NEGATION TRIGGERS AND THEIR SCOPE

Sabine Rosenberg

A thesis
in
The Department
of
Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements
For the Degree of Master of Computer Science
Concordia University
Montréal, Québec, Canada

September 2013

CONCORDIA UNIVERSITY
School of Graduate Studies

This is to certify that the thesis prepared

By: **Sabine Rosenberg**
Entitled: **Negation Triggers and their Scope**

and submitted in partial fulfillment of the requirements for the degree of

Master of Computer Science

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining commitee:

_____ Chair
Dr.

_____ Examiner
Dr. Eusebius Doedel

_____ Examiner
Dr. Leila Kosseim

_____ Thesis Supervisor
Dr. Sabine Bergler

Approved _____
Chair of Department or Graduate Program Director

_____ 2013 _____
Dr. Christopher Trueman, Dean
Faculty of Engineering and Computer Science

Abstract

Negation Triggers and their Scope

Sabine Rosenberg

Recent interest in negation has resulted in a variety of different annotation schemes for different application tasks, several vetted in shared task competitions. Current negation detection systems are trained and tested for a specific application task within a particular domain. The availability of a robust, general negation detection module that can be added to any text processing pipeline is still missing. In this work we propose a linguistically motivated trigger and scope approach for negation detection in general. The system, NEGATOR, introduces two baseline modules: the scope module to identify the syntactic scope for different negation triggers and a variety of trigger lists evaluated for that purpose, ranging from minimal to extensive. The scope module consists of a set of specialized transformation rules that determine the scope of a negation trigger using dependency graphs from parser output. NEGATOR is evaluated on different corpora from different genres with different annotation schemes to establish general usefulness and robustness. The NEGATOR system also participated in two shared task competitions which address specific issues related to negation. Both these tasks presented an opportunity to demonstrate that the NEGATOR system can be easily adapted and extended to meet specific task requirements. The parallel, comparative evaluations suggest that NEGATOR is indeed a robust baseline system that is domain and task independent.

Acknowledgments

This thesis could not have been completed without the patience and never ending encouragement from my supervisor Professor Sabine Bergler. Of course, I am entirely grateful to my husband Elio Bidinost for his unflappable support and for encouraging me to embark on this journey one calls "graduate school". I would also like to thank Michelle Khalifé for her positive energy and sound advice along the way. A big thank you to all my colleagues in the CLaC lab for their support and good cheer.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Negation is a linguistic phenomenon that has often been reduced to a logic one: negation in logic reverses the truth value of a proposition (1a) into (1b). The unary operator ¬ triggers this behaviour and it is unambiguous over which proposition its influence extends, i.e its *scope* (underlined).

(1) *(a)* $flat(earth) \equiv false$

(b) $\neg \underline{flat(earth)} \equiv true$

(c) $\neg\underline{(\neg\ flat(earth))} \equiv flat(earth) \equiv false$

The following strict laws of logic further characterize a negated logic proposition (Horn, 1989):

1. **Law of Contradiction (LC)**: a statement cannot be both true and false at the same time.
2. **Law of Excluded Middle (LEM)**: a statement must be either true or false.
3. **Law of Double Negation**: allows the ¬ operator to cancel itself out without any effect on the proposition under its scope (1c).

In the field of linguistics, negation also has an explicit negation operator (i.e. *not*). As in logic, its domain of influence has to be clearly demarcated. (Huddleston and Pullum, 2002, pg 792) describe the *scope of negation* as the relevant components of the sentence which are under the semantic influence of a negation operator - the lexical items which contribute to the determination of the truth value of the proposition. In natural language the most direct translation from logical negation would be *sentence negation*, where the clearest phrasing would be "it is not the case that [proposition] ". Example (2) demonstrates how the negated proposition in (2a) may be translated into natural language (2b).

(2) *(a)* $\neg$ *flat(earth)*

(b) *It is not the case that the earth is flat.*

In natural language, the determination of negation scope is not simply a matter of negating the entire sentence. Rather, there are often components of a given sentence which are not under the influence of the negation operator. Example (3) from (Huddleston and Pullum, 2002, pg 793) demonstrates that depending on how the scope of a negation is allocated, the resulting interpretation is different. The affirmative statement is stated in (3a). (3b) and (3c) present two possible determinations for the negation scope (underlined). The resulting meanings of the two negated statements are very different. In (3b), the statement implies that Liz did not delete the backup file and had the intention not to. Whereas in (3c), Liz did delete the file but not intentionally. Example (3) shows that the different positions of the adjunct *intentionally*, relative to the negation operator *n't* results in a different scope of negation. Consequently, expressions of negation in natural language are not so syntactically simple: the scope is not as obvious as in logic.

(3) (a) Liz intentionally deleted the backup file

(b) Liz intentionally did[n't] delete the backup file.

(c) Liz did[n't] intentionally delete the backup file.

It is well documented by experts in the field of linguistics (Horn, 1989; Huddleston and Pullum, 2002; Quirk et al., 1985; Givón, 1993) to name a few, that the syntactic structure of the sentence is a fundamental basis for determining the scope of a negation operator. In fact, (Huddleston and Pullum, 2002) dedicate over 60 pages to describing the well established syntactic patterns which help to identify the negation scope. *Verbal negation* is the most common variant in English and typically occurs when a negation operator i.e. *not* is grammatically associated with the main verb in a sentence. An example of verbal negation is illustrated in (4). The negation operator *not* (in square brackets) is associated with the main verb *kill* and the resulting scope of negation is the verb phrase (underlined). There are quite a few other variants of negation, which are described in detail later in the thesis. The common thread amongst all of them is that they can be identified by the inherent structural features in a sentence.

(4) John did [not] kill the goat.

At first glance, one might assume that interpreting what a given negation in natural language *expresses* could be reduced to finding the negation operator, detecting the scope using syntactic analysis and reversing the polarity of all items within the identified negation scope. Actually it is far more complex. The following three scenarios demonstrate instances where the interpretation of the negation is not straightforward:

A sentence containing a negation operator does not necessarily follow the traditional rules of logic. As illustrated in (5) the law of double negation does not necessarily apply: (5 b) does not necessarily equate to (5 a).

(5) (a) She is happy.

(b) She is not unhappy.

Negation in natural language need not just be an operator whose function is to reverse the polarity of the statement as demonstrated in the next two examples. Negation can express *less than* or *in between* when used in a scalar context as illustrated in (6). The interpretation of this sentence could easily mean that John has either one or two children.

(6) John does not have three children.

Negation can also be used as a *contrastive* device i.e. where the negation is used to disagree or emphasize a part of the statement and not actually negate it. As seen in (7), the place is defined as *massive*, and therefore it is also *big*.

(7) That place is not big, it is massive.

Interpreting what a given negation in a sentence *means* as illustrated in (5)-(7) can only be determined after the correct determination of negation terms and their corresponding scope. The research in this thesis focuses on this preliminary phase which occurs before any semantic interpretation, and therefore the semantics of these issues is beyond the scope of the thesis.

Another important issue to consider is that in natural language, there are multiple forms of negation operators which may or may not imply existence denied. At the highest structural level, as defined by (Givón, 1993), negations in natural language occur in two forms: morphological negations where the root of a word is modified by a negating prefix (i.e. *dis-, non-, in-, un-*) or suffix (*-less*), and syntactic negation where clauses are negated by explicit operators (i.e. *not, never, no, without*) or implicit negations: verbs and nominalizations which imply a negative context in their complements (i.e. *failed, prevented*).

In this thesis, these explicit markers of negation are referred to as *negation triggers*. Givón points out that "more than just logic must be at issue" to explain the triad: (8 a-c), where (8 b) is not synonymous with (8 c) (Givón, 2001, p370). We add the implicit negation (8 d), where the matrix verb lexically encodes the negation of the complement. Givón's observation means that all these different forms of negation are functionally different and that different applications may have to treat them differently in order to capture the subtle variations in meaning and interpretation.

(8) (a) I am happy.

(b) I am not happy.

(c) I am unhappy.

(d) I miss being happy.

Negation is a frequent phenomenon in text. (Tottie, 1991) reports that negations are twice as frequent in spoken text (27.6 per 1,000 words) as in written text (12.8 per 1,000 words). (Elkin et al., 2005) find that 12% of the concepts in 41 health records are identified as negated by annotators. (Nawaz et al., 2010) report that more than 3% of the BioMedical abstracts of the GENIA (Kim et al., 2008) are negated. (Councill and Velikovich, 2010) annotate a corpus of product reviews with negation information and find that 19% of the sentences contain negations. Information Extraction systems often face the issue of being able to determine if textual information can be classified as affirmative, negated, or speculative. For example, sentiment analysis systems need to detect negation for accurate polarity classification. Similarly, medical information extraction systems need to differentiate between affirmed, negated, and speculated medical conditions. It is no wonder then, that the automatic detection of varying types of negation phenomena is a problem encountered in a wide variety of document understanding tasks, including but not limited to medical data mining, general fact or relation extraction, question answering and sentiment analysis.

Advances in negation detection have been most evident within the BioNLP domain. Two influential annotation efforts are the GENIA event corpus (Thompson et al., 2011) and BioScope (Szarvas et al., 2008). GENIA is a careful annotation effort of a selection of domain relevant events defined for a particular task description. It annotates *negative events*, which include *down regulation*, but do not cover all linguistic negations, if they are not judged to be of importance for the task described. BioScope, in turn, annotates negation and speculative language more generally, however, still uses biomedical journal data. While the BioScope negation annotations do transfer to other genres, the data is domain specific and thus does not extend well to texts from other genres for statistical systems. It is not clear whether negation systems developed exclusively on the BioScope corpus would transfer successfully to other domains. A more recent negation annotation effort outside of the BioMedical Domain was accomplished by (Morante and Daelemans, 2012a). This corpus consists of two Conan Doyle stories[1] (*The Hound of the Baskervilles* and *The Adventure of Wisteria Lodge*). Importantly, the availability of this corpus allows for negation systems to be developed and tested for yet another text genre.

The premise of this thesis is that any treatment of negation in natural language has to address both the determination of negation scope and the consideration of multiple forms of negation triggers. We believe that these two tasks benefit from being addressed independently but also in context of each other. Thus, NEGATOR, the negation system presented in this thesis, introduces baseline versions for each of the two essential ingredients: the scope module to identify the linguistically motivated scope for different negation triggers and a variety of trigger lists evaluated for that purpose, ranging from minimal to extensive.

[1]Website of the Conan Doyle corpus: http://www.clips.ua.ac.be/BiographTA/corpora.html

Three main types of triggers are considered: *explicit* triggers like *not* and *no* are words that indicate negation only; *implicit* triggers like *fail to* and *absence of* which lexically encode negative polarity together with other lexical semantics; and *affixal* triggers like *insufficient* and unaffected, that encode negative polarity but with idiosyncratic scope.

The NEGATOR scope detection module presented in this thesis is intended to be a robust, domain independent and linguistically motivated approach to negation detection. It therefore consists of a set of heuristics that determine the scope of a negation trigger by identifying general, well established syntactic patterns. These syntactic patterns are acquired from the *constituent trees* and *collapsed syntactic dependency relations* (de Marneffe et al., 2006) made available when parsing a given sentence using i.e. the Stanford Lexicalized Parser (Klein and Manning, 2003). Constituent trees and dependency relations are based on the *Phrase Structure* and *Dependency Structure* formalisms.

Phrase Structures: The phrase structure formalism, introduced by Chomsky (1957) is based on a set of rewrite rules, which when applied, construct a syntactic tree representing a grammatical sentence. The resultant syntactic tree is a data structure originating from *terminal* nodes (the leaves) and concluding in the *root* node. A simple sample grammar shown in Table 1 allows symbols on the right side of the arrows to be combined into the ones on the left side. A combination of symbols is known as a phrase or *constituent*; phrase labels can be found on either side of an arrow for a given grammar rule, rendering the grammar recursive.

NP	→	NNP		
NP	→	DT	NNS	
VP	→	VBZ	NP	
S	→	NP	VP	PU

Table 1: Example Set of Grammar Rules

Consider the sentence "John has no enemies." In order to analyze its syntax, the first step is to acquire the part of speech of each word, a first level of abstraction which maps to the symbols necessary to combine together according to the rules in Table 1. The recursive application of these rules can be consequently represented by a syntactic tree shown in Figure 1.

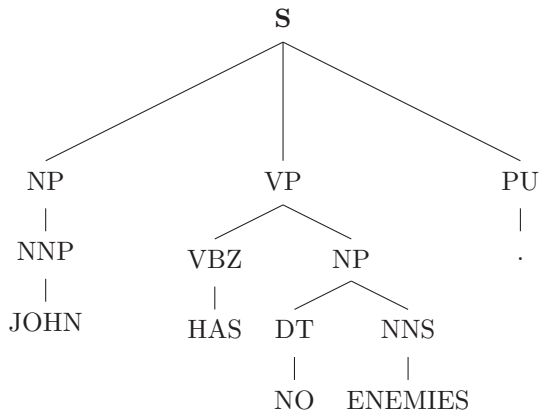

Figure 1: Syntactic Tree for " John has no enemies."

Dependency Structures: The dependency structure formalism differs from phrase structures in that a sentence is represented as a collection of *dependency* relations between single words (Mel'čuk, 1988). These dependencies are generally typed grammatical relations, such as *direct object*, *nominal subject*, *adverb modifier* etc.... In this representation, a dependency relation is formalized as binary, directed grammatical relationship involving two words: *the head and the dependent*. Every *dependent* is allowed exactly one *head*. Consequently, a sentence is represented as a graph, where the nodes correspond to the words and the edges correspond to the *dependency* relations between them.

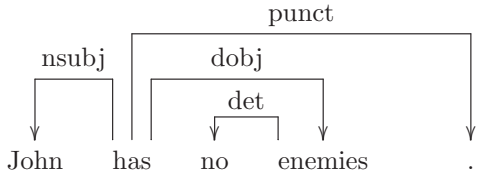

Figure 2: Dependency graph representation of "John has no enemies."

Continuing with the example sentence "John has no enemies.", the resulting graph is demonstrated Figure 2. Four dependency relations are observed: *nsubj(has,John), dobj(has, enemies)* and *punct(has,.)* with the main verb as the head node. In the relation *det(enemies, no)*, *enemies* is the head node. In practice, these dependency relations are automatically extracted from the phrase structure parse trees by the parser. The Stanford dependency scheme (de Marneffe et al., 2006) contains a total of 55 grammatical relations. Each dependency relation is written as: *abbreviated_dep_name(governor, dependent)*. The *govenor* corresponds to the more generic term *head* node. The Stanford dependency parser provides dependency output of one of four types: *basic, collapsed, collapsed with propagation of conjunct dependencies, and collapsed preserving a tree structure*. The *basic* format outputs a tree structure, meaning that there are no crossing dependencies (each word in the sentence

is a dependent of one word). In the *collapsed* representation, dependencies involving prepositions, conjuncts, as well as information about the referent of relative clauses are collapsed in order to get direct dependencies between content words. For example, given the phrase *based in Montreal*, the resulting dependencies in the *basic* representation are: *prep(based,in) and pobj(in Montreal).* In contrast, the *collapsed* format will *collapse* the two relations into one single relation: *prep_in(based,Montreal).* In this representation, a directed graph is output, since additional dependencies are considered which do not abide by the tree structure. The *collapsed with propagation* option propagates conjunct relations. In the *collapsed preserving a tree structure* representation, those dependencies that break the tree are removed. The NEGATOR scope module presented in this thesis uses the *collapsed* representation[2].

These dependency relations indicate only a local notion of scope: the direct dependency between the *governor* and the *dependent*. For many applications, this notion of *local* scope is not useful as it does not necessarily capture the full extent of the negation scope. Rather, capturing all the relevant syntactic constituents from the parse tree proves to be a more productive option. The NEGATOR scope heuristics are designed according to this second notion, to extract the relevant constituents, rather than single terms. Specifically, once a scope heuristic has identified a particular dependency relation, it will then use this dependency relation in order to determine which constituents in the associated parse tree correspond to the negation scope.

Surprisingly, this approach is not the common basis of work on negation. This thesis demonstrates its effectiveness for negation and evaluates it on different corpora from different genres to establish general usefulness and robustness. These corpora used for evaluation, differ not only in terms of domain, but also in terms of the definition of the extent of the negation scope. This parallel, comparative evaluation suggests that NEGATOR is indeed robust, domain and task independent.

A related notion to negation scope is *focus*, which is described as the part of the scope that is more prominently or explicitly influenced by the negation trigger (Huddleston and Pullum, 2002). The question that arises when considering the *focus of negation* is what is the intended opposition in Example (9)?

(9) I didn't [get that book from Mary].
¬ `get(I, book, from Mary)`

The negation trigger is *not*, the scope of the negation is the entire verb phrase (in square brackets), but which aspect of the verb phrase is definitely *intended* to be interpreted as false, that is which of the following statements in (10) is most likely entailed?

[2]Appendix A contains the definitions for the dependency relations used in this thesis.

(10) (a) *Focus:* from Mary

`have(I, book)` $\wedge$ `source(book,` $\neg$ `Mary)`

(b) *Focus:* that book

$\neg$ `have(I, book)` $\wedge$ `have(book, Mary)`

The two possible entailments are both valid depending on the interpretation. Here, if the focus is (10 a): *from Mary*, it would be likely that the speaker has possession of the book, but received it some other way. If the focus is (10 b): *that book*, the speaker does not have possession of it. Usually, context is necessary to determine focus. This notion of focus is not syntactically determined as shown in (10) but pragmatically and it correlates with pronunciation stress, as discussed in linguistics by (Han and Romero, 2001). The difference of scope and focus of negation are elaborated by (Partee, 1993), and have been used for computational use by (Blanco and Moldovan, 2011).

The NEGATOR trigger-scope module has subsequently been used in two shared tasks which address specific issues related to negation. The *SEM 2012 pilot task on *Detecting the Focus of Negation* was dedicated to identifying the *focus* element contained in the scope span of a negation trigger. The QA4MRE pilot task on *Processing Modality and Negation for Machine Reading* (Morante and Daelemans, 2012b) at CLEF 2012 involved detecting negated and modalized[3] events. Both these tasks presented an opportunity to not only further assess the performance of the NEGATOR system in a formalized setting, but to also extend its capabilities to meet the specific task requirements and thus show its adaptability.

[3]if an event or situation is determined to not be certain nor factual then it is modalized.

Chapter 2

Related Work

2.1 Foundational Work on Negation

Identifying negation in text has always been considered an important support task within the biomedical and sentiment and opinion analysis domains. The early negation detection methods considered negation phenomena mainly from a task and domain driven perspective. As there were no publicly available gold standards annotated with negation, researchers developed approaches for detecting negation using custom annotated data from within their domain. Consequently, the type of negation phenomena annotated and detected was very narrowly defined. Despite these constraints, these early methods highlight some fundamental issues and challenges, such as the difficulties involved in determining the accurate scope of negations. In this section, some of these foundational approaches are discussed and the results and issues found are highlighted.

Identifying negated concepts: Recent research in identifying negated concepts originated in the medical domain, motivated by the need for clinical reports and discharge summaries to be reliably interpreted and indexed. Despite the availability of effective automatic indexing methods, IR systems did not differentiate between present and negated concepts. Furthermore, since negation triggers were by default treated as stop words, they were just ignored. The first systems developed to address this issue were rule based and use lexical information but not the syntactical structural information of a sentence.

(Mutalik et al., 2001) developed 'Negfinder', a three-step pipeline system that detects negated UMLS[1] concepts in dictated medical documents. The first and second step detects and encodes UMLS concepts present in a document. The third *lexing/parsing* step, performed by a lexical scanner, uses regular expressions to identify 60 negation triggers. A grammar comprised of corpus specific context free rules is then used to associate these

[1]The Unified Medical language system is a helpful resource for identifying concepts for medical indexing.

negation triggers with preceding or succeeding UMLS concepts present in a sentence. The authors discuss a number of challenges in the development of the system: a negation trigger may be a single word (i.e *not*) or a complex verb phrase (i.e. *could not be correctly identified*) - and these need to be distinguished; certain verbs when preceded by *not* will negate the subject or object concepts but others will not (i.e. *X is not seen* vs *X did not increase*). Another challenge discussed is that a single negation trigger may scope over several concepts, but not necessarily all concepts in the sentence. These issues were addressed without using the syntactic structure of the sentence. Consequently, Negfinder reliably identifies negated concepts in medical text when they are located near the negation triggers. They observe that in their corpus "One of the words *no*, *denies/denied*, *not*, or *without* was present in 92.5 % of all negations."

(Chapman et al., 2001) use a similar approach and developed 'NegEx'[2] a publicly available module that is still maintained and updated. NegEx is a regular expression based algorithm for determining whether a finding or disease mentioned in the text is present or absent. They identified and compiled a list of 272 domain specific negation triggers (single words and phrases). The negation triggers are divided into two groups: phrases that seem to indicate negation but are actually double negatives (i.e. *not ruled-out*), and triggers that are true negations (i.e. *not , without, did not exhibit,...*). The algorithm first identifies if a negation trigger is present in the sentence. Next, the scope of a found negation trigger term is determined: up to 5 tokens preceding or succeeding it, and if a UMLS concept is found within this window, it is considered to be negated. The evaluation of the system was done on a gold standard consisting of 560 discharge summaries annotated by physicians. They report performance measures of 84% for precision and 78% for recall. Among the system's weaknesses, the authors report that detecting the scope of *not* is not a straightforward task. (11 a) indicates the clinical finding *infection* is absent, however in (11 b), the *not* negates the term *source* and not *infection*. In both cases NegEx will mark 'infection' as being negated since the algorithm does not take into consideration the syntactic structure of the sentence.

(11) (a) This is not an infection

(b) This is not the source of the infection

Their findings show that negation triggers appear in clinical reports occur according to Zipf's law, there are a few very common ones (*i.e. no, without, no evidence of*), more medium frequency negation triggers and a very large number of low-frequency triggers. They note that having a small list of very common triggers can capture a large portion of negated concepts.

[2]http://code.google.com/p/negex/

ConText (Harkema et al., 2009) is an extension of NegEx developed as a processing resource within the GATE NLP framework (Cunningham et al., 2011). It uses the same approach based on regular expressions, negation triggers and contextual information. In contrast to NegEx it not only detects negation but also *temporal* and *historical* information, in order to determine whether a clinical condition mentioned is negated, hypothetical, historical or experienced by someone other than the patient. They developed separate sets of trigger terms for each property. In order to determine if a condition is negated, the algorithm checks if it falls within the scope of a negation trigger. The determination of scope is slightly different from NegEx in that the scope extends to the right of a trigger and ends either at domain specific termination terms or at the end of the sentence. The system was evaluated on 6 different types of clinical reports: radiology, emergency department, surgical pathology, echocardiogram, operative procedures, and discharge summaries. They collected 240 reports, that were manually annotated by a certified medical professional from that domain. Half of the data set was used for development and the other half was used for testing. They report an average precision of 94% and average recall of 92%. The authors report that in general, negation triggers have the same interpretation across the different report types.

