words. In other words, the system bootstraps its own information extraction tool from a small sample of carefully chosen sentences.

An evaluation was conducted by sampling 98 sentences and inspecting manually the output from the information extraction module, revealing a Precision and Recall of 75% and 56%, respectively.

2.1.13 Summary

In the general Swanson linking paradigm, open discovery is conducted by extracting all relations $a \rightarrow b_i$ from $L(a)$. For every b_i, all relations $b_i \rightarrow c_j$ are then extracted from $L(b_i)$. The set of all $a \rightarrow b_i \rightarrow c_j$ relations is then are presented to the user as potential discoveries, sorted according to some ranking metric. Most of the systems presented in the chapter prior to Bitola adhere to this scheme, with various degrees of human interaction throughout the process.

In most pre-Bitola systems, various co-occurrence metrics were used to mine for relations between concepts. A relation $x \rightarrow y$ is postulated if x and y exhibit a high degree of co-occurrence in $L(x)$, either in terms of absolute frequency of co-occurrence, or in terms of unlikelihood given the statistical promiscuity of the two concepts. While a few systems use the sentence as the domain for counting co-occurrences, most systems count co-occurrences across entire abstracts.

The most widely adopted ranking metric is Linking Term Count (LTC), i.e. the number of b terms connecting a to c_j for any j. LTC has also been shown to yield high performance in a quantitative evaluation of ranking metrics (see 2.1.8).

Many LBD systems remove from C all terms that are already known to be in a relation with a, so that the output only consists of new discoveries. This is normally done by removing any (a, c) pairs that exhibit higher degrees of co-occurrence than a predefined threshold (normally 1 co-occurrence) in $L(a)$.

A significant innovation in literature-based discovery was the adoption of standardised concepts from knowledge resources, which is believed to have improved the precision of the predications made by the system, thus lessening the workload for the human user. However, no empirical evaluation has thoroughly verified the claim that system performance has indeed improved.

A recent trend in LBD is the utilization of information extraction techniques to extract relations from text, rather than using statistical measures to extract them. This is believed to further increase the performance of the system, but performance will be limited by the performance of the IE tools employed. The concept-relation model of these systems also allow for more complex discovery types than the simple ABC model, employing instead discovery patterns, discovery browsing or reasoning by analogy.

Smalheiser [2012] critiques the usage of relation extraction in LBD and claims that while reasoning over explicit relations may lead to so-called *incremental discoveries*, that is, discoveries that lie close to the existing knowledge and therefore are less interesting, they are not able to lead to any *radical discoveries*, that is discoveries that seem unlikely at time of discovery. He also claims that human discoveries, both incremental and radical, tend to be on a higher level, using analogies and abstract similarities rather than explicit relations, and that the benefit from using relation extraction therefore is minimal[4].

Literature-based discovery systems can be divided into three generations, according to the types of knowledge resources they use: First generation systems use only information directly extracted from text, concepts are words or n-grams, and co-occurrence metrics are used for relation detection. Second generation systems use standardised concepts, but still rely on co-occurrence metrics for relation detection. Third generation systems use information extraction tools to extract concepts and relations from text, and reason in various ways over the concept-relation space.

As mentioned earlier, it is believed that newer generation systems perform better than older generation systems, and although no standardised evaluation procedure has been developed to verify this claim, there are indications that it is correct. However, the newer generation systems are dependent on largely domain-specific tools and resources, making it hard to port these approaches to domains without the required resources and tools. Nevertheless, the additional precision in output afforded by third generation system is highly valuable, and the this thesis bases its approach on the hypothesis that the quality improvement is worth the additional work of developing information extraction tools for the domain.

2.1.14 Research Gaps

Somewhat surprisingly, few LBD systems use full paper texts. Schuemie et al. [2004] show that 30-40% of all information contained in a section is new to that section, meaning that significant amounts of knowledge is lost when only looking at abstracts and index terms of a paper. The need for full text data is also pointed out by Cameron et al. [2013]. The reason for not using full text seems to be that paper abstracts and index terms are available in xml format through the Pubmed API, while full paper texts require accessing rights and are normally stored as pdf.

Literature-based discovery lacks a tradition for systematic evaluation. It is admittedly difficult to evaluate discoveries, as they are per definition unknown, but there are possible techniques, as discussed in 2.1.8. The field would benefit from a shared task that could be used to compare systems even across generations.

Literature-based discovery research has been focused almost exclusively on the biomedical domain. Very little work has been conducted into making the techniques applicable across domains, and the exact relationship between quality of domain resources and tools and usability of the resulting literature-based discovery system is not known.

[4]Smalheiser's critique also extends to many of the widely employed co-occurrence based methods. The argument is that research should focus on developing methods that rank interesting, rather than frequent, relations highly.

2.2 Information Extraction

This sub-chapter gives an introduction to Information Extraction (IE), and gives an overview of some of the challenges encountered when developing an information extraction system. It is not intended to be a comprehensive survey of the field, but is included to give the reader sufficient background to understand this thesis. Two information extraction systems, MetaMap and SemRep, are also presented, due to their importance for literature-based discovery.

Information extraction is a task of Natural Language Processing (NLP) that is traditionally defined as extracting structured data from natural language text, which is said to be "unstructured". Information Extraction can be divided into a set of sub-tasks of increasing complexity, extracting increasingly complex information structures.

2.2.1 Information Extraction Tasks

One subtask of Information Extraction is (Named) Entity Recognition (ER), which aims to extract entity mentions from text, and classify them into predefined categories. The categories of interest depend on the domain. In a business/financial news type domain, categories such as *Person*, *Organization* and *Location* are used, whereas biomedical entity recognition normally focuses on categories from the UMLS. ER tasks can be of varying granularity, from simple binary classification of entities and non-entities, to mapping to concepts in a multi-level ontology.

Relation Extraction (RE) is another sub-task of information extraction that aims to extract the relations that hold between Entities in the text. Most relation extraction systems extract binary relations between pairs of Entities in the form of *(subject, predicate, object)* triples. Relation extraction systems normally extract relations from a predefined set of relations developed for the specific domain or task. In the business/financial news type domain, possible relations include *WorksFor(Person, Organization)* and *LocatedAt(Organization, Location)*.

Event Extraction (EE) can be defined as the task of extracting information in structures more complex than relation triples. Event extraction can for instance involve extracting temporal or spacial information about the extracted relations, or involve relations with more complex argument structures, such as n-ary relations or relations that take other relations as arguments. In event extraction, information is normally extracted as events with a number of arguments, where an argument belongs to a specific type, such as *Agent* or *Theme*[5] and is itself an entity or another event. For sentence (1), the information in Figure 2.1 can be extracted.

[5]The terms used for argument types are normally taken from the linguistic literature on thematic roles. The agent argument is normally the argument that undertakes an action, and the theme is normally the argument that is affected by the event.

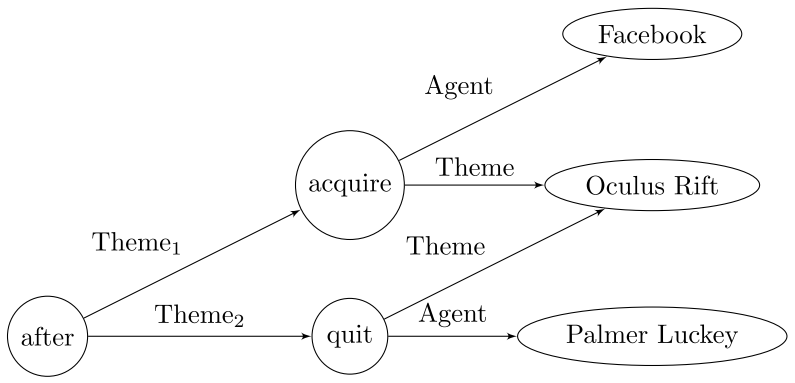

Figure 2.1: Events extracted, represented as a graph. Events are represented as circles.

(1) Following the acquisition of Oculus Rift by Facebook, founder Palmer Luckey left the company.

In the information extraction field, research was originally focused on the simplest tasks, but as performance reached levels sufficient for application, some researchers shifted their attention towards more complex tasks. Therefore, while a lot of research is still being conducted on improving entity recognition and relation extraction, event extraction has been given more attention in the later years. For domains with a rich information extraction tradition, such as the biomedical domain, development of applications has followed an incremental approach, where event extraction systems are only being developed now that reasonably well-performing entity recognition and relation extraction systems exist.

2.2.2 Information Extraction Techniques

Information extraction systems can be roughly divided into two types: Knowledge-based systems that depend on hand-written rules and procedures, and data-driven systems based on machine learning. Knowledge-based systems normally consist of hand-written patterns that the system tries to match against some representation of the natural language text (surface form, part-of-speech sequence or phrase/dependency structure) and/or heuristic rules that are used to rank candidates. Knowledge-based systems tend to yield high precision but low recall. Their development requires significant human effort in developing and adjusting the rules. There also exist hybrid tools that combine data-driven and knowledge-based system in various ways.

Data-driven systems use supervised machine learning techniques to learn how to perform relation extraction. These systems require significant amounts of annotated data to learn from, and the development of such corpora requires significant human effort, but development can be sped up with automated tools. Data-driven systems tend to yield better recall but lower precision than knowledge-based systems. Due to the cost of developing annotated training data, there has been a recent trend towards unsupervised or weakly supervised techniques, where little or no annotated training data is required, but these systems tend to perform worse than supervised systems.

Data-driven systems normally treat entity recognition as a sequence labelling task, where every word is labelled as either not belonging to an entity, as signalling an entity of a certain class or as belonging to the same entity as the previous word (for multi-word entities). Sequence models such as Hidden Markov Models (HMMs) and Conditional Random Fields (CRFs) are popular choices of machine learning algorithms, as are Support Vector Machines (SVMs). Commonly used features include part-of-speech, words or n-grams present within a context window, shape features such as capitalization, case alternations, punctuation, chunking information, and presence of the word in a *gazetteer*. A gazetteer is a list of entries for a given category, such as geographical names, political entities and name lists.

In data-driven systems, relation extraction is normally handled by treating each ordered pair of entities in a sentence (or a larger span if intra-sentential relations are allowed) as a separate classification task. For each ordered pair of entities, the system classifies the relation as negative or as one of the predefined relation types. While any machine learning algorithm can be used, SVMs are most common for this task.

Event Extraction can be conducted similarly to relation extraction, but in this case, the entity recognition stage also extracts triggers for events. During the second stage, classification is performed between pairs of events and pairs of events and entities, and classification determines the argument type that holds between the pair, rather than the relation type.

Commonly used features for relation/argument detection include the types of the entities, the dependency path or word, lemma or part-of-speech chain that connects them, and the words, lemmas or parts-of-speech of the entities themselves.

Most information extraction systems that perform multiple subtasks are so-called *pipeline systems*. In a pipeline system, the output from one component is used as the input to the next component. This architecture is conceptually simple, and makes it is easy to develop and modify each component separately. The drawback is that each component has to make greedy decisions using the information it has available, even if new evidence is uncovered during later processing steps. Errors therefore propagate through the system.

The performance issues caused by the pipeline architecture can be avoided by adopting a *joint architecture*, where a joint model for simultaneous extraction of entities and relations/arguments is learned from the data, so that evidence from all sub-tasks can influence the choices made for the other tasks, leading to a globally optimal choice. Although some research has been conducted into developing joint information extraction systems, the state-of-the-art is still dominated by pipeline systems.

2.2.3 MetaMap

MetaMap is an entity recognition tool for the biomedical literature that extracts entities and likes them to UMLS, a standardised vocabulary for biomedicine, augmented with a semantic network [Aronson and Lang, 2010]. MetaMap uses the SPECIALIST parser, a rule-based parser that draws heavily on the rich lexical resources available through UMLS [McCray et al., 1993]. After the linguistic preprocessing step, all noun phrases found are further analysed in the following way (see [Aronson, 2006] for a detailed description):

Variant generation Variants are generated for the noun phrase itself, as well as for any single word and sequence of words within the noun phrase that occurs in the SPECIALIST lexicon. Variants are generated by recursively substituting words with their synonyms, abbreviations/acronyms and derivational variants (e.g. cancerous $\Rightarrow$ cancer).

Candidate identification Any UMLS meta-thesaurus entry with a string matching any of the variants is identified as a candidate.

Mapping evaluation The candidates are evaluated, and the best scoring candidate is chosen as the UMLS concept for the noun phrase. Evaluation uses a linguistically motivated heuristic procedure, that is a linear combination of the following scores, all ranging from 0 to 1:

- **Centrality** is 1 if the head word of the NP is included in the mapping, 0 otherwise.
- **Variation** measures the difference between candidate and the noun phrase, where identical phrases score 1, and the score decreases the more steps were required during variant generation.
- **Coverage** measures how much of the input text is used in the mapping.
- **Cohesiveness** measures how gapped the mapping text strings are.

In summary, MetaMap is a knowledge-based entity recognition system that draws heavily on the extensive lexical resources available for the biomedical domain.

2.2.4 SemRep

SemRep is a commonly used relation extraction tool for the biomedical domain. SemRep is a pipeline system that uses the SPECIALIST parser and the Xerox Part-of-Speech tagger [Cutting et al., 1992] for linguistic preprocessing, and MetaMap as the entity recongition component. Manually written rules identify phrases that trigger potential relations. In case of a match, the system tries to find the arguments for the relation, exploiting semantic restrictions (i.e. the relation *treats* requires its object to be of semantic type *disease or syndrome*) that are specified in UMLS, and hand-written syntactic restrictions (i.e. the object of a relation triggered by a nominalization must be the object of the preposition *of* headed by the nominalization).

2.3 OCEAN-CERTAIN and Oceanographic Climate Science

Oceanographic Climate Science (OCS) is a cross-disciplinary field that studies the interaction between climate change and the oceanic processes, and therefore draws researchers from areas such as biology, chemistry, physical oceanography and the social sciences. OCEAN-CERTAIN is a EU 7th framework programme research project that works in oceanographic climate science. OCEAN-CERTAIN aims to understand the effects of global warming on the oceanic *food web* and the *biological pump*.

The *food web* describes the feeding patterns in an ecological system in terms of who eats whom (see Figure 2.2 for an example). Species are divided into *trophic levels* according to their relative position in the food web. On the lowest trophic level are the *autotrophs*, who produce biomass from inorganic material, normally through photosynthesis, a process that consumes CO_2 and sunlight, transforming it into biomass. Because of this, autotrophs are also called *primary producers*. The biomass produced by the primary producers flows through the food web as organisms are consumed by species on a higher trophic level.

Phytoplankton are the most common primary producers in the oceans. Several different species of phytoplankton exist, with different physical characteristics. Some of the phytoplankton species are *calcifying*, in that they from a calcium carbonate exoskeleton around their body. This exoskeleton is called a *coccosphere*, and is a composition of calcium carbonate plates called *coccoliths*. Whether calcifying or non-calcifying phytoplankton species dominate in an environment is dependent on several factors, such as the availability of mineral nutrients.

As photosynthesis requires access to CO_2 and sunlight, the phytoplankton communities live near the surface of the ocean. When phytoplankton die, most of their biomass remains in the surface layer of the ocean, and is eventually released back into the atmosphere. However, the shell of calcifying phytoplankton species ballasts the biomass, making some of the biomass sink into the deep ocean, eventually storing some of the carbon in the ocean floor sediment. Also, when phytoplankton are consumed by species living deeper in the ocean, additional biomass is transported downwards. This downwards vertical flux of carbon is referred to as the *biological pump*, and has significantly mitigated the effects of man-made CO_2 emissions on global climate by absorbing parts of the CO_2 emissions.

However, climate change and human activity creates *stressors* on the food web, such as ocean acidification, ocean pollution and overfishing. Stressors have various direct effects on some of the species in the environment, and indirect effects are propagated throughout the food web. A single stressor is therefore likely to affect the entire ecosystem. As the biological pump is dependent on the species composition of phytoplankton and the species that feed on them, stressors also affect the efficiency of the biological pump. Even though the structure of the food web and the direct effects of stressors might be well known,

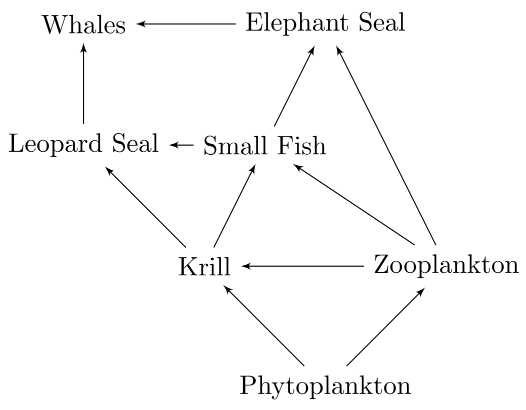

Figure 2.2: A simplified version of the Arctic food web

the eventual effects of various stressors are hard to predict in this highly complex system. Prediction is an even harder task when taking multiple simultaneous stressors into account.

Of particular interest in OCS is the discovery of *feedback loops.* A feedback loop is a situation where a change in one variable causes a chain of events that causes a new change to the original variable. In a negative feedback loop, the change to the original variable caused by the loop is the opposite of the change that initiated the feedback loop, causing the feedback loop to *stabilize*, i.e. stop. In a positive feedback loop, the change to the original variable caused by the loop is the same as the change that initiated the feedback loop, causing the feedback loop to continually increase in intensity. Figure 2.3 illustrates a very simple positive feedback loop, where global warming decreases the efficiency of the biological pump, which in turns increases the intensity of global warming, thus creating a self-reinforcing loop. If, on the other hand, global warming were to increase the efficiency of the biological pump, then the biological pump would decrease the intensity of global warming, causing the loop to stabilize, meaning that the global warming process will stop at a certain point. Because of the likelihood and consequences of the existence of feedback loop between global warming and the biological pump, exploring the potential existence of such a feedback loop is one of the goals of oceanographic climate science.