(Elkin et al., 2005) implemented a rule based system to identify negated concepts in electronic health records, which was part of a larger expert system whose purpose was to assign a level of certainty (positive, negative or uncertain) to concepts. Their approach, like earlier systems, was concerned with identifying negation patterns within the medical domain. Negation assignment was performed by the 'automated negation assignment grammar'. Their approach was to use an ontology of operators and their associated rules. Operators were terms which belonged to one of two distinct sets. The first set implied a starting negation (i.e. *no, denies, rules out ...*). The second set indicated the termination of the assignment of negation in a phrase (i.e. *other than*). They also implemented a rule base which contained rules on how and when a negation trigger should be applied to existing concept(s) in the sentence. Their research differs from other early systems in that they used full medical evaluations which have a reported higher occurrence of negated concepts than in surgical reports or discharge summaries. They also used a different resource, SNOMED-CT[3] for identifying concepts, as opposed to UMLS.

The previously described systems all focused on having rich set of domain specific negation triggers. They also successfully identified a comprehensive set of heuristics to determine negation triggers in context. However, an issue with these early systems was incorrect determination of scope: namely when the concept is separated by more than a few words from the negation trigger.

[3]Systematized Nomenclature of Medicine-Clinical Terms: http://www.snomed.org

Consequently, (Huang and Lowe, 2007) were motivated to build a system based on syntactic information. The goal of this system was to detect negated noun phrases in radiology reports. They specifically chose not to use UMLS/SNOMED-CT derived concepts in order to focus on evaluating and determining negation in a more general manner. The first step in their implementation involved constructing a negation grammar by manually identifying negation triggers, negation phrases (multi-word triggers) and patterns in thirty reports. They also studied linguistic literature in order to gain insight into a more linguistic perspective for the determination of negation patterns. Negation triggers were then classified based on their syntactical categories (i.e. *no: determiner, without: preposition,...*). Using these categories, they constructed the associated grammar rules by identifying the syntactic structural patterns for locating a negated noun phrase in the parse tree given an existing negation trigger. They used the remaining 470 reports in the training set for this phase of development. The resultant set of structural grammar rules forms the basis of their system, to locate negated noun phrases in an automatic manner. The system achieves a precision of 98.6% and a recall of 92.6% on the gold standard of 120 Reports that contained 2976 noun phrases of which 310 were negated.

Identifying negated polar expressions: Sentiment analysis is an active field of research that focuses on the automatic detection and treatment of opinions in natural language processing applications.[4] Recently, systems developed for sentiment analysis tasks have recognized that the resolution of negation is an important support task. In a sentiment analysis task, negation is most commonly considered as a device used to reverse the polarity of an expression. The polarity of the statement *'This is a good camera'* should be the opposite of its negation *'This is **not** a good camera'*. This issue has been tackled using a variety of approaches.

(Pang et al., 2002), assume a very simple approach, whereby all words between the negation term (i.e. *not*) and the first punctuation mark are negated. They modelled these negated words as a new separate feature by adding the tag 'NOT' to these words. In this setup, the words considered for negation modification are unrestricted, no matter if they occur in a polar expression[5] or not. Later work explores more sophisticated approaches to using negations. (Kennedy and Inkpen, 2005) for instance, who developed a document level polarity classifier, which includes features based on *contextual valence shifters*[6](Polanyi and Zaenen, 2004), which are words that change (*shift* or *reverse*) the polarity or intensity of an expression. Again, a simple scope model for negation is chosen: a polar expression is thought to be negated if the negation word directly precedes it. These early approaches

[4]refer to (Pang and Lee, 2008) for a comprehensive overview of Sentiment Analysis.
[5]an expression that is positive or negative.
[6]i.e. *never, nowhere, little, hardly, most.*

report that detecting negation does not have a significant impact on the performance of machine learning methods for sentiment classification. This is in part due to their methods for modelling the scope of negations.

The survey described in (Wiegand et al., 2010), demonstrates that it is widely understood that the detection of negation scope is necessary. The survey stresses the usefulness for using syntactic knowledge and fine-grained linguistic analysis in order to model the scope of negation expressions and extract the relevant features for machine-learning or rule based sentiment analysis systems. They also highlight that the effectiveness of negation models can vary with different corpora because of specific language constructions/ patterns present in different contexts (issue of language usage).

An extension of the work presented in (Kennedy and Inkpen, 2005) is described in (Kennedy and Inkpen, 2006). In this system, the benefits of detecting negation scope are demonstrated empirically. Importantly, a parser is used to extract the relevant syntactic structures in order to compute negation scope. Final results show that the modelling of negation is important and relevant. Their work was based on the theoretical model proposed by (Polanyi and Zaenen, 2004), whereby the model would assign scores to polar expressions (i.e. a positive value for a positive expression and a negative value for a negative expression) and if this expression is considered to be negated then this score is inverted.

(Wilson et al., 2005) also consider negation scope using more advanced methods. The main goal of the system presented was to identify the *contextual*[7] polarity of an expression. The correct determination of the scope of a negation trigger is highlighted as an important support task. The system defines and incorporates a more advanced approach to the modelling of negations encoded as features. They do this by not only identifying the scope of any identified negations, but also by attempting to categorize and characterize them. They do this in part by differentiating between *local* and *long distance* negation features. The *local* feature checks whether a negation trigger is a local negation: if it occurs in a fixed window of four words preceding the polar expression. In contrast, syntactic dependency relations are used to identify the *longer distance* dependencies between the negation trigger and the polar expression (i.e. *does not* look [very good], *no one* thinks it is [good]). They also highlight the need for disambiguating negation triggers that do not necessarily function as negations within certain contexts like: (*not just*, *not only* ...). Given this more fine-grained modelling of negations, they report significant improvement. In (Wilson et al., 2009), the experiments of (Wilson et al., 2005) are extended by more detailed analysis of the effectiveness of their defined feature classes.

[7]To determine the in-context polarity of an expression : i.e. the word *excellent*, has a *prior* polarity value of positive. However, within the sentence *The movie was not excellent*, the *contextual polarity* of the phrase in which *excellent* appears, no longer has positive polarity - due to the negation term *not*.

The fine grained Multi-perspective Question Answering (MPQA) Opinion corpus[8] (Wiebe et al., 2005) used in (Wilson et al., 2005; Wilson et al., 2009) is a publicly available corpus consisting of 10,657 sentences in 535 documents of English newspaper articles. It also contains annotations of *subjective expressions*: any word or phrase used to express an opinion, emotion, speculation etc. A general label for such states is the *private state* (Quirk et al., 1985). For the system described in (Wiebe et al., 2005), the MPQA corpus annotations were extended to include the associated contextual polarity feature.

(Wilson et al., 2005; Wilson et al., 2009) also compiled a **prior polarity subjective lexicon** in order to be able to look up the default polarity values of expressions in their experiments. They incorporated over 8000 *subjectivity clues* which are words that may be used to express private states. They compiled the lexicon starting with a list of clues from (Riloff and Wiebe, 2003), and expanded it using a dictionary, a thesaurus, negative word lists from the General-Inquirer 2000[9] and from the research discussed in (Hatzivassiloglou and McKeown, 1997). All clues have an associated polarity value (positive, negative, both or neutral). 59.7% are allocated with negative prior polarity. Since this is also a publicly available resource,[10] we used this lexicon as a basis for the NEGATOR trigger lists described in Chapter 3.

(Choi and Cardie, 2008) combine different kinds of negation triggers (i.e. content negators: *eliminated, lacked, denied...*) with lexical polarity items through various compositional semantic models, both heuristic and machine learned, to improve sentiment analysis at a phrasal level. In this work the scope of negation was either left undefined or determined from surface level syntactic patterns similar to the syntactic patterns described in (Moilanen and Pulman, 2007). With this more fine grained modelling of negation, their evaluation reports far better results to a normal bag-of words approach.

2.2 Systems and Tasks dedicated to negation

Recently, the importance of processing negation as an independent task has gained recognition by the NLP research community. The success of several initiatives such as *The Negation and Speculation in Natural Language Processing Workshop* (Morante and Sporleder, 2010) and the *Special Issue on Modality and Negation* (Morante and Sporleder, 2012) demonstrate this trend. There have also been shared tasks dedicated to resolving various negation phenomena: the *SEM 2012 Shared Task: *Resolving the Scope and Focus of Negation* (Morante and Blanco, 2012) and The QA4MRE pilot task on *Processing Modality and*

[8] http://mpqa.cs.pitt.edu/

[9] http://wjh.harvard.edu/ inquirer/spreadsheet_guide.htm

[10] http://mpqa.cs.pitt.edu/

Negation (Morante and Daelemans, 2012b). Most useful have also been the careful annotation efforts of gold standards including various negation phenomena. This shift in the field has led to many more systems being developed primarily for the automatic detection of negation triggers and scope. More recent systems focus their approaches on using syntactic information in order to determine the scope of negations. However, most initiatives train and test their systems on one gold standard situated in a specific domain. Thus, the availability of a robust, general negation detection module that can be added to any text processing pipeline is still sparse. In this section, the recent annotation efforts, new systems developed and finally the relevant shared tasks dedicated to negation are described. These initiatives are fundamental to providing a solid basis for the development and testing of the NEGATOR system described in this thesis.

Recent annotation efforts: One of the first formal corpus annotation efforts which included annotations related to negation and speculation phenomena was the GENIA event corpus (Kim et al., 2008). The corpus consists of 1000 MEDLINE abstracts in which 36,858 biological events[11] have been identified. The events in this corpus are annotated with negation and speculation. In the case of negation, events are marked with the label `exists` or `non-exists`, since negation at the bio-event level is defined as the non-existence of the event. This indication of non existence can be explicit (i.e. presence of a negation marker) or implicit (i.e. through semantic inference). (Thompson et al., 2011) is an initiative which extends the annotations in the GENIA event corpus (Kim et al., 2008). These extensions involve annotating the events with *meta-knowledge* elements including LEXICAL POLARITY. This element will identify whether or not an event is negated (has positive or negative polarity).

The availability of the relevant GENIA (Kim et al., 2008) annotations formed the basis for the negation and speculation subtasks in two shared task competitions on biological event extraction (Kim et al., 2009; Kim et al., 2011). In these tasks negation and speculation is defined as correctly identifying speculation and negation instances and the events that these instances have scope over. The highest ranking participant (out of 6) in the task discussed in (Kim et al., 2009), was a system that applies syntax-based heuristics developed by (Kilicoglu and Bergler, 2009). The approach discussed in (Kilicoglu and Bergler, 2009) is to analyze the dependency path between an event trigger and the speculation and/or negation cues in order to determine whether the event is within the scope of the cues.

Detecting linguistic negation scope as an independent task was still largely ignored until the release of the BioScope corpus. The BioScope corpus (Szarvas et al., 2008; Vincze

[11] In the most general form, a textual event is an action, relation, process or state expressed in the text (Sauri, 2008). Consequently, a *bio event* is a textual event specialized for the biomedical domain.

et al., 2008)[12] is a freely available corpus consisting of biological and medical texts. The corpus consists of 3 subcorpora: full papers (9) and abstracts (1273) from the GENIA corpus (Collier et al., 1999), and clinical (radiology) reports (1954). In total there are 20,000 sentences. Every sentence is annotated with information about negation and speculative-language: triggers and linguistic scope. In BioScope, negation is defined as expressing the non-existence of something as demonstrated in (12a). Here, *without* is determined to be a negation trigger and the negation scope is the prepositional phrase. Speculative statements express the *possible* existence of something exemplified in (12b), where *suggests* is a speculation trigger and the resulting scope is the verb phrase.

(12) (a) Mildly hyper inflated lungs [without] focal opacity.

(b) This result [suggests] that the valency of Bi in the material is smaller than +3.

The scope of a keyword is determined by syntax, and is extended to the largest syntactic unit to the right of the cue, including all the complements and adjuncts of verbs and auxiliaries.

Since the GENIA Event and BioScope corpus share 958 abstracts, their negation annotations have been compared by (Vincze et al., 2011). Their study shows that the scope annotations in BioScope are not directly useful for detecting the certainty status of events in GENIA. However, as the scope annotations in BioScope are based on linguistic principles, they more easily adaptable to non-biomedical domain applications.

(Nawaz et al., 2013) conduct a detailed analysis for identifying negated bio-events given gold standard annotations. Importantly, they identify three key aspects to consider for achieving better performance in the task of negated bio-event finding. These aspects are: the compilation of the negation trigger list, the design and selection of suitable features, and the choice of machine learning algorithm. Their consideration for what constitutes a negation trigger stems from a task driven view-point. Thus, they are interested in identifying negation triggers specific to the given domain. They highlight that context and the annotation/information perspective (i.e. linguistic vs. biological perspective) are key factors to consider. The major contribution of the work discussed in (Nawaz et al., 2013), are the detailed experiments that were conducted. They combine different feature sets, different machine algorithms and various trigger lists, which are then subsequently run on three different gold standards.

(Morante, 2010) provides a detailed description of the various negation cues and their corresponding scopes found in biomedical texts, based on the cue and scope annotations found in the BioScope corpus. The paper also discusses issues related to the ambiguity of cues (i.e. what constitutes a negation cue given particular contexts) and scope. The detailed descriptions and examples support our view that the determination of scope depends on

[12] http://www.inf.u-szeged.hu/rgai/bioscope

the part of speech feature of the cue and the syntactic structure of the sentence.

The BioScope corpus was subsequently provided as the training data set for the biological track of the 2010 CoNLL Shared task on *Learning to Detect Hedges and their Scope in Natural Language Text* (Farkas et al., 2010).

As an early corpus for negation, BioScope has dominated the field, as acknowledged by (Morante and Daelemans, 2012a). To provide text from a new domain, however, (Morante and Daelemans, 2012a) present a corpus of two Conan Doyle stories[13] (*The Hound of the Baskervilles* and *The Adventure of Wisteria Lodge*) annotated with negation triggers and scope. The interpretation of negation scope differs from the one defined in BioScope. Specifically, the gold scope annotations include all arguments relating to an event being negated. The resulting negation scope spans represent the entire proposition being negated. Also, unlike in BioScope, they annotate affixal negation. Example (13) illustrates a gold annotated sentence, whereby all the arguments relating to the event *drive* are negated.

(13) We did [not] drive up to the door but got down near the gate of the avenue .

Example (14) illustrates a gold annotated sentence illustrating the relevant scope for an instance of affixal negation (underlined).

(14) . . . he said that he had indeed seen the [un] happy maiden.

(Blanco and Moldovan, 2011) consider a more semantic oriented view for negation detection, by annotating the negation *focus*. Specifically, they extended the Propbank (Palmer and Gildea, 2005) corpus with annotations relating to the negation *focus*: "that part of the negation scope that is most prominently or explicitly negated". 3993 verbal negations signalled with the MNEG label[14] were identified in the Propbank corpus. Subsequently, according to specific criteria, a specific semantic relation was chosen to also be the *focus* and an annotation labelled as: -NOT was added for that relation. According to the authors, the annotation of focus allows the derivation of the implicit positive meaning of negated statements. For example, for the sentence *They didn't release the UFO files until 2008.*, if the focus of negation is determined to be *until 2008*, the implicit positive meaning of the sentence will be: *They released the UFO files in 2008.*

Negation Scope Resolvers trained on BioScope: The availability of the BioScope corpus led to the development of quite a few systems dedicated to resolving the scope of

[13]Website of the Conan Doyle corpus: http://www.clips.ua.ac.be/BiographTA/corpora.html

[14]The Propbank corpus is annotated with verbal propositions and their arguments. The relations between the verb and its arguments are referred to as semantic relations. Propbank has a wide set of possible labels, and it is out of the scope of this thesis to define them. However, for the following example generic semantic relation labels are used for illustration purposes. The sentence: *The cow didn't eat grass with a fork.* Typical semantic relations will encode AGENT (the cow, eat), THEME (grass,eat) INSTRUMENT (with a fork, eat) and NEGATION (n't eat). In Propbank, the 'MNEG' label refers to the generic NEGATION label.

negation. (Morante et al., 2008) pioneered the research on negation scope. They approached the task as a chunking problem to determine whether a word in the sentence is inside or outside the negation scope. The system described in (Morante et al., 2008) is extended in (Morante and Daelemans, 2009). Here, the scope finding task is also modelled as a classification task, subdivided into two parts. The first subtask it to detect if a token is a match or is a part-match to a negation trigger. This involves classifying the tokens in the sentence as either being the beginning, end, inside or outside of a negation trigger. This approach allows for the detection of multi-word triggers (i.e. *instead of...*). The second subtask is to resolve the scope of an identified negation trigger. They used three classifiers which predict whether a token is the first token, last token or neither in the scope sequence. Each token in a sentence is paired with the identified negation trigger (classified in the previous phase) in the sentence. This pair represents an *instance*. The features used for the classifiers included token level tags (i.e. lemma, pos tag) and information regarding the token to the left, and the three next tokens to the right. A fourth classifier, a meta learner then uses the predictions of the three classifiers to make the final predictions. Post processing rules are implemented to build the final scope sequences. The system was evaluated on the three BioScope subcorpora, evaluated using PCS scores. Percentage of fully correct scopes was introduced by (Morante and Daelemans, 2009). With PS being the number of correct scopes produced by the system and S the number of gold scopes, PCS can be expressed with: $PCS = PS \ / \ S$. They report 66.7%, 41% and 70.75% for the abstracts, full texts and clinical reports respectively.

(Li et al., 2010) developed a negation scope finding system trained and tested on the BioScope corpus. As their focus is only on identifying negation scope, they extracted the negation triggers for their systems directly from the gold standard. In contrast to (Morante and Daelemans, 2009), their approach identifies whether constituents are in the scope of negation rather than single tokens. They achieve this by using the syntactic information (i.e. parse trees) made available by first parsing the sentences. Their method is to model the task as a semantic parsing problem. Specifically, the identified negation trigger is regarded as a predicate and the constituents which belong to the scope are the semantic arguments to this predicate. They implement a few heuristics to identify the candidate arguments from the parse tree. They then employed other heuristics to prune out potential candidate arguments. Finally, they use a binary classifier using features extracted from the parse tree to identify whether a candidate constituent (argument) is within the scope of negation or not. Their experiments include using gold parse trees, and automatically parsing sentences. They report PCS scores for their final system on automatically parsed sentences of 81.84%, 64.02% and 89.79% for abstracts, full papers and clinical reports respectively.

(Apostolova et al., 2011) developed a rule set extracted from the BioScope corpus for

identifying negation and speculation scope . Their approach was to use syntactic information made available by the parse tree, similar to (Li et al., 2010). Unlike other systems, their approach was to detect negation triggers and their scope simultaneously. 70% of the BioScope was used for training and 30% for testing. Their system is essentially a set of rules consisting of lexico-syntactic patterns which identify negation and speculation scopes. Instead of manually compiling the rules like the hedge detection system discussed in (Kilicoglu and Bergler, 2010), they attempted to automatically extract the lexico-syntactic rules from the BioScope corpus. Their process involved first parsing each sentence in the training set. The rules were then extracted by identifying the subtree rooted at the closest ancestor of the negation trigger found. Finally, some post processing are applied to make the extracted rules more general. In order to test their rule set, they developed the 'ScopeFinder' module. The PCS scores for negation on the test set (30% of the BioScope corpus) are reported as 80.63%, 71.26% and 85.56% for abstracts, full papers and clinical reports respectively.

(Councill and Velikovich, 2010) present a negation detection similar to the system described in (Morante and Daelemans, 2009). The main differences are that in the trigger detection phase they utilize a dictionary of 35 explicit negation triggers (i.e. *not, cannot, never*) instead of being machine learned. (Councill and Velikovich, 2010) only use one classifier, and this classifier incorporates features (i.e. dependency relations) from a dependency parser. The work described in (Councill and Velikovich, 2010) was part of a larger effort to improve the accuracy of sentiment analysis in online reviews. As there existed no gold standard for negation in the Sentiment Analysis domain, they developed and annotated a custom corpus of 268 Product Reviews. Their negation detection system was trained on both BioScope and the Product Reviews corpora. They report a PCS score of 53.7% on the BioScope Abstracts and 39.8% on the Product Reviews corpus. Cross training results are also reported showing that the system has better results for the Product Review Corpus when first trained on BioScope. The authors determine that this indicates that scope boundaries are more difficult to predict in the Product Reviews Corpus. (Councill and Velikovich, 2010) also ran their sentiment analysis system using their negation module. They report that the results improve by 29.5% and 11.4% for positive and negative sentiment respectively. This is an important initiative as it is one of the few which was developed and tested in two distinct domains.

The *SEM 2012 Pilot Task on *Resolving the Scope and Focus of Negation* (Morante and Blanco, 2012) consisted of two independent subtasks dedicated to the detection of various negation phenomena. The first, involved detecting the negation triggers, the scope and negated events in the Conan Doyle corpus (Morante and Daelemans, 2012a). The second was to detect the focus of negation using the gold standard prepared by (Blanco and

Moldovan, 2011). A total of 12 runs were submitted for the scope task and 2 for the focus task.

Most systems submitted for the scope task were machine learning systems. Only a few systems implemented a rule-based approach. Overall, syntax information was widely used either in the form of heuristics or incorporated into the learning models. The top ranking system for both the negated event subtask and the global task in the closed track was the 'UiO1' system is an adaptation of another system (Velldal et al., 2012). It combined SVM (Support Vector Machine) cue classification with SVM-based ranking of syntactic constituents for scope resolution. The approach was then extended to identify negated events by first classifying negations as factual or non-factual, and then applying an SVM ranker over candidate events. The top ranking system in the open track, the 'UiO2' system combines SVM cue classification with CRF-based sequence labeling. Two systems in the open track were developed as rule based systems. The first, 'UCM1', used a lexicon and WordNet for the identification of negation triggers. Heuristics based on parse tree information are implemented for the determination of negation scope. The second, 'UCM-2' implemented an algorithm to detect negation triggers and their scope by traversing dependency structures.

The only official participant (who submitted both runs) in the *detecting focus of negation* task was submitted by (Rosenberg and Bergler, 2012). The NEGATOR trigger-scope module described in this thesis, was extended with custom focus heuristics. The system implemented by the task organizer, described in (Blanco and Moldovan, 2011): uses a supervised learning approach. Each sentence containing a verbal negation from Propbank (Palmer and Gildea, 2005) is considered an instance. The decision to be made is which of the semantic relations present in the sentence corresponds to the 'focus'. The available pre-existing annotations such as syntactic information and semantic role labels available from PropBank were used in the development of their system. They report an accuracy of 61.38% on the BASIC baseline and 65.50% on the FOC-DET system.