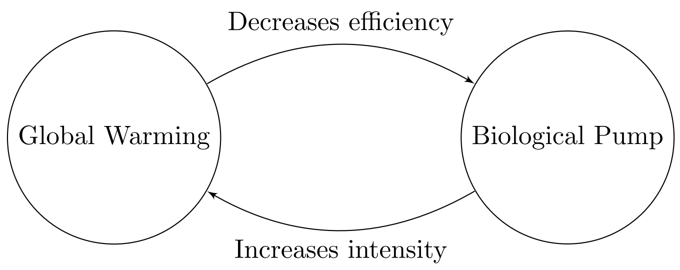

Figure 2.3: A positive feedback loop

Chapter 3

Developing an Annotated Corpus

Development of information extraction systems is normally supported by the development of a corpus of manually annotated data. Firstly, developing the annotated corpus is useful because an annotation scheme that defines which kinds of concepts and relations should be annotated needs to be developed. Successful annotation schemes are normally developed iteratively as they tend to need modification when applied to real text. Secondly, the annotated corpus is an important resource, as a large amount of annotated data is required for training data-driven systems. An annotated corpus is also required for testing the performance of any developed system. Having an annotated corpus available is also beneficial for the developers of a knowledge-driven system, because studying the patterns of expression occurring in the annotated corpus helps the developers create realistic rules for the systems. Furthermore, a publicly available annotated corpus will be of benefit for everyone working on natural language processing in oceanographic climate science or related domains. It was therefore decided to develop a pilot annotated corpus as the next step in the research process.

During textual annotation, spans of text are first annotated as belonging to a category. For instance, in an annotation scheme for event extraction focused around persons and transfer of ownership, the sentence "At that moment, Obi-Wan Kenobi gave Luke a lightsaber" can be annotated as shown in Figure 3.1. The annotated text spans are referred to as *triggers* for their respective categories. If a corpus for relation or event extraction is developed, the relations or argument structures of the events are annotated after annotating triggers. An ownership-transfer event extraction annotation scheme may for instance focus on the arguments of the transfer events, marking arguments such as *agent* (the entity initiating the transfer), *beneficiary* (the entity that gain possession of something due to the transfer) and *theme* (the entity that is transferred). Figure 3.2 gives the same example sentence annotated with arguments.

A graphical annotation tool, *brat* (brat rapid annotation tool) is used for annotation. Brat is publicly available open source software, and can be downloaded from `http://brat.nlplab.org/`. Brat runs as a server, and can be accessed through a web browser. Annotations in brat are stored in file separate from the source text, using a stand-off annotation format, so that the original text file remains unchanged. All images of annotated text in this thesis are taken from brat.

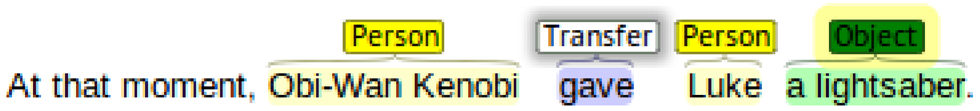

Figure 3.1: Example text span annotation

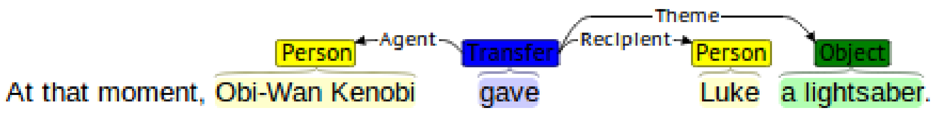

Figure 3.2: Example argument annotation

3.1 Annotation Scheme

When developing an annotated corpus, the first step is to develop an *annotation scheme*. An annotation scheme specifies which types of entities and events should be annotated, and which categories (e.g. person, transfer, agent, theme) to use during annotation. This largely defines the type of information that the information extraction system is able to extract, so the downstream components need to be taken into account when developing the annotation scheme.

While there are an infinite number of potentially interesting relations expressed in natural language, an annotation scheme must realistically limit its treatment to a manageable subset, as there is a trade-off between expressibility of the annotation scheme and complexity of annotation. A more complex annotation scheme increases the cognitive load on the annotator, thus increasing the time required to perform the annotation and reducing the consistency of the annotated data. A more complex annotation scheme is also harder to replicate automatically. As a result, annotation schemes tend to vary widely between domains, as different domains have different relations of interest.

Our annotation scheme is developed in an incremental fashion, with the potential to gradually expand to include more relations of interest. The scheme presented here should therefore be considered likely to be expanded at a later stage. The initial relations of interest were chosen to fulfil the following criterion:

$$\begin{array}{c} increase(atmospheric\ CO_2) \rightarrow \\ decrease(ocean\ pH) \rightarrow \\ decrease(calcifying\ phytoplankton\ ratio) \rightarrow \\ decrease(biological\ pump\ efficiency) \rightarrow \\ increase(atmospheric\ CO_2) \\ \hline \models positive\_feedback(atmospheric\ CO_2) \end{array}$$

Figure 3.3: A causal chain that represents a positive feedback loop.

$$\begin{array}{c} increase(atmospheric\ CO_2) \rightarrow \\ decrease(ocean\ pH) \rightarrow \\ increase(calcifying\ phytoplankton\ ratio) \rightarrow \\ increase(biological\ pump\ efficiency) \rightarrow \\ decrease(atmospheric\ CO_2) \\ \hline \models negative\_feedback(atmospheric\ CO_2) \end{array}$$

Figure 3.4: A causal chain that represents a negative feedback loop.

The relations must be so specific as to be useful for making discoveries in the domain, but still so general as to facilitate the extraction of knowledge from related, but different, domains. The latter part was a desideratum from the domain experts, as they felt that they were already sufficiently familiar with their own domain, but knowledge from similar domain, such as climatology, could lead to new insights.

Domain experts pointed out that feedback loops are of particular interest in oceanographic climate science, as stated in chapter 2.3, and the discovery of these was therefore selected as the goal for the LBD system. Formulating a scheme to discovery feedback loops can be facilitated by the observation that *feedback loop* is an abstract property of some causal chains: If the result of a causal chain and the initiating change are the same, then the causal chain is a positive feedback loop. If the result or a causal chain and the initiating change are opposite, then the chain is a negative feedback loop. Feedback loops can in other word be inferred from causal chains, as illustrated in figures 3.3, 3.4 and 3.5. Information extraction should therefore focus on extraction of *events* and causality between elements, so that a reasoning step can be able to combine events to form causal chains, and discover feedback loops in the causal chains.

Discussions with the domain experts revealed that feedback loops of interest in the domain hold between directional changes to quantitative variables, such as *increase in atmospheric* CO_2. Directionality in the change is required, because it might be the case that for instance *increase in atmospheric* CO_2 causes a positive feedback loop, whereas *decrease in atmospheric* CO_2 does not. Information extraction should therefore extract information about directional changes to quantitative variables to use as elements in the causal chains.

The resulting system of concepts and relations that the information extrac-

$$\begin{array}{c} \mathit{increase}(\mathit{atmospheric}\ CO_2) \rightarrow \\ \mathit{decrease}(\mathit{ocean}\ pH) \rightarrow \\ \mathit{decrease}(\mathit{krill\ population}) \rightarrow \\ \mathit{decrease}(\mathit{leopard\ sealpopulation}) \rightarrow \\ \mathit{decrease}(\mathit{whale\ population}) \rightarrow \\ \hline \not\models \mathit{negative\_feedback}(\mathit{atmospheric}\ CO_2) \end{array}$$

Figure 3.5: A causal chain that does not represent any feedback loop.

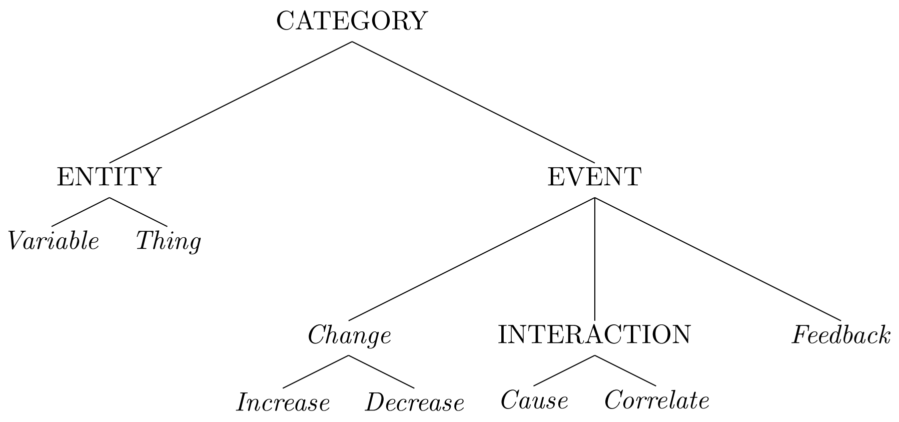

Figure 3.6: Type Hierarchy for the Annotation Scheme

tion system should extract therefore consists of quantitative variables, directional changes to these variables and causal interactions between such directional changes, as well as explicitly mentioned feedback loops. Information is extraction as *domain rules* of the form $\mathit{direction}_1(\mathit{variable}_1) \rightarrow \mathit{direction}_2(\mathit{variable}_2)$, such as $\mathit{increase}(\mathit{ocean\_CO_2}) \rightarrow \mathit{decrease}(\mathit{ocean\_pH})$, that represent the established facts in the domain.

Corpus annotation revealed that the interactions between changes of quantitative variables that are expressed in the literature are often weaker than causal. The interaction type *correlation* was therefore included in the annotation scheme, as it provides clues about the existence of feedback loops, albeit weaker than causal interactions. Domain rules based on correlations are on the form $\mathit{direction}_1(\mathit{variable}_1) \Leftrightarrow \mathit{direction}_1(\mathit{variable}_2)$.

Figure 3.6 presents the primary categories of interest in the annotation scheme. Types written in italics are categories that are used during annotation, and types written in capital letters are abstract types used to show the high level relations between the types. The following sections will describe each of the categories further.

3.1.1 Variables

Variable in our annotation scheme is 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 things, counts, frequencies and ratios.

Only variables involved in quantitative changes are used to infer feedback loops, but a scientific article can mention many variables that are not involved in any change. As these are not of interest, variables that do not occur in any event are not annotated.

(1) a. Calcification increases [the acidity of seawater].
  b. [The acidity of seawater] can be measured ...
  c. ... changes in [the network of global biochemical cycles].

In the example (1) above[1], only the text span in (1-a) is annotated as a variable in our scheme. In (1-b), the entity is measurable, but is not involved in a change, and the entity in (1-c) is, although involved in a change, not quantifiable along any ordered axis.

Variables are annotated with the maximal text span, including noun phrase modifiers, to incorporate as much information as possible about the variable. This design choice is based on the observation that it is easier to discard superfluous information than to acquire new information in the post-processing steps. As an exception to this rule, some constructions that are judged to not convey any additional information about the variable, including appositions and non-restrictive relative clauses are never annotated. The annotation in (2-a) is therefore preferred to the annotation in (2-b).

(2) a. [Chlorophyll biomass in the surface ocean] is regulated by ...
  b. [Chlorophyll biomass] in the surface ocean is regulated by ...

At the moment, the category *variable* covers all types of quantitative variables, but we plan to include a more fine-grained distinction of variable types in the future, possibly linking to existing domain ontologies.

3.1.2 Thing

Thing is a catch-all entity category that has evolved to fill multiple roles the annotation process revealed structures that could not be handled by the annotation scheme. The category *thing* is used in three cases:

1. Some text spans cannot reasonably be classified as a variable by itself, but can be interpreted as a variable when lexical information encoded in the trigger for the change it undergoes is factored in. In example (3), *coccoliths* is not a quantitative variable by itself, and can therefore not be annotated with the *variable* tag. However, *larger* can be interpreted as

[1]In the examples, brackets indicate annotated or emphasised text spans.

meaning *increase in size*, which when combined with *coccoliths* creates the quantitative variable *coccolith size*. The *thing* category is used to annotate arguments such as *coccoliths* in these cases.

2. Some text spans signal both a change and a variable. The most common example of such a text span is *ocean acidification*, which can be interpreted as *decrease in ocean ph*. For lack of a better annotation category, these cases are also annotated as *thing* in the current scheme.

3. When a coordinate structure consists of several variables, but also some conjuncts that cannot reasonably be classified as *variable*, *thing* is used to annotated the non-variable conjuncts. Example (4) illustrates this case, where the conjuncts *temperature* and *irradiance* are quantitative variables, whereas *nutrient distributions* is not quantitative. *Nutrient distributions* is therefore annotated as *thing* rather than *variable*.

(3) Coccoliths were larger in the high nutrient experiment.

(4) Changes in temperature, nutrient distributions and irradiance ...

The heterogeneity of the thing category increases the cognitive load on the annotator, and it has become clear that *thing* is used inconsistently between annotators. Additionally, the experiments with automatic annotation conducted in this thesis (chapters 4 and 5) reveals that automatic annotation tools have problems learning the *thing* category. Because of this, the thing category is to be removed in the new version of the annotation scheme. In the new annotation scheme, case 2 will be marked as change events without any theme argument, and the other cases will be marked as variables. The reasoning component will then be responsible for interpreting them in context. For more information, see chapter 6.

3.1.3 Change Events: Change, Increase and Decrease

A change event describes a change in the value of a quantitative 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 is underspecified in the text. Example (5) shows some change events from the corpus.

(5) a. [INCREASE Increasing] concentrations of CO2 ...
b. The [DECREASE reduced] degree of calcification is ...
c. Calcification was strongly [CHANGE affected] by [INCREASE elevated] CO2.
d. Indeed, Δ18O [CHANGE changed] linearly with ...

Changes are only annotated if they apply to a variable, and only if they represent quantitative changes. In Example (6-a) *change* is not annotated, as it does not apply to any variable. In Example (6-b), *change* is not annotated, because it does not signal a quantitative change, but rather a change in which variables are present.

(6) a. Sea-ice retreat due to global change makes Arctic Ocean ...
b. ... proxy for various variables that change with location and time.

The variable that is affected by the change event is marked as the *theme* argument of the change event. As apparent from the description above, every change event is required to take a theme argument. Quite differently from variables and things, change events are annotated with the minimal text span that is required to signal the change. Words that bear mostly grammatical functions are not annotated as part of the change event trigger. The annotation in (7-a) is therefore preferred to the annotation in (7-b). This is done because it reflects how a single word root can be used as a change event trigger across multiple grammatical contexts, as illustrated by example (8).

(7) a. [INCREASE Increases] in [THEME phytoplankton growth] ...
b. [INCREASE Increases in] [THEME phytoplankton growth] ...

(8) a. ... thus [INCREASE:Participle increasing] [VARIABLE:DirectObject CO_2 levels] significantly.
b. This [INCREASE:Noun increase] in [VARIABLE:PrepositionalObject CO_2 levels] ...
c. [VARIABLE:Subject CO_2 levels] [INCREASE:Verb increased] ...

In some cases, the text states explicitly that a variable is in a state, but the state can be interpreted as resulting from an underlying change. If this happens, the explicit state is annotated as a change event, as in example (9), where "high" is interpreted as the result of an increase from low to high carbon.

(9) In the [INCREASE high] [VARIABLE carbon] experiment ...

3.1.4 Interaction Events: Cause and correlate

Cause events signal that one change event causes the other change event. Cause events take two arguments: *Agent*, signalling the causing event, and *theme*, signalling the caused event. Example (10) illustrates some well-formed cause events. Note that the arguments are change events, and therefore have internal structure.

(10) a. The regression suggests [AGENT stronger nitrogen recycling], [CAUSE resulting in] [THEME higher carbon export].
b. [THEME Expression of genes increased] [CAUSE in response to] [AGENT increasing CO_2].
c. [AGENT Increasing CO2 levels] [CAUSE cause] [THEME Δ18ODIC to increase].

Correlate event signal that two change events are correlated, which is taken to be a weaker form of interaction than causality. As change events involve variables, correlations ultimately hold between two variables. Correlate takes two arguments, *theme* and *co-theme*.

From a statistical standpoint, correlations are bidirectional relations, but we have observed that there is a tendency to focus on one of the changes as initiating the the change in the other, giving correlations a quasi-directional interpretation[2]. If there is an observable directionality to the correlation, the initiating event is marked as the theme argument and the resulting change event is marked as the co-theme. Otherwise the syntactic object of the correlation trigger is selected as co-theme, and the other event as the theme as a default. In (11-a), no directionality is observed. However, in (11-b) and (11-c) there is a directionality to the correlation, so arguments are assigned according to the status of the involved change events.

(11) a. [THEME Calcite Δ18O decreases] strongly [CORRELATE with] [CO-THEME increasing concentrations of CO_3^{-2}].
b. [CO-THEME Expression of genes decreased] [CORRELATE in] [THEME the high CO_2 treatment].
c. [CORRELATE Associated with] this [THEME increase in opal] there was a [CO-THEME decrease in CaCO3%].

The annotation scheme does not distinguish between directional and non-directional correlations, which is problematic, because the reasoning component cannot then exploit this distinction. Also, the resulting inconsistent usage of theme and co-theme makes it hard for automated systems to pick the correct argument types, as illustrated by the experiments in later chapters. Therefore, in the version of the annotation scheme currently under development, directional correlations take theme and co-theme arguments, while non-directional correlations simply take two theme arguments instead.

Cause and correlate events are only annotated if there is an explicit trigger for the cause or correlation, and interactions are only annotated within the scope of a single sentence. No interaction is therefore annotated in Example (12).