The QA4MRE pilot task on *Processing Modality and Negation* (Morante and Daelemans, 2012b) involved detecting negated and modalized *events*. The main objective of the QA4MRE[15] evaluation task (Peñas et al., 2012) at CLEF 2012 was to develop a methodology for evaluating Machine Reading[16] systems through question answering and reading comprehension tests. Processing negation and modality is very relevant for tasks like question answering: extracted information that falls within the scope of a negation trigger (i.e. *not, no, never,...*) cannot be presented as asserted information. Similarly, information falling within the scope of a modal trigger (i.e. *could, should, would,...*) cannot be presented

[15] http://celct.fbk.eu/QA4MRE/

[16] a task with the main objective of developing systems that are capable of automatically 'understanding' texts utilizing an unsupervised approach (Etzioni et al., 2006).

as factual or certain. Consequently, one of the two pilot tasks offered in parallel to the main QA4MRE task was *Processing Modality and Negation for Machine Reading* (Morante and Daelemans, 2012b). The goal of this task was to determine whether a pre-annotated event[17] present in the text is within the scope of a negation, in the scope of a modal, within the scope of both, or within the scope of neither a modal nor a negation trigger. The test data set for the pilot task consisted of 8 English documents part of the larger test set for the main QA4MRE Task. Four topics were covered: *AIDS, Climate Change, Music and Society, and Alzheimer's Disease*. Three groups participated with a total of six runs. The two top ranking runs were submitted by (Rosenberg et al., 2012).The NEGATOR trigger-scope module described in this thesis, was adapted to not only detect the scope of negation, but also for the scope of modal triggers as well. A custom module was also developed in order to detect whether an event was within a modal and/or negated context. The 'desancis' team developed a rule-based system defined in JAPE (Java Annotation Patterns Engine). The system consists of three different components: The VG module tags verbal groups with identified characteristics i.e. tense, aspect, voice, and modality; the MODNEG module tags particles that may be related to modality and/or negation and finally the LABELER module consists of rules that use contextual information about modality and negation that then label the event accordingly. The 'JUCSENLP' team developed a system that relies on a word list of modality and negation triggers. The label assigned to an event depends on whether the event is preceded in the sentence by the modality and negation triggers contained in the list.

[17] An event is defined here as any of the main verbs mentioned within the text.

Chapter 3

Negation Triggers

This chapter describes the NEGATOR negation trigger lists and the NEGATOR trigger detection module. In order to motivate the linguistic variety present in negation triggers, the research presented in this thesis explores a much more extensive set of triggers than is usually considered in the literature. In the last section, the NEGATOR trigger lists along with different sized negation trigger lists from the biomedical domain are evaluated in terms of usefulness on four different gold standards from different genres.

3.1 Different forms of Negation

3.1.1 Explicit Negation Triggers

The list of *explicit* negation triggers form a consensus subset of triggers that are most frequently included in the detection of negation phenomena in any text genre. Figure 3 lists these triggers categorized by their lexical category:

- **Adverbs** : *not, nor, neither, never.*
- **Contractions**: *n't used with auxiliary verbs (aren't, isn't, couldn't,...)*
- **Determiners** : *no, neither (neither side of the brain is dominant over the other).*
- **Pronouns**: *none, nobody, nothing, nowhere, no one, neither (neither of us believes it)*
- **Prepositions**: *without, except*
- **Multiword expressions**: *rather than, with the exception of, by no means, turned down, instead of, and on the contrary)*

Figure 3: Explicit Negation Trigger Categories

In any given sentence, an *explicit* negation trigger does not necessarily only function as a negation marker (Huddleston and Pullum, 2002). As illustrated in (15a), the negation trigger *no* has multiple responsibilities. It marks negation and expresses quantification by being a determiner in the noun phrase. In (15b), the contraction *isn't* is considered in contemporary language to be a verbal suffix, and thus is a necessary part of the verb. In (15c), the *neither ... nor ...* pattern functions as establishing a conjunction between two negated clauses (marked by neither and then nor respectively). In (15d) the negated pronoun *Nobody* also serves as the main subject of the sentence.

(15) (a) *John had **no** money.*
(b) *The report is**n't** complete*
(c) *I am **neither** a liberal **nor** a conservative.*
(d) ***Nobody** finished their homework on time.*

Example (15) highlights the notion that the identified lexical category of an explicit negation trigger is an important feature to consider. It not only further characterizes the negation trigger but is also an important indicator for determining the extent of its scope in the sentence. In the case of multi word expressions, the lexical category of the last word in the expression (i.e. *than* in *rather than* is a preposition) is the determinant. Thus, when an explicit negation trigger is identified in the text, its lexical category will be added as a feature to the annotation by the NEGATOR trigger detection module (this will be discussed in more detail in section 3.2).

The NEGATOR *explicit* negation trigger list does not account for any special patterns relating to specific interpretations of negation phenomena. For example, the presence of an *explicit* negation trigger often indicates a reversal of the polarity of the items contained within the negation scope, as shown in example (16b) (scope is underlined). In contrast, as shown in the statement (16c), the pattern *not just* is a device used for emphasis, and therefore the scope will be *just*. The author's intention in (16c) is to highlight that John found *both* the book *and* his keys. Regardless of these different interpretations, in both (16b) and (16c), the *not* needs to be identified as an *explicit* negation trigger, as its main function in the sentence is to mark the presence of some negation phenomenon.

(16) (a) *John found the book.*
(b) *John did **not** find the book.*
(c) *John did **not** just find the book, he also found his keys.*

Therefore, the NEGATOR *explicit* negation trigger list contains only the terms listed in Figure 3 in their most general form, without any consideration for specific interpretation features.

3.1.2 Implicit Negation Triggers

Negation also occurs in another lexicalized form as illustrated in (17 a) where there are two *implicit* negations *repress* and *prevent* are loosely paraphrased as *does not allow* in (17 b).

(17) (a) FTZ-F1 represses Hr39 expression to prevent competition.

(b) FTZ-F1 does not allow Hr39 expression and thus does not allow competition.

There is no pre-existing universal trigger list for *implicit* negation, and they are only partially annotated in current gold standard annotations that include annotations for explicit negation and related phenomena, such as TIMEBANK (Pustejovsky et al., 2003), or MPQA (Wiebe et al., 2005). In contrast to *explicit* negation, *implicit* negation is often interpreted and categorized from a domain specific perspective and not necessarily as negation phenomena. Although, there is sufficient evidence in the literature that implicit negation is worthwhile considering and has been actively used in specific tasks. (Choi and Cardie, 2008) identify *content negation triggers*, i.e. (*lack, hamper, deny*) within the domain of Sentiment Analysis. (Harabagiu et al., 2006) recognize a class of *indirectly licensed* negations which include triggers like: (*fail, deny, refuse*), which are used for the task of detecting contradictions in text. The BioScope (Szarvas et al., 2008) corpus considers a small subset of *implicit* negations but only when the implied meaning is interpreted as the non-existence of something i.e. (*fail, failure, absent, absence, lack of*). The GENIA event corpus (Thompson et al., 2011) applies further constraints by not only annotating negations in specific contexts, but also categorizing them according to domain requirements (i.e. as *events* which fall into the category of *negative regulation*).

The focus of the research presented in this thesis was to develop a domain independent and general approach to the detection of negation. Thus, a list of *implicit negation* triggers was compiled. Specifically, they all impart negative polarity regardless of their domain dependent semantics. How this resulting negativity is interpreted within a specific context can be modelled according to the requirements of the application. The NEGATOR *implicit* negation trigger list was constructed using the Subjectivity Lexicon[1] described in (Wiebe et al., 2005). All the verbs and nominalizations which contain the negative prior polarity feature and are determined to also be *covertly negative lexical items* were extracted from the Subjective lexicon. (Huddleston and Pullum, 2002), define implicit negations as *covertly lexical items*. These *covertly lexical items* all belong to one or more of the following categories:

[1] Section 2.1

Categories of *covertly lexical items* (Huddleston and Pullum, 2002)

- **failure, avoidance, and omission**
- **prevention and prohibition**
- **denial**
- **doubt**
- **counter-expectation**
- **unfavourable evaluation**

Any lexical item that belongs to one or more of these categories, all have clausal or clausal type complements, that trigger entailments or implicatures involving negation in the subordinate clause. The resultant NEGATOR list of *implicit negation* triggers consists of 112 verbs, their nominalizations and any adjective derivatives like (*deny, denial, prevent, prevention fail, failure, reject, rejection, absence, absent ...*).

3.1.3 Affixal Negation Triggers

Finally, negations which may also be explicitly part of the lexical semantics of a word when added through negation prefixes like (*dis-, non-, un-, im-, in-, ir-,il-, a-*) or postfixes/infixes like (*-less*) (Tottie, 1991). Affixal negation triggers are different from *explicit* and *implicit* negation triggers in that they always have a local scope. Affixal negation is purely morphological (Huddleston and Pullum, 2002). In Example (18), *un* is the negation trigger and the corresponding negation scope is *happy*. Even with the affixal negation present in example (18), one can still infer that the person that is unhappy is still a maiden. This maiden however is unhappy. If the sentence was formed as *he said that he had indeed seen the **not** happy maiden*, this inference could be false.

(18) ...he said that he had indeed seen the un[happy] maiden.

A third NEGATOR trigger list was compiled containing terms with negation affixes, using the same Subjectivity Lexicon as for the *implicit* negation trigger list. All the relevant *clues* (the majority of which are adjectives) were extracted according to the guidelines presented in (Huddleston and Pullum, 2002) and shown in the following list:

Guidelines for negation affixes (Huddleston and Pullum, 2002)

- **in-, im(m)-, ir(r)-, il(l)-, un- and a-**: Mainly adjectives, along with the nouns and adverbs derived from these adjectives *(i.e. impatient, impatience, impatiently).*

- **dis-**: The only category that along with adjectives *(i.e. disagreeable)* also contains a few verbs *(i.e. dislike, disobey, distrust)*.
- **non-**: Nouns derived from verbs *(i.e. non-recognition)* and adjectives *(i.e. non-governmental)*.
- **-less**: Adjectives *(i.e. carelessness, hopeless)*.

As observed in this list, the part of speech information is an important feature to consider when identifying a given affix of a word to have the intended meaning of negation. However, not all words with the above mentioned prefixes and suffixes express negation of the root term: like *in* which can also denote *within* (i.e. inside), *un* when added to verbs usually is interpreted as a reversal (i.e. tie: untie), or words like *united*, where the correct prefix is not *un*. Consequently, these guidelines are not foolproof. By the process of manual extraction we ensured that each chosen term met the requirements (being a term containing a negation affix).

Research within the domain of Sentiment Analysis use affixal negation terms quite extensively (Moilanen and Pulman, 2007; Wilson et al., 2005), and thus the Subjective Lexicon is an appropriate resource to use as there are numerous and varied entries for affixal negation. The resultant word list contains 701 terms labeled as *selfNeg* triggers. This trigger list contains only a fraction of the possible words with negation affixes. We have conducted preliminary research for more automatic strategies (i.e. using resources such as WordNet[2]). However, for the evaluations presented in this thesis, the NEGATOR *selfNeg* trigger list proved to be sufficient.

3.2 NEGATOR Trigger Detection module

In the previous section, the construction of the *explicit, implicit and selfNeg* NEGATOR trigger lists was described. All the wordlists are formatted in XML. Each entry in a given list has a negation type (`explicitNeg,implicitNeg, selfNeg`). The NEGATOR trigger detection module is the first of the two core components presented in this thesis. For negation detection, this component uses the NEGATOR trigger lists as input. This module is not specifically designed for only negation triggers. Rather, it is capable of detecting any type of triggers, as long as the wordlist has been compiled into the required XML format (i.e. other negation trigger lists, modal triggers, valence shifters . . .).

The NEGATOR trigger detection GATE (Cunningham et al., 2011) module requires the following standard preprocessing tasks to be done before the module is run:

[2]WordNet (http://wordnet.princeton.edu/) is an online lexical database. English nouns, verbs, adjectives, and adverbs are organized into sets of synonyms, each representing a lexicalized concept.

1. Sentence Splitting: Divide the text in a document into individual sentence units.

2. Tokenization: Break apart each sentence unit into individual tokens.

3. Part of Speech Tagging: Annotate each token with its corresponding lexical category (i.e verb, noun, punctuation. . .).

These preprocessing tasks are done using the standard ANNIE plugins (Cunningham et al., 2011) on the given data set. The NEGATOR trigger detection module takes as input a typed XML trigger list. It then iterates over every token in the given text and upon finding a string match will create and output a negation trigger annotation for that token (or token span if the token span is a multiword expression). A separate annotation set is created for a each specified negation trigger type. The resulting sets of negation annotations are visible to any down stream processing in the GATE negation detection pipeline. The features present for any negation trigger annotation are its type, the lexical category, the string, and its start and end offset relative to the document. Example (19) contains an *explicit* negation trigger: *not*, and the resultant annotation is shown in Figure 4.

(19) . . . TNF-alpha mRNA induction by PMA does ***not*** *correlate with NF-kappa B binding activities . . .*

explicit negation trigger

Ann_Type	explicitNegTrigger
Type	explicitNeg
POS	RB
String	not
Start Offset	1194
End Offset	1197

Figure 4: NEGATOR Explicit Trigger Annotation

Example (20) contains an *implicit* negation trigger: *lacking*, and the resultant annotation is shown in Figure 5.

(20) Oncogenic forms of NOTCH1 ***lacking*** *either the primary binding site for RBP-Jkappa or nuclear localization . . .*

implicit negation trigger	
Ann_Type	implicitNegTrigger
Type	implicitNeg
POS	VBG
String	lacking
Start Offset	44
End Offset	51

Figure 5: NEGATOR Implicit Trigger Annotation

Example (21) illustrates a sentence which contains the *selfNeg* trigger: *Un-*. The local scope is the root of the word containing the negation affix: *expectedly*. In the case of *selfNeg* trigger annotations, the module will not only annotate the affixal negation trigger for (21) as shown in Figure 6, but will also generate a separate annotation for the resultant local scope. This local scope annotation is shown in Figure 7. The scope annotation is linked to the trigger annotation by both having entries for the *original token identifier* - a reference id to the original token annotation.

(21) ***Un****expectedly, a second, stronger RBP-Jkappa-binding site, . . .*

affixal negation trigger	
Ann_Type	selfNegTrigger
Type	selfNeg
String	un
Original_String	Unexpectedly
Original_Token_ID	332
Start Offset	1064
End Offset	1066

Figure 6: NEGATOR Affixal Trigger Annotation

affixal negation scope	
Ann_Type	selfNegScope
Type	selfNeg
String	expectedly
Original_String	Unexpectedly
Original_Token_ID	332
Start Offset	1066
End Offset	1076

Figure 7: NEGATOR Affixal Scope Annotation

The NEGATOR trigger detection module will mark all terms identified as negations in the text, regardless of their surrounding context. For example, in cases of double negation,

if a negation trigger is within the scope of another trigger, both terms will be annotated as negation triggers. As illustrated in (22), *exclude* is marked as a negation trigger by NEGATOR despite the fact that it is within the scope of the *not*.

*(22) This does **not exclude** the diagnosis of pertussis.*

(23) exemplifies a scenario where the negation trigger is within an uncertain context, because *will* is in the future tense, and therefore the action of *excluding reflux* has not occurred yet. NEGATOR regardless of this context will mark *exclude* as a negation trigger.

*(23) Voiding cystogram will be performed to **exclude** reflux.*

The purpose of the NEGATOR trigger detection module and the three wordlists is to mark any occurrences of negation phenomena identified in the text. Therefore, by not applying constraints such as surrounding context at this point allows for a more general approach. Importantly, it is the domain and task requirements which should determine the existence and necessity of such constraints.

3.3 Trigger Lists Experiment and Results

In this section we present and discuss the results of an experiment conducted in order to assess the performance and generality of the NEGATOR negation trigger lists. Specifically, the NEGATOR trigger lists are compared with four other lists on four datasets from different text genres.

3.3.1 Issues with evaluating the triggers

Trigger lists have the advantage of being easily expanded and modified for specific applications. There is a more or less universal understanding to what constitutes an *explicit* negation trigger. The *explicit negation* trigger list aims to detect any explicit negation occurrences in a text with high accuracy and coverage. The assumption here is that if a particular explicit negation trigger from the NEGATOR trigger list does not occur in a given application, it will not affect the overall performance (contracted negation forms, for instance, are frequent in the news domain, but are absent in lifescience journals). However, this assumption is not true for the *implicit* or *selfNeg* trigger lists. What constitutes an *implicit* or *selfNeg* trigger varies according to the choice of the annotators. Therefore, if a gold standard does not annotate a given trigger from any of these lists, all these triggers will incur precision errors. In contrast, if a gold standard annotates an *implicit* or *selfNeg* trigger and it is not contained in the NEGATOR lists, then recall errors will occur. Currently, available corpora annotated with negation will not necessarily annotate all instances of a

given negation trigger, because of domain and task specific considerations. As illustrated in (24), there are two sentences from GENIA which both contain the term *unchanged*. This term is identified as a gold trigger in (24 a) but not in (24 b).

*(24) (a) Human RAR alpha expression was **unchanged** in H9 and CEM cells, and elevated in U937 cells, after PMA stimulation.*

*(b) This was correlated with an **unchanged** level of the active form of the cytosolic inhibitor protein IkappaB-alpha.*

(25 a) illustrates a case from the BioScope corpus, where the *not* is not marked as a negation trigger. Rather, the multiword expression *not known* is annotated as a speculative trigger instead.

BioScope and GENIA do not annotate negation triggers when the trigger is *deactivated* by another word like (*only, clear, evident . . .*)(Szarvas et al., 2008; Nawaz et al., 2013). As illustrated in (25 b), the term *only* deactivates the *Not*. In essence *only* term nullifies any negation function brought on by the presence of the negation trigger.

*(25) (a) . . . its role in the inflammatory process is **not known**.*

*(b) **Not only** was this effect observed at the mRNA level . . .*

Regardless of these types of mismatches, we still pursued to evaluate the three NEGATOR trigger lists on four different datasets.

3.3.2 Evaluation of NEGATOR Negation Trigger Lists

(Nawaz et al., 2013), provide an in-depth analysis on the detection and characteristics of negated bio-events. In their documented experiments they use four separate trigger lists compiled from the biomedical domain:

1. **C40**: their own compilation of 40 negation triggers
2. **cBioInfer**: a list of 25 triggers extracted from BioInfer (Pyysalo et al., 2007)
3. **cBioScope** a list of 28 triggers from BioScope compiled by (Morante, 2010)
4. **cCore**, a list of the 20 most frequent triggers in 1000 randomly selected negated bio events from BioInfer, the GENIA Event Corpus, and the BioNLP09 ST corpus (Kim et al., 2008) and (Kim et al., 2009).

These four trigger lists are shown in Table 2. They all contain various forms of negation, and each one has a balance of similar, distinct and common triggers in comparison to each other. The evaluation presented here uses these four lists along with the three NEGATOR trigger lists.

Name	Size	Elements
C40	40	absence, absent, barely, cannot, deficiency, deficient, except, exception, fail, failure, impair, inability, inactive, independent, independently, insensitive, instead, insufficient, lack (noun), lack (verb), limited, little, loss, lose, lost, low, negative, neither, never, no, none, nor, not, prevent, resistance, resistant, unable, unaffected, unchanged, without
cBioScope	28	absence, absent, cannot, could not, either, except, exclude, fail, failure, favor over, impossible, instead of, lack (noun), lack (verb), loss, miss, negative, neither, never, no, no longer, none, not, rather than, rule out, unable, with the exception of, without
cBioInfer	25	abolished, absence, cannot, defective, deficient, despite, differ, different, differential, distinct, failure, independent, independently, lack, negligible, neither, no, nor, not, protected, separately, simultaneously, unable, unlike, without
cCore	20	absence, fail, inability, independent, independently, insensitive, insufficient, lack (noun), lack (verb), little, neither, no, nor, not, resistant, unable, unaffected, unchanged, without

Table 2: The four Negation Trigger Lists from (Nawaz et al., 2013)

These four wordlists were converted into the XML format required by the NEGATOR trigger detection module. This ensures that all triggers detected in the text are produced with in the same input and preprocessing parameters. To assess generality of the trigger lists, all the lists were run on four different data sets annotated with varying levels of negation phenomena including negation triggers:

- The QA4MRE Pilot Task Test Set from the PilotTask on Negation and Modality (Morante and Daelemans, 2012b) for news and current affairs.
- BioScope (Szarvas et al., 2008) for 1273 biomedical journal abstracts.
- *SEM 2012 Scope of Negation Training set (Morante and Blanco, 2012) (The Hound of the Baskervilles by Sir Arthur Conan Doyle).
- GENIA Event Corpus (Thompson et al., 2011) for 1000 biomedical abstracts.

3.3.2.1 Results

All the trigger lists have been compiled independently of task specific annotations, and therefore a custom evaluation scheme was defined. All triggers are assumed to be good indicators of linguistic negation in principle, and thus the overlap with gold annotations is classified into three categories:

`TP (true positives)`: indicates that a word from a trigger list and a matching gold annotation exists in a particular sentence

`FN (false negatives)`: indicates that a gold annotation trigger in a particular sentence is missing in a trigger list.

`FP (false positives)`: indicates that the trigger from a trigger list is present in a sentence, but is not annotated as a trigger by the gold standard.

The performance of the each trigger list for each data set is evaluated according to the following measures:

`Precision`: `TP/(TP+FP)` indicates the ratio for how many triggers retrieved by a given trigger list were actually relevant.

`Recall`: (`TP/TP+FN`) indicates the ratio for how many relevant triggers were retrieved by a given trigger list.

`F-score`: the harmonic mean between the Precision and Recall is calculated as (`2*(Precision*Recall)/(Precision+Recall)`)

Tables 3 - 6 compare the results of the NEGATOR lists (combined have a total of 852 triggers), with the other four lists. Each table presents the results from a distinct data set. In all data sets, *explicit* negation occurs most frequently. As expected, all trigger lists perform overall very well in terms of recall. As discussed earlier, the NEGATOR trigger detection module does not apply any rules for determining domain specific context features in the process of identifying a negation trigger in a sentence. This leads to a number of *false positive* occurrences for all lists. Table 3 shows the results from running the trigger lists on

	TP	**FN**	**FP**	**GOLD**	**Precision**	**Recall**	**F-score**
NEGATOR	1746	4	2260	1750	.43	.99	.60
C40	1724	26	1165	1750	.60	.98	.74
cBioScope	1704	46	638	1750	.73	.97	.83
cBioInfer	1633	117	1011	1750	.61	.93	.74
cCore	1682	68	453	1750	.79	.96	.87

Table 3: Performance summary of Negation Trigger Lists on BioScope Abstracts

the BioScope Abstracts. BioScope contains no gold affixal negation triggers, and a small subset of gold implicit negation triggers. Even though the recall measures for all the lists is very encouraging, the precision measure for all lists drops significantly. This is in part because a trigger present in a list is never marked as a gold negation trigger in BioScope. For example: the cCore, cC40 and cBioInfer lists contain the trigger terms: *independent, independently*. These terms are considered negation triggers in the GENIA corpus, but not in BioScope. Combined there are 102 occurrences of these terms in BioScope. The same situation occurs for the triggers: *inhibit, inhibiting and inhibitor* present in the NEGATOR *implicit* trigger list which occur (combined) 633 times in BioScope. Triggers from the NEGATOR *selfNegTrigger* list also contribute 15%to the *false positive* count.

The NEGATOR *explicit* trigger list is the only list which contains both *nor* and multi-word expressions i.e. (*with the exception of, rather than*). Consequently, the NEGATOR lists have a better recall measure than the other lists. The best performer overall on BioScope is the cCore list with an 87% F-score. This list does very well in recall, yet more importantly it has a much smaller amount of *false positives*. In contrast, the NEGATOR lists performs less well because of the relatively low precision measure. This is expected, as the NEGATOR lists are larger and more general than the cCore list.

	TP	**FN**	**FP**	**GOLD**	**Precision**	**Recall**	**F-score**
NEGATOR	1202	315	2033	1517	.37	.79	.50
C40	1302	215	1019	1517	.56	.86	.68
cBioScope	1103	414	736	1517	.60	.73	.66
cBioInfer	1165	352	981	1517	.54	.77	.63
cCore	1248	269	479	1517	.72	.82	.77

Table 4: Performance summary of Negation Trigger Lists on Genia Event Corpus

In contrast to BioScope, the results in Table 4 indicate that there are a greater number of *false negatives* from all the trigger lists when run on the GENIA corpus. The gold *polarity clue* triggers in the GENIA event corpus are highly domain specific (*i.e multi-word expressions such as: returned to basal levels, persists at high levels . . .*), and there are many which occur as *polarity clue* triggers only once or twice in the entire dataset.

The GENIA corpus annotates a number of affixal negations (*independent, inactive, insensitive, insufficient*). One would expect then that the NEGATOR lists should also have better recall performance, given its extensive *selfNeg* trigger list. However, the NEGATOR *selfNeg* trigger list does not contain the terms *independent* or *independently*, which make up 6.5% of GENIA gold negation triggers. Although, NEGATOR is able to redeem itself, as 26% of the gold triggers in GENIA are *implicit* triggers also present in the NEGATOR list. Again, there are a great number of *false positives* proportionally for all the trigger lists. The reasons for these *false positives* are consistent with the observations made with BioScope. Also, the GENIA annotations do not adhere to a general linguistic approach to negation. Rather, a *polarity clue with a negative value* is only marked when they affect a domain specific bio-event directly.

	TP	**FN**	**FP**	**GOLD**	**Precision**	**Recall**	**F-score**
NEGATOR	968	16	309	984	.76	.98	.86
C40	702	282	169	984	.80	.71	.75
cBioScope	698	286	162	984	81	.71	.76
cBioInfer	628	356	54	984	.92	.64	.75
cCore	614	370	105	984	.85	.62	.72

Table 5: Performance summary of Negation Trigger Lists on *SEM Task Training Set

In comparison with the other trigger lists, the NEGATOR lists have better recall on the datasets from *SEM and QA4MRE (Tables 5 - 6). This is in part due to the existence of contractions (*n't*) which make up 6.5% of the total number of gold triggers in the *SEM data set. Typically, *n't* does not occur in the biomedical data sets and therefore is not a part of the lists compiled by (Nawaz et al., 2013). The *SEM data set also contains a fair number of affixal negation triggers (16%). Since the NEGATOR *selfNeg* trigger list is more extensive and general than the others, it is the overall best performer with a 86% F-score.

	TP	**FN**	**FP**	**GOLD**	**Precision**	**Recall**	**F-score**
NEGATOR	91	6	113	97	.45	.94	.61
C40	60	37	51	97	.54	.62	.58
cBioScope	59	38	47	97	.56	.61	.58
cBioInfer	56	41	42	97	.57	.58	.58
cCore	56	41	36	97	.61	.58	.60

Table 6: Performance summary of Negation Trigger Lists on QA4MRE Test Set

The overall insight from this experiment is that in the Zipfian distribution of negation (Chapman et al., 2001), less is more and larger trigger lists have to be carefully assessed against a training set for performance tuning[3]. Yet in the absence of satisfactory training data, the lists tested here give a very satisfactory baseline. The performance of the different lists is close and varies in predictable ways. Thus, the cCore list, which is the smallest list drawn from biomedical texts out performs larger lists due to a smaller number of *false positives*. But its performance out of domain plummets. In contrast, the NEGATOR lists which perform quite consistently across the different domains in terms of recall, have low precision against any gold standard. Unlike in information retrieval where one usually accepts the harmonic mean as indicative, here one may want to maximize coverage or precision dependent on the task. Comparing the expected behaviour of different trigger lists on different genres is an effective gauge for such fine-tuning.

[3]Precision is here defined only with respect to matching the gold standard annotations, which frequently do not include certain negation forms. For applications that prioritize recall, the *false positive* inclusion in precision errors may be too stringent.

Chapter 4

Negation Scope

This chapter describes the approach for determining the *negation scope* of a negation trigger. The basic premise for our approach to negation scope detection was to develop a general and linguistically motivated negation scope detection module, which could be used in any text processing pipeline for various applications. Thus, the NEGATOR scope detection module was developed using formal well-established patterns, based on syntax, as discussed extensively in (Givón, 1993; Huddleston and Pullum, 2002). The module consists of heuristics based on identifying these syntactic patterns. This rule-based approach proves to be quite effective, extensible and flexible. These heuristics are implemented using the parse tree and dependency graphs. The final implementation, developed as a stand alone module for the GATE environment has heuristics for both *narrow* or *wide* negation scope models, as different applications down stream can select different interpretations.

4.1 The Syntax of Negation

The scope of a negation trigger is reflected in and determined by the syntactic structure of the sentence. In this section, several issues along with a few prevailing syntactic patterns are discussed. These insights were integral in forming the linguistic foundation for the implementation of the NEGATOR scope detection module.

There are two main considerations when determining the negation scope for *explicit negation* triggers. (Huddleston and Pullum, 2002, pg 788-790) distinguish these considerations as two distinct forms of negation: *Verbal and Non Verbal*. They are described in some depth in the following paragraphs:

4.1.0.1.1 Verbal Negation is the most common type of negation in English and occurs when the negation trigger (i.e. *not, never*) is grammatically associated with the main verb. In Example (26), the negation trigger is associated with the main verb *promise* and the

resulting scope of negation is the verb phrase.

(26) She did[n't] promise to help him. [Verbal]

This interpretation is referred to as *narrow scope*, as it does not include the subject (*She*) in the scope of negation. As demonstrated in Example (27a) there are at least two possible interpretations for the sentence "The French King is not bald." The *narrow scope* interpretation: (27b), presupposes the existence of the *King of France* (the subject) and consequently the negation trigger *not* affects only the part of the statement referring to his *baldness*. In contrast the *wide scope* interpretation: (27c), also denies the existence of the *King of France*. (Givón, 2001, pg 379) discusses that Linguists will usually consider the subject as presupposed and exclude it from the syntactic scope of negation as illustrated in Example (27b). In our implementation of the NEGATOR scope detection module, we adopt by default this more general *narrow scope* interpretation. However, as will be demonstrated later in this chapter, the NEGATOR scope detection module is flexible in its design, and we quite easily were able to implement *wide scope* heuristics.

(27) (a) The French King is not bald.
(b) $\exists$ *x: FrenchKing(x)* $\wedge$ $\neg$ *bald(x).* [narrow scope]
(c) $\neg$ $\exists$ *x: FrenchKing(x)* $\wedge$ *bald(x).* [wide scope]

4.1.0.1.2 Non Verbal Negation occurs when the negation trigger is grammatically associated with a dependent of the main verb. In (28a), the negation trigger *no* combined with *body* forms the negative pronoun *nobody*, and is a dependent of the verb *promise*. Here, the scope of negation is determined to be the entire clause, since the possibility of the action occurring (defined by the main verb) will also be denied due to the implied absence (non-existence) of the subject. In (28b), the negation trigger *no* is grammatically associated with the direct object and therefore the scope is determined to be *money*.

(28) (a) [No]body promised to help him. [Non Verbal]
(b) She promised him [no] money. [Non Verbal]

Sentence(28b), repeated in (29a) is considered to be equivalent in meaning to the *verbal negation* in (29b) (Huddleston and Pullum, 2002). Therefore, in our approach to negation scope detection, in cases like (29a), we will interpret the sentence as a verbal negation, and include the verb into the scope of the negation.

(29) (a) She promised him [no] money. [Non Verbal]
(b) She did [not] promise him any money. [Verbal]

The determination of negation scope for *explicit negation* triggers is most often connected with the linear ordering of lexical units (Huddleston and Pullum, 2002, pg 474). This is also exemplified in examples (26 - 28), where the scope of negation is determined to be all the lexical units that follow the negation trigger: defining those lexical units as *not asserted*. In example (26), the negation trigger is in the matrix clause and yields *clausal* negation. However verbal negation does not always equate to clausal negation. In example (30), the negation trigger is within the subordinate clause and therefore the scope of negation does not include the main verb.

(30) (a) She promised [not] to help him.

There exist quite a few cases where these general syntactic patterns of clausal or sub clausal negation may be overridden. Example (31a) demonstrates the coordination of two clauses. Here the scope of the negation trigger is only over the first clause and not over the second one. In Example (31b), the negation scope is in fact narrowed down to the complement of the verb *think* - which corresponds to *I think that he is not coming*. This phenomenon is called *transferred negation* (Quirk et al., 1985) or *negative raising* (Horn, 1989); where a number of verbs (i.e. *believe, think, suppose ...*), when negated, allow the narrowing of negation scope to their complement clause. In (31c): another case of non verbal negation where the *not* scopes only over the subordinate clause which here precedes the main clause.

(31) (a) Liz did[n't] delete the backup file and Sue wrote the report.
(b) I do [not] think he is coming.
(c) [Not] an accomplished dancer, he moved rather clumsily.

There are other lexical items in natural language that are also scope-bearing: like: quantifiers (*some, all, any ...*), modal auxiliary verbs (*should, may, could, ...*). Negation may combine in a given sentence with other scope bearing elements like these. Again, the default case is that the one which comes first will generally have scope over the one which comes later, as depicted in example (32).

(32) (a) He has[n't] got [many] friends. [negation out-scopes the quantifier].
(b) [Many] people did[n't] attend the show. [quantifier out-scopes the negation].

However, there are again cases where this default pattern is overridden. In (33), the negation trigger has a wide scope reading (it out-scopes *Everybody*), and its scope only extends until the end of the first coordinated clause.

(33) Everybody did[n't] support the proposal, but most did.

When both a quantifier and a negation are present in the sentence as in (33), the NEGATOR scope detection module will determine a *narrow* scope interpretation for negation, and interaction with quantifiers has yet to be implemented.

4.1.0.1.3 Syntax of Affixal Negation: Here, the scope of an affixal negation trigger is defined as *constituent negation* (Tottie, 1991). The negation scope is only over the root word and does not affect any complements of the affixed term, thus in Example (34a) the will is strengthened by explicitly negating the possibility of its *wavering*.

(34) (a) We will never bend our [un]wavering will. [constituent negation]
(b) We will never bend our [un]wavering will. [wide scope negation]

This is the default option for determining the negation scope of an affixal negation trigger in the NEGATOR scope detection module. However, as the component is designed to be flexible, the option for determining a wide scope interpretation as exemplified in (34b) was also implemented. These extensions implemented in NEGATOR scope detection module will be described and demonstrated in the next section.

4.1.0.1.4 Syntax of Implicit Negation: When the negation trigger is a verb *(fail)* or a nominalization *(absence of)*, the scope of negation is determined to be the complement clause as depicted in (35).

(35) The student [failed] the exam.

4.2 NEGATOR Scope Heuristics

The implementation for identifying the negation scope of a given negation trigger is rule based. The basis for these heuristics are inspired by the implementation discussed in (Kilicoglu and Bergler, 2009). This system was designed according to BioNLP09 Shared Task requirements. The BioNLP09 Shared Task defines the problem of negation scope detection as identifying negated *bio-events*, and not the linguistic scope of a negation trigger. In contrast, the NEGATOR scope detection module has a more general purpose, to detect the syntactic scope of a negation trigger. The heuristics defined in (Kilicoglu and Bergler, 2009) were used as a basis for the heuristics defined here, and were adapted and extended quite extensively. The theoretical foundation for the heuristics that were developed follow closely the syntactic patterns relevant to negation described in (Huddleston and Pullum, 2002, pg 799-812). NEGATOR relies not only on available lexical information associated with the negation trigger but also on the syntactic structure of the sentence. The syntactic structure is derived by first parsing the text for the *constituent trees* and subsequently the

collapsed syntactic dependency relations after running the dependency extraction module (de Marneffe et al., 2006).

A comprehensive set of heuristics was implemented for *narrow* negation scope, where the subject constituent is excluded. (Morante and Daelemans, 2012a) have recently presented new a corpus of Conan Doyle stories annotated with negation triggers and scope. They use a *wide* scope model for negation. Specifically, they include all the arguments relating to the event being negated, thus rendering the entire proposition to be negated, including the subject. NEGATOR's *narrow* scope model is adapted with a few added extensions for including the subject constituent. These extended heuristics for *wide* scope were also implemented using *collapsed syntactic dependency relations*. The scope heuristics were developed on Wall Street Journal texts from the MPQA Opinion corpus (Wiebe et al., 2005). Even though the MPQA Opinion corpus does contain specific features in their annotations related to the phenomena of negation, it does not contain gold annotations specifically for negation triggers and negation linguistic scope. Thus, the heuristics were implemented by analyzing identified syntactic patterns present in sentences containing negations, and not according to specific gold standard annotations.

Given a sentence with a negation trigger, the parse tree and its corresponding dependency graph, the next two paragraphs illustrate how NEGATOR determines the scope of a negation.

Example of determining *narrow* negation scope: Consider the sentence in (36a):

(36) (a) ***Sentence:*** *The woman did* ***not*** *give the book to the boy*

(b) ***Candidate scope:*** *The woman [$_{VP}$ did not give the book to the boy]*

(c) ***Narrow scope:*** *The woman did not [$_{VP}$ give the book to the boy]*

The *narrow* syntactic negation scope of an identified negation trigger is determined in a two step process. The first step involves identifying the corresponding scope heuristic. Each heuristic specifies a syntactic dependency relation path between a given negation trigger and another lexical term in the sentence. This lexical term is considered to be the term *directly affected* by the negation trigger. The negation trigger in (36a) is the adverb *not*. The parse tree for (36a) is depicted in Figure 8, and the dependency graph in Figure 9.

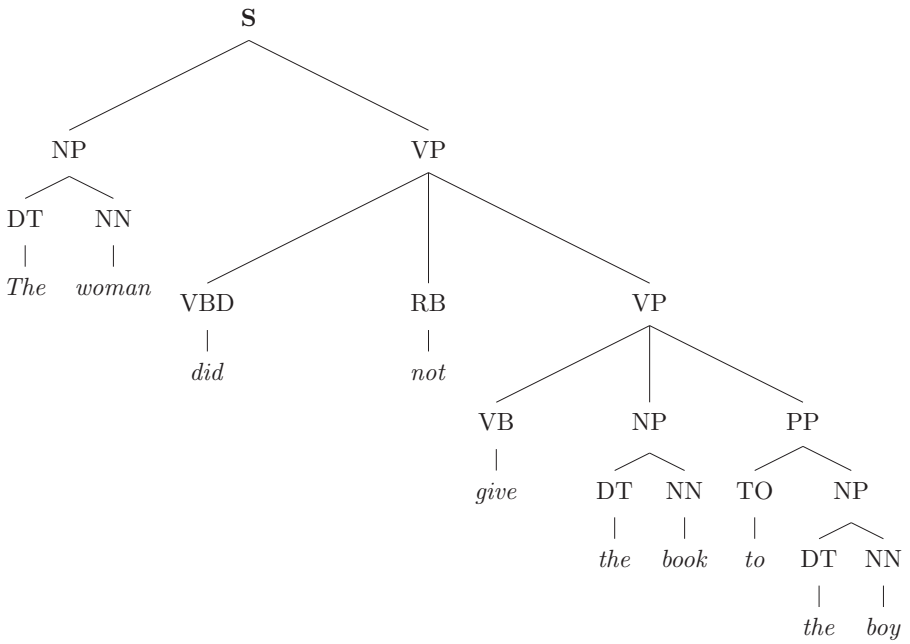

Figure 8: Parse Tree for: The woman did not give the book to the boy

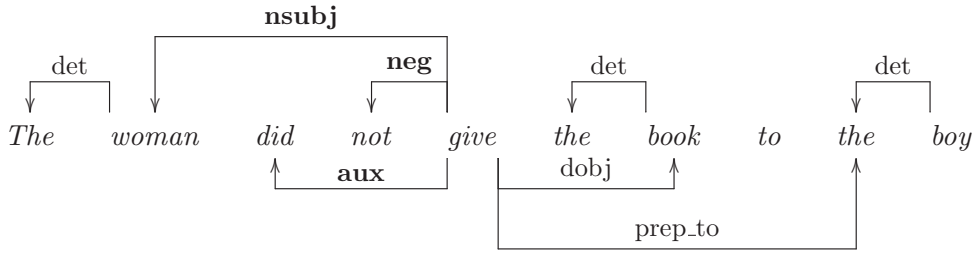

Figure 9: Dependency Graph for: The woman did not give the book to the boy

Figure 9 shows that there exists the *neg* relation between the verb *give* and *not*. Thus, the heuristic corresponding to the *neg* relation is identified. The next step requires the parse tree of the given sentence, Figure 8. The task here is to determine from this associated parse tree which constituent contains both the negation trigger (the dependent in the *neg* relation) and in this case the verb (the governor in the *neg* relation). The parse tree in Figure 8 specifies that this constituent is the verb phrase *[$_{VP}$ did not give the book to the boy]* , (36b). This verb phrase is determined to be the *candidate narrow* scope span.

Usually, scope is to the right of the trigger term, and since the negation trigger is not determined to be a part of the negation scope, the inner verb phrase constituent *[$_{VP}$ give the book to the boy]* is extracted. This constituent is identified as the *narrow scope span*, (36c).

Example of determining *wide* negation scope: Example (37a) presents sentence from (36) and its resulting *narrow* scope negation span is repeated in (37b).

(37) *(a)* ***Sentence:*** *The woman did not give the book to the boy*

(b) ***Narrow scope:*** *The woman did not [$_{VP}$ give the book to the boy]*

(c) ***Ext. Scope:*** *[$_{NP}$ The woman] did not give the book to the boy*

(d) ***Ext. Scope with aux:*** *[$_{NP}$ The woman] [did] not give the book to the boy*

(e) ***Final Wide scope:*** *[$_{NP}$ The woman] [did] not [$_{VP}$ give the book to the boy]*

In order to identify the *wide* negation scope span for (37a), the identification of the constituent containing the subject is required. This is accomplished by identifying an additional heuristic which specifies a dependency path starting from the lexical term that was *directly* affected by the negation trigger. This term is a member of the dependency relation used for the determination of the *narrow* scope, here between *give* and *not*. Therefore, the next step requires identification of a dependency relation between *give* and the subject *woman*. Figure 9, shows the *nsubj* relation between *give* and *woman*. Next, the noun phrase constituent containing *woman* is extracted for the first scope span extension (37c).

(Morante and Daelemans, 2012a) add the auxiliary term *did* into the wide scope span. As observed in Figure 9, the *aux* relation between *give* and *did* establishes this connection. Another scope span for this term is created, as shown in (37d). The final *wide* negation scope is illustrated in (37e), which consists of the *narrow* scope and the two additional scope spans.

Examples (36) and (37) use both the parse tree and the corresponding dependency graph to identify the scope of a negation trigger in a sentence. The next section describes in detail each of the heuristics implemented. All the heuristics follow the same approach as in the previous examples: by identifying distinct dependency relation paths starting with those associated with the negation trigger, and consequently extracting and pruning the relevant constituent(s) from the parse tree in order to represent the resultant negation scope (wide and narrow).

The remainder of this section describes the implemented heuristics organized according to the possible lexical categories of a negation trigger.[1] The implementation of the *wide* scope is closely coupled with that of *narrow* scope and for every lexical category of a negation trigger, we present the heuristics for the determination of the relevant *narrow* scope spans and optionally[2] those for *wide* scope.

[1] The reader may refer to Appendix A for the definitions of the dependency relations used in the heuristics described in the next subsections.

[2] *Wide* negation scope is optional, in the sense that the user may choose to only run the *narrow* scope, or they may choose to also have *wide* scope. In the next section we will discuss the details of the implementation of the NEGATOR Scope Detection GATE module which allows for this functionality.

4.2.1 Adverbs *(i.e. not, never, neither)*

There are three distinct heuristics for determining the *narrow* scope when the negation trigger is an adverb. The first is the most common: when the *neg* dependency relation is identified. The second is when the negation trigger participates in a coordinate construction (*but not*) which is identified by the collapsed *conj_neg_cc* dependency relation. The final heuristic discussed here is to detect the *passive noun subject constituent* in instances of verbal negation. For each *narrow* scope heuristic, the additional heuristics for determining the *wide* scope are also described.

4.2.1.1 The *neg* relation

4.2.1.1.1 narrow scope: In the majority of cases the explicit negation triggers (*not, never*), are identified by the parser as a negation term, and consequently the *neg* dependency relation is created which connects this negation term with the word that it grammatically modifies. This *neg* dependency relation is used to identify the narrow scope. Usually, the negation trigger will modify the main verb in the sentence, although there are cases where it will modify other terms in the sentence. Regardless of the *type* of the modified term, the *same* heuristic will be used. Therefore, this heuristic has been implemented in a modular fashion, and no matter what the modified term is, the constituent which dominates both the negation trigger and the modified term is determined to be the candidate narrow negation scope. The following list exemplifies the most general cases for when the *neg* dependency relation heuristic will be used to determine the *candidate narrow* negation scopes and the final narrow pruned scope (underlined):

i: *neg* relation identified between the negation trigger and the main verb.

(38) ***Sentence:*** *. . . the creditors do not see any signals of concrete support from the G-7.*
Dep Relation: neg(see, not)
Candidate Narrow Scope: *. . . the creditors [$_{VP}$ do not see any signals of concrete support from the G-7].*
Narrow Scope: *. . . the creditors [$_{VP}$ do [not] see any signals of concrete support from the G-7].*

ii: *neg* relation identified between the negation trigger and the verb in the subordinate clause.