(12) The [VARIABLE mineral nutrients] were [INCREASE added] to the experimental culture. At the end of the incubation period, [VARIABLE bacterial growth] was found to have [INCREASE increased] by 236% compared to the control cultures.

Interaction event triggers, as opposed to change events, are annotated as phrases, rather than single words. This is because functional words are often required for a head word to signal an interaction event. Compare the two sentences in example (13).

(13) a. Phytoplankton biomass is changing [CAUSE in response to] climate change.
b. The [CHANGE response] of the nitrogen flux measurements may be due to ...

[2]From a discourse point of view, this makes sense, as some entities are more focused/topical than others.

It is not uncommon for interactions to hold between a change event and a variable, or directly between two variables, rather than two change events, as illustrated in example (14). In these cases, the variable is interpreted as undergoing an implicit change. Sometimes, when there is an correlation between two variables with no change event specified, the text explicitly states whether the correlation is positive or negative, in which cases this is annotated as one of the more specific categories *positive_correlate* or *negative_correlate*, as in example (14-c).

(14) a. . . . found a significant [CORRELATE correlation] between [THEME:VARIABLE suspended POC] and [CO-THEME:VARIABLE potential respiration].
b. An [THEME:INCREASE increase in the abundance of Planctonomycete-specific 16S rRNA] [CORRELATE is indicative of] [CO-THEME:VARIABLE anammox bacteria].
c. [THEME:VARIABLE AT] and [CO-THEME:VARIABLE CT fluxes] were [+CORRELATE positively correlated].

It is also quite common that an interaction lies implicit in a change event, as in example (15). In these cases, no text span is explicitly annotated as an interaction event. Instead, one of the change events acts as a cause or correlations, taking the other change event as the agent or co-theme argument, in addition to the normal theme argument.

(15) a. Thus, [INCREASE increased] [VARIABLE food production] [INCREASE heightens] [VARIABLE the magnitude of the terrestrial Si pump].
b. This suggests that [INCREASE more acidified] future [VARIABLE oceans] will [INCREASE increase] [VARIABLE the rate of dissolution of organic matter].

3.1.5 Grammatical structures: And, or, refexp

In addition to the main entity and event categories that have been discussed above, some auxiliary categories have been defined to handle common language constructs, as these facilitate extracting the information of interest. There are two Coordination Event categories, *and* and *or*, that are used to annotate conjunctions and disjunctions of events and entities. These are only annotated when they connect multiple annotated text spans, and take these text spans as *part* arguments. Example (16) illustrates a conjunction of variables. The change event takes the conjunction as the theme argument, which again takes the variables as theme arguments.

(16) . . . thus [INCREASE increasing] [PART:VARIABLE carbon] [THEME:AND and] [PART:VARIABLE nitrogen concentrations].

In addition, there is one category, RefExp, which is used to handle referring expressions such as *that*, *it*, *such changes* etc., that refer to entities and events in previous sentences. These take the previously mentioned text span as a *coref* argument, and can itself be used as an argument for an event as though it were the element it refers back to. Example (17) illustrates this.

(17) a. A statistically significant [COREF:INCREASE increase] was observed in [THEME:VARIABLE phytoplankton biomass].
b. [THEME:REFEXP This increase] can be [CAUSE attributed to] [AGENT:DECREASE decreased predator activity].

Referring expressions, like entities, are only annotated if they actually occur as arguments in an event structure, such as a change or interaction event.

3.1.6 Feedback loops

The discovery support system will focus on the discovery of feedback loops by chaining quantitative changes. However, there are some cases where feedback loops are mentioned directly in the text, as in example (18). The feedback tag is used to annotate these cases, and feedback events take the two involved variables as theme and co-theme arguments, where the choice between them is arbitrary[3].

(18) Our model suggests [+FEEDBACK a positive feedback] between [THEME temperature] and [CO-THEME atmospheric CO_2content].

The annotation scheme distinguishes feedback loops of unspecified direction (*feedback*), positive feedback loops (*postive_feedback*) and negative feedback loop (*negative_feedback*) by means of different tags.

3.1.7 Negation

Negated changes and interactions are annotated just like actual events, but with a "negative" attribute specified on the event, as in example (19). The trigger for the negation is not annotated in the present annotation scheme.

(19) a. [VARIABLE The transport efficiencies of various ballasts] did not [CHANGE:NEG vary] significantly after 1000 m.
b. [VARIABLE production rates] were [CHANGE:NEG unaltered].
c. [VARIABLE The observed variables] did showed no [CORRELATE:NEG relationship with] [VARIABLE sea-water ph].

3.2 Annotation Process

Development of the annotation scheme proceeded in an iterative fashion: The initial annotation scheme and guidelines were developed in cooperation with

[3]This is also likely to change in the next version of the annotation scheme, as it seems better for feedback events to take two theme arguments, given that there is no difference in focus of the two variables.

the domain experts. A set of 12 abstracts from journal articles selected by domain experts as representative for the field were annotated by one annotator (Erwin Marsi) and reviewed by another annotator (Elias Aamot), leading to a discussion and revision of the annotation scheme. A description of the annotation scheme developed at that stage formed the basis for a workshop paper [Marsi et al., 2014b]. Subsequently, one annotator (Elias Aamot) annotated 8 full journal articles also selected by the domain experts as representative of the field. Another annotator (Erwin Marsi) revised some of the annotations, and some minor adjustments were made to the annotation scheme, resulting the guidelines presented here. Two additional papers were annotated to create evaluation materials for the experiment presented in chapter 4.

The resulting corpus therefore consists of 10 full text papers. The annotation process revealed that the *Methods and Materials* sections did hardly contain any relations of interest, so these sections were not annotated in the corpus. Due to copyright restrictions and the wish make the corpus publicly available, all papers in the corpus are taken from the open access journal PLoS One[4]. Copyright and reuse restrictions poses a large challenge to acquiring enough diverse textual material for developing the corpus.

It is usual to have multiple annotators annotate the same material to improve annotation quality and also to calculate inter-annotator agreement. However, the manpower constraints associated with this master project did not allow for a systematic measurement of inter-annotator agreement. However, the review process revealed that the annotators did mostly agree, except for some specific categories, such as *thing*. It was measured that an annotator familiar with the annotation guidelines is able to annotate approximately 30-35 sentences per hour of concentrated work.

3.3 Conclusion

This chapter presented answers to research questions 1 and 2, discussing the type of annotations scheme required to extract information that supports discovery in the domain. Several challenges have been mentioned: Annotation speed and consistency, computational learnability, acquiring textual material, and a continually changing annotation scheme, requiring revision of previously annotated text.

To increase annotation consistency and learnability, several changes to the annotation scheme are planned, including changing the argument types of *correlate* and phasing out *thing*. Presently, an annotator is annotating new material using the updated annotation scheme, and a revision of the existing corpus is planned if no problems are encountered with the new annotation scheme.

[4]`http://www.plosone.org/`

Chapter 4

Automatic Annotation by Pattern Matching

As described in chapter 3, the reason to develop an annotated corpus is to facilitate the development of information extraction tools. Conveniently, information extraction tools can reduce the work required to create an annotated corpus, by automatically annotating at least parts of the text. If conducted properly, this can lead to a bootstrapping process where gradually better information extraction tools make it possible to annotate larger quantities of literature, which again can give rise to better information extraction tools. This chapter develops an information extraction system based on pattern matching with patterns manually extracted from the annotated corpus of 8 journal articles. This system is subsequently evaluated as an annotation assistance system, to see whether primitive automatic systems can assist human annotators. The same system is also used to compare the strengths and weaknesses of machine learning systems to pattern matching systems in chapter 5.

Figure 4.1 illustrate the conceptual functionality of the pattern matching system developed in this chapter. Automatic annotation is performed on unseen text by pattern matching. The annotation patterns are developed based on the pilot annotated corpus. If the system is used as an intelligent annotation assistant, the automatically annotated text is then given to an annotator for manual correction, whereas if the system is used in an LBD pipeline, the annotated text is then sent to the domain rule extraction component.

This chapter is structured as follows: Section 4.1 explains the types of annotation patterns used by the system and the manual process of developing the annotation patterns. Section 4.2 explains briefly how the pattern matching system was implemented. Section 4.3 explains the methodology and results for the evaluation of the system. Finally, section 4.4 discusses the implication of the results produced by the evaluation, viewing the system as an annotation helper.

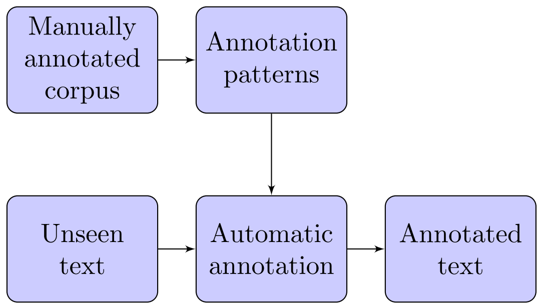

Figure 4.1: Flow diagram for the pattern.

TRIGGER *increase*
VAR X dobj T ; X prep "in" pobj S

Figure 4.2: Example structural pattern consisting of two sub-patterns.

4.1 Designing the Patterns

The pattern-matching system developed in this chapter uses two different types of patterns. The simplest type of patterns, *surface patterns*, which uses simple string matching. A surface pattern consists of a set of sub-patterns, and every sub-pattern must be found in the target string for the pattern to be considered matched. Each string is normalized to lower case and non-letter characters are removed, to improve recall of this kind of patterns. As will be explained in 4.1.2, these patterns are used for detection of interaction events.

The second type of pattern used in the systems is *structural patterns*, which are matched against the dependency structure of each sentence. Each structural pattern consists of a set of sub-patterns[1], where each sub-pattern defines a path in the dependency structure. A path is linear sequence of nodes and edges, alternately. In addition to the sub-patterns, each pattern consists of a trigger word, which has a special purpose during pattern matching (see chapter 4.2). Each structural pattern also specifies which kind of event it detects, and what category (variable or thing) the argument it takes belongs to.

Edges in a sub-pattern path are constrained by *dependency type*, such as *nsubj, dobj* or *prep*. Node element can be constrained to a specific word lemma, such as *"of", "in"* or *"trend"*, assigned to a variable (X or Y), constrained to match the trigger of the pattern (*'T'*), or specify the end-anchor (*'S'* or *'N'*). The end anchor specifies the root of the sub-tree in the dependency structure that becomes the argument of the event if the pattern matches.

[1]For ease of implementation, and due to the fact that no patterns in the data required more, a pattern can only consist of 1 or 2 sub-patterns in the current implementation, but this can easily be generalized.

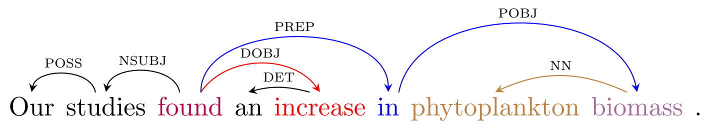

Figure 4.3: Pattern matching the dependency structure.

The sub-graph spanned by the sub-pattern paths must form a connected graph, so every path must share at least one node with at least one other path, and this property is exploited for efficiency during pattern matching (see chapter 4.2). The node shared by two paths is called the *join*, and must be either a variable, the end-anchor, or the trigger.

As an example, consider the structural pattern in figure 4.2 with the trigger *increase*. The pattern consists of two sub-patterns, *X dobj T* and *X prep "in" pobj S*, and signals an argument of type variable ("VAR"). The join of the two sub-patterns is the node constrained by the variable 'X' An example of a successful matching is shown in figure 4.3, where the red and blue paths signal the two sub-patterns, and the brown sub-tree is the sub-tree rooted by the end anchor. The variable *X*, constrained to the node "found", joins both paths, and the word *biomass* is the last element in the path of the second pattern, which specifies it to be the end-anchor. This matching gives rise to the annotation below (Example (1)), as the trigger word is annotated as the change event trigger, and the sub-tree rooted at the anchor is annotated as a variable, and chosen as the theme argument of the change event.

(1) Our studies found an [INCREASE increase] in [THEME:VARIABLE phytoplankton biomass].

As will be seen in chapter 4.1.1, structural patterns are used for detection of change events. As explained in the annotation guidelines, variables and things are not annotated unless a part of an event, and interaction events are only annotated if they connect two arguments. Therefore, pattern matching is conducted as follows: First, patterns are applied to detect change events. All arguments that are detected by the patterns are annotated as variables or things. Whenever there are more than two change events in a sentence, patterns to detect interaction events are applied between every subsequent pair of change events in the sentence.

Annotation patterns were manually developed by the protocol described in the following sections, in order to create annotation patterns that realistically represent the desired relations, rather than writing rules based on idealistic preconceptions. Annotation pattern were developed in two phases, first developing patterns for change events (*change*, *increase* and *decrease*) and their arguments, then expanding the system to accommodate interaction events.

The resulting set of patterns is available online at `https://github.com/EliasAamot/MasterData`.

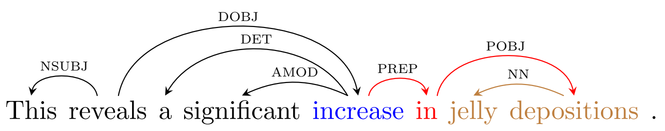

Figure 4.4: Example Dependency Parse Tree; The event trigger is coloured blue, the theme argument is coloured brown, and the path from the trigger to the theme argument is coloured red.

4.1.1 Patterns for Change Events

As a preprocessing step, the entire corpus was parsed to dependency format using the Stanford Parser [Klein and Manning, 2003], and all change event trigger words were extracted from the annotated corpus. The set of patterns was developed using the following procedure: For every trigger word, search through the annotated corpus for sentences containing the corresponding lemma. Ignore all the sentences where the lemma occurs, but is not annotated as the trigger for an event. For every sentence where the lemma occurs as a trigger for an annotated event, store the dependency path from the trigger to the theme argument as a sub-pattern, keeping all closed class words as strings, and abstracting all open class words to variables. In the dependency tree illustrated in figure 4.4, the extracted annotation pattern would be "*T prep "in" pobj S*" with the trigger "increase".

If there is an obvious parsing error, use the dependency path that would hold in the correct parse tree. If the path is too cluttered by complex phenomena, such as conjunctions, to give rise to a realistically useful pattern, ignore it. This does not hurt recall significantly, because for every complex case, there normally exists a simple version of the same underlying pattern in the data.

When all patterns for a lemma have been developed, the next step is to look through all sentences where the lemma occurs again, and tally the number of true and false positives each pattern generates on the development corpus. If a pattern yields a high number of false positives, add more elements or sub-patterns to make the pattern more specific if there is an obvious way of doing so. Otherwise, exclude the pattern in the final set of annotation patterns. There is no absolute threshold on the ratio of false to true positives that are accepted, as the types of errors and availability of data should be taken into consideration as well, but the developer should keep in mind that a system to assist human annotation should strive for high precision, but worry less about recall.

The pattern development process revealed that while quite unambiguous patterns exist for increase and decrease events, the patterns for change events are much more ambiguous, and the potential theme argument strongly influences whether a trigger word corresponds to an actual change event or not.

The procedure above only works for single-word triggers. While the corpus has been annotated to make most change event triggers single words, there are some examples of multi-word triggers, especially triggers of the type *more ADJ* or *less ADJ*. In these cases, the most discriminating word (the ADJ in the example cases) is used as the trigger word, and the pattern is augmented by a sub-pattern requiring the presence of the other word(s).

Negation is handled by the following heuristic: Certain triggers, such as *unaltered* and *stability*, are specified by the developer as initially negated. All other triggers are by default non-negative. When a match has been found, a secondary pattern matching procedure checks for the existence of a negation word or the determiner *no* in the immediate sub-tree of the trigger. If this is found, the negative polarity of the trigger is reversed.

4.1.2 Patterns for Causes and Correlations

As described in the annotation guidelines, change events in one sentence can be connected either implicitly, where one event takes the other as an agent or co-theme argument, or explicitly, where there is a text span annotated as *cause* or *correlate* connecting the change events. Ideally, an automatic annotation system should be able to replicate both kinds of connections. However, the pattern matching system described in this chapter focuses on only the explicit connectors cause and correlate, because they are easy to detect using surface patterns, whereas probabilistic inference is most likely required to uncover implicit connections, due to the lack of unambiguous trigger words. As the current system is primarily intended as a tool to assist annotation, and only secondarily as a full automatic annotation system, the explicit connections are also the most interesting, as a human can connect two change events by an agent or co-theme relation more quickly and easily than first annotating a text span as an explicit interaction event, and then connecting that span to each of the involved events.

Visual inspection of the dependency paths between the triggers for interaction events and their arguments revealed that these paths tended to be much longer than the corresponding paths for change events. The length make it hard to formulate specific dependency patterns that still generalize for the type of syntactic relation that holds between the interaction trigger and its arguments. Also, longer dependency chains are more susceptible to parsing errors, which tend to make the patterns useless in practice. Because of this, the interaction event detection component uses surface matching, rather than structure matching.

When a pattern is attempted matched against a target text, the matching process works as follows: Interaction events hold between two pairs of change events, so matching is not performed on sentences less than two interaction events. For each adjacent pair of change events in the sentence, the sentence is divided into three parts: The string before the text span of the first change event, the string between the text spans of the change events and the string after the text span of the change event. For instance, the sentence in example (2) gives rise to the parts below.

TRIGGER 2
BETWEEN due to

TRIGGER 1
BEFORE due to

Figure 4.5: Patterns created by the example sentences in (3).

(2) The results seem to indicate that the [$_{\text{CHANGE}}$ increased biological activity] is a consequence of the [$_{\text{CHANGE}}$ nutrient additions] described above.