(39) ***Sentence:*** *He claimed that Bob had not offered bribes to any official.*
Dep Relation: neg(offered, not)
Candidate Narrow Scope: *He claimed that Bob [$_{VP}$ had not offered bribes to any official].*
Narrow Scope: *He claimed that Bob [$_{VP}$ had [not] offered bribes to any official].*

iii: *neg* relation identified between the negation trigger and a noun. In this case, NEGATOR will also check if the governor node of the *neg* dependency relation is a member of a *cop* relation. If this condition is satisfied and the dependent node has the infinitive form: *to be*, then not

only will the scope span to the right of the trigger be identified but the copular verb as well, resulting in two discontinuous scope spans. If no such *cop* dependency relation exists, then the resultant scope span will be the one identified by the *neg* relation.

(40) ***Sentence:*** *Taiwan is not a U.N. member.*
Dep Relation: neg(member, not)
Dep Relation: cop(member, is)
Candidate Narrow Scope: *Taiwan [$_{VP}$ is not a U.N. member].*
Narrow Scope: *Taiwan [$_{VP}$ is [not] a U.N. member].*

iv: *neg* relation identified between the negation trigger and a preposition:

(41) ***Sentence:*** *Not for the first time, she felt utterly betrayed.*
Dep Relation: neg(for, Not)
Candidate Narrow Scope: *[$_{PP}$ Not for the first time], she felt utterly betrayed.*
Narrow Scope: *[$_{PP}$[Not] for the first time], she felt utterly betrayed.*

v: *neg* relation identified between the negation trigger and an adjective. In this case, NEGATOR will also check if the governor node of the *neg* dependency relation is also a member of a *cop* relation. If this condition is satisfied and the dependent node has the infinitive form: *to be*, then not only will the scope span to the right of the trigger be identified but the copular verb as well, resulting in two discontinuous scope spans. If no such *cop* dependency relation exists, then the resultant scope span will be the one identified by the *neg* relation as illustrated in Example (42)[3].

(42) ***Sentence:*** *He seemed not entirely honest.*
Dep Relation: neg(honest, not)
Candidate Narrow Scope: *He seemed [$_{ADJP}$ not entirely honest].*
Narrow Scope: *He seemed [$_{ADJP}$[not] entirely honest].*

vi: *neg* relation identified between the negation trigger and and subject:

(43) ***Sentence:*** *Not all students regarded it as a success.*
Dep Relation: neg(Not, students)
Candidate Narrow Scope: *[$_{NP}$ Not all students] regarded it as a success.*
Narrow Scope: *[$_{NP}$ [Not] all students] regarded it as a success.*

vii: *neg* relation identified between the negation trigger and an *is*:

(44) ***Sentence:*** *There is not a tree in the garden.*
Dep Relation: neg(is, not)
Candidate Narrow Scope: *There [$_{VP}$ is not a tree in the garden].*
Narrow Scope: *There [$_{VP}$ is [not] a tree in the garden].*

[3]The *cop* relation does exist between *honest* and seem, in Example (42), however *seem* does not have the infinitive form *to be* and therefore will not be a part of the resultant negation scope

4.2.1.1.2 wide scope: The narrow scope for a negation trigger is identified by the *neg* dependency relation, the corresponding wide scope heuristics are implemented to identify either the corresponding subject or pleonastic pronoun constituents. In the majority of cases, the *nsubj* relation is identified between the governor member of the *neg* relation and the subject term. However, this is not always the case, and therefore the following list (i-iii) exemplifies the possible scenarios:

i: The *nsubj* relation is identified between the governor term of *neg* relation and subject. The noun phrase constituent which dominates this subject is determined to be a wide scope span. If the subject found is the governor of a collapsed *prep* relation,*rc_mod* relation or a *part_mod* relation, then the noun phrase constituent is extracted which contains also the prepositional phrase, relative clause or participial clause, respectively. Additionally, the *aux* relation is identified between the governor term of the *neg* relation and the corresponding auxiliary item. This *aux* term is marked as a second wide scope span. Examples (45-47) illustrate the above mentioned cases. The final aggregated wide scope spans are underlined, and the negation trigger in square brackets.

(45) ***Sentence:*** *On the other hand, for now, the creditors do not see any signals of concrete support from the G-7.*

Dep Relation for narrow scope: neg(see, not)
Narrow Scope: *. . . the creditors do [not] see any signals of concrete support from the G-7.* — *narrow*

Dep Relation for wide scope I: nsubj(see,creditors)
Wide Span I: *. . . [$_{NP}$ the creditors] do not see any signals of concrete support from the G-7.*
Dep Relation for wide scope II: aux(see,do)
Wide Span II: *. . . the creditors ⟨do⟩ not see any signals of concrete support from the G-7.*
Wide Scope: *. . . the creditors do [not] see any signals of concrete support from the G-7.* — *wide*

Example (46) demonstrates a case where the relative clause headed by 'who' needs to be included in the wide scope span and therefore the noun phrase which contains both the relative clause and subject is extracted:

(46) ***Sentence:*** *The girl who wore the white shirt did not wave to us.*

Dep Relation for narrow scope: neg(wave, not)
Narrow Scope: *The girl who wore the white shirt did [not] wave to us.* — *narrow*

Dep Relation for wide scope I: nsubj(wave,girl) and rc_mod(girl,wore)
Wide Span I: *[$_{NP}$ The girl who wore the white shirt] did not wave to us.*
Dep Relation for wide scope II: aux(wave,did)
Wide Span II: *The girl who wore the white shirt ⟨did⟩ not wave to us.*
Wide Scope: *The girl who wore the white shirt did [not] wave to us.* — *wide*

Example (47) demonstrates a case where the participial verb clause headed by *sitting* needs to be included in the wide scope span and therefore the noun phrase which contains both the this clause and subject is extracted:

(47) ***Sentence:*** *The girl sitting at the table did not wave to us.*

Dep Relation for narrow scope: neg(wave, not)
Narrow Scope: *The girl sitting at the table did [not] wave to us.* — *narrow*

Dep Relations for wide scope I: nsubj(wave,girl) and part_mod(girl,sitting)
Wide Span I: $[_{NP}$ *The girl sitting at the table] did not wave to us.*
Dep Relation for wide scope II: aux(wave,did)
Wide Span II: *The girl sitting at the table* ⟨*did*⟩ *not wave to us.*
Wide Scope: *The girl sitting at the table did [not] wave to us.* — *wide*

ii: The *expl* relation identified between the governor term of *neg* relation and the existential *There*. Simply, the dependent term: *There* is determined to be the additional wide scope span as illustrated in (48) .

(48) ***Sentence:*** *There is not a tree in the garden.*

Dep Relation for narrow scope: neg(is, not)
Narrow Scope: *There is [not] a tree in the garden.* — *narrow*

Dep Relation for wide scope: expl(is,There)
Wide Span: ⟨*There*⟩ *is not a tree in the garden.*
Wide Scope: *There is [not] a tree in the garden.* — *wide*

iii: If there is no subject or pleonastic pronoun linked directly with governor term of the *neg* relation, it may be the case that the negated clause is part of a pair of coordinate clauses, and the subject or pleonastic pronoun is located in the first non-negated co-ordinated clause. In this case, as exemplified in (49) the narrow scope is determined as usual. However, identifying the subject constituent requires an extra step than that of example (45). Specifically, a collapsed *conj* dependency relation is identified between the governor term of the *neg* relation and the corresponding term in the first coordinate clause. If this term is found, then the next step is to detect *nsubj* relation between the coordinate term and the subject. If successful, the noun phrase constituent containing both the subject and prepositional phrase is determined to be the wide scope span. Additionally, the *aux* relation is identified between the governor of the *neg* relation and the corresponding auxiliary item, (49).

(49) ***Sentence:*** *The shadow has departed and will not return.*

Dep Relation for narrow scope: neg(return, not)
Narrow Scope: *The shadow has departed and will [not] return.* — *narrow*

Dep Relations for wide scope I: conj_and(departed,return) and nsubj(departed,shadow)
Wide Span I: *[$_{NP}$ The shadow] has departed and will not return.*
Dep Relation for wide scope II: aux(return,will)
Wide Span II: *The shadow has departed and ⟨will⟩ not return.*
Wide Scope: *The shadow has departed and will [not] return.* — *wide*

4.2.1.2 The *conj_neg_cc and conj_and* relations

4.2.1.2.1 narrow scope: The negation triggers (*not, never*) may also participate in a coordinate construction. This heuristic is used when no *neg* relation is found with the negation trigger. The most common cases are *but not* or *and not* as illustrated in the following list:

i: In the case of *but not* the collapsed *conj_neg_cc* dependency relation will be is identified between the two terms in the sentence connected by the *but* conjunction term, and the resultant candidate narrow scope of negation is the constituent dominating the *dependent* member of the dependency relation and the negation trigger. This case is demonstrated in Example (50). The final narrow scope span is the narrow candidate scope without the negation trigger present.

(50) ***Sentence:*** *They had invited Jill but not her husband.*
Dep Relation: conj_negcc(Jill, husband)
Candidate Narrow Scope: *They had invited [$_{NP}$ Jill but not her husband].*
Narrow Scope: *They had invited [$_{NP}$ Jill but [not] her husband].*

ii: In the case of *and not*, a collapsed *conj_and* dependency relation between the *not* and a term in the non-negated co-ordinate clause is identified. The resultant candidate narrow scope of negation is the constituent dominating both terms of this relation. This case is demonstrated in Example (51). The final narrow scope span is the narrow candidate scope without the negation trigger present.

(51) ***Sentence:*** *It was the cat and not the dog who caught the mouse..*
Dep Relation: conj_and(cat, not)
Candidate Narrow Scope: *It was [$_{NP}$ the cat and not the dog who caught the mouse].*
Narrow Scope: *It was [$_{NP}$ the cat and [not] the dog who caught the mouse].*

4.2.1.2.2 wide scope: The heuristic for determining wide scope is when the governor of the *conj_neg_cc* or *conj_and* relations are both direct objects. In this case, the approach is to identify the governor (the main verb) of the *dobj* relation, and if found to check whether this term is also connected to the subject constituent by the *nsubj* dependency relation. If found, then the dependent

term of the *nsubj* relation (the subject) is extracted and the constituent which dominates the subject and the governor of the original *conj* relation is determined to be the candidate wide scope span. The wide scope span will be the candidate wide scope up to but not including the governor conjunct term. Example (52) illustrates this case:

(52) ***Sentence:*** *They had invited Jill but not her husband.*
Dep Relation for narrow scope: conj_negcc(Jill, husband)
Narrow Scope: *They had invited Jill but [not] her husband.*] *narrow*
Dep Relation for wide scope: dobj(invited, Jill) and nsubj(invited, They)
Wide Span:*[S They had invited Jill but not her husband.]*
Wide Scope: *They had invited Jill but [not] her husband.*] *wide*

4.2.1.3 The *nsubjpass* relation

4.2.1.3.1 narrow scope: In passive sentence constructions the object of the verb is in the subject position. Therefore, in cases of verbal negation not only is the *narrow* scope the span to the right of the trigger, but also the passive subject constituent is a second discontinuous *narrow* scope span. The passive noun subject is identified by the *nsubjpass* dependency relation between the verb and the passive subject. The candidate narrow scope of negation is the parent noun phrase constituent containing the passive noun subject. If the passive subject found is the governor of a collapsed *prep* relation: then the noun phrase constituent is extracted which contains also the prepositional phrase. Example (53) illustrates the default case and the final narrow scope contains two discontinuous scope spans, (the right span having been determined by a previous heuristic):

(53) ***Sentence:*** *Extensions can't be granted for filing tax returns due Oct. 31.*
Dep Relation: nsubj_pass(granted,Extensions)
Candidate Narrow Scope: *[NP Extensions] [VP ca[n't] be granted for filing tax returns due Oct 31].*
Narrow Scope: *[NP Extensions] [VP ca[n't] be granted for filing tax returns due Oct. 31].*

For this heuristic there is no need to determine a wide scope span as the original subject (if there is one) will usually be already in the scope of negation triggered by the *neg* dependency relation.

4.2.2 Determiners *(i.e. no,neither)*

In the majority of cases the explicit negation trigger *no* is identified by the parser as a determiner, and consequently the *det* dependency relation is created which connects this determiner and the word that it grammatically modifies. The following list exemplifies the most general cases for when the *det* dependency relation heuristic will be used to determine the candidate narrow negation scopes (in angle brackets), and the narrow pruned scope (underlined). For each narrow scope heuristic, also described are the corresponding heuristics (if any) for determining the wide scope.

4.2.2.1 The *det* relation between neg trigger and subject

The *det* relation is identified between the negation trigger and the subject. In (54), the interpretation is that since the subject of the sentence did not complete the act of *giving the book to the boy*, both the subject and the verb phrase are in the negation scope. By default a wide negation scope is assigned here, not just the constituent dominated by the negation trigger and the subject. This is accomplished by further checking for the *nsubj* dependency relation between the subject and the main verb. If found, the candidate scope of negation is the constituent which dominates the negation trigger and the main verb. Example (54) illustrates this case:

(54) ***Sentence:****No woman gave the book to the boy.*
Dep Relation: det(woman, No)
Dep Relation: nsubj(gave, woman)
Candidate Narrow Scope: *[$_S$⟨No woman gave the book to the boy].*
Narrow Scope: *[$_S$[No] woman gave the book to the boy].*

4.2.2.2 The *det* relation between neg trigger and direct object

4.2.2.2.1 narrow scope: The *det* relation is identified between the negation trigger and the direct object. Here, the heuristic will also check if there exists one of the collapsed *prep* dependency relations (*prep_at*) between the direct object and the preposition. If found: the attached prepositional phrase will be included in the candidate negation scope, exemplified in (55 a). Otherwise, just the constituent which dominates the negation trigger and the direct object will be the candidate narrow scope as shown in (55 b). For both cases, a second scope span is may be created in order to include the verb in the narrow negation scope. This scope span will be identified by the *dobj* relation between the direct object and the main verb. The resultant second narrow scope span will be the verb (see (55 b)).

(55) (a) ***Sentence:*** *The woman gave no book to the boy.*
Dep Relation: det(book, no)
Dep Relation: prep_at(gave, boy)
Dep Relation: dobj(gave, book)
Candidate Narrow Scope: *The woman [$_{VP}$ gave no book to the boy].*
Narrow Scope: *The woman [$_{VP}$ gave [no] book to the boy].*

(b) ***Sentence:*** *The woman had no books.*
Dep Relation: det(books, no)
Dep Relation: dobj(had, books)
Candidate Narrow Scope: *The woman had [$_{NP}$ no books].*
Narrow Scope: *The woman had [$_{NP}$[no] books].*

4.2.2.2.2 wide scope: Given that the *narrow* scope for a negation trigger is identified by the *det* dependency relation with the direct object: the corresponding wide scope heuristics identify the corresponding subject. In the majority of cases, the *nsubj* relation is identified by identifying the

nsubj relation between the governor of the *dobj* relation and the subject. A less frequent case is when the negated clause is part of a pair of coordinate clauses, and identifying the subject requires the extra step of first identifying one of the *conj* dependency relations. The following list illustrates the two different cases:

i: The *nsubj* relation is identified between the governor term of *dobj* relation and subject. The noun phrase is extracted which contains the subject, representing the wide scope span as seen in (56).

(56) ***Sentence:*** *The woman had no books.*
Dep Relations for narrow scope: det(books, no) and dobj(had, books)] *narrow*
Narrow Scope: *The woman had [no] books.*]
Dep Relation for wide scope: nsubj(had,woman)]
Wide Span: *[$_{NP}$ The woman] had no books.*] *wide*
Wide Scope: *The woman had [no] books.*]

ii: The second heuristic for determining wide scope applies when the governor of the *dobj* relation has no relation to the subject but is part of a pair of coordinated clauses. Here, the first step is to check if this governor term is a dependent of a *conj* relation. If yes, then the next step checks whether the corresponding governor term is linked to the subject by the *nsubj* relation. If found, then the noun phrase containing the subject is the candidate wide scope span. Example (57) illustrates this case:

(57) ***Sentence:*** *Jane was late for work and had no excuse.*
Dep Relations for narrow scope: det(excuse, no) and dobj(had, excuse)] *narrow*
Narrow Scope: *Jane was late for work and had [no] excuse.*]
Dep Relations for wide scope: conj_and(was, had) and nsubj(was, Jane)]
Wide Span: *[$_{NP}$ Jane] was late for work and had no excuse.*] *wide*
Wide Scope: *Jane was late for work and* had *[no] excuse.*]

4.2.2.3 The *det* relation between neg trigger and prepositional object

4.2.2.3.1 narrow scope: The *det* relation identified between the negation trigger and the prepositional object. In this case as observed in (58), the resultant candidate narrow scope of negation is determined to be the constituent (usually a prepositional phrase) which dominates the negation trigger and the prepositional object.

(58) ***Sentence:*** *The woman gave the book to no boy.*
Dep Relation: det(boy, no)
Candidate Narrow Scope: *The woman gave the book [$_{PP}$ to no boy].*
Narrow Scope: *The woman gave the book [$_{PP}$to [no] boy].*

4.2.2.3.2 wide scope: Given that the narrow scope for a negation trigger is identified by the *det* dependency relation with the prepositional object: the corresponding wide scope heuristic is used if there is a corresponding subject. The process is implemented similar to example (56), except that in the majority of cases identifying the subject requires at least two steps. First: to identify the main verb by a *prep* relation between the governor of the *det* relation and the verb. If found: the *nsubj* relation is identified between the verb (the governor of the *prep* relation) and the subject. Upon success: the wide scope span is the noun phrase constituent which contains this subject. A second wide scope span is again marked for the verb as illustrated in (59):

(59) ***Sentence:*** *The woman gave the book to no boy.*
Dep Relation for narrow scope: det(boy, no)
Narrow Scope: *The woman gave the book to [no] boy.* — *narrow*
Dep Relation for wide scope I: prep_to(gave, boy) and nsubj(gave, woman)
Wide Span: *[$_{NP}$ The woman] gave the book to no boy.*
Wide Span II: *The woman ⟨gave⟩ the book to no boy.*
Wide Scope: *The woman gave the book to [no] boy.* — *wide*

4.2.2.4 The *det* relation identified with negation trigger

4.2.2.4.1 narrow scope: The *det* relation is also identified between the negation trigger and other terms. Example (60) illustrates when the governor term of the *det* relation is also connected to a copula verb. Example (61) shows the case when the governor term is part of a sentence which contains an pleonastic pronoun. Determining the narrow scope span is the same for both examples: the constituents which dominate both the negation trigger and governor terms are extracted, and the narrow scope spans are those constituents without the negation trigger.

(60) ***Sentence:*** *That is no use to us at the moment.*
Dep Relation: det(use, no)
Candidate Narrow Scope: *That is [$_{NP}$ no use to us at the moment].*
Narrow Scope: *That is [$_{NP}$ [no] use to us at the moment].*

(61) ***Sentence:*** *There is no sheep dog running in the fields .*
Dep Relation: det(dog, no)
Candidate Narrow Scope: *There is [$_{NP}$ no sheep dog running in the fields].*
Narrow Scope: *There is [$_{NP}$ [no] sheep dog running in the fields].*

4.2.2.4.2 wide scope: Determining the wide scope span in the case of example (60) requires identifying the *nsubj* relation, where the governor is the governor term of the *det* relation, and the dependent is the subject. If present, the noun phrase constituent is the first wide scope span. Additionally, the *cop* relation is identified, whose governor is the governor term of the *det* relation, and this copula is marked as a second wide scope span. Example (62) illustrates this case.

(62) ***Sentence:*** *That is no use to us at the moment.*

Dep Relation for narrow scope: det(use, no)
Narrow Scope: *That is [no]* <u>*use to us at the moment*</u>. — *narrow*

Dep Relation for wide scope I: nsubj(use,That)
Wide Span I: *[$_{NP}$ That] is no use to us at the moment.*
Dep Relation for wide scope II: cop(use, is)
Wide Span II: *That ⟨is⟩ no use to us at the moment.*
Wide Scope: <u>*That is*</u> *[no]* <u>*use to us at the moment*</u>. — *wide*

Determining the wide scope span in the case of example (61) requires two steps. First, the *nsubj* relation is identified where the dependent term is the governor term of the *det* relation. In this case, the governor term is also linked to the pleonastic pronoun: *There*, and consequently the *expl* dependent relation is also identified between the governor term of the *nsubj* relation with the pleonastic pronoun. If found, both terms (usually *There is*) represent the candidate wide scope span. Example (63) illustrates this case:

(63) ***Sentence:*** *There is no sheep dog running in the fields.*

Dep Relation for narrow scope: det(dog, no)
Narrow Scope: *There is [no]* <u>*sheep dog running in the fields*</u>. — *narrow*

Dep Relations for wide scope: nsubj(is,dog) and expl(is,There)
Wide Span: *⟨There is⟩ no sheep dog running in the fields.*
Wide Scope: <u>*There is*</u> *[no]* <u>*sheep dog running in the fields*</u>. — *wide*

4.2.3 Pronouns *(i.e. nothing,nobody)*

This category is similar to the *determiner* class of heuristics. Depending on which position the negation trigger occupies, a different heuristic will be used. If the pronoun is in subject position: the *nsubj* dependency relation is used. If the pronoun is a direct object, the *dobj* dependency relation is used. Finally if the pronoun occupies the prepositional object position, the corresponding collapsed *prep* dependency relation is used. These three dependency relations will all connect the negated pronoun with the main verb. Each of the these cases are delineated in the following list, where the candidate narrow negation scopes are in angle brackets), and the narrow pruned scopes are underlined. For each narrow scope heuristic, the corresponding heuristics (if any) are described for determining the wide scope.

4.2.3.1 The *nsubj/nsubjpass* relation between negation trigger and main verb

In example (64), the *nsubj* relation identified between the negated pronoun (the subject) and the main verb. By default a wide negation scope is assigned here and the resultant candidate scope of negation results in the entire phrase.

(64) ***Sentence:*** *None of the children went swimming today.*
Dep Relation: nsubj(went, None)
Candidate Narrow Scope: *[$_{S}$ None of the children went swimming today].*
Narrow Scope: *[$_{S}$ [None]* <u>*of the children went swimming today*</u>*].*

It may be the case that the constituent in the subject position in the sentence is a *passive* subject constituent. As illustrated in example (65), the *nsubjpass* relation identified between the negated pronoun (the passive subject) and the main verb. Again, a wide negation scope is assigned here and the resultant candidate scope of negation results in the entire phrase.

(65) ***Sentence:*** *Nothing had been heard of him.*
Dep Relation: nsubjpass(heard, Nothing)
Candidate Narrow Scope: *[$_S$ Nothing had been heard of him].*
Narrow Scope: *[$_S$[Nothing] had been heard of him].*

4.2.3.2 The *dobj* relation between negation trigger and main verb

4.2.3.2.1 narrow scope: The *dobj* relation is identified between the negated pronoun (the direct object) and the main verb. The resultant candidate narrow scope of negation results in the verb phrase headed by the main verb. The narrow scope results in two discontinuous scope spans in order to include the main verb in the final negation scope. This case is illustrated in (66):

(66) ***Sentence:*** *Bill gives nothing away.*
Dep Relation: dobj(gives, nothing)
Candidate Narrow Scope: *Bill [$_{VP}$ gives nothing away].*
Narrow Scope: *Bill [$_{VP}$ gives [nothing] away].*

4.2.3.2.2 wide scope: There are two different cases for determining the additional wide scope spans. In the first case, the *nsubj* relation is identified between the governor of the *dobj* relation (the main verb) and the subject. If found, then the wide scope span is the noun phrase containing the subject as illustrated in (67):

(67) ***Sentence:*** *Bill gives nothing away.*
Dep Relation for narrow scope: dobj(gives, nothing)
Narrow Scope: *Bill gives [nothing] away.* — *narrow*
Dep Relation for wide scope: nsubj(gives, Bill)
Wide Span: *[$_{NP}$ Bill] gives nothing away.*
Wide Scope: *Bill gives [nothing] away.* — *wide*

In the second case, (if the first case fails) it is possible that the negated phrase is part of a pair of coordinate clauses. Therefore, an extra step is required to identify the subject. Specifically, first: a collapsed *conj* relation is identified between the corresponding conjunction term and the main verb. If found, then the *nsubj* relation between the governor of the *conj* relation and the corresponding subject is identified, and the resulting wide scope span is the noun phrase containing the subject as illustrated in (68):

(68) ***Sentence:*** *Bill is stingy and gives nothing away.*
Dep Relation for narrow scope: dobj(gives, nothing)
Narrow Scope: *Bill is stingy and gives [nothing] away.* — *narrow*

Dep Relation for wide scope: conj_and(stingy, gives) and nsubj(stingy, Bill)
Wide Span: *[$_{NP}$ Bill] is stingy and gives nothing away.*
Wide Scope: *Bill is stingy and gives [nothing] away.* — *wide*

4.2.3.3 A collapsed *prep* relation between negation trigger and main verb

4.2.3.3.1 narrow scope: A *prep* relation is identified between the negated pronoun (the prepositional object) and the main verb. The resultant candidate narrow scope of negation is the verb phrase headed by the main verb. This case is illustrated in (69):

(69) ***Sentence:*** *Bill gave the book to nobody.*
Dep Relation: prep_to(gave, nobody)
Candidate Narrow Scope: *Bill [$_{VP}$ gave the book to nobody].*
Narrow Scope: *Bill [$_{VP}$ gave the book to [nobody]].*

4.2.3.3.2 wide scope: The heuristics for the additional wide scope spans are the same as illustrated in (67): the *nsubj* relation is identified between the main verb and the subject or then as in (68), the corresponding *conj* relation needs to be identified before.

4.2.4 Prepositions *(i.e. without, rather than, instead of)*

Here, the relevant collapsed *prepc* dependency relation is identified. If the negation trigger is *without* then the *prepc_without* dependency relation will be used. In the case of the negation triggers *rather than* or *instead of*, the *prepc_than* or *prepc_instead_of* will be used respectively. In all these cases, the candidate scope is determined to be the constituent which dominates the negation trigger and the prepositional complement or the prepositional object. In each case, the final scope is determined to be the candidate scope span without the negation trigger. The following list delineates examples for each of these cases:

4.2.4.1 prepc_without

The prepc_without relation is identified, and the negation trigger *without* is the preposition which modifies the dependent term. The resultant candidate narrow scope of negation results in the prepositional phrase headed by *without*.

(70) ***Sentence:*** *The patient was able to tolerate food without nausea or vomiting.*
Dep Relation: prepc_without(tolerate, nausea)
Candidate Narrow Scope: *The patient was able to tolerate food [$_{PP}$ without nausea or vomiting].*
Narrow Scope: *The patient was able to tolerate food [$_{PP}$ [without] nausea or vomiting].*

4.2.4.2 prepc_than

The *prepc_than* relation is identified, and the second part of this multi-word negation trigger *than* is the preposition which modifies the dependent term. The resultant candidate narrow scope of negation is the prepositional phrase starting with *rather than*:

(71) ***Sentence:*** *John went to the disco rather than going home.*
Dep Relation: prepc_than(went, going)
Candidate Narrow Scope: *John went to the disco [$_{PP}$ rather than going home].*
Narrow Scope: *John went to the disco [$_{PP}$ [rather than] going home].*

4.2.4.3 prepc_instead_of

prepc_instead_of relation identified, and the second part of this multi-word negation trigger *of* is the preposition which modifies the dependent term. The resultant candidate narrow scope of negation is the prepositional phrase starting with *instead of*.

(72) ***Sentence:*** *John ate chips instead of carrots.*
Dep Relation: prepc_instead_of(chips, carrots)
Candidate Narrow Scope: *John ate chips [$_{PP}$ instead of carrots].*
Narrow Scope: *John ate chips [$_{PP}$ [instead of] carrots].*

4.2.4.4 Default heuristic for prepositions

There is also a default heuristic in case any of the aforementioned *prep_c* heuristics are not found. Instead, if there exists the *pobj* between the negation trigger which is a preposition and its prepositional object. In this case, the candidate narrow scope is determined to be the constituent which dominates the negation trigger and the prepositional object. The final narrow scope is determined to be the candidate narrow scope span without the negation trigger as illustrated in example (73).

(73) ***Sentence:*** *Jane likes chips with or without salt.*
Dep Relation: pobj(without,salt)
Candidate Narrow Scope: *Jane likes chips [$_{PP}$ with or without salt].*
Narrow Scope: *Jane likes chips [$_{PP}$ with or [without] salt].*

4.2.4.5 Wide scope for negation triggers that are prepositions

The above mentioned cases were heuristics for determining the narrow scope spans. In all these cases there is just one possible heuristic implemented for wide scope. If the governor term of the corresponding *prep_c* relation is a member of the *nsubj* relation: then the corresponding noun phrase which contains the subject (the dependent term in the *nsubj* relation) is the additional wide scope span. It is not the case that the wide span heuristic will always be used. Example (74) illustrates a the corresponding wide scope: the extension of example (71).

(74) ***Sentence:*** *John went to the disco rather than going home.*

Dep Relation for narrow scope: prepc_than(went, going)
Narrow Scope: *John went to the disco [rather than] going home.* — *narrow*

Dep Relation for wide scope: nsubj(John,went)
Wide Span: ⟨*John*⟩ *went to the disco rather than going home.*
Wide Scope: *John went to the disco [rather than] going home.* — *wide*

4.2.5 Verbs with non-finite complement *(i.e. failed to)*

4.2.5.1 narrow scope:

This heuristic is implemented for verbs that are also negation triggers. The *xcomp* dependency relation is identified when the negation verb trigger is linked to a non finite embedded verb. The resultant narrow candidate scope of negation is the verb phrase headed by the negation verb trigger and the final narrow scope is the same scope span, except without the negation trigger as exemplified in (75):

(75) ***Sentence:*** *The White House failed to act on the domestic threat from al Qaeda prior to September 11, 2001.*
Dep Relation: xcomp(failed, act)
Candidate Narrow Scope: *The White House [$_{VP}$ failed to act on the domestic threat from al Qaeda prior to September 11, 2001].*
Narrow Scope: *The White House [$_{VP}$ [failed] to act on the domestic threat from al Qaeda prior to September 11, 2001].*

4.2.5.2 wide scope:

There is one heuristic implemented for the additional wide scope span: as illustrated in (76): to identify if the negation verb trigger is the governor of an *nsubj* relation. If successful, then the dependent token (the subject constituent) is extracted. The resultant wide scope span is the noun phrase which dominates this subject constituent.

(76) ***Sentence:*** *The White House failed to act on the domestic threat from al Qaeda prior to September 11, 2001.*

Dep Relation for narrow scope: xcomp(failed, act)
Narrow Scope: *The White House [failed] to act on the domestic threat. . .*. — *narrow*

Dep Relation for wide scope: nsubj(failed, House)
Wide Span: ⟨*The White House*⟩ *failed to act on the domestic threat . . . .*
Wide Scope: *The White House [failed] to act on the domestic threat. . .*. — *wide*

4.2.6 Verbs with direct object *(i.e. avoid)*

4.2.6.1 narrow scope:

A second heuristic implemented for verbs that are also negation triggers. The *dobj* dependency relation was already used in the case of negated pronouns where the dependent term of the dependency relation is the pronoun (refer to (66)). However, in the case where a negation verb trigger is identified as having a direct object in its complement, the *dobj* dependency relation is created: linking the verb with the direct object. Here, the negation verb trigger is the governor term in the relation. Example (77) demonstrates this case, where the resultant candidate narrow scope of negation is the verb phrase headed by the implicit negation verb *avoid*, and the final scope is the same scope span except without the negation trigger.

(77) ***Sentence:*** *We managed to avoid any further delays.*
Dep Relation: dobj(avoid, delays)
Candidate Narrow Scope: *We managed to [$_{VP}$ avoid any further delays].*
Narrow Scope: *We managed to [$_{VP}$ [avoid] any further delays].*

4.2.6.2 wide scope:

There are two heuristics implemented for the additional wide scope span: the first as illustrated in (78): where the negation verb trigger is the governor of an *nsubj* relation. The resultant wide scope span is the noun phrase which dominates this subject constituent (the dependent node).

(78) ***Sentence:*** *The plane narrowly avoided disaster when one of the engines cut out on take-off.*
Dep Relation for narrow scope: dobj(avoided, disaster)
Narrow Scope: *The plane narrowly [avoided] disaster when. . . .* — *narrow*
Dep Relation for wide scope: nsubj(avoided, plane)
Wide Span: ⟨*The plane*⟩*narrowly avoided disaster when. . . .*
Wide Scope: *The plane narrowly [avoided] disaster when. . . .* — *wide*

The second heuristic, is when the negation verb trigger is the dependent term in the *xcomp* relation and the governor term is the outer verb. If true, the next step is to identify the *nsubj* relation between this governor term the governor term and the subject constituent. As illustrated in Example (79), the resultant wide scope span is the noun phrase which dominates this subject constituent:

(79) ***Sentence:*** *We managed to avoid any further delays.*
Dep Relation for narrow scope: dobj(avoid, delays)
Narrow Scope: *We managed to [avoid] any further delays.* — *narrow*
Dep Relation for wide scope: xcomp(managed, avoid) and nsubj(managed, We)
Wide Span: ⟨*We*⟩ *managed to avoid any further delays.*
Wide Scope: *We managed to [avoid] any further delays.* — *wide*

4.2.7 Nominalizations with of *(i.e. absence of)*

4.2.7.1 narrow scope:

A heuristic implemented for negation triggers identified as nominalizations. In the case where the nominalization (the negation trigger) is followed by *of*, the collapsed dependency relation: *prep_of* is identified between the negation trigger and the prepositional object. The resultant candidate narrow scope of negation is the constituent dominating the two member terms as exemplified in (80):

(80) ***Sentence:*** *In the absence of a reducing agent, the proteins remain folded...*
Dep Relation: prep_of(absence,agent)
Candidate Narrow Scope: *In [$_{NP}$ the absence of a reducing agent], the proteins remain folded...*
Narrow Scope: *In [$_{NP}$ the [absence] of a reducing agent], the proteins remain folded...*

There are also cases where the negation trigger which is a nominalization is involved in a *conj* relation with another term. If this non-negated term takes the first position in the coordinated phrase, then it will be the governor of both the collapsed *prep* and *conj* relations. The negation trigger can only identified by this *conj* relation as the dependent node. If such a *conj* relation is identified and if the governor is also a governor of the *prep_of* relation, the candidate scope of negation will be the constituent which dominates the governor node and the prepositional object as exemplified in (81):

(81) ***Sentence:*** *Each grid cell contains a value which indicates the presence or absence of an obstacle in the corresponding place.*
Dep Relations: conj_or(presence,absence) and prep_of(presence,obstacle)
Candidate Narrow Scope: *Each grid cell contains a value which indicates [$_{NP}$ the presence or absence of an obstacle in the corresponding place].*
Narrow Scope: *Each grid cell contains a value which indicates [$_{NP}$ the presence or [absence] of an obstacle in the corresponding place].*

4.2.7.2 wide scope:

It is not always the case that there will exist a wide scope span, as in example (80). However, in the second example (81), the term *which* should be considered to be the additional wide scope span. The subject constituent needs to be identified by the *nsubj* relation. The first step is to identify the *dobj* relation between the main verb (the governor of this relation) with the implicit negation trigger. If this particular relation is not found, the *dobj* relation between the main verb and the governor term of the *conj* relation needs to be identified. If any two are found, the second step is to identify the *nsubj* relation between the main verb and the subject constituent. Upon success, the subject constituent is marked as the additional wide scope span. Also, the main verb will also be marked as a second wide scope span as illustrated in Example (82):

(82) ***Sentence:*** *Each grid cell contains a value which indicates the presence or absence of an obstacle in the corresponding place.*

Dep Relations for narrow scope: conj_or(presence,absence) and prep_of(presence,obstacle)
Narrow Scope: *... indicates the presence or [absence] of an obstacle in the corresponding place.* — *narrow*

Dep Relation for wide scope I: dobj(indicates, absence)
Wide Span I: *Each grid cell contains a value ⟨which⟩ indicates the presence or absence of an obstacle in the corresponding place.*
Dep Relation for wide scope II: nsubj(indicates, which)
Wide Span II: *Each grid cell contains a value which ⟨indicates⟩ the presence or absence of an obstacle in the corresponding place.*
Wide Scope: *Each grid cell contains a value which indicates the presence or [absence] of an obstacle in the corresponding place.* — *wide*

4.2.8 verbs with a causal complement *(i.e. denied that)*

4.2.8.1 narrow scope

In the case where a verb that is identified as a negation trigger has a clausal complement, the *ccomp* dependency relation will be identified between this trigger and the dependent node. The resultant candidate narrow negation scope will be the constituent which dominates both the negation trigger and dependent node as exemplified in (83).

(83) ***Sentence:*** *She denied that she had taken money.*
Dep Relation: ccomp(denied, taken)
Candidate Narrow Scope: *She [$_{VP}$ denied that she had taken money].*
Narrow Scope: *She [$_{VP}$ [denied] that she had taken money].*

4.2.8.2 wide scope

There is one heuristic implemented for the additional wide scope span: as illustrated in (84): to identify if the negation verb trigger is the governor of an *nsubj* relation. If successful, then the dependent token (the subject constituent) is extracted. The resultant wide scope span is the noun phrase which dominates this subject constituent.

(84) ***Sentence:*** *She denied that she had taken money.*
Dep Relation for narrow scope: ccomp(denied, taken)
Narrow Scope: *She [denied] that she had taken money.* — *narrow*

Dep Relation for wide scope: nsubj(denied, She)
Wide Span: *⟨She⟩ denied that she had taken money.*
Wide Scope: *She [denied] that she had taken the money.* — *wide*

4.2.9 Adjectives *(i.e. unable,negative)*

There are two heuristics implemented for adjectives that are also negation triggers. The first is when the negation adjective trigger is the governor node of an *xcomp* dependency relation The second is when the negation adjective trigger is the dependent node in the *amod* relation.

4.2.9.1 The *xcomp* relation

4.2.9.1.1 narrow scope: The *xcomp* dependency relation is identified when the negation adjective trigger is linked to a non finite embedded verb. The resultant candidate scope of negation is verb phrase headed by the negation adjective trigger and the final scope is the same scope span except without the negation trigger as exemplified in (85):

(85) ***Sentence:*** *John was unable to go to school today.*
Dep Relation: xcomp(unable, go)
Candidate Narrow Scope: *John was* $[_{ADJP}$ *unable to go to school today].*
Narrow Scope: *John was* $[_{ADJP}$*[unable] to go to school today].*

4.2.9.1.2 wide scope: If there is a *nsubj* relation between the negation trigger with the subject then the noun phrase which contains this subject is extracted and determined to be the additional wide scope span as illustrated in (86):

(86) ***Sentence:*** *John was unable to go to school today.*
Dep Relation for narrow scope: xcomp(unable, go)
Narrow Scope: *John was [unable] to go to school today.* — *narrow*
Dep Relation for wide scope: nsubj(unable, John)
Wide Span: ⟨*John*⟩ *was unable to go to school today.*
Wide Scope: *John was [unable] to go to school today.* — *wide*

4.2.9.2 The *amod* relation

4.2.9.2.1 narrow scope: The *amod* dependency relation is identified when the negation adjective trigger is linked to the corresponding noun. This term is usually the dependent node in the relation. The resultant scope span is the constituent which dominates both members of the *amod* relation with the negation trigger removed from the span as illustrated in (87). This heuristic will also be frequently used in the cases where *affixal negation triggers* are required to have a wider scope interpretation.

(87) ***Sentence:*** *Negative examination.*
Dep Relation: amod(examination, Negative)
Candidate Narrow Scope: $[_{NP}$ *Negative examination].*
Narrow Scope: $[_{NP}$ *[Negative] examination].*

4.2.9.2.2 wide scope: There is one heuristic implemented for the additional wide scope span, illustrated in (88), where the governor term in the *amod* relation participates in a *det* relation. The resultant wide scope span is the corresponding determiner.

(88) ***Sentence:*** *This negative result generally signifies that the resonant frequency of the shaft is reached.*

Dep Relation for narrow scope: amod(result, negative)
Narrow Scope: *This [negative] result generally signifies that the resonant frequency of the shaft is reached.* — *narrow*

Dep Relation for wide scope: det(result, This)
Wide Span: ⟨*This*⟩ *negative result generally signifies that the resonant frequency of the shaft is reached.*
Wide Scope: *This [negative] result generally signifies that the resonant frequency of the shaft is reached.* — *wide*

4.2.10 Conjunctions *(i.e. neither, nor)*

The explicit negation trigger *nor* is identified as a type of conjunction between coordinating phrases and is very often part of the *Neither... nor* or the *not ... nor* constructions. In both cases there will be a distinct negation scope span for each coordinate phrase containing one of the negation triggers. Determining the negation scope of the *nor* trigger separately is achieved by identifying the *conj_nor* dependency relation. If the first coordinate phrase contains the negation trigger *not* , then the *neg* dependency relation will be identified, however if the first coordinate phrase contains 'neither' then either the *advmod* or the *pre_conj* dependency relation will be identified.

4.2.10.1 conj_nor

4.2.10.1.1 narrow scope In the case of the *not... nor*, NEGATOR will first determine the narrow negation scope of the first coordinate phrase, and since the negation trigger is *not*, the *neg* dependency relation will be identified and the candidate narrow scope determined as with any other *not*. For the second coordinate phrase: the *conj_nor* dependency relation is identified between the two terms in the sentence connected by the *nor* conjunction. The resultant candidate narrow scope of negation is the constituent dominating the dependent member of the dependency relation and the negation trigger. (89) illustrates a case where the first coordinate phrase contains the negation trigger *not* and the second contains *nor*. The final narrow scope spans for both phrases are the candidate scopes but only to the right of the relevant negation trigger.

(89) ***Sentence:*** *He didn't attend the meeting nor was he informed of its decisions.*
Dep Relation: neg(attend, not)
Dep Relation: conj_nor(attend, was)
Candidate Narrow Scope: *He [$_{VP}$ didn't attend the meeting nor was he informed of its decisions].*
Narrow Scope: *He [$_{VP}$ did[n't] attend the meeting [nor] was he informed of its decisions].*

4.2.10.1.2 wide scope The wide scope heuristic implemented here is to identify the subject constituent. In general, this subject should be linked to the governor term of the *neg* dependency relation. Consequently, the *nsubj* relation needs to be identified where the governor term of the *neg*

relation is also the governor of the *nsubj* relation. The subject is extracted if this relation exists, and the corresponding wide scope span is the noun phrase which dominates the subject. As well, if there exists an *aux* relation, where the same governor term is also governor, then a second wide scope span will be created, containing the auxiliary term. This case is illustrated in (90):

(90) ***Sentence:*** *He didn't attend the meeting nor was he informed of its decisions.*
Dep Relations for narrow scope: neg(attend, not) and conj_nor(attend, was)] *narrow*
Narrow Scope: *He did[n't]* *attend the meeting* *[nor]* *was he...*.]
Dep Relation for wide scope I: nsubj(attend, He)]
Wide Span I: ⟨*He*⟩ *didn't attend the meeting nor was he....*
Dep Relation for wide scope II: aux(attend, did)] *wide*
Wide Span II: *He* ⟨*did*⟩*n't attend the meeting nor was he....*
Wide Scope: *He did[n't]* *attend the meeting* *[nor]* *was he...*.]

4.2.10.2 pre_conj

4.2.10.2.1 narrow scope In the case of the *Neither... nor*, there are two possible dependency relations to identify the scope of the *Neither* trigger: either the *preconj* dependency relation: when the *governor* is not a verb or the *advmod* dependency relation when the governor is a verb. In both cases the candidate narrow scope will be the constituent which dominates both the negation trigger *Neither* and the governor term. This candidate scope will include the second coordinate clause containing *nor* and therefore the final scope span will become two discontinuous spans: the first one being the original candidate scope up to but not including the nor trigger and the second will be all constituents to the right of the *nor* trigger. Example (91) demonstrates such a case:

(91) ***Sentence:*** *She found it neither surprising nor alarming.*
Dep Relation: preconj(surprising, neither)
Candidate Narrow Scope: *She found it* $[_{ADJP}$ *neither surprising nor alarming].*
Narrow Scope: *She found it* $[_{ADJP}$ *[neither]* *surprising* *[nor]* *alarming].*

4.2.10.2.2 wide scope If the candidate narrow scope is determined by the *advmod* relation where the governor term is a verb, then the additional wide scope span(s) will be determined in the same approach as to the *not... nor...* case. However, if the *preconj* relation was used, then usually the final wide scope will consist of the entire sentence (with the negation triggers pruned out) as exemplified in (92):

(92) ***Sentence:*** *She found it neither surprising nor alarming.*
Dep Relations for narrow scope: preconj(surprising, neither)] *narrow*
Narrow Scope: *She found it [neither]* *surprising* *[nor]* *alarming*.]
Wide Scope: *She found it* *[neither]* *surprising* *[nor]* *alarming*.] *wide*

4.2.10.3 nor not in a coordinate phrase

The last heuristic discussed here is in the case where a sentence starts with *nor* and there is no other coordinating negation trigger in the sentence. In this case as illustrated in (93), the *cc* relation is identified where the negation trigger is the dependent node. The candidate narrow scope of negation is the constituent which dominates both members of the relation. Usually this results in the final narrow scope being the entire sentence with the trigger pruned out. Therefore, there is no wide scope heuristic for this case.