BEFORE *The results seem to indicate that the*

BETWEEN *is a consequence of the*

AFTER *described above.*

Interaction events between non-adjacent change events were ignored during pattern development, because the procedure for surface matching will detected multiple interactions for the same trigger, in this case. Non-adjacent interaction events are quite rare in the development data, so it is suspected that only little coverage is lost by ignoring them.

The annotation patterns for interaction events were developed according to the following protocol: For every trigger in the annotated corpus, a pattern with one sub-pattern corresponding to the trigger at the location it occurs is extracted. Then the sentence is studied, to look for other words that are deemed to be required indicators for the interaction. If any such words are encountered, each of these is made into a separate sub-pattern. It is important to note here that the multiple occurrences of the same trigger phrase can give rise to several patterns depending on their relative position with respect to their argument events. For instance, the sentences in example (3) *due to* gives rise to the two patterns in figure 4.5, where one matches if *due to* is found in the *between* part, and the other matches if *due to* is found in the *before* part.

(3)
a. [$_{\text{CAUSE}}$ Due to] [$_{\text{CHANGE}}$ decreased] nutrient availability, we observed [$_{\text{CHANGE}}$ diminished] bacterial growth.
b. We observed [$_{\text{CHANGE}}$ diminished] bacterial growth [$_{\text{CAUSE}}$ due to] [$_{\text{CHANGE}}$ decreased] nutrient availability.

Additionally, every cause event trigger must specify which of the change event arguments is selected as the agent argument. The number following the keyword *trigger* specifies whether the first or second change event is annotated as the agent.

The patterns were preliminarily tested by running them on the annotated corpus. This revealed that patterns consisting of only a single preposition tended yield more False Positives than True Positives. Single preposition patterns were therefore removed from the final set of patterns, increasing precision significantly at the cost of recall.

4.1.3 Ignored Structures

The patterns developed as explained above represent a simple, prototype pattern-matching system. While it should be sufficient to get an indication of how well a pattern-matching can perform on the task of automatic annotation, several complex aspects of the annotation are ignored that need to be considered in a complete system. This section lists the annotation structures that are ignored by the current version of the pattern matching system.

Conjunctions : The annotation tags for conjunction and disjunctions (*and* and *or*) are ignored.

Referring Expressions : Both "RefExp" annotation tags, and their "co-ref" arguments are ignored.

Changing Things : The cases where a *thing* tag is used to annotate a text span that is both a change and a variable, such as *ocean acidification*, are ignored.

Interactions between two variables : Interaction events that hold between two variables with no explicit change are ignored.

Interactions between Variable and Change Event : Interaction events that hold between one variable with no explicit change and one complete change event are also ignored.

Implicit Agent/Co-themes : The system never treats change events as implicit interaction events, and all agent/co-theme links between change events with no explicit interaction events connecting them are ignored.

Interactions between non-adjacent Change Events : As stated earlier, interaction events are only detected between adjacent change events, because simple pattern matching is not able to disambiguate which change event is the correct argument.

Feedback : Feedback categories are ignored by the system, as no feedback events occurred in the development data.

Many of these structures could theoretically be handled by pattern matching without any additional tools[2], but doing so would require significant engineering effort to develop and tune the patterns to yield an optimal precision/recall trade-off, handle special cases and to combine the different pattern matching processes.

[2]The Stanford Parser is able to handle detection and resolution of referring expressions, but tools developed or customized for resolution of the specific kind of referring expressions used in this annotation task will most likely yield much better performance.

Also, it is suspected that additional (statistical) techniques are required to detect some of the structures. For these reasons, none of these structures are handled by the prototype system presented in this chapter.

4.2 Implementation

The pattern matching-based automatic annotatino system has been implemented as a Python program, which uses arbitrary patterns written in a human-readable pattern language, as illustrated in 4.2 (structural pattern) and 4.5 (surface pattern). When annotating new textual material, the Stanford Parser is used as a pre-processing step which tokenizes, lemmatizes, conducts sentence splitting and parses the sentences to derive the dependency structure for each sentence. Pattern matching is then conducted. Output is provided as annotations in brat format.

The first component in the pattern matching pipeline uses structural matching to extract change events from the papers. For efficiency reasons, only patterns with triggers that match a lemma in the sentence are applied. Pattern matching tries to match one sub-pattern at the time, until either all sub-patterns have matched or one sub-pattern fails to match. The pattern matching system always tries to match the sub-pattern containing the trigger first, starting matching in the dependency tree from the node representing the trigger word. The pattern matching itself is conducted as a breadth-first search. For subsequent sub-patterns, matching starts from the join node. By starting at the most constrained node of a sub-pattern (the trigger or join), the search space becomes maximally constrained, consequently reducing search time.

When a structural pattern matches, the sub-tree rooted by the node signalled as the end-anchor by the pattern is extracted as the argument. Sometimes, however, the trigger word is in the sub-tree rooted by the end-anchor, illustrated in figure 4.6, in which case the trigger must be removed from the sub-tree that represents the argument, because there can be no overlap between the text span of the event trigger and the argument trigger. In the pattern language developed for this system, the end-anchor symbol "S" signals that the entire sub-tree should be used to form the argument, while the end-anchor symbol "N" signals parts of the sub-tree, such as the sub-tree rooted by the trigger must be pruned away.

Because structural matching is used for change events only, patterns can only have a single end-anchor, which represents the *theme* argument. The system can be extended to open for multiple end-anchors that signal different argument types if structural matching is to be used for extraction of other event types.

Brat annotations must occupy a continuous span of text without any gaps, and for various reasons, such as pruning of the sub-tree, certain syntactic constructions and errors in tokenization, the resulting dependency structure does not necessarily correspond to a continuous text span. A simple heuristic is used to map the dependency sub-tree to a continuous text span; the longest continuous string is extracted as the text span for the argument.

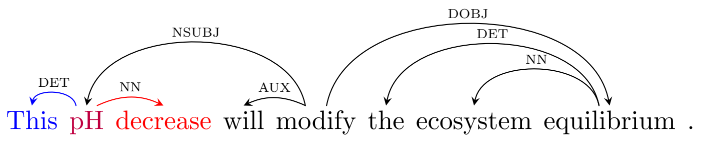

Figure 4.6: Example of structure where the trigger is in the sub-tree of the end-anchor. Extracted argument in blue, matched pattern in red.

After detection of change events, a surface matching component tries to detect interaction events in all sentences containing at least two change events. For every pair of subsequent interaction events, the system splits the sentence into three parts, as described above (chapter 4.1.2), and tries to perform a matching for every pattern, one sub-pattern at the time. Matching is done on a word-to-word basis, so only whole words are matched, ensuring that patterns such as *due* don't match words such as *residue*.

Except for some minor optimizations, such as only attempting full pattern matching if the trigger word occurs as a lemma in the sentence, efficiency has not been considered during system development. The pattern matching system spends approximately 2 seconds to perform pattern matching for a single paper, so there is likely some room for improvement, but the running time is nevertheless dominated by the time required to parse the papers using the Stanford Parser, which is in the order of 1 minute per paper.

4.3 Experimental evaluation

4.3.1 Evaluation Method

Two papers recommended by the domain expert were downloaded from PLoS and annotated manually to serve as a held-out test data. The pattern matching system was then applied to unannotated versions of the same papers. A human judge then compared the output from the pattern-matching system to the gold standard, based on the protocol described below. Note that the protocol is designed to reflect usefulness as a annotation assistance tool, rather than usefulness as an automatic annotation software.

Change Events

Two change events are considered *equivalent* iff they occur in corresponding sentences, the event types are the same and the text spans of both the event trigger and the theme argument exhibit at least a partial overlap. Assuming, a gold standard annotation as in example (4), then (5-a) contains an equivalent Change Event, whereas (5-b) does not, because the event type is different, and (5-c) does not contain an equivalent change event, because the theme arguments do not exhibit any overlap.

(4) In this experiment, we observed a significant [INCREASE increase] in [VARIABLE deep sea primary production].

(5) a. In this experiment, we observed [INCREASE a significant increase] in deep sea [VARIABLE primary production]
b. In this experiment, we observed a significant [DECREASE increase] in [VARIABLE deep sea primary production]
c. In this experiment, [VARIABLE we] observed a significant [INCREASE increase] in deep sea primary production.

Two change events are considered *semi-equivalent* iff they occur in corresponding sentences, the event types are the same and the text spans of the event triggers exhibit at least a partial overlap, but the text spans of the theme arguments exhibit no overlap. In the examples above, (5-c) contains a change event that is semi-equivalent to the gold standard change event.

A change event in the output is counted as a true positive iff an equivalent change event occurs in the gold standard, and as a false positive iff no equivalent or semi-equivalent sentence occurs in the gold standard. Change events in the output that have a corresponding semi-equivalent change event in the gold standard are not counted. This is because it is not correct, and can therefore not count as a true positive, but is still helpful for a human annotator, as it gives an indication of a true change event.

A change event in the gold standard is considered a false negative if no equivalent or semi-equivalent change event occurs in the output.

Sometimes the pattern matching system creates a *heap* of change events, which is several change event that have the exact same event trigger span, but completely different arguments. An example of a heap is given in figure 4.7. Heaps occur when multiple patterns with the same trigger match at the same position. An entire heap is counted a true positive if at least one of the change events is equivalent to a change event in the gold standard, and the entire heap is counted as a false positive if no change event in the heap is equivalent to a change event in the gold standard. This evaluation procedure is used because a human can easily determine the correct event in a heap with a correct event, making such a heap useful for the annotator.

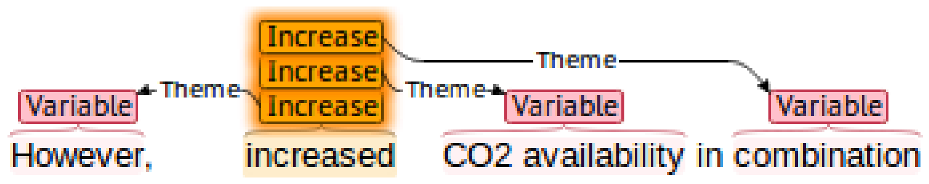

Figure 4.7: An Example of a Heap produced by the pattern matching system.

A change event that is semantically correct, but does not have a corresponding change event in the gold standard because the entire text span is annotated as a variable or thing is not counted during evaluation. Figure 4.8illustrates this situation. While both events are entailed by the sentence, only the increase is annotated in the gold standard.

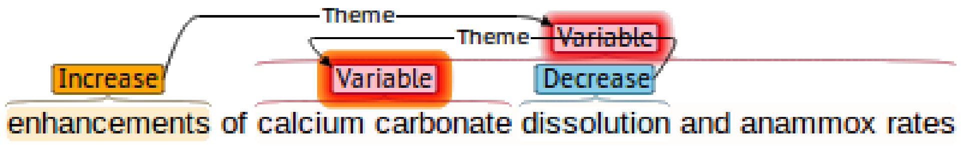
Theme
Theme
Variable
Increase
Variable
Decrease
enhancements of calcium carbonate dissolution and anammox rates

Figure 4.8: Example of an event embedded in a variable.

Interaction Events

Two interaction events are *equivalent* iff they occur in corresponding sentences, the triggers have the exact same text span and type, and they have at least one equivalent or semi-equivalent argument. Assuming a gold standard annotation as in (6), the interaction event in (7-a) is not equivalent to the gold standard, because neither the theme nor agent arguments are equivalent or semi-equivalent. The interaction event in (7-b) is, in fact, equivalent to the event in the gold standard, because the event span and type are the exact same, and the theme arguments are equivalent. The interaction event in (7-c) is not equivalent, because both the event trigger text span and the event type are different.

(6) In our low light experiment, [AGENT addition of iron] [CAUSE resulted in] [THEME increased phytoplankon growth].

(7) a. In our low light experiment, [THEME addition of iron] [CAUSE resulted in] [AGENT increased phytoplankon growth].
b. In our [AGENT low light] experiment, addition of iron [CAUSE resulted in] [THEME increased phytoplankon growth].
c. In our low light experiment, [AGENT addition of iron] [CORRELATE resulted] in [THEME increased phytoplankon growth].

An interaction event in the output is counted as a true positive if it exists an equivalent interaction event in the gold standard, and as a false positive otherwise. An interaction event in the gold standard is counted as a false negative if it does not have any equivalent interaction event in the output.

For heaps of interaction event, the same evaluation criteria as for change event heaps are used.

Other Details

The gold standard can be revised during evaluation if errors or inconsistencies in the manual annotation is detected. Error and inconsistencies are expected

Metric	Restricted	Full
Precision	0.91	0.91
Recall	0.62	0.50
F-score	0.74	0.65

Table 4.1: Scores for pattern matching system

to be found during any systematic study of corpus, and even more so as the systematic application of patterns tend uncover events that human annotators may have missed.

To be explicit: A variable/thing is never counted alone, as the entire change event is considered one unit for the purpose of counting true and false positives and negatives. On the other hand, for interaction events, the interaction event trigger and its argument arrows are counted as one unit, while the arguments are counted separately, being individual change events.

False negatives are counted in two alternative ways: Once counting the restricted set of structures that are handled by the system only, as described in the protocol above, and once for the full set of structures, i.e. including the structures described above in chapter 4.1.3. For the second count, all occurrences of structures in the gold standard that are not handled by the system are added to the first count. The first count should therefore provide an estimate of the performance of the system on the task it is created to perform, and the second count estimates its performance on the complete annotation task.

4.3.2 Results

The human judge counted 221 true positive events, 21 false positive events, and 134 false negative events in the first count. When comparing to the full gold standard, 223 false negatives were found. An overview of the derived evaluation metrics is found in table 4.1.

For each of the three general error types (false positives, false negatives among treated structures, false negatives among ignored structures), errors were further grouped into sub-types to provide an overview of the causes for errors in the system. The frequencies of the error types are given in tables 4.2, 4.3 and 4.4. The structures that were ignored by the pattern matching systems were discussed in chapter 4.1.3, while the rest of this section will explain the error types for false positives and false negatives among treated structures in more detail.

The following error types exist for false negatives:

Missing Trigger An event was not detected by the pattern matching system because the trigger word/phrase did not exist in the pattern base.

Prepositional Inter. Event An interaction event was not detected because the trigger was a single preposition. As described in chapter 4.1.2, these

Error Type	**Frequency**
Missing Trigger	52
Prepositional Inter. Event	26
Parsing Error	16
Follow-up errors (Inter.)	15
Missing Pattern	14
Lack of Conj-treatment	9
No tokenization of -	2
Total	134

Table 4.2: Error Types by Frequency (False Negatives)

Error Type	**Frequency**
Co-ordinations	43
Inter. Event between two variables	14
Implicit Agent/Co-theme	12
Inter. Event between variable and Change Event	10
Changing Things	10
Total	89

Table 4.3: Error Type by Frequency (Ignored Structures)

Error Type	**Frequency**
Not a Variable	9
Not a Change	4
Total	13

Table 4.4: Error Types by Frequency (False Positives)

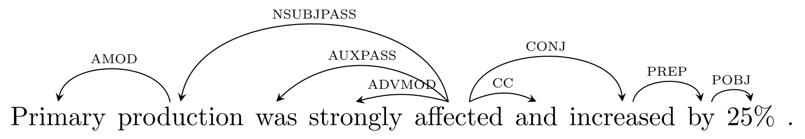
NSUBJPASS
AMOD
AUXPASS
ADVMOD
CONJ
CC
PREP
POBJ
Primary production was strongly affected and increased by 25% .

Figure 4.9: Example of conjunction. The pattern *T nusbjpass S* does not match the relation between *increase* and *primary production* due to the *conj* edge.

triggers were not included in the system due to the extremely high rate of false positives.

Parsing Error A change event was not detected because the parser assigned the wrong grammatical structure to the sentence.

Follow-up Errors An interaction event was not detected because of errors in change event detection. Either the system failed to detect a required change event, or an error in text span mark-up caused the trigger of the interaction event to be contained in the text span of an entity.

Missing Pattern A change event was not detected even though the trigger existed in the pattern base, because the exact structural pattern did not exist in the pattern base for that trigger.

Lack of Conj-treatment Failure of patterns to take conjunction edges in the dependency graph into account cause a change event to not be detected. See figure 4.9 for an example.

No tokenization of - A pattern could have been successfully applied if the parser had split up words divided by a hyphen/dash into multiple tokens.

The following error types exist for false positives:

Not a Variable The event detected by the system cannot be considered an acceptable change event, because the theme argument is not a quantitative variable, nor can it be interpreted as one in the context of the event trigger word.

Not a Change The event detected by the system cannot be considered an acceptable change event, because the text span cannot be interpreted as signalling a change in the given context.

4.4 Discussion

In a task as difficult as automatic annotation, a precision score of 0.91 can be considered quite high. A recall of 0.62 (or 0.5 for the full task) is also good, especially for a system that targets precision over recall. While the numbers seem to be good for a prototype system, there are no baselines or upper bounds that the numbers can be compared to. If conducted properly, inter-annotator agreement should have been measured during corpus annotation, but lacking manpower precluded this. What the evaluation does show, is a lot of room for improvement. The analysis of error types provides a starting point for discussing potential improvements:

Missing Trigger Recall can be improved significantly by adding more trigger words to the pattern base as they more papers are annotated. Because the total number of unique words increases logarithmically with the amount of text, as stated by Zipf's Law, the expected number of new triggers decreases exponentially with the number of papers annotated.

Prepositional Interaction Events As discussed earlier, interaction events with single prepositions as triggers yield a high rate of false positives, which is not desirable in a "high precision, low recall" setting. However, the distances between the trigger and the arguments of these kinds of interaction events tend to be shorter for other interaction events, making it likely that structural pattern matching can yield productive results for this kind of interaction events.

Parsing Error Most parsing errors are systematic and predictable, and additional patterns can be developed the match the most common misparsed structures. The most common kind of parsing error is related to PP-attachment, where the PP attaches to the main verb rather than the NP. This can be handled by by adding an additional pattern. See figure 4.10 for an illustration.

Follow-up errors Can be improved by improving the recall of the change event matching part, and by improving the rules for extracting spans of text.

Missing Pattern Missing patterns can be added in as they are encountered. If a drastic improvement is desired, one can group words with similar argument structures together, and use the same structural patterns for every trigger in the group. For instance, when words like *stimulate* and *enhance* all have the structural patterns *T dobj S* and *T nsubjpass S*, then it is likely that *improve* will have both the structural patterns, even if only one of them occur in the data.

Lack of conj-treatment Can be treated by creating additional patterns that take *conj* edges into account, or by improving the underlying search mechanism of the structure matching component to match patterns even across *conj* edges.

No tokenization of - By tokenizing words divided by a dash/hyphen into separate tokens, these cases can be handled. However, in this case, care must be taken to change the patterns that depend on hyphenated words being tokenized as a single word, such as the pattern for "up-regulate".

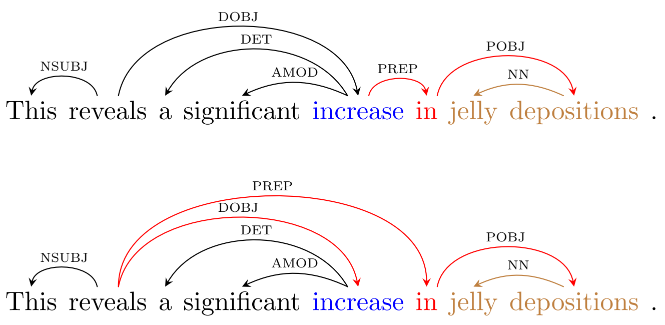
DOBJ
DET
AMOD
NSUBJ
PREP
POBJ
NN
This reveals a significant increase in jelly depositions .
PREP
DOBJ
DET
AMOD
NSUBJ
POBJ
NN
This reveals a significant increase in jelly depositions .

Figure 4.10: Example of a typical parsing error: PP attachment. Above is the correct parse tree, below is the same sentence with a parsing error. Red shows pattern required to handle this.

Not a variable While it is hard even for a human to understand what exactly constitutes a quantitative variable, some heuristic could be developed for pruning away text spans that are clearly not a quantitative variable. One simple heuristic is to create a list of stop words, containing such words as ”model”, ”system”, ”state”, ”structure” and ”data”, that often are found as syntactic arguments of common trigger words, but are not quantitative variables.

Not a change Handling this case requires a sophisticated machinery, because it involves nuances that are problematic even for humans.

It is important to keep in mind that the evaluation procedure was designed to reflect usefulness as an intelligent annotation assistant, not as an automatic annotation system, the former being a much easier task. Despite attaining high scores in the evaluation procedure presented here, it is less useful as an automatic system due to the amount of manual post-editing that is required to bring the output of the system up to human quality annotation. Studies conducted on a random sample of output sentences (n=50), revealed that approximately 60% of all change events that were considered true positives or ignored during evaluation required manual post-editing to yield human standard annotation. Three kinds of post-editing was required: 1) manual adjustment of the text span of the variable/thing, 2) the deletion of superfluous/incorrect events in event heaps and 3) the deletion of change events that are entirely contained in the text span of a variable or thing. The random sample revealed no cases of change event triggers that required post-editing, most likely because change event triggers are normally single words.

Reducing the amount of post-processing required appears at least as important as improving the evaluation score of the system. The amount of heaps can be reduced by allowing only one pattern match to be taken into account for a given trigger word. Preferably, the system can rank the matches according to some metric and pick the best match for a given trigger. A simpler method would be for the developer to rank the patterns for each trigger manually, giving the most specific patterns higher rank. The current algorithm for extracting the text span for the theme argument given a pattern match is very primitive, and there is a lot of room for improvement. One particular situation that often causes error in the test data is when a comma intervenes between the head and non-restrictive relative clause, and the current system picks the relative clause rather than the head, because it is longer. The third kind of post-editing can be avoided by automatically removing any change event that is contained entirely within the text span of a variable.

Despite being heavily dependent on manual post-editing, there are several indications that the pattern matching system can be used beneficially as a tool for annotation assistance. While it is hard to quantify the extent of this, the pattern matching system was able to discover several events that the human annotator had missed when developing the gold standard annotation. The pattern matching system can therefore contribute by increasing the consistency and thus quality of the human annotated data.

Furthermore, an experiment on annotating the test data based on the output of the pattern matching system revealed that a human annotator could annotate at a rate of approximately 75 lines per hour of concentrated work, more than doubling of speed compared to an annotator working from scratch. It can be objected that the improved speed was due to the fact that annotator was already familiar with the papers, but the annotator had annotated several papers multiple times in the course of the corpus development process (due to changes in the annotation scheme and line splitting errors), without experiencing any significant boost in speed in these cases.

4.5 Conclusion

In the context of research question 3, the results can be considered positive, as they show that even at this primitive level, an intelligent tool can speed up and improve the quality of the annotation effort. Although the quality is far from sufficient to conduct fully automatic annotation with no human interaction, the relatively high scores attained by the system seem to imply that it is possible to develop a high-quality information extraction system based on hand-written patterns, if enough effort is put into it. However, there are many cases of ambiguous triggers, so it seems likely that some probabilistic component will be required to handle all cases. Also, certain aspects of the annotation scheme, such as implicit change events are likely to require probabilistic reasoning to accommodate.

Another observation was that determining the scope of the entity argument was shown to be non-trivial, and some research should be conducted into the best way of doing that. Two challenges are associated with this: First, inappropriate parts of the sub-tree rooted by the anchor must be pruned away. This can most likely be done by a simple rule-based system, but some research must be conducted into exactly which types of sub-trees should be pruned away. Second, the dependency sub-structure must be associated to a continuous text span. The current heuristic of picking the longest continuous text span in the dependency sub-structure has been shown to be insufficient, as it fails to give the required weight to the head word.

Chapter 5

Automatic Annotation by Machine Learning

Machine learning systems represent a completely different approach to automatic annotation than pattern matching, by exploiting statistical information from a training data corpus to learn how to optimally conduct automatic annotation. In this chapter, an existing machine learning-based information extraction system is adapted to the OCS domain by training on the annotated corpus to see how such a system compares to pattern matching systems, using the system developed in the previous chapter for comparison.

Section 5.1 describes TEES, the machine learning system that was adapted to evaluated the performance of machine learning system on this task, and also motivates the choice of TEES as a benchmark system. Section 5.2 explains the evaluation procedure and shows the results. Section 5.3 discusses the results, provides an in-depth error analysis, and compares the performance of TEES as a benchmark system to the performance of the pattern matching system developed in the previous chapter. Section 5.4 summarises the main findings and concludes this chapter.

5.1 TEES

The Turku Event Extraction System (TEES) [Björne et al., 2009] was chosen as the preferred candidate for domain adaptation to OCS because it has shown state-of-the-art performance over several years of development, can extract events with arbitrary structures, is trainable with different data sets and the source code is publicly available[1]. TEES has been developed for the biomedical domain, and has attained high scores on a number of shared tasks, including BioNLP 2009 (1st place) [Björne et al., 2009], BioNLP 2011 (1st place in 4/8 tasks) [Björne and Salakoski, 2011] and Drug-Drug Interactions 2011 Challenge

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

(4th place) [Björne et al., 2011].

TEES is a pipeline event extraction system with two main components. The first component, *Trigger Detection*, which classifies every word as either a trigger for an annotation category or as not a trigger. The subsequent *Edge Detection* component examines every ordered pair of triggers, and classifies the relation between the triggers as a relation (Agent, Theme, Co-theme and Part in our scheme) or as a non-relation. As relations thus can be inferred between any pair of triggers, not just between an entity and an event, this lets the system extract arbitrary event structures. A final component, *Unmerging*, performs some cleaning up of the extracted structure. All components are SVM-based machine learning components.

At the current stage of development, TEES is not able to detect intra-sentential relations. However, our annotation scheme contains one relation category that can hold between sentences, namely the *Coref* relation. Because of this limitation of TEES, experiments are conducted ignoring the *Coref* argument category and the corresponding trigger category *RefExp*.

TEES is designed for a task where most triggers are single words, and most triggers extracted by the system are therefore single words. No multi-word triggers are extracted unless an exactly equivalent trigger phrase is found in the training data. This means that most entities (*variable* and *thing*) in the test data will not be extracted fully, because their length make it unlikely that an exact equivalent string is annotated in the training data. Entities in the output are therefore normally represented by a single trigger word, which a post-processing step can potentially expand by taking the entire dependency sub-tree in a procedure similar to the one used by the pattern matching system.

TEES is implemented as a publicly available Python program, developed mainly by Jari Björne at the University of Turku. Mr. Björne has been very helpful in providing support and bug-fixing for TEES so that this experiment could be realized.

TEES uses several external tools in its pipeline. The efficient SVM$^{\text{multiclass}}$ implementation[2], is used, with linear kernels. For a description of the features used in TEES, see [Björne et al., 2009]. For linguistic preprocessing, the Charniak-Johnson parser[Charniak and Johnson, 2005] is used with the McClosky-Charniak model adapted for biomedicine [McClosky and Charniak, 2008], and the output is converted to the Stanford dependency format [de Marneffe and Manning, 2008].

TEES is primarily designed for event extraction tasks where entities are given in the system input. The pipeline contains wrappers for the BANNER biomedical entity extraction system [Leaman and Gonzalez, 2008], for use in cases where entities are not given in advance. The inclusion of BANNER implies that although the system is able to conduct entity recognition in the trigger detection step, it is not expected to perform comparably to a specialized ER system on the task.

During trigger detection, every token is classified separately, and subsequent

[2]`http://www.cs.cornell.edu/people/tj/svm_light/svm_multiclass.html`

tokens that are assigned to the same class are merged to a multi-token trigger iff the merged string occurs as a trigger in the training data. The design choice works well with single word triggers, but is likely to yield decreased performance for multi-word triggers.

TEES uses the so-called Interaction XML (IXML) format for data exchange, both internally and for interfacing with other programs. For the experiments presented below, a pipeline of Python scripts were therefore developed to perform the preprocessing and transform the results to IXML, as well as to evaluate the results by comparing the IXML output by TEES to the IXML of the test set. In our pipeline, Stanford CoreNLP tools are used for linguistic preprocessing. The choice of Stanford CoreNLP over the general domain Charniak-Johnson parser was made due to ease of use and the fact that Stanford CoreNLP provides output directly in the Stanford dependency format, rather than based on an evaluation of performance.

5.2 Experimental Evaluation

5.2.1 Experimental design

To get the most out of the limited data available, 5-fold cross-validation was conducted. Cross-validation is a technique for assessing the performance of a machine learning system (or statistical model in general) under limited data. In k-fold cross validation, the data is divided into k roughly equal partitions, and validation is conducted k times, each time using a different partition as test data and the other partitions as training data. With 5-fold cross validation over 10 papers, each fold consists of two papers, meaning that 2 papers are used for testing and the other 8 are used as training data.

Comparing the output from each fold to the gold standard test set for that fold, a confusion matrix is filled out. A confusion matrix is an N-by-N matrix, where N is the number of categories in the classification scheme. The rows correspond to categories in the gold standard, whereas the columns correspond to categories in the system output, and the value of a cell is the number of times an item of the row category in the gold standard has been classified as an item of the column category in the system output. A confusion matrix can therefore function as a basis for investigating the kinds of error a system makes. In the experiment presented here, separate confusion matrices were calculated for triggers (Variable, Thing, Increase, Decrease, Cause, Correlate, And and Or) and for relations (Agent, Theme, Co-theme, Part), as elements can be misclassified within these groups, but not across them.

An element in the output is matched to an element in the gold standard as long as there is any degree of overlap between the text spans, just like in the PMS evaluation. This is necessary because TEES outputs mostly single word triggers that may correspond to larger text spans in the gold standard.

From a confusion matrix, the precision, recall and f-score can easily be calculated with respect to any category. These evaluation metrics can also be calculated by taking the average over the precision, recall and f-score metrics for every category. This kind of global average performance is called the macro performance. The macro score gives unproportionally large influence to categories that occur very rarely, such as *or*. A weighted average performance, called the micro performance, where every category is weighted according to its proportion in the gold standard examples, is therefore an at least equally interesting metric.

Entities recognition (detection of variables and things) is likely to be harder in our annotation scheme than in the biomedical domain: The triggers do not have any identifying surface characteristic and they are semantically diverse. Perhaps even more problematically, text strings that may in fact represent quantitative variables are often left unannotated, because they do not occur in any event, as required by the annotation scheme. This diversity and lack of consistency is believed to make entities very hard to learn, especially for a pipeline system like TEES. Because of this, a separate experiment was also conducted, in which the entities were provided in the input to the system, simplifying the task to only detecting event triggers and their arguments. This problem formulation corresponds closely to the shared tasks for which TEES has been developed, such as the Genia Event Extraction task in the BioNLP shared tasks [Kim et al., 2009]. As the pattern matching system did not perform too well on interaction triggers and event argument selection, this second experiment can shed some light on whether the machine learning approach taken by TEES suits better for those tasks. The first experiment will be referred to as the *full experiment*, the second as the *event extraction experiment*.

5.2.2 Results, Full Experiment

Running a 5-fold cross-validation on TEES yielded the confusion matrices shown in Tables 5.1 and 5.2. Tables 5.3 and 5.4 present the evaluation metric scores per category.

5.2.3 Results, Event Extraction Experiment

Running a 5-fold cross-validation on the dataset where Entities (Variable and Things) are provided as input to TEES yielded the following confusion matrices shown in Tables 5.5 and 5.6. Tables 5.3 and 5.4 present the evaluation metrics for every category as well as global scores.

5.3 Discussion

TEES attained micro f-scores of 0.39 and 0.20 on triggers and arguments respectively, in the full experiment. In the event extraction experiment, TEES achieved micro f-scores of 0.55 and 0.49. Considering the training data limited amount of training data and the fact that TEES was applied to a domain that it was not developed for, the results are within the expected range. TEES was

		Predicted										
	■	None	Variable	Thing	Increase	Decrease	Change	Cause	Correlate	And	Or	Sum
True	None	0	659	53	68	30	41	9	3	4	0	867
	Variable	715	494	9	7	3	2	2	2	0	0	1234
	Thing	173	24	33	0	1	1	1	0	0	0	233
	Increase	269	18	0	253	5	1	2	1	0	0	549
	Decrease	187	4	0	1	123	0	0	1	0	0	316
	Change	229	4	2	2	0	44	0	1	0	0	282
	Cause	163	3	1	3	0	4	2	0	0	0	176
	Correlate	108	0	1	6	0	0	2	17	0	0	134
	And	150	0	0	0	0	0	0	0	4	0	154
	Or	10	0	0	0	0	0	0	0	0	0	10
	Sum	2004	1206	99	340	162	93	18	25	8	0	

Table 5.1: Confusion matrix for Triggers, full experiment.

	Predicted					
		None	Agent	Theme	Co-theme	Part	Sum
True	None	0	15	393	14	8	430
	Agent	293	6	5	0	0	304
	Theme	1140	1	260	3	0	1404
	Co-theme	128	1	6	8	0	143
	Part	346	0	0	0	8	354
	Sum	1907	23	664	25	16	

Table 5.2: Confusion matrix for Arguments, full experiment.

	Full Experiment			Event Extraction		
	Precision	Recall	F-score	Precision	Recall	F-Score
Variable	0.41	0.40	0.40	-	-	-
Thing	0.33	0.14	0.20	-	-	-
Increase	0.74	0.46	0.57	0.80	0.61	0.70
Decrease	0.76	0.39	0.52	0.83	0.61	0.70
Change	0.47	0.16	0.24	0.68	0.52	0.59
Cause	0.11	0.01	0.02	0.25	0.05	0.08
Correlate	0.68	0.13	0.22	0.50	0.11	0.18
And	0.50	0.03	0.06	0.79	0.30	0.43
Or	0.00	0.00	0.00	0.00	0.00	0.00
Macro	0.44	0.25	0.32	0.55	0.31	0.40
Micro	0.50	0.32	0.39	0.69	0.46	0.55

Table 5.3: Evaluation Metrics by category, Trigger categories.

	Full Experiment			Event Extraction		
	Precision	Recall	F-score	Precision	Recall	F-Score
Agent	0.26	0.02	0.04	0.40	0.10	0.16
Theme	0.39	0.19	0.23	0.66	0.53	0.58
Co-theme	0.32	0.06	0.10	0.38	0.08	0.13
Part	0.80	0.02	0.04	0.73	0.25	0.37
Macro	0.44	0.04	0.07	0.54	0.24	0.33
Micro	0.43	0.13	0.20	0.62	0.40	0.49

Table 5.4: Evaluation Metrics by category, Argument categories.

		Predicted								
		None	Increase	Decrease	Change	Cause	Correlate	And	Or	Sum
True	None	0	41	19	22	18	9	11	0	120
	Increase	112	391	11	20	0	1	0	0	535
	Decrease	91	22	194	8	0	1	0	0	316
	Change	114	17	5	151	2	0	0	0	289
	Cause	152	3	3	13	9	4	0	0	184
	Correlate	93	13	1	7	7	15	0	0	136
	And	108	0	1	0	0	0	46	0	155
	Or	9	0	0	0	0	0	1	0	10
	Sum	673	487	234	221	36	30	58	0	

Table 5.5: Confusion matrix for Triggers, Event Detection Experiment.