(93) ***Sentence:*** *Nor does the apparent correlation with Alzheimer 's prove anything.*
Dep Relation: cc(does, Nor)
Candidate Narrow Scope: *[$_{SINV}$ Nor does the apparent correlation with Alzheimer's prove anything].*
Narrow Scope: *[$_{SINV}$ [Nor] does the apparent correlation with Alzheimer's prove anything].*

4.2.11 Exception cases

not only, not even and not just are exception cases and the scope of the explicit negation trigger is limited to *only, even and just* respectively.

4.2.12 Wide scope heuristics for affixal negation

NEGATOR also contains heuristics that can be allocated for the terms containing the affixal negation triggers[4] by any of the aforementioned heuristics, if the correct conditions are met.

4.3 The NEGATOR Scope Detection Module

The heuristics for determining the narrow or wide syntactic negation scope of an explicit, implicit or affixal negation trigger is implemented as a GATE (Cunningham et al., 2011) component: the NEGATOR Scope Detection Module. This module is intended to be run in a GATE pipeline after the NEGATOR Trigger Detection module. It assumes that sentence, token and negation trigger annotations already exist, and are used as input to the module. As input, it requires GATE parse tree and dependency relation annotations to also be present within the current GATE session. For the research presented in this thesis the Stanford and Charniak Parsers were the only parsers used. Subsequently, the Dependency module from (Klein and Manning, 2003; de Marneffe et al., 2006) implemented as a standard GATE plugin extracts the dependency graphs. The NEGATOR Scope Detection Module

[4] i.e. unhappy: the affixal trigger is *un*, and the *term* is *unhappy*.

will consequently identify and determine the negation scope using the previously described heuristics, for any negation triggers identified in a given text.

The NEGATOR Scope Detection Module will by default use only the narrow scope heuristics. However, the module is implemented with different options that can be set at runtime. Specifically, one can select: (1): The trigger *type* (`explicit, implicit or selfNeg`) that negation scope should be determined for and (2): Either the narrow or wide scope option.

Therefore, the following scenario is possible: the user may choose to run one instance of the module with wide scope on `explicit` negation triggers and in parallel have another instance of the module running with *narrow* scope on `implicit` negation triggers.

Example (94) contains an explicit negation trigger *not* and an implicit negation trigger *inhibitor* determined and annotated by the NEGATOR trigger detection module. If NEGATOR module is set with the default scope option, it will subsequently create two distinct *narrow* scope annotations for each negation trigger identified. The resultant GATE annotations are shown in Figures 10 -11.

*(94) Okadaic acid, an **inhibitor** of phosphatases 1 and 2A, did **not** overcome the defect in these subclones.*

explicit negation scope

Ann_Type	Negation Scope
Type	explicitNegScope
Negation_Trigger	*not*
Text_Span	overcome the defect in these subclones
Constituent_ID	1196
Start Offset	1380
End Offset	1418

Figure 10: Explicit Scope Annotation

implicit negation scope

Ann_Type	Negation Scope
Type	implicitNegScope
Negation_Trigger	*inhibitor*
Text_Span	of phosphatases 1 and 2A
Constituent_ID	1180
Start Offset	1346
End Offset	1370

Figure 11: Implicit Scope Annotation

The NEGATOR Scope Detection Module will trigger the heuristic corresponding to the *neg* dependency relation, extract the respective constituents from the parse tree, prune the negation trigger from the candidate scope and finally output the annotation as shown in Figure 10. For the implicit scope annotation, the heuristic corresponding to the *prep_of* dependency relation will be triggered, the relevant constituents from the parse tree will be extracted, the candidate scope will be pruned accordingly and the annotation as shown in Figure 11 will be produced. An additional feature in both annotations shown is the *Constituent_ID*. This feature identifies the constituent in the parse tree annotations that

contains the relevant negation scope span. This feature is useful in any down stream processing tasks that make use of the scope annotations as it allows one to immediately reference the corresponding node in the parse tree.

4.4 Negation scope summary

In this chapter, the NEGATOR narrow and wide negation scope heuristics have been described. Importantly, the approach to developing these heuristics was to primarily use the syntactic structure of a sentence: the available constituent trees and grammatical relations. The resultant set of heuristics forms a solid foundation for any application needing to detect negation phenomena. The heuristics are general and stable enough to perform well enough on different text genres, given that they were not developed according to a specific gold standard. An important feature of the NEGATOR scope detection module is that it is flexible enough to support both on one hand the narrow and wide negation scope models as well as incorporating scope patterns for the varying types of negation. The performance and usefulness of the NEGATOR scope heuristics will be assessed in the next chapter. Two evaluations will be described on two different gold standards. The NEGATOR trigger- scope modules were also used in two different shared tasks that were dedicated to the detection, identification and modelling of the phenomena of negation. The results and discussion pertaining to these shared tasks are also described in the next chapter.

Chapter 5

Evaluation

5.1 Evaluation of NEGATOR trigger-scope modules

NEGATOR is evaluated on two different gold standards: BioScope for Biological Texts and the Conan Doyle Test set for fiction. Both these data sets annotate negation triggers and linguistic scope. The gold standards do not only differ in text genre. They consider different negation triggers according to domain and task specific requirements, and use different models for annotating negation scope. The BioScope continuous scope annotations assume the *narrow* scope model and unconventionally includes the negation trigger in the resultant scope span (95a). In contrast, the Conan Doyle gold scope annotations include all arguments relating to the event being negated. These discontinuous negation scope spans represent the entire proposition that is negated, including the subject but excluding the negation trigger (95b). Occurrences of affixal negation are also assigned a *wide* scope interpretation.

(95) *(a)* *BioScope*
Once again, the Disorder module does [not] contribute positively to the prediction.

(b) *Conan Doyle*
We did [not] drive up to the door but got down near the gate of the avenue.

Given these corpus specific annotations, the NEGATOR trigger-scope module is set up with two distinct configurations. For the determination of negation triggers in BioScope, NEGATOR uses the NEGATOR *implicit and explicit*[1] trigger lists. For Conan Doyle, NEGATOR additionally uses the *selfNeg* trigger list. The cCore trigger list is used by NEGATOR as an alternative method for the detection of negation triggers in both gold standards. The NEGATOR scope module incorporates heuristics for both the *narrow* and *wide* scope interpretations. In the BioScope evaluation, NEGATOR's *narrow* scope heuristics are employed resulting in the scope annotation shown in (96a).

[1] As BioScope does not annotate *affixal* negation, therefore this list was not used.

(96) (a) NEGATOR *narrow*

Once again, the Disorder module does [not] contribute positively to the prediction.

(b) NEGATOR *wide*

We did [not] drive up to the door but got down near the gate of the avenue.

The resulting NEGATOR scope span is equivalent but not identical to the corresponding BioScope scope span[2], as NEGATOR does not include the negation trigger in the scope span. In this evaluation, the BioScope trigger and scope annotation (95a) and is considered a complete match with NEGATOR's (96a). In the Conan Doyle evaluation, NEGATOR's *wide* scope heuristics are employed. The resulting NEGATOR trigger and scope annotations (96b) are a complete match with that of Conan Doyle in (95b).

The results for the evaluation on the Conan Doyle test set were obtained by using the evaluation script from the 2012 *SEM shared task. The following three categories are used to classify the overlap between the gold annotations and the NEGATOR annotations:

`True Positives`: NEGATOR scope span is a complete match with the gold scope annotation.

`False Positives`: NEGATOR predicts a non-existing scope annotation in the gold standard.

`False Negatives`: There exists a gold scope annotation and NEGATOR fails to find it, or when NEGATOR predicts a partial scope span match.

The performance is then evaluated according to the following three measures:

`Precision`: True Positives/(TruePositives+False Positives)

`Recall`: True Positives/(True Positives+False Negatives)

`F-score`: (2*Precision*Recall)/(Precision+Recall)

The results for the evaluation on the BioScope corpus were obtained by a custom evaluation script. This script uses the same methods (the category classifications and measurements) as those defined in the 2012 *SEM shared task evaluation script.

5.1.1 Results

A summary of the performance of the NEGATOR trigger-scope module on the BioScope corpus and the Conan Doyle data set is shown in Table 7. The evaluation is done by using the precision and recall measures and their harmonic mean, the f-score. The unit being measured are the complete negation scope spans allocated by NEGATOR corresponding to a negation trigger.

The results show that the NEGATOR scope heuristics perform very well in terms of recall. Using the larger, more extensive NEGATOR trigger lists results in better recall for

[2]Please refer to Appendix B.1 for a more detailed discussion.

	NEGATOR trigger lists			cCore trigger lists		
	precision	**recall**	**f-score**	**precision**	**recall**	**f-score**
Conan Doyle Test Set	.71	.66	.68	.71	.39	.50
BioScope Abstracts	.41	.77	.54	.82	.75	.78
Full Texts	.47	.69	.56	.75	.63	.69
Clinical Reports	.88	.82	.85	.95	.82	.88

Table 7: Evaluation Summary of NEGATOR on BioScope and Conan Doyle Corpus

both corpora. However, in the BioScope corpora, precision suffers. The cCore list is far better suited for the BioScope corpus, demonstrated by the higher precision measures. In contrast, for the fiction data the cCore list does not add any benefits as the precision remains the same. Thus, using different trigger lists has substantial influence in the performance of NEGATOR on a particular corpus. Careful tailoring of a trigger list is most beneficial and necessary to meet domain specific requirements. The f-score results show that when given a more well suited trigger list (i.e. the cCore list for BioScope), the NEGATOR trigger-scope module performs even better. This is because the NEGATOR scope heuristics are designed to allocate the linguistic scope to any type of trigger. The f-score results also show that the performance on the two data sets is in fact very close. This insight is most supportive for the argument that NEGATOR functions consistently across different text genres due to general and linguistically motivated approach.

5.1.1.1 Results Comparison for BioScope

Table 8 shows the performance of NEGATOR compared with other state of the art systems run on BioScope. The only comparable measure found between the reported results of these systems and NEGATOR is the PCS[3] score. NEGATOR's recall measure is directly comparable with the published PCS scores.

	Abstracts	**Full Texts**	**Clinical Reports**
NEGATOR **with** NEGATOR **lists**	76.97	68.51	82.22
NEGATOR **with cCore list**	74.62	63.22	81.77
(Morante and Daelemans, 2009)	66.70	41.00	70.75
(Apostolova et al., 2011)	80.63	71.26	85.56
(Li et al., 2010)	81.84	64.02	89.79

Table 8: Performance comparison over PCS scores (%) of NEGATOR with other systems on BioScope

The results show that the NEGATOR trigger-scope module regardless of trigger lists used has comparable recall results with (Apostolova et al., 2011) and (Li et al., 2010). In contrast

[3]Percentage of fully correct scopes was introduced by (Morante and Daelemans, 2009). With *PS* being the number of correct scopes produced by the system and *S* the number of gold scopes, PCS can be expressed with: $PCS = PS \ / \ S$.

to (Morante and Daelemans, 2009), NEGATOR and the three other systems rely on structured syntactic information in the task of negation scope finding. This result therefore explains the close results and indicates the appropriateness and effectiveness of a linguistically motivated approach. NEGATOR with the NEGATOR lists has better recall performance than (Li et al., 2010) in the full papers subcorpus. This result shows that NEGATOR's approach is surely competitive, since negation systems have traditionally claimed to have the most difficulty with the full papers subcorpus.

All systems generate the highest scores on the clinical reports subcorpus. This subcorpus is composed of an average of three sentences per report, which are shorter and syntactically simpler than those in the other two subcorpora. Also, 77% of negations are from the negation trigger *no*. The resulting scope span, determined by NEGATOR, for the majority of these sentences is the entire noun phrase as illustrated in Example (97). Both (97a) and (97b) are determined to be matches with the BioScope annotations.

(97) (a) [No] focal pneumonia.

(b) [No] radiographic evidence of acute cardiopulmonary disease.

NEGATOR's weaker performance in comparison to (Apostolova et al., 2011) and (Li et al., 2010) in the clinical reports subcorpus is mainly due to two error cases. The first is a scope error with the negation trigger *negative*. This error is due to a different interpretation of the determination of negation scope between BioScope and NEGATOR. As illustrated in (98), NEGATOR determines the noun modified by the adjective *negative* to be the negation scope. However, BioScope will consistently in this case only mark *negative* as the scope of negation and not the modified noun.

(98) (a) Negative examination. UTI.

implicit negation

BioScope	***offset***	***scope***
cue	*9-17*	*Negative*

(b)

xscope	*9-17*	*Negative*
NEGATOR*:*	***offset***	***scope***
implicitNeg	*9-17*	*Negative*
implicitNegScope	*18-29*	*examination*

The second error is often triggered by the negation trigger *no*. It occurs when NEGATOR fails to determine the fully correct scope span. As illustrated in (99), NEGATOR determines the noun phrase to be the scope span, while BioScope takes a wider approach and includes the verb phrase (whose main verb is *identified*) in the scope span. We believe that the NEGATOR scope in this example has been correctly determined. The parse tree indicates that the negation *no* only affects *hydronephrosis or cortical scarring*, since the complete noun

phrase constituent is *no hydronephrosis or cortical scarring*. The verb phrase containing *indicated ...* is a sibling of the aforementioned noun phrase. The statement is interpreted as something was indicated - just not hydronephrosis or cortical scarring.

(99) (a) *There is no hydronephrosis or cortical scarring identified on the current exam.*

explicit negation

BioScope	***offset***	***scope***
cue	*138-140*	*no*
xscope	*138-207*	*no hydronephrosis or cortical scarring identified on the current exam*
NEGATOR*:*	***offset***	***scope***
explicitNeg	*138-140*	*no*
explicitNegScope	*141-176*	*hydronephrosis or cortical scarring*

NEGATOR also has weaker performance in the Abstracts data set. There is one major error case which incurs approximately 20% of mismatches. As illustrated in (100), NEGATOR and BioScope differ in the approach to annotating the scope, where NEGATOR will include the verb *had* as part of the scope. As stated in the BioScope Annotation guidelines, the BioScope `xscope` annotations usually extend to the right of the negation trigger. There are only a few exception cases where they will annotate to the left. NEGATOR has made a clear decision to interpret sentences as the one exemplified in 100 as verbal negations. These cases are when the negation trigger affects the direct object (*effect*) , and consequently the verb *had* is linked to the direct object. The sentence in 100 could be interpreted without any change in meaning as Ascorbate and AZT also **did not have any** effect ....

(100) (a) *Ascorbate and AZT also had no effect on NF-kappa B activation following TNF-alpha- or PMA-induced stimulation of U1 promonocytic cells.*

explicit negation

BioScope	***offset***	***scope***
cue	*975-977*	*no*
xscope	*975-1082*	*no effect on NF-kappa B activation following TNF-alpha- or PMA-induced stimulation of U1 promonocytic cells*

(b)

NEGATOR*:*	***offset***	***scope***
explicitNeg	*975-977*	*no*
explicitNegScope	*971-974*	*had*
explicitNegScope	*976-1082*	*effect on NF-kappa B activation following TNF-alpha- or PMA-induced stimulation of U1 promonocytic cells*

5.1.1.2 Results Comparison for Conan Doyle

The Conan Doyle corpus was used for the task of *Resolving the Scope of Negation Cues and Detecting Negated Events* at *SEM 2012 Shared Task (Morante and Blanco, 2012). The NEGATOR trigger scope module did not participate in the *SEM 2012 Task, but was evaluated using the publicly available evaluation script. Tables 9 and 10 compares NEGATOR's performance with the published results from the open track[4].

	precision	**recall**	**f-score**
NEGATOR **with** NEGATOR **lists**	69.97	96.58	81.04
NEGATOR **with cCore list**	73.95	53.79	62.43
UOslo2	89.17	93.56	91.31
UGroningen, run 2	88.89	84.85	86.82
UCM1	89.26	91.29	90.26
UCM2	81.34	64.39	71.88
UGroningen, run 1	86.90	82.95	84.88

Table 9: Performance comparison of cue detection between NEGATOR and official participants in *SEM 2012 Task 1 Open Track.

Table 9 shows the results for cue detection. The NEGATOR system using the NEGATOR trigger list has the lowest precision measure below the almost 10% below the average of 87%. This is in part because NEGATOR's extensive *implicit* trigger list identifies 30 implicit negations and the gold standard only 1. However, using the cCore list results in only marginally better precision. Consequently, there are also a number of `false positives` in instances where NEGATOR will annotate an *explicitNeg* trigger where the gold standard does not. The cCore list also contains triggers which the gold standard does not, i.e. *little*. In contrast, NEGATOR with the NEGATOR trigger lists is the best performer in recall with 96.58%. 16% of the cues in the gold standard are affixal negation triggers, and by using the NEGATOR comprehensive *selfNeg* trigger list, all these instances are retrieved. In comparison, the cCore list with very few relevant affixal negation triggers generates a lower recall. The low precision measure resulting from using the NEGATOR lists generates a lower f-score than most other systems, however the overall f-score is still better than when using the cCore list. Using the smaller cCore list for this gold standard is not beneficial for either the precision or recall measures. Rather, tailoring the larger, more extensive NEGATOR lists to task requirements would achieve better results.

Table 10 shows the results for identifying scope spans given a correctly identified trigger. NEGATOR with the NEGATOR trigger lists remains the best performer in recall, while

[4]In the open track, systems could make use of any external resource or tool. The tools used could not have been developed or tuned using the annotations of the test set. The NEGATOR trigger-scope module has been developed to use other tools and resources than those provided in the training set. Thus, NEGATOR would belong in the open track.

	precision	recall	f-score
NEGATOR **with** NEGATOR **lists**	70.67	66.14	68.32
NEGATOR **with cCore list**	71.11	38.55	49.99
UOslo2	85.71	62.65	72.39
UGroningen, run 2	76.12	40.96	53.26
UCM1	82.86	46.59	59.64
UCM2	67.13	38.55	48.98
UGroningen, run 1	46.38	12.85	20.12

Table 10: Performance comparison of scope detection between NEGATOR and official participants in *SEM 2012 Task 1 Open Track.

generating a comparatively low precision due to the extensive number of `false positives`. NEGATOR's overall f-score of 68.32% shows that in scope detection it performs quite competitively and effectively. The best performer 'UOslo2' has an f-score of 72.39%, while the second published best is 'UCM1' with 59.64%, almost 10% lower than NEGATOR. This result supports and validates NEGATOR's general and linguistically motivated approach to scope detection.

For both negation cues and scope detection, it has been observed that NEGATOR has comparatively weaker performance in precision. There are two major reasons for the high number of `false positives` generated by the NEGATOR module. The first as illustrated in (101) occurs when NEGATOR annotates a negation trigger and its scope but the gold standard will never consider this negation trigger:

(101) *(a)* *"... but you deny that you kept the appointment ."*

implicit negation

Conan	***offset***	***scope***
cue	-	-
scope	-	-

(b)

NEGATOR	***offset***	***scope***
implicitNeg	*7334-7338*	*deny*
implicitNegScope	*7330-7333*	*you*
implicitNegScope	*7339-7668*	*that you kept the appointment*

A second reason is that the gold standard will not annotate a negation trigger given a particular context. Example (102) is a case where NEGATOR will mark the negation trigger *not* and *only* (an exception case) as the scope span. The gold standard will not even mark the trigger *not* and consequently there is no gold negation scope span. Patterns like *not only* or *not just* do not indicate a negation in terms of non-existence, but rather they are used as a device for emphasis. Thus we believe that including these patterns and not completely ignoring them can prove useful to some down stream tasks which do not only consider one interpretation of negation.

(102) (a) *"Not only was his body that of a giant. . . ."*

explicit negation

Conan	***offset***	***scope***
cue	-	-
(b) *scope*	-	-
NEGATOR	***offset***	***scope***
explicitNeg	*12059-12062*	*Not*
explicitNegScope	*12063-12067*	*only*

The comparisons of the NEGATOR trigger-scope module with state of the art negation systems on both the Conan Doyle and BioScope corpus has proved it to be a solid contender. It has been shown that it performs competitively, especially in recall, for different text genres with distinct annotation schemes.

5.2 Shared Tasks

NEGATOR was adapted to two recent pilot tasks which address specific issues related to negation The QA4MRE pilot task on *Processing Modality and Negation* (Morante and Daelemans, 2012b) involved detecting negated and modalized *events*. The *SEM 2012 pilot task on *Detecting the Focus of Negation* (Morante and Blanco, 2012) was dedicated to identifying the *focus* item contained in the scope span of a negation trigger. These pilot tasks presented an opportunity to showcase the NEGATOR system in a more applied and task specific setting. Furthermore, the data sets used in these tasks are from yet other text genres then the datasets used in the previous scope evaluations. Consequently, assessing NEGATOR's performance on other text genres can only be beneficial.

5.2.1 QA4MRE Pilot Task

The goal of this task was to determine whether an event present in the text is within the scope of a negation, in the scope of a modal or within the scope of both. As the task was framed as an annotation task, the output of the system required each event to be allocated either a `MOD, NEG, NEGMOD` or `NONE` label. The test data set for the pilot task is composed of eight English documents divided into four topics: *AIDS, Climate Change, Music and Society, and Alzheimer's Disease*, part of the larger test set for the main QA4MRE Task.

The NEGATOR system was adapted and extended according to the task requirements. A list of modality triggers (taken from (Kilicoglu, 2012)) was added as an additional trigger list. The NEGATOR scope module was extended to cover the *aux* dependency relation and conditional clauses as modality contexts. Example (103) illustrates how the NEGATOR system will identify a label for a predetermined event. It identifies *might* and *not* to be

modal and negation triggers respectively. Subsequently, the corresponding scope span will be identified for each trigger. Finally, the event to label is *come*. NEGATOR determines that this event is both in a negated and modal context and will label the event as `NEGMOD`, which is a match to the gold label for that event. Please refer to (Rosenberg et al., 2012) for a more detailed description and discussion of how NEGATOR was adapted and extended for the task .

(103) ***Sentence:*** *Half of Europe's electricity might not come from fossil fuels.*
ModalTrigger: *might*
Modal Scope: *Half of Europe's electricity might not come from fossil fuels.*
explicitNeg Trigger: *not*
explicitNeg Scope: *Half of Europe's electricity might not come from fossil fuels.*
NEGATOR ***Labeled Event:*** *come:* $LABEL$ = `NEGMOD`
Gold Labeled Event: *come:* $LABEL$ = `NEGMOD`

Two different variants were submitted for the task. The *narrow* variant, considered only the direct members of the dependency relation that triggered a relevant scope annotation. In contrast, the *greedy* variant, considered the entire scope annotation in which an event is contained within.