		Predicted					
		None	Agent	Theme	Co-theme	Part	Sum
True	None	0	41	365	18	32	456
	Agent	268	30	6	0	0	304
	Theme	657	3	742	2	0	1398
	Co-theme	119	1	11	12	0	143
	Part	266	0	0	0	88	354
	Sum	1310	75	1142	32	120	

Table 5.6: Confusion matrix for Arguments, Event Detection Experiment.

among the best-performing systems on BioNLP's Genia Event Extraction task, with an f-score of 0.51 [Nédellec et al., 2013], although the values are not directly comparable due to differences in the tasks, evaluation methods and Annotation Schemes. While the results are far from perfect at the moment, it is possible that TEES will perform acceptably in oceanographic climate science given some tweaking and additional training data.

Comparing the results of the full and event extraction experiments, it is not surprising that the results for the event extraction experiment are better, as it is a simplified task. The improvement is especially large in the recall of argument categories, which makes sense, as detecting arguments depends on a successful trigger detection phase. Improved performance across all other categories can be attributed to the fact that the provided entities are good clues for the existence of other trigger types in the sentence.

Manual inspection of the system output reveals that a significant portion of the entities in the system output occur independently, without participating in any event. Such entities do not conform to the annotation guidelines, and cannot be used by the downstream reasoning component. A post-processing step would therefore have to consider every independent entity, and either remove it, or force the system to detect an event to which it can belong. The relatively much higher precision than recall of entity categories seems to indicate that it would be better to treat detected entities as indications that there are events present in the sentence than to delete them, but a more systematic study of the independent entities produced by TEES must be conducted before a definite conclusion can be reached. This touches upon one of the main weaknesses of a pipelined system; a decision must be made greedily at a stage in the pipeline without regards to information that might be uncovered later. The types of annotations required by our scheme seem to require a system that is able to reason jointly over events and entities.

5.3.1 Error Analysis by Category

In both the full and event extraction experiment, it can be seen from tables 5.3 and 5.4 that the micro evaluation metric scores are significantly better than the macro scores. This reveals that the system achieves worse performance on the less frequent categories, possibly indicating insufficient training data to learn the characteristics of the less frequent categories. Another possibility might be that TEES learns the distribution of the categories in the training data, and therefore favours the more common categories.

The micro precision performance over entity categories is somewhat lower than the micro over all categories for the full experiment (0.40 vs. 0.50 Precision), with performance much better for variables than for things. The low performance on the thing category is likely due to that category's role as a "catch-all category" that encompasses three very different kinds of objects (see chapter 3.1.2 for details). Keeping the perceived difficulty of the task in mind, system performance in the variables category is relatively good, with 80% of micro precision across all categories, and a recall of 0.36, equating to 113% of micro recall. With the apparent difficulties with entity recognition discussed earlier in this chapter in mind, the results for the variable category are surprisingly good.

For both experiments, performance is highest on the change event triggers, with increase and decrease outperforming change. This is likely due to the fact that these categories have a relatively small set of trigger words, and that most trigger words are unambiguous indicators of a change event. As was discovered during development of patterns, triggers for the change category are more ambiguous, explaining the weaker performance in this category.

System performance is lower for causes and correlations, which is expected as these categories are of a higher order than change Events (They take change events as arguments), so errors committed during detection of entities and change events propagate to the interaction event detection step. As these categories are similar in argument structure and context, one would expect performance to be roughly equal for both categories. Surprisingly, system performance is much higher on correlations than causes. This is the opposite of what happened during pattern development, when it was found that single word prepositional correlation triggers were too ambiguous to be used as patterns. A possible explanation for this performance is that most correlations have single word triggers, with a large portion of them exhibiting similar grammatical properties (as they are prepositions), whereas causes tend to have multi-word triggers, which TEES struggles to handle, and cause triggers are more grammatically diverse.

TEES yields high precision for *and*, which is most likely due to the restricted set of trigger words, and their clarity. The low recall is likely due to the fact that only conjunctions that participate in events are annotated, leaving most conjunctions in the training data as negative examples. The system is unable to learn the *or* category, most likely due to the sparsity of the category in the training data.

Performance over the argument categories shows significant improvement from the full experiment to the event extraction experiment, indicating that the performance on these categories depend strongly on the lower-order (trigger) categories. Part stands out as having very high precision, which is likely due to the fact that coordination events are unambiguous indicators of such relations, whereas change and interaction events can take arguments of any of the other three relation categories, leading to more ambiguity. Of the three remaining categories, performance is consistently better for the theme category. This is perhaps caused by the high number of examples in the training data.

5.3.2 Analysis of the confusion matrix

The patterns of confusion that are revealed by the confusion matrices are in no way surprising: The tendency of confusion between change event types is to be expected, as there are some ambiguities that are hard for humans to clarify as well. For instance, in example (1) does *dissolution* signal an increase or decrease? When CaCO3 is dissolved, it becomes less of it in its original location. However, if CaCO3 is dissolved in water, it means that there will be a higher level of CaCO3 in the water after the dissolution process.

(1) The dissolution of CaCO3 is one of the ways ocean acidification can, potentially, greatly affect the ballast of aggregates.

The relatively high degree of confusion between cause and correlation is also to be expected as humans also tend to disagree on the exact category in many of the cases. For instance, in example (2), does *with* signal a cause or a correlation?

(2) The carbon:nitrogen ration increases with decreasing vertical flux.

It might seem surprising that there is a noticeable confusion between change events and interaction events, but, as discussed in chapter 3.1.4, it is common for change events to express causes and correlations implicitly, leading them to take similar arguments. This leads to a high degree of overlap in contexts, making it harder for a classifier to discern the categories. The confusion is especially strong from cause to change, which is not surprising, given the existence of words such as *effects* which can signal both a change and a cause.

For argument types, there is a significant confusion between theme and co-theme, which is natural as these two are treated interchangeably by human judges in certain situations, for instance in undirected correlations. The relatively high degree of misclassification of gold standard agents as themes is somewhat surprising, but most likely due to the much higher frequency of theme arguments, and the significant overlap of contexts where agents and theme arguments occur.

In the full experiment, the high degree of confusion between variable and thing is not surprising, given that they occur in similar contexts, and humans also tend to disagree on the correct classification. For instance, in example (3) should *the ballast of aggregates* be annotated as a thing, or as a variable meaning something like "the weight of the aggregate ballasts"?

(3) Variability in nutrients gives rise to changes in the ballast of aggregates.

It is not surprising that there is some degree of confusion between the entity categories and the other categories, as entities with large text spans sometimes contain text that could have been annotated as an event if it were not a part of an annotated entity. In example (4), *dissolution* could have given rise to a decrease, but because *calcium carbonate dissolution* is annotated as a variable, the internal event is not annotated. TEES could potentially discover the embedded event, leading to a confusion between variable and a decrease event.

(4) The aeolian process causes enhancement of calcium carbonate dissolution and anammox rates.

5.3.3 Modifying the training data

As explained above, it is believed that entity recognition is particularly difficult for TEES, due to, among other things, the inconsistency in the annotation of entities. A follow-up experiment was therefore conducted to see if performance could be improved by removing all sentences that did not contain any

	Full data			Restricted data		
	Precision	Recall	F-score	Precision	Recall	F-Score
Variable	0.41	0.40	0.40	0.26	0.56	0.35
Thing	0.33	0.14	0.20	0.37	0.32	0.34
Increase	0.74	0.46	0.57	0.62	0.62	0.62
Decrease	0.76	0.39	0.52	0.57	0.49	0.53
Change	0.47	0.16	0.24	0.42	0.46	0.44
Cause	0.11	0.01	0.02	0.29	0.09	0.14
Correlate	0.68	0.13	0.22	0.45	0.10	0.16
And	0.50	0.03	0.03	0.38	0.08	0.14
Or	0.00	0.00	0.00	0.00	0.00	0.00
Macro	0.44	0.19	0.27	0.37	0.30	0.33
Micro	0.50	0.31	0.39	0.40	0.47	0.43

Table 5.7: Evaluation metrics for triggers on full and restricted input data

	Full data			Restricted data		
	Precision	Recall	F-score	Precision	Recall	F-Score
Agent	0.26	0.02	0.04	0.13	0.05	0.07
Theme	0.39	0.19	0.25	0.25	0.25	0.25
Co-theme	0.32	0.06	0.10	0.41	0.10	0.16
Part	0.50	0.02	0.04	0.24	0.05	0.08
Macro	0.37	0.07	0.12	0.26	0.11	0.16
Micro	0.39	0.13	0.19	0.24	0.18	0.21

Table 5.8: Evaluation metrics for arguments on full and restricted input data

annotations, under the belief that this will remove some of the inconsistency in annotation of variables, and by balancing the ratio of positive to negative examples in the training data.

The test data was not modified, in order to give an assessment of the system performance on a realistic task. The results are presented in in tables 5.7 and 5.8, with the performance given full input data given for ease of comparison.

Limiting the training data yields a significantly higher recall at the cost of precision, which is expected, as the reduced number of negative examples encourages detecting more triggers. The f-scores are somewhat better in the limited data experiments, due to the fact that the f-measure favours balanced scores. However, in text mining, textual data is normally abundant, making high precision more important than high recall. It therefore seems that the full training data is preferable to limiting the training data.

5.3.4 Comparing TEES to the PMS

To shed light on research question 4, the performance of TEES, acting as a representative machine learning system, was compared to that of the pattern matching system developed in chapter 4. However, the evaluation criteria used for the evaluation of the pattern matching system does not allow for a direct comparison of the results. To accommodate a reasonable comparison, the output from the pattern matching system was converted to IXML, and the evaluation script originally applied to the output from TEES was applied to it, comparing it to the gold standard annotation of the same papers.

The confusion matrices from this experiment are shown in tables 5.9 and 5.10. The evaluation metrics calculated per category, as well as the micro and macro metrics are presented in tables 5.11and 5.12, with the results from TEES on the full experiment also provided for ease of comparison.

From these tables it can be seen that the pattern matching system clearly outperforms TEES for most trigger categories. A notable exception is *correlate*, where the pattern matching system performs poorly, as has been discussed earlier. It should not be surprising that TEES also achieves better performance on the category *and*, as the pattern matching system ignores coordination categories.

For argument categories, TEES outperforms the pattern matching system on agent and co-theme. This is likely due to the fact that the pattern matching system ignores many types of interactions (for instance interactions where one argument is an entity), and that interactions are only detected between adjacent pairs of change events in the pattern matching system. The pattern matching system can be improved by handling additional types of interactions, but it seems likely that some statistical reasoning is required to find the correct arguments for a detected interaction event trigger.

The pattern matching system exhibits less intra-category confusion than TEES does[3]. There is some confusion between variable and thing, and vice versa, which is to be expected, given the relative overlap between the two categories. There is also a surprisingly common confusion with gold standard variables predicted as decrease. This is likely due to the word *dissolved*, which can trigger a decrease event, but often is annotated as a part of a variable, such as *dissolved organic carbon*. Many predicated variables are matched to gold standards events, but this is most likely due to errors in the scoping of the variables by the pattern matching system.

There are several differences between the approach taken in TEES and the PMS that can shed light on the differences in performance: Whereas the TEES pipeline starts by extracting all triggers, and then tries to connect the different triggers through relations, the PMS starts by focusing on the change events, and only postulate variables and things if a change event is found. The latter approach seems better fitted to our annotation scheme, which ignores isolated entities, whereas the former seems a better match for BioNLP challenges,

[3]It should be noted that there is less data to judge from. Additional types of intra-category confusion could possibly surface with more test data.

	Predicted										
		None	Variable	Thing	Increase	Decrease	Change	Cause	Correlate	And	Or	Sum
True	None	0	67	5	4	6	18	5	0	0	0	105
	Variable	197	209	1	0	6	0	0	0	0	0	413
	Thing	19	2	6	0	0	0	0	0	0	0	27
	Increase	33	2	0	122	0	0	0	0	0	0	157
	Decrease	32	6	0	0	92	0	0	0	0	0	130
	Change	30	2	0	0	0	56	0	0	0	0	88
	Cause	22	1	0	0	0	4	13	0	0	0	40
	Correlate	46	2	0	0	0	0	0	2	0	0	134
	And	37	4	0	0	0	0	0	0	0	0	41
	Or	2	0	0	0	0	0	0	0	0	0	2
	Sum	418	295	12	126	104	78	18	2	0	0	

Table 5.9: Confusion matrix for Triggers, pattern matching system.

		Predicted					
		None	Agent	Theme	Co-theme	Part	Sum
True	None	0	14	185	2	0	301
	Agent	57	4	3	0	0	64
	Theme	294	0	140	0	0	434
	Co-theme	55	0	0	0	0	55
	Part	94	0	0	0	0	94
	Sum	500	18	328	2	0	

Table 5.10: Confusion matrix for Arguments, pattern matching system.

	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
And	0.50	0.03	0.06	0.00	0.00	0.00
Or	0.00	0.00	0.00	0.00	0.00	0.00
Macro	0.44	0.25	0.32	0.61	0.35	0.44
Micro	0.50	0.32	0.39	0.77	0.48	0.60

Table 5.11: Comparison of trigger categories, TEES and PMS.

	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
Part	0.80	0.02	0.04	0.00	0.00	0.00
Macro	0.44	0.04	0.07	0.16	0.10	0.12
Micro	0.43	0.13	0.20	0.31	0.22	0.26

Table 5.12: Comparison of argument categories, TEES and PMS.

where isolated biomedical entities (such as Proteins, Genes, Diseases and Drugs) are also extracted. Furthermore, trigger extraction for entities is easier in the biomedical domain, because the biomedical entities fall into clearly defined semantic categories, whereas the entities in our annotation scheme (quantitative variables) have no single defining semantic characteristic, making it difficult to detect them without the presence of an event to guide the choice.

5.4 Summary and conclusion

The discussion above reveals that there is a strong tendency for confusion among categories that humans tend to disagree over. This is an expected results, as these are cases where there is no clear indicator that can be used by humans or the computer to make a judgement. Furthermore, categories where humans tend to disagree tend to have less consistent annotation, even when there is only a single annotator, making it even harder for the computer to learn the correct behaviour.

It was also observed that performance on the argument type categories is worse than on trigger type categories. This is likely due to the fact that errors are propagated throughout the pipeline, and is an argument against pipelined approaches to information extraction for the task.

There is a tendency for categories with more training examples to have better performance than less frequent categories. This seems to indicate that system performance can be improved by adding more data, but it is uncertain how much improvement can be gained from simply increasing the amount of training data for the system without improving the underlying system qualitatively. If one wants experiment further with using TEES or machine learning systems in general, it might be useful to conduct learning curve experiments that plot system performance as a function of training data size, to estimate how much more data is needed by examining the gradient of the performance function.

TEES is designed to perform a different task than the one required by our annotation scheme, and some of the design decisions are therefore sub-optimal. This is likely the case for most existing information extraction systems. If a machine learning-based approach is to be used for automatic annotation in oceanographic climate science, it is recommended to build a new system that takes the specifics of the task into consideration during design.

Comparing TEES to the pattern matching system developed in chapter 4 reveals that the pattern matching systems outperforms TEES on most categories, but that there are some areas in which the pattern matching system can learn from TEES. There are several reason why this comparison must be taken with a grain of salt: The PMS is very primitive, and some additional development effort is likely to improve its performance noticeably. TEES is not designed for this task, and developing a system specifically with the annotation scheme in mind, is likely to yield better results for a machine learning system. Finally data is limited, and performance of a machine learning system is likely to improve as the amount of available data increases. Nevertheless, this seems to indicate that a pattern matching system performs better than machine learning systems for the domain, but the fact that TEES outperformed the PMS on some categories reveal that there are some aspects of machine learning approaches that should be exploited. Chapter 7.2.1 discusses a system that takes the best from both approaches.

Chapter 6

Domain rules

After automatic annotation, the next step for the information extraction component is to extract domain rules from the annotated text. A script written by Erwin Marsi is applied to extract domain rules. The script extracts domain rules exactly as stated in the annotated text, without any post-processing or extraction of additional information. This chapter will discuss what kind of additional information must be extracted by the domain rule extraction component in order to enable automatic reasoning and discovery support.

6.1 Domain rules and reasoning

Table 6.1 presents a manually picked sample of automatically extracted rules from two of the manually annotated papers. All domain rules extracted from the manually annotated corpus are available at `https://github.com/EliasAamot/MasterData`. In the notation used in this chapter, $\uparrow$ signifies an increase, $\downarrow$ signifies a decrease, and $\updownarrow$ signifies a change, which is underspecified as to direction.

Based on the presented rules, a human expert can postulate a feedback loop using the following argumentation: Increased CO_2 in the ocean leads to increased concentrations of hydrogen ions (1), increased concentrations of hydrogen ions leads to decreased pH (2), decreased pH leads increased dissolution of particulate inorganic carbon ballasting the aggregates (3), which again decreases their settling time (4). Changes in the settling time for these particles leads to increased residence time for the particles in the water column (5), and increased residence times in the water column causes a decrease in the carbon flux to the deep-sea (6), i.e. the biological pump. Therefore, by modus ponens, increased CO_2 in the ocean leads to decreased efficiency of the biological pump, which can be used to infer a feedback loop if the additional domain rule $\downarrow$ *biological pump* $\Rightarrow \uparrow$ *CO_2 in the ocean* can be supplied by the domain expert.

This pattern of reasoning can be replicated automatically by a production system, where the system assumes any change event as the premise, and uses

1. $\uparrow$ CO2 in the ocean $\Rightarrow \uparrow$ [the concentrations of carbon dioxide $\wedge$ carbonic acid $\wedge$ hydrogen ions $\wedge$ bicarbonate ions]
2. $\uparrow$ hydrogen ions $\Rightarrow \downarrow$ pH
3. $\downarrow$ pH $\Rightarrow \uparrow$ the dissolution of particulate inorganic carbon ballasting the aggregates
4. $\uparrow$ the dissolution of particulate inorganic carbon ballasting the aggregates $\Rightarrow$ [$\downarrow$ their settling velocity $\wedge$ $\uparrow$ their residence time in the upper twilight zone]
5. $\updownarrow$ the settling velocity of the particles $\Rightarrow \uparrow$ residence time in the water column
6. $\uparrow$ residence time in the water column $\Rightarrow \downarrow$ the carbon flux to the deep-sea

Table 6.1: A sample of rules extracted from the annotated corpus

domain rules to explore the consequences of the premise event. If the system has the change event X in its working memory (X is postulated as the premise or has been inferred), Y can be inferred if there exists a domain rule $X \rightarrow Y$.

It is not clear exactly how correlations can best be used for reasoning. One possible way is to treat them as bi-directional causation, effectively reading every rule of the form $X \leftrightarrow Y$ as $X \rightarrow Y$and $Y \rightarrow X$. Another possibility is to introduce fuzziness or degree of belief into the production system. Then, if the system believes X with a certainty π, it can infer Y with certainty π if there is a domain rule $X \rightarrow Y$, or with certainty $\pi\eta$ if there is a domain rule $X \Leftrightarrow Y$, where $\eta < 1.00$, so that certainty is decreased when correlations are used for production of new inferences. This latter approach models the fact that correlations do give clues about the consequences of a change, albeit weaker than the clues given by causations.

For successful reasoning along the lines above, additional information must be extracted from the annotated text, as well as other knowledge sources, so that the reasoning component can solve the following three problems: Variable unification, aspect identification and ensuring reasoning validity. Each of these challenges will be discussed in this sub-chapter.

6.1.1 Variable unification

Given that the system knows $\uparrow X_1$, and there is a domain rule $\uparrow X_2 \rightarrow \downarrow Y$, the system must be able to identify the strings X_1 and X_2 as representing the same variable in order to infer $\downarrow Y$. The process of checking for equivalence will be referred to as *variable unification.* By following the chain of reasoning laid out above, one can see that variable unification can be performed by simple string matching until the chaining of rules 4 and 5, where the variables *their settling* *velocity* and *the settling velocity of the particles* must be successfully unified. For most rules, it seems that simple string matching is not sufficient for variable unification.

The unification procedure must take into account syntactic variation in the variables, unifying variables such as *energy dissipation* and *dissipation in light energy*, and be able to ignore irrelevant modifiers, unifying *the availability of light* and *light*, but retain distinguishing modifiers, in order to not incorrectly unify the distinct variables *atmospheric CO2* and *CO2 in the ocean*. Furthermore, the unification procedure must be able to take synonyms into account, unifying *biological pump* and *vertical carbon flux*, and abbreviations, unifying *PIC* and *particulate inorganic carbon*.

The most reliable way to perform variable unification is to map variable text strings to concepts in a domain ontology, and only unify variables that map to the same concept. The problem is that developing a broad-coverage domain ontology is very manpower intensive, especially so if potentially relevant related fields such as atmospheric science are to be covered as well. Combining several ontologies that exists for parts of the domain, such as ontologies over living species, is most likely the best approach. It is conceivable that variable unification can be performed with other techniques as well, for instance by transforming the dependency structures of the variables to a canonical form using a series of operations, and then comparing canonical forms, or by use of some corpus-based similarity measure, but these approaches are likely only worth exploring if creating an ontology proves too expensive.

6.1.2 Aspect identification

Many entities have several quantitative axes that can undergo a change, and identifying exactly which axis is affected is crucial for variable unification. Compare the various aspects of *phytoplankton* (individual size, species diversity, total biomass weight) that are affected in the sentences below (1), even though the variable text span itself is identical. The example below illustrates the tendency of natural language to omit information that can be retrieved from the context. Retrieving this contextual/discourse information resembles in many ways a co-reference resolution task. A special component is likely needed to handle these cases, which are rather common in natural language.

(1) a. …leading to an observed decrease in zooplankton size, but, somewhat counter-intuitively, an increase was observed in phytoplankton.
  b. …leading to an observed decrease in zooplankton species diversity, but, somewhat counter-intuitively, an increase was observed in phytoplankton.
  c. …leading to an observed decrease in zooplankton biomass, but, somewhat counter-intuitively, an increase was observed in phytoplankton.

In addition to information contained in the variable text span, and information extracted implicitly from context, the predicate provides strong clues about which axis of the theme argument is affected, as can be seen in example (2), where variables with identical text spans are clearly affected along different axes (physical size of each coccolith plate vs. number of coccolith plates per plankton). One of the main purposes of the annotation category *thing* was to signal that the text span required information from the predicate to be factored in order to identify it as a variable. However, it appears that the predicate commonly contributes to the interpretation of an entity text span, albeit the degree of contribution varies. It is therefore hard to make a binary distinction between when the predicate contributes to the interpretation of the argument ("thing") and when it doesn't ("variable"). Because of this, the *thing* category is to be phased out in the updated annotation scheme under development, and all cases are annotated as variables. The aspect identification component will then be responsible for combining the information from the various sources (text span, predicate, inferred implicit information) in an optimal way.

(2) a. [VARIABLE Coccoliths] were [INCREASE larger].
b. . . . creating [INCREASE more] [VARIABLE coccoliths].

6.1.3 Sound reasoning

Even if we assume fully functional variable identification and unification module, there are several challenges for sound reasoning in the current event extraction protocol:

Underspecified change

While implicitly encoded information most often causes problems for entity extraction, there is at least one case where event extraction is troubled by implicit information as well, and that is the use of the underspecified *change* when there is a known directionality to the change that is understood by a human reader. This is detrimental to sound reasoning when such change events are extracted as preconditions for domain rules, as this creates a potentially incorrect generalisation of the intended rule. Compare (3-a), the extracted rule given in the examples above, to (3-b), the semantically correct one. The former allows the inference of *increased residence time in the water column* from *increased settling velocity of the particles*, which is incorrect, as an increase in speed will *decrease*, not increase, the time an object stays in a location, i.e. the residence time.

(3) a. $\updownarrow$ the settling velocity of the particles $\Rightarrow$ $\uparrow$ residence time in the water column
b. $\downarrow$ the settling velocity of the particles $\Rightarrow$ $\uparrow$ residence time in the water column

In case of an underspecified change in the consequent of a rule, the soundness of reasoning is not affected, but the output of the rule is less specific than it could have been, thus reducing coverage. Compare (4-a) to the hypothetical wrongly extracted rule in (4-b). The erroneous rule (4-b) yields the conclusion *changed the carbon flux to the deep-sea*, which also is entailed by the conclusion of the correct rule *decreased carbon flux to the deep-sea*, so reasoning is sound but has less coverage. However, in a large database of text, it is likely that the same domain rule is stated multiple times, making completeness less critical than soundness.

(4) a. $\uparrow$ residence time in the water column $\Rightarrow$ $\downarrow$ the carbon flux to the deep-sea
    b. $\uparrow$ residence time in the water column $\Rightarrow$ $\updownarrow$ the carbon flux to the deep-sea

As the discussion above shows, there is a need for a component to disambiguate underspecified *change* events, at least those that occur in the precondition of a rule. While this task might to be quite hard, it would be interesting to see whether a simple statistical machine learning system could attain acceptable performance on this task, casting it as a trinary classification task (increase, decrease or truly bidirectional).

Modality and meta-knowlege

All the sentences in (5) give rise to the domain rule $\downarrow$*pH* $\Rightarrow$ $\downarrow$*primary production* in the present annotation and extraction scheme, but it is clear to a human reader that the degree of support for the rule is quite different between the sentences. Such aspects of modality and meta-knowledge are not handled by the current annotation scheme, but modal aspects of information such as speculation and negation have been recently given some attention by the information extraction community, as evident by the BioNLP 2013 Shared Task, where negation/speculation was one defined sub-task [Nédellec et al., 2013]. Current state-of-the-art performance is 0.25 f-score, which is probably not good enough for real-world application, but hopefully this will improve as the community starts to focus on the task. An annotation scheme and pilot annotation for modal information and meta-knowledge in biomedicine is presented in [Thompson et al., 2011], the approach of which can be replicated if the annotation scheme developed here is to be expanded to incorporate meta-knowledge.

(5) a. Our results indicate that decreased pH negatively affects primary production rates.
    b. We explore the hypothesis that decreased pH negatively affects primary production rates.
    c. The present study contradicts the claim of X et al. that decreased pH negatively affects primary production rates.

Although not a linguistic phenomenon, a related problem to modality is the trustworthiness of the information. A comprehensive treatment of this is clearly outside the scope of OCEAN-CERTAIN, but it might be worthwhile to look into bibliometric heuristics, such as the number of citing papers and the reputation of the journal and author(s).

Modality and/or trustworthiness can potentially be incorporated in the reasoning component envisioned here by incorporating degree of belief in a statement, as was proposed when discussing reasoning with correlations.

Rule context

Contextual information about the applicability of a domain rule is also ignored by the current system. Many of the domain rules extracted by the system only apply given certain contextual restrictions, such as geographic or conceptual location, time period limitations, or assuming the presence of certain contextual variables. Some contextual preconditions are presented in (6). Both reasoning with contextual preconditions and the extraction of these from text are issues that will be addressed in future OCEAN-CERTAIN work.

(6)
- a. [PRECONDITION Under Fe limitation], increased pCO2 had no influence on C fixation whereas [PRECONDITION under Fe enrichment], primary production increased with increasing pCO2 levels.
- b. This is, to the best of our knowledge, the first report on the effect of ocean acidification on sediment oxygen fluxes [PRECONDITION in the Arctic].
- c. We find that, [PRECONDITION during summer and autumn] POC export to the deep ocean was 2-4 times higher than observed for the rest of the year.

6.1.4 Challenges and the annotation scheme

At the moment, the annotation scheme does not take the challenges posed to the reasoning component into account, but focuses on annotating the explicitly stated information only. Incorporating additional information into the annotation scheme, such as speculation, full disambiguation of the *change* category and mapping to a ontology of variable types, will likely prove valuable for the development of tools to overcome the challenge posed to the reasoning/domain rule extraction component.

However, when expanding the annotation scheme it is important to keep the cognitive load into the consideration, so as to not hurt annotation consistency. Some problems of inconsistency can be mitigated by proper training of annotators, and by retaining the option to be less specific when the annotator is not sure about the details, but the cost of expanding the annotation scheme must always be kept in mind.

For a corpus that will be used as data for several systems, it is important to design the annotation scheme in a way that lets each system extract data at the level of specificity it needs. This can easily be done by a structured type hierarchy. That way, if a system architecture is used that contains separate automatic annotation and variable identification components, where the automatic annotation component only extracts the high-level category *variable* because the variable identification component identifies the exact sub-type of variable, then the automatic annotation component can treat all the specific sub-types of *variable*as belonging to the super-type *variable* during system training.

6.2 Towards a full system: PMS and rule extraction

Although experiments show that the pattern matching system is not yet good enough for automatic annotation, and, as this chapter illustrates, the domain rule extraction component is still incomplete, it is interesting to see what kind of results they produce on completely unseen text, yielding an evaluation of the entire system pipeline in its current state.

6.2.1 Experimental set-up

A large collection of paper abstracts was downloaded from Mendeley[1] as all papers returned when querying the Mendeley API with a list of ten keywords deemed relevant to the domain, such as *ocean acidification*, *carbon flux*, *global warming* and *biological pump*. From the collection N=425 papers were randomly sampled. Manual inspection revealed that the disciplines of the sample papers included medicine, oceanography, (earth and ocean) biology and materials science[2]. These domains are all sufficiently close to harbour useful knowledge for oceanographic climate science.

The pattern matching system was applied for automatic annotation of the paper abstracts. No manual editing of system output was performed. The rule extraction script was applied to extract rules from the automatic annotations. The domain rules output by the system were informally analysed to get a feel of system performance.

6.2.2 Results and discussion

Of 425 paper abstracts, the system was able to extract domain rules from 46. While it is likely that a portion of the abstracts don't contain textual material that give rise to a domain rule, it is likely that the low percentage of papers containing domain rules is due to the low recall of the pattern matching system. Some of the extracted domain rules are presented below. All extracted domain rules are available at `https://github.com/EliasAamot/MasterData`.

[1]Mendeley is "a free reference manager and academic social network", that has a large database of paper metadata, including paper abstracts available through a developer API.

[2]The author's limited familiarity with these domains may have caused some misclassification.

- **Domain rule extracted:** $\uparrow$ light-absorbing organic matter in the atmosphere $\Leftrightarrow$ $\uparrow$ C
- Evidence for the atmospheric presence of C(brown) comes from (1) spectral aerosol light absorption measurements near specific combustion sources, (2) observations of spectral properties of water extracts of continental aerosol, (3) laboratory studies indicating the formation of light-absorbing organic matter in the atmosphere, and (4) indirectly from the chemical analogy of aerosol species to colored natural humic substances.

The extracted domain rule is semantically entailed by the sentence, but the distance from surface form to domain rule is quite large, so the rule is most likely correctly extracted due to luck, rather than clever system design.

- **Domain rule extracted:** $\updownarrow$ the biostructure and productivity of lakes $\Rightarrow$ $\updownarrow$ the loading of allochthonous organic carbon
- Large variations in the loading of allochthonous organic carbon (e.g., due to climatic variations) may have considerable effects on the biostructure and productivity of lakes.

The directionality of the causation is wrong, but the change events themselves are correct. This is because of the *due to*, which should connect to *climatic variations*, which also should be extracted as an change event. *The biostructure of lakes* is not a quantitative variable, but *productivity of lakes* is, so the rule does contain some potentially useful knowledge. Conjunction treatment is also required for full rule extraction.

- **Domain rules extracted:**
  - $\uparrow$ lupin and wheat $\Rightarrow$ $\uparrow$ organic matter
  - $\uparrow$ lupin and wheat $\Rightarrow$ $\uparrow$ lupin roots to soil
  - $\uparrow$ lupin roots to soil $\Rightarrow$ $\uparrow$ pH soil
- The response of soil pH change to the addition of organic matter depended on the type of plant materials and starting pH. The net effect of addition of lupin and wheat shoots to acid soils (pH¡5) caused soil pH to increase, the addition of lupin roots to soils caused soil pH to decrease slightly, whilst with a higher pH soil (6.5) the wheat straw and lupin shoots raised pH and pH was unchanged for soil with addition of lupin roots.

The preprocessing component has made an error in sentence splitting, causing two sentences to be merged, lading to guaranteed parsing errors. Multiple domain rules are semantically entailed by the sentences, but none of the extracted rules are correct. It appears that extraction up to change event level is successful, but errors occur in the formation of interaction events, which is not surprising, given the primitive treatment of these events in the pattern matching system, where only two adjacent change events can interact.

In general, the system is able to extract some correct and useful domain rules, but most significant portion of the rules contain semantically wrong or incomprehensible parts. The latter is illustrated in the examples below (7).

(7)
- a. $\uparrow$ Alternatively $\Rightarrow$ $\uparrow$ Equatorial
- b. $\updownarrow$ the nanotube transport $\Rightarrow$ $\updownarrow$ which
- c. $\downarrow$ they $\Rightarrow$ $\uparrow$ the intracellular pH.

This experiment illustrates that inspection of the extracted domain rules is a useful perspective for the evaluation of information extraction systems for literature-based discovery in oceanographic climate science, and also shows that, if improved as discussed in 7.2.1, the pattern matching system will likely be able to support the extraction of useful domain rules.

Chapter 7

Conclusions and further work

This chapter summarizes the main findings in this thesis and identifies directions for future work.

7.1 Conclusions

This thesis has aimed to lay down ground work for developing literature-based discovery systems in oceanographic climate science, focusing mainly on the information extraction component. An annotation scheme was developed targeting the discovery of feedback loops, in response to research goals 1 and 2, and a corpus was manually annotated in accordance to the annotation scheme, fulfilling research goal 3. Subsequently, two approaches to automatic annotation were compared, resulting in two prototype automatic annotation tool as required by research goal 4, although much work still remains before high quality automatic annotation can be achieved. Finally, domain rules were extracted from the automatically annotated text, fulfilling research goal 5, and the perceived challenges to reasoning were identified.

This section summarizes how the research questions have been answered during the course of the thesis work.

What kinds of information should be extracted from the scientific literature in order to support discovery in the domain?

Feedback loops were identified as the discoveries targeted in the domain, and these can be inferred from change events and causal or correlative interactions between these. Sometimes feedback loops can also be extracted directly from text. Chapter 3 describes this in greater detail.

Reasoning over the change events and interactions can be performed manually by inspecting the system output, but preferably some automatic reasoning is incorporated to increase the efficiency and consistency of the discovery process. Automatic reasoning requires extraction of additional information, in order to successfully unify identical variables and ensure sound reasoning. Details of this is given in chapter 6.

What challenges are present for corpus annotation?

Corpus annotation is time consuming, and errors/inconsistencies are likely. Details in the annotation scheme affect time and quality, and it is therefore important to adjust the annotation scheme with this in mind. Keeping the corpus up-to-date with changes in the annotation scheme is also challenging, in terms of manpower costs.

Legal and technical aspects related to acquiring textual material and distributing the corpus are also important to keep in mind, although they were not focused on in this thesis.

Can intelligent tools be developed that improve annotation speed and quality?