5.2.1.1 Results

Tables 11 and 12 show the official results evaluated by the task organizers. The evaluation was conducted using the standard measures of precision, recall and their harmonic mean the f-score. Each of the four possible labels (`NEG, MOD, NEGMOD, NONE`) allocated to a pre-annotated event is evaluated individually. Additionally two methods were used for calculating the overall performance of a system: the *Macroaverage*[5] and the *Microaverage*[6] shown in Table 11. The macroaveraged measure is balanced across both runs with 64%

	Narrow	**Greedy**
Macroaveraged f-score:	0.6368	0.6196
Microaveraged f-score:	0.7117	0.6750
Overall Accuracy:	0.7130	0.6688

Table 11: Averaged Results for QA4MRE Pilot Task on Processing Modality and Negation

for the *narrow* variant and 62% for the *greedy* variant. The *narrow* variant has overall better performance. The methodology for allocating the labels in the *narrow* variant was

[5]The macroaverage measure calculates the precision by taking the average of the precision values calculated individually for each category. For recall, it will take the average of the recall values. The macroaverage f-score is the harmonic mean of the macroaverage precision and recall.

[6]The Microaverage f-score is the harmonic mean of the Microaverage precision and recall measures.

Narrow Variant			
	Precision	**Recall**	**f-score**
MOD	0.6660	0.6646	0.6653
NEG	0.7826	0.5625	0.6545
NEGMOD	0.4865	0.4390	0.4615
NONE	0.7529	0.7789	0.7657

Greedy Variant			
	Precision	**Recall**	**f-score**
MOD	0.5862	0.7532	0.6593
NEG	0.8182	0.5625	0.6667
NEGMOD	0.3373	0.6829	0.4516
NONE	0.8091	0.6180	0.7008

Table 12: Global Results for QA4MRE Pilot Task on Processing Modality and Negation

not overly generous in the `MOD/NEG/NEGMOD` labelling of the events. Table (12) therefore shows that the *narrow* variant performs better in detecting the `NONE` label correctly for an event. Not surprisingly then, the *greedy* variant has the same or better recall measures for all other labels. The detection of the `NEGMOD` label for an event is the worst performer for both variants. This is due to the manner in which the `NEGMOD` label is allocated: the correct determination of an event being labeled `NEGMOD` is reliant on it having been previously correctly allocated the `NEG` label and the `MOD` label. Any errors that occur in the allocation of the `NEG` or `MOD` label to an event will propagate through to the `NEGMOD` labelling task. In both runs the f-scores for both the `NEG` and `MOD` labels are fairly stable: 65%-66%.

There were three teams with a total of six runs who submitted for this pilot task. The scores per run are provided in Table 13 in terms of overall macroaveraged f-score, overall accuracy and f-score per label from (Morante and Daelemans, 2012b).

	Overall Results		**f-score per label**			
Run	**Macro f-score**	**Accuracy**	**NONE**	**MOD**	**NEG**	**NEGMOD**
CLaC 1	0.6368	0.7130	0.7657	0.6653	0.6545	0.4615
CLaC 2	0.6196	0.6688	0.7008	0.6593	0.6667	0.4516
desancis 1	0.5339	0.6551	0.7478	0.5307	0.5000	0.3571
desancis 2	0.5043	0.6342	0.7247	0.5511	0.4275	0.3137
desancis 3	0.5027	0.6125	0.6985	0.5272	0.4409	0.3441
JUCSENLP 1	0.3378	0.6262	0.7219	0.5933	0.0000	0.0360

Table 13: Overall results per run for QA4MRE Pilot Task 2012

As shown in Table 13 negator ranked first by the wide margin of an macroaveraged f-score of approximately 10% less as compared to the next best competitor. It is also very encouraging to see that NEGATOR performed well for all four label categories.

5.2.2 Negation Focus

*Sem 2012 Task 2 on Focus Detection (Morante and Blanco, 2012) builds on recent negation scope detection capabilities and introduces a gold standard (Blanco and Moldovan, 2011) to identify the focus item, namely the sub-string of the scope which is targeted by the negation. The Focus of negation is annotated on PropBank (Palmer and Gildea, 2005), accounting for verbal, analytical and clausal relations to a negation trigger; the role most likely to correspond to the focus was selected as focus. Thus, the focus is always the full text span of the chosen semantic role. A sample annotation from the gold standard is given in (104), where PropBank semantic roles are labelled *A1, M-NEG*, and *M-TMP* and focus is underlined (until June).

(104) $\langle A\ decision_{A_1}\rangle\ is\langle n't_{M-NEG}\rangle\ expected\ \langle \underline{until\ June}\ _{M-TMP}\rangle$

Since the dataset for the pilot task is small, we adopted a linguistic approach and used NEGATOR with the extensive trigger list and narrow scope module to identify negation and its scope on the raw text, side-stepping PropBank. NEGATOR was extended to include some simple focus heuristics. The baseline heuristic is defined as: the last item in the scope is declared (word or constituent) to be the negation focus. This simple heuristic was inspired by (Huddleston and Pullum, 2002) on prosodic focus placement. Please see (Rosenberg and Bergler, 2012) for a more detailed description and discussion of the extensions applied to NEGATOR for this task.

5.2.2.1 Results

Table 14 shows the overall performance of the system which is almost balanced between precision and recall with an f-score of 58%.

Test Set			**Development Set**		
Precision	60.00	[405/675]	**Precision**	59.65	[303/508]
Recall	56.88	[405/712]	**Recall**	57.06	[303/531]
F-score	58.40		**F-score**	58.33	

Table 14: System Results for *SEM 2012 Task on Focus Detection

The fact that performance on the test set surpassed its performance on the development set is a strong indicator of the strength and generality of the chosen approach. That one could adapt the simple trigger list plus scope heuristic system to this new task with very low effort showcases the generality and adaptability of linguistically inspired, deeper semantic processing. The NEGATOR system was the only participant in the this task. Thus, one cannot compare the NEGATOR results with other teams. However, one can compare the NEGATOR results to the system implemented by the task organizer, described in (Blanco

and Moldovan, 2011). They report an accuracy of 61.38% on their BASIC baseline and 65.50% on the more extended FOC-DET system. These results support the notion that the adapted NEGATOR system is quite a stable prototype, given that it was developed from a more general perspective by not relying on the available semantic features.

Chapter 6

Conclusion

In this work, NEGATOR is presented, a lightweight, linguistically inspired negation module. It addresses two fundamental tasks for identification of negation phenomena in text: trigger detection (based on a list of triggers provided) and scope determination based on dependency graph information from parser output.

Three main types of negation triggers are considered: explicit triggers like *not*, implicit triggers like *fail to*, which lexically encode negative polarity; and affixal triggers like *unaffected*, that encode negative polarity but with idiosyncratic scope. We compiled a trigger list for each of these trigger types, by extracting the relevant terms according to specific criteria from the Subjectivity Lexicon described in (Wiebe et al., 2005). The NEGATOR trigger detection module, using any or all of these trigger lists identifies and annotates occurrences of negation triggers in text. In order to assess the performance and generality of the NEGATOR trigger lists, we compare the NEGATOR trigger lists with four other lists from (Nawaz et al., 2013) on four data sets from the BioMedical, fiction and news and current affairs genres. Given that none of the lists were specifically trained on any of these data sets, they all give a satisfactory baseline. The smallest list, 'cCore' drawn from biomedical texts is the best performer overall due to incurring less *false positive* instances. The NEGATOR lists perform consistently across all domains in terms of recall, but its larger lists incur a significant drop in precision. The overall insight is that the careful tailoring of lists to meet domain requirements is beneficial and necessary. Given sufficient training data this is definitely possible.

The second core component, the NEGATOR scope detection module, is intended to be a robust, domain independent and linguistically motivated approach to negation scope detection. It is well documented in the literature that syntactic scope is determined to a large part, by the lexical category of the negation trigger over the rest of the parse tree. Thus, we propose a set of specialized heuristics that covers all triggers in the extended

NEGATOR trigger lists. The final implementation of this module is highly flexible, as it contains heuristics for both *narrow* and *wide* scope models as different applications down stream may require different interpretations.

The NEGATOR trigger and scope detection modules are evaluated on two different gold standards: *BioScope* for Biological Texts and the *Conan Doyle* Test set for fiction. Both these data sets annotate negation triggers and linguistic scope. The gold standards do not only differ in text genre, rather they consider different negation triggers according to task specific requirements, and use different models (*narrow* and *wide* respectively) for annotating negation scope. NEGATOR was not trained nor adapted for either dataset or specific annotation scheme. NEGATOR's performance is compared with other state of the art negation detection systems. In both instances, NEGATOR proves to be a solid contender, performing competitively, especially in recall. Even though NEGATOR is not the ultimate performer in either one, something much more important is demonstrated: that its an out-of-domain, out-of-task behaviour is consistent and robust on both. This insight supports the argument that a general, linguistically motivated approach to negation detection is valid, especially one that can easily be tuned to verify and implement different representations of negation phenomena.

The NEGATOR module was adapted and extended for two pilot tasks which address specific issues related to negation: The *SEM 2012 pilot task on *Detecting the Focus of Negation* (Morante and Blanco, 2012) and The QA4MRE pilot task on *Processing Modality and Negation for Machine Reading* (Morante and Daelemans, 2012b) at CLEF 2012. Both these tasks presented not only other text genres for NEGATOR to be run on, but also provided an opportunity to prove the capabilities of NEGATOR in a formalized setting. The NEGATOR system was the best performer in the QA4MRE pilot task, and the only performer in the Focus Task. Thus, our approach to the NEGATOR trigger-scope module demonstrates that it is possible to create a domain independent module based on solid foundations and all the while maintaining performance at a competitive level.

Bibliography

E. Apostolova, N. Tomuro, and D. Demner-Fushman. 2011. Automatic extraction of lexico-syntactic patterns for detection of negation and speculation scopes. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2011)*, Portland, Oregon, USA. Association for Computational Linguistics.

E. Blanco and D. Moldovan. 2011. Semantic representation of negation using focus detection. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2011)*, Portland, OR, USA.

W. Chapman, W. Bridewell, P. Hanbury, G.F. Cooper, and B. Buchanan. 2001. A simple algorithm for identifying negated findings and diseases in discharge summaries. *Journal of Biomedical Informatics*, 34(5):301-310.

E. Charniak and M. Johnson. 2005. Coarse-to-fine n-best parsing and MaxEnt discriminative reranking. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL05)*.

Y. Choi and C. Cardie. 2008. Learning with compositional semantics as structural inference for subsentential sentiment analysis. In *Empirical Methods in Natural Language Processing (EMNLP)*.

N. Collier, H. Park, N. Ogata, Y. Tateishi, C. Nobata, T.Ohta, T. Sekimizu, H. Imai, K. Ibushi, and J.Tsujii. 1999. The genia project: corpus-based knowledge acquisition and information extraction from genome research papers. In *In Ninth Conference of the European Chapter of the Association for Computational Linguistics (EACL-99*, pages 271–272.

I.G .and R. McDonald Councill and L. Velikovich. 2010. What's great and what's not: learning to classify the scope of negation for improved sentiment analysis (nesp-nlp 2010). *Proceedings of the Workshop on Negation and Speculation in Natural Language Processing*.

H. Cunningham, D. Maynard, K. Bontcheva, V. Tablan, N. Aswani, I. Roberts, G. Gorrell, A. Funk, A. Roberts, D. Damljanovic, T. Heitz, M.A. Greenwood, H. Saggion, J. Petrak, Y. Li, and P. Wim. 2011. *Text Processing with GATE (Version 6).* GATE (April 15, 2011).

M.-C. de Marneffe and Ch. D. Manning, 2011. *Stanford typed dependencies manual.*

M.-C. de Marneffe, B. MacCartney, and Ch.D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *LREC.*

P. Elkin, S. Brown, B. Bauer, C. Husser, W. Carruth, L. Bergstrom, and D.W. Roedler. 2005. A controlled trial of automated classification of negation from clinical notes. *BMC medical informatics and decision making*, 5.

O. Etzioni, M. Banko, and M. J. Cafarella. 2006. Machine reading. In *Proceedings of the 21st National Conference on Artificial Intelligence*, volume 2, pages 1517–1519.

R. Farkas, V. Vincze, G.Móra, J. Csirik, and G.Szarvas. 2010. The CoNLL-2010 Shared Task: Learning to detect hedges and their scope in natural language text. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning.*

T. Givón. 1993. *English Grammar: A Function- Based Introduction.* John Benjamins Publishing Co., Amsterdam, NL.

T. Givón. 2001. *Syntax, Volume I.* John Benjamins Publishing Co., Philadelphia.

C-H. Han and M. Romero. 2001. Negation, focus and alternative questions. In K. Megerdoomian and L.A. Bar-el, editors, *Proceedings of the West Coast Conference in Formal Linguistics XX*, Somerville, MA. Cascadilla Press.

S. Harabagiu, A. Hickl, and F. Lacatusu. 2006. Negation, contrast and contradiction in text processing. In *Proceedings of the 21st AAAI.*

H. Harkema, J.N. Dowling, T. Thornblade, and W. Chapman. 2009. Context: An algorithm for determining negation, experiencer, and temporal status from clinical reports. *J. of Biomedical Informatics*, 42.

V. Hatzivassiloglou and K.R. McKeown. 1997. Predicting the semantic orientation of adjectives. In *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics (ACL 97)*, pages 174–181, Madrid, Spain. Association for Computational Linguistics.

L.R. Horn. 1989. *A Natural History of Negation.* University of Chicago Press, Chicago.

Y. Huang and H.J. Lowe. 2007. A novel hybrid approach to automated negation detection in clinical radiology reports. *Journal of the American Medical Informatics Association : JAMIA*, 14(3):304-311.

R.D. Huddleston and G.K. Pullum. 2002. *The Cambridge Grammar of the English language*. Cambridge University Press, Cambridge, UK; New York.

A. Kennedy and D. Inkpen. 2005. Sentiment classification of movie and product reviews using contextual valence shifters. In *Workshop on the Analysis of Informal and Formal Information Exchange during Negotiations, FINEXIN 2005*, Ottawa, Ontario, Canada.

A. Kennedy and D. Inkpen. 2006. Sentiment classification of movie reviews using contextual valence shifters. *Computational Intelligence*, 22.

H. Kilicoglu and S. Bergler. 2009. Syntactic dependency based heuristics for biological event extraction. In *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing: Shared Task (BioNLP '09)*.

H. Kilicoglu and S. Bergler. 2010. A high-precision approach to detecting hedges and their scopes. In *Proceedings of the Fourteenth Conference on Computational Natural Language Learning — Shared Task (CoNLL '10)*.

H. Kilicoglu. 2012. *Embedding Predications*. Ph.D. thesis, Concordia University, Montreal, Quebec.

J.-D. Kim, T. Ohta, and J. Tsujii. 2008. Corpus annotation for mining biomedical events from literature. *BMC Bioinformatics*, 9(10).

J.-D. Kim, T. Ohta, S. Pyysalo, Y. Kano, and J. Tsujii. 2009. Overview of BioNLP'09 shared task on event extraction. In *Proceedings of the BioNLP 2009 Workshop Companion Volume for Shared Task*.

J.-D. Kim, Y. Wang, T. Takagi, and A. Yonezawa. 2011. Overview of genia event task in bionlp shared task 2011. In *Proceedings of BioNLP Shared Task 2011 Workshop at ACL-HLT*.

D. Klein and C.D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*.

J. Li, G. Zhou, H. Wang, and Q. Zhu. 2010. Learning the scope of negation via shallow semantic parsing (COLING '10). In *Proceedings of the 23rd International Conference on Computational Linguistics*.

I.A. Mel'čuk. 1988. *Dependency Syntax: Theory and Practice*. State University Press of New York.

K. Moilanen and S. Pulman. 2007. Sentiment composition. In *Proceedings of Recent Advances in Natural Language Processing*.

R. Morante and E. Blanco. 2012. *SEM 2012 Shared Task: Resolving the Scope and Focus of Negation. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics (*SEM 2012)*.

R. Morante and W. Daelemans. 2009. A metalearning approach to processing the scope of negation. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning*, CoNLL '09, pages 21–29, Stroudsburg, PA, USA. Association for Computational Linguistics.

R. Morante and W. Daelemans. 2012a. Conandoyle-neg: Annotation of negation cues and their scope in conan doyle stories. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC-2012)*.

R. Morante and W. Daelemans. 2012b. Processing modality and negation. a pilot task of the QA4MRE lab. In *CLEF 2012 Notebook papers*.

R. Morante and C. Sporleder, editors. 2010. *Proceedings of the Workshop on Negation and Speculation in Natural Language Processing (NeSp-NLP 2010)*.

R. Morante and C. Sporleder. 2012. Modality and negation: An introduction to the special issue. *Computational Linguistics*, 38(2):223–260.

R. Morante, A. Liekens, and W. Daelemans. 2008. Learning the scope of negation in biomedical texts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, Stroudsburg, PA, USA.

R. Morante. 2010. Descriptive analysis of negation cues in biomedical texts. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*.

P. G. Mutalik, A. Deshpande, and P.M. Nadkarni. 2001. Use of general-purpose negation detection to augment concept indexing of medical documents: a quantitative study using the umls. *Journal of the American Medical Informatics Association : JAMIA*, 8(6):598-609.

R. Nawaz, P. Thompson, and S. Ananiadou. 2010. Evaluating a meta-knowledge annotation scheme for bio-events. In *Proceedings of the Workshop on Negation and Speculation in Natural Language Processing (NeSp-NLP '10)*, pages 69–77.

R. Nawaz, P. Thompson, and S. Ananiadou. 2013. Negated bio-events: analysis and identification. *BMC Bioinformatics*, 14.

M .and P. Kingsbury Palmer and D. Gildea. 2005. The proposition bank: An annotated corpus of semantic roles. *Computational Linguistics*, 31.

B. Pang and L. Lee. 2008. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2(1-2):1–135.

B. Pang, L. Lee, and S. Vaithyanathan. 2002. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of EMNLP*, pages 79–86.

B. Partee. 1993. On the "scope of negation" and polarity sensitivity. In E. Hajicova, editor, *Functional Approaches to Language Description*.

A. Peñas, E. Hovy, P. Forner, A. Rodrigo, R. Sutcliffe, C. Sporleder, C. Forascu, Y. Benajiba, and P. Osenova. 2012. Overview of QA4MRE at CLEF 2012: Question Answering for Machine Reading Evaluation. In *CLEF 2012 Evaluation Labs and Workshop Online Working Notes)*.

L. Polanyi and A. Zaenen. 2004. Contextual lexical valence shifters. In *Proceedings of the AAAI Spring Symposium on Exploring Attitude and Affect in Text: Theories and Applications*.

J. Pustejovsky, P. Hanks, R. Sauri, A. See, R. Gaizauskas, A. Setzer, D. Radev, B. Sundheim, D. Day, L. Ferro, and M. Lazo. 2003. The TIMEBANK corpus. In *Proceedings of Corpus Linguistics*.

S. Pyysalo, F. Ginter, J. Heimonen, J. Bjorne, J. Boberg, J. Jarvinen, and T. Salakoski. 2007. Bioinfer: a corpus for information extraction in the biomedical domain. *BMC Bioinformatics*, 8.

R. Quirk, S. Greenbaum, G.Leech, and J. Svartvik. 1985. *A Comprehensive grammar of the English language*. Longman.

E. Riloff and J. Wiebe. 2003. Learning extraction patterns for subjective expressions. In *Proceedings of the 2003 conference on Empirical methods in natural language processing (EMNLP 2003)*, Stroudsburg, PA, USA. Association for Computational Linguistics.

S. Rosenberg and S. Bergler. 2012. UConcordia: CLaC Negation Focus Detection at *Sem 2012. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2:*

Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012), Montréal, Canada. Association for Computational Linguistics.

S. Rosenberg, H. Kilicoglu, and S. Bergler. 2012. CLaC Labs: Processing Modality and Negation. Working Notes for QA4MRE Pilot Task at CLEF 2012. In *CLEF (Online Working Notes/Labs/Workshop)*.

R. Sauri. 2008. *A factuality profiler for eventualities in text*. Ph.D. thesis, Brandeis University, Waltham, MA, USA.

G. Szarvas, V. Vincze, R. Farkas, and J. Csirik. 2008. The bioscope corpus: annotation for negation, uncertainty and their scope in biomedical texts. In *Proceedings of the Workshop on Current Trends in Biomedical Natural Language Processing(BioNLP '08)*.

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

G. Tottie. 1991. *Negation in English Speech and Writing: a Study in Variation*. San Diego: Academic Press.

E. Velldal, L. vrelid, J. Read, and S. Oepen. 2012. Speculation and negation: Rules, rankers, and the role of syntax. *Computational Linguistics*, 38(2):369–410.

V. Vincze, G. Szarvas, R. Farkas, G. Móra, and J. Csirik. 2008. The bioscope corpus: biomedical texts annotated for uncertainty, negation and their scopes. *BMC Bioinformatics*, 9(Suppl 11):S9.

V. Vincze, G. Szarvas, G. Móra, T. Ohta, and R. Farkas. 2011. Linguistic scope-based and biological event-based speculation and negation annotations in the bioscope and genia event corpora. *Journal of Biomedical Semantics*, 2(5):1–11.

J. Wiebe, T. Wilson, and C. Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation*, 39(2-3).

M. Wiegand, B. Roth, D. Klakow, A. Balahur, and A. Montoyo. 2010. A survey on the role of negation in sentiment analysis. In *Proceedings of the Workshop on Negation and Speculation in Natural Language Processing (NeSp-NLP 2010)*.

T. Wilson, J. Wiebe, and P. Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT- EMNLP 2005)*.

T. Wilson, J. Wiebe, and Hoffmann P. 2009. Recognizing contextual polarity: An exploration of features for phrase-level sentiment analysis. *Computational Linguistics*.

Appendix A

Stanford Dependency Relations

The following dependency relation definitions and examples are extracted directly from the Stanford typed dependencies manual (de Marneffe and Manning, 2011)[1]

***advmod*: adverbial modifier**

An adverbial modifier of a word is a (non-clausal) RB or ADVP that serves to modify the meaning of the word.

"Genetically modified food"	advmod(modified, genetically)
"less often"	advmod(often, less)

***amod*: adjectival modifier**

An adjectival modifier of an NP is any adjectival phrase that serves to modify the meaning of the NP.

"Sam eats red meat"	amod(meat, red)

***aux*: auxiliary**

An auxiliary of a clause is a non-main verb of the clause, e.g. modal auxiliary, "be" and "have" in a composed tense.

"Reagan has died"	aux(died, has)
"He should leave"	aux(leave, should)

[1] freely available from the following website: http://nlp.stanford.edu/software/dependencies_manual.pdf

***auxpass*: passive auxiliary**

A passive auxiliary of a clause is a non-main verb of the clause which contains the passive information.

"Kennedy has been killed"	auxpass(killed