Even the primitive pattern matching system developed in chapter 4 was able to yield an at least two-fold increase in annotation speed when used as an intelligent annotation assistant. Although the extent was not quantified, quality was also improved by increasing consistency. More sophisticated automatic annotation tools are likely to yield even larger increases in annotation speed, as the amount of human post-editing required decreases.

How do pattern-matching and machine learning approaches differ in their performance for automatic annotation?

For the annotation scheme developed here, there is an interdependency between entities and events, as well as between interaction and change events, making the annotation scheme a bad fit for a pipeline machine learning system. The pattern matching approach, on the other hand, can be tailored to work well with the annotation scheme in most cases.

There are several cases were probabilistic reasoning is likely to be required, such as when detecting the presence of an implicit causal or correlative relation between two change events, and it is likely that probabilistic reasoning can aid pattern matching on most sub-tasks, for example by weighting the probability that a match for a given pattern is a false positive against the likelihood that the proposed argument is a variable.

Joint learning might work well for the annotation scheme, given enough data, but it is believed that a pattern matching system that incorporates some probabilistic reasoning should be able to outperform any pure machine learning system.

What are the expected challenges for reasoning with the domain rules?

The extracted domain rules do not contain sufficient information to for automatic reasoning. For instance, the system is unable to perform variable unification of non-identical strings, making the system unable to chain domain rules like $\uparrow X \rightarrow\downarrow Y_1$ and $\downarrow Y_2 \rightarrow\uparrow Z$, if the strings $X_1$and X_2 are different even though the represent the same variable.

Additional components must be developed to gather the additional information required for successful reasoning, either from the scientific literature, from domain knowledge resources such as ontologies, or by questioning the human user.

It is also uncertain how to best handle correlations in the reasoning component, and whether to include modality/meta-knowledge and trustworthiness into the reasoning component.

7.2 Future work

This section points out recommended directions for future research and engineering effort based on the findings in this thesis. The most logical next step is the development of a high quality information extraction system. Section 7.2.1presents a description of the proposed system. Section 7.2.2 presents future work towards the development of reasoning and discovery support components. Finally, section 7.2.3mention some other areas that could benefit from future work.

7.2.1 Architecture for an IE system

Based on what has been learned from the experiments made in this thesis, I recommend the development of a hybrid, but primarily pattern matching-based, information extraction system with the following characteristics:

The system should be able to extract patterns from data automatically, following the protocol described in chapter 4.1.1, and assign a strength score to each pattern based on the performance of the pattern on the annotated corpus. Separately, a machine learning component could be used to learn the likelihood that an NP represents an entity. The plausibility of this was illustrated by the fact that TEES was able to learn, to a certain extent, entity triggers. Patterns should only apply if the combined score of the pattern and the variable exceed a certain threshold[1]. The largest weight should likely be put on the pattern, as event triggers are easier to classify correctly than entity triggers, as illustrated by the TEES experiment. The threshold and weighting should preferably be learned from data.

[1]In reality, different change event triggers tend to take different types of entity arguments, but building a model that takes this into account is likely infeasible with limited data.

To avoid constructing event heaps in such a system, the most highly ranked event should be chosen in case of a conflict, where the ranking metric is the combination of pattern and variable strength.

Interaction events have been shown to be more difficult to learn, and are very diverse. It is likely beneficial to keep the current surface matching paradigm more or less without any changes. However, a probabilistic component should be introduced to choose which change event is taken as argument if there are multiple possible change events.

It is most likely necessary to introduce separate structure matching components for some types of interaction events, namely interaction events that hold between two variables and interaction events that hold between one variable and one change event. The former should be modelled as a structure matching pattern with two distinct end-anchors, the latter can be modelled as a structure matching pattern with one argument, the variable, whereupon a separate component connects it to its change event argument. As stated earlier, it is also likely that single preposition triggers for correlations will need structure matching patterns to avoid over-application.

Although this is just a speculation, I believe that a machine learning component can be trained to successfully detect whether an implicit interaction holds between two change events. The event triggers and dependency chain between the triggers seem like good features for such a task.

Feedbacks can easily be included in the same way as interactions between variables.

Coordination poses several problems for the current system, and accounting for these is required in a complete system. Theoretically, this can be done without adding any new patterns by making some procedural changes to the pattern matching engine, but in many coordinate cases the parser performance is not good enough to yield the desired results, so improving the parser or restructuring the output in some way may be required. That, however, is an entirely different research area.

The observant reader will remark that the architecture proposed above is not fully joint, but I believe that no significant reduction in performance will be caused by this: The strength of a variable is taken into account during change event detection, so these categories are already modelled jointly. Interaction events are not modelled jointly with change events, but this corresponds with the nature of annotation scheme, in which strong interaction event triggers are ignored unless they actually connect change events, so information about interaction events is not going to help detection of change events.

Referring expressions are quite problematic. Due to the specialised use of referring expressions in the annotation scheme, it is unlikely that any existing coreference resolution tool performs well on the task. A system would therefore have to be developed from scratch. How the output from this system can best be combined with the other components is a matter requiring investigation, as referring expressions are not annotated independently.

7.2.2 Reasoning

A potential design for the reasoning component was proposed, where reasoning is modelled as a production system. This is in no way the only approach that can be taken. A modified version of discovery browsing (as described in chapter 2.1.10) can for instance be used, and feedback loops can be detected in the graph space by searching for cycles in the graph, for instance by a simple breath-first search. The domain rules can also be formulated in first-order logic, and a theorem prover can be used to look for feedback loops. Comparing the weaknesses and strengths of the different reasoning systems is the most central research goal for reasoning.

As mentioned earlier, open research questions also include how to best handle correlations during reasoning, whether to incorporate trustworthiness or model information during reasoning, how to best perform variable unification, aspect identification and disambiguation of rule context, and how to best extract and use information about domain rule context requirements.

7.2.3 Other work

The experiments show that performance on higher level systems is inevitably linked to the performance of lower level systems. Parser performance is a noteworthy example of this. A systematic evaluation of parser performance in the domain is therefore likely to be worthwhile, as choosing the optimal parser can improve performance for all systems downstream in the pipeline. It is also worth looking into domain adaptation of parsing models, or, if possible, the development of new parsing tool for the domain, as has been done in biomedicine with significant results [Miyao et al., 2009].

Bibliography

Ahlers, C. B., Hristovski, D., Kilicoglu, H., and Rindflesch, T. C. (2007). Using the literature-based discovery paradigm to investigate drug mechanisms. *AMIA ... Annual Symposium proceedings / AMIA Symposium. AMIA Symposium*, pages 6–10.

Aronson, A. R. (2006). Metamap: Mapping text to the umls metathesaurus. Technical report, National Library of Medicine.

Aronson, A. R. and Lang, F.-M. M. (2010). An overview of MetaMap: historical perspective and recent advances. *Journal of the American Medical Informatics Association : JAMIA*, 17(3):229–236.

Björne, J., Airola, A., Pahikkala, T., and Salakoski, T. (2011). Drug-Drug Interaction Extraction from Biomedical Texts with SVM and RLS Classifiers. In *Proceedings of the 1st Challenge Task on Drug-Drug Interaction Extraction*, pages 35–42, Huelva, Spain.

Björne, J., Heimonen, J., Ginter, F., Airola, A., Pahikkala, T., and Salakoski, T. (2009). Extracting Complex Biological Events with Rich Graph-Based Feature Sets. In *Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task*, pages 10–18, Boulder, Colorado. Association for Computational Linguistics.

Björne, J. and Salakoski, T. (2011). Generalizing Biomedical Event Extraction. In *Proceedings of BioNLP Shared Task 2011 Workshop*, pages 183–191, Portland, Oregon, USA. Association for Computational Linguistics.

Cairelli, M. J., Miller, C. M., Fiszman, M., Workman, T. E., and Rindflesch, T. C. (2013). Semantic MEDLINE for discovery browsing: using semantic predications and the literature-based discovery paradigm to elucidate a mechanism for the obesity paradox. *AMIA ... Annual Symposium proceedings / AMIA Symposium. AMIA Symposium*, 2013:164–173.

Cameron, D., Bodenreider, O., Yalamanchili, H., Danh, T., Vallabhaneni, S., Thirunarayan, K., Sheth, A. P., and Rindflesch, T. C. (2013). A graph-based recovery and decomposition of swanson's hypothesis using semantic predications. *J. of Biomedical Informatics*, 46(2):238–251.

Charniak, E. and Johnson, M. (2005). Coarse-to-fine n-best parsing and maxent discriminative reranking. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 173–180, Stroudsburg, PA, USA. Association for Computational Linguistics.

Cohen, T., Widdows, D., Schvaneveldt, R. W., Davies, P., and Rindflesch, T. C. (2012a). Discovering discovery patterns with predication-based Semantic Indexing. *Journal of Biomedical Informatics*, 45(6):1049–1065.

Cohen, T., Widdows, D., Vine, L., Schvaneveldt, R., and Rindflesch, T. C. (2012b). Many Paths Lead to Discovery: Analogical Retrieval of Cancer Therapies. In Busemeyer, J., Dubois, F., Lambert-Mogiliansky, A., and Melucci, M., editors, *Quantum Interaction*, volume 7620 of *Lecture Notes in Computer Science*, pages 90–101. Springer Berlin Heidelberg.

Cory, K. (1997). Discovering Hidden Analogies in an Online Humanities Database. In *Computers and the Humanities.*

Cutting, D., Kupiec, J., Pedersen, J., and Sibun, P. (1992). A practical part-of-speech tagger. In *Proceedings of the Third Conference on Applied Natural Language Processing*, ANLC '92, pages 133–140, Stroudsburg, PA, USA. Association for Computational Linguistics.

de Marneffe, M.-C. and Manning, C. D. (2008). The stanford typed dependencies representation. In *Coling 2008: Proceedings of the Workshop on Cross-Framework and Cross-Domain Parser Evaluation*, CrossParser '08, pages 1–8, Stroudsburg, PA, USA. Association for Computational Linguistics.

Digiacomo, R. A., Kremer, J. M., and Shah, D. M. (1989). Fish-oil dietary supplementation in patients with Raynaud's phenomenon: A double-blind, controlled, prospective study. *The American Journal of Medicine*, 86(2):158–164.

Gordon, M., Lindsay, R. K., and Fan, W. (2001). Literature Based Discovery on the World Wide Web. In *ACM Transactions on Internet Technology*, pages 261–275, New York, USA. ACM Press.

Gordon, M. D. and Dumais, S. (1998). Using latent semantic indexing for literature based discovery. *J. Am. Soc. Inf. Sci.*, 49(8):674–685.

Gordon, M. D. and Lindsay, R. K. (1996). Toward discovery support systems: a replication, re-examination, and extension of Swanson's work on literature-based discovery of a connection between Raynaud's and fish oil. *J. Am. Soc. Inf. Sci.*, 47(2):116–128.

Hristovski, D., Friedman, C., Rindflesch, T. C., and Peterlin, B. (2006). Exploiting semantic relations for literature-based discovery. *AMIA ... Annual Symposium proceedings / AMIA Symposium. AMIA Symposium*, pages 349–353.

Hristovski, D., Friedman, C., Rindflesch, T. C., and Peterlin, B. (2008). Literature-Based Knowledge Discovery using Natural Language Processing. In Bruza, P. and Weeber, M., editors, *Literature-based Discovery*, volume 15 of *Information Science and Knowledge Management*, chapter 9, pages 133–152. Springer, Heidelberg, Germany.

Hristovski, D., Kastrin, A., Peterlin, B., and Rindflesch, T. C. (2010). Combining Semantic Relations and DNA Microarray Data for Novel Hypotheses Generation. In Blaschke, C. and Shatkay, H., editors, *Linking Literature, Information, and Knowledge for Biology*, volume 6004 of *Lecture Notes in Computer Science*, pages 53–61. Springer Berlin Heidelberg.

Hristovski, D., Stare, J., Peterlin, B., and Dzeroski, S. (2001). Supporting discovery in medicine by association rule mining in Medline and UMLS. *Studies in health technology and informatics*, 84(Pt 2):1344–1348.

Ijaz, A. Z., Song, M., and Lee, D. (2009). Mkem: A multi-level knowledge emergence model for mining undiscovered public knowledge. In *Proceedings of the Third International Workshop on Data and Text Mining in Bioinformatics*, DTMBIO '09, pages 51–58, New York, NY, USA. ACM.

Kim, J.-D., Ohta, T., Pyysalo, S., Kano, Y., and Tsujii, J. (2009). Overview of bionlp'09 shared task on event extraction. In *Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task*, pages 1–9, Boulder, Colorado. Association for Computational Linguistics.

Klein, D. and Manning, C. D. (2003). Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 423–430, Stroudsburg, PA, USA. Association for Computational Linguistics.

Kostoff, R. N. (2007). Validating discovery in literature-based discovery. *Journal of Biomedical Informatics*, 40(4):448–450.

Leaman, R. and Gonzalez, G. (2008). Banner: An executable survey of advances in biomedical named entity recognition. In *In Pac Symp Biocomput.*

Lee, S., Choi, J., Park, K., Song, M., and Lee, D. (2011). Inferring hidden relationships from biological literature with multi-level context terms. In *Proceedings of the ACM Fifth International Workshop on Data and Text Mining in Biomedical Informatics*, DTMBIO '11, pages 27–34, New York, NY, USA. ACM.

Lindsay, R. K. and Gordon, M. D. (1999). Literature-based discovery by lexical statistics. *Journal of the American Society for Information Science*, pages 574–587.

Marsi, E., Ožtürk, P., and Aamot, E. (2014a). Deliverable d1.1: Literature-based knowledge discovery in climate, marine and environmental science. Technical report, NTNU.

Marsi, E., Öztürk, P., Aamot, E., Sizov, G., and Ardelan, M. V. (2014b). Towards text mining in climate science: Extraction of quantitative variables and their relations. In *Proceedings of the Fourth Workshop on Building and Evaluating Resources for Health and Biomedical Text Processing.*

McClosky, D. and Charniak, E. (2008). 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, pages 101–104, Stroudsburg, PA, USA. Association for Computational Linguistics.

McCray, A. T., Aronson, A. R., Browne, A. C., Rindflesch, T. C., Razi, A., and Srinivasan, S. (1993). UMLS knowledge for biomedical language processing. *Bulletin of the Medical Library Association*, 81(2):184–194.

Miyao, Y., Sagae, K., Saetre, R., Matsuzaki, T., and Tsujii, J. (2009). Evaluating contributions of natural language parsers to protein-protein interaction extraction. *Bioinformatics (Oxford, England)*, 25(3):394–400.

Nédellec, C., Bossy, R., Kim, J.-D., Kim, J.-J., Ohta, T., Pyysalo, S., and Zweigenbaum, P. (2013). Overview of bionlp shared task 2013. In *Proceedings of the BioNLP Shared Task 2013 Workshop*, pages 1–7, Sofia, Bulgaria. Association for Computational Linguistics.

Pratt, W. and Yetisgen-Yildiz, M. (2003). LitLinker: capturing connections across the biomedical literature. In *Proceedings of the 2nd international conference on Knowledge capture*, K-CAP '03, pages 105–112, New York, NY, USA. ACM.

Schuemie, M. J., Weeber, M., Schijvenaars, B. J., van Mulligen, E. M., van der Eijk, C. C., Jelier, R., Mons, B., and Kors, J. A. (2004). Distribution of information in biomedical abstracts and full-text publications. *Bioinformatics*, 20(16):2597–2604.

Smalheiser, N. R. (2012). Literature-based discovery: Beyond the ABCs. *J. Am. Soc. Inf. Sci.*, 63(2):218–224.

Smalheiser, N. R., Torvik, V. I., and Zhou, W. (2008). Arrowsmith two-node search interface: a tutorial on finding meaningful links between two disparate sets of articles in MEDLINE. *Computer methods and programs in biomedicine*, 94(2):190–197.

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

Swanson, D. R. (1988). Migraine and magnesium: eleven neglected connections. *Perspectives in Biology and Medicine*, 31(4):526–557.

Swanson, D. R. and Smalheiser, N. R. (1997). An interactive system for finding complementary literatures: a stimulus to scientific discovery. *Artificial Intelligence*, 91(2):183–203.

Thompson, P., Nawaz, R., McNaught, J., and Ananiadou, S. (2011). Enriching a biomedical event corpus with meta-knowledge annotation. *BMC Bioinformatics*, 12(1):393+.

Weeber, M., Klein, H., de Jong van den Berg, L. T., and Vos, R. (2001). Using concepts in literature-based discovery: Simulating Swanson's Raynaud-fish oil and migraine-magnesium discoveries. *Journal of the American Society for Information Science and Technology*, 52(7):548–557.

Wilkowski, B., Fiszman, M., Miller, C. M., Hristovski, D., Arabandi, S., Rosemblat, G., and Rindflesch, T. C. (2011). Graph-based methods for discovery browsing with semantic predications. *AMIA ... Annual Symposium proceedings / AMIA Symposium. AMIA Symposium*, 2011:1514–1523.

Wren, J. D. (2004). Extending the mutual information measure to rank inferred literature relationships. *BMC bioinformatics*, 5.

Wren, J. D., Bekeredjian, R., Stewart, J. A., Shohet, R. V., and Garner, H. R. (2004). Knowledge discovery by automated identification and ranking of implicit relationships. *Bioinformatics (Oxford, England)*, 20(3):389–398.

Yetisgen-Yildiz, M. and Pratt, W. (2006). Using statistical and knowledge-based approaches for literature-based discovery. *Journal of Biomedical Informatics*, 39(6):600–611.

Yetisgen-Yildiz, M. and Pratt, W. (2009). A new evaluation methodology for literature-based discovery systems. *Journal of biomedical informatics*, 42(4):633–643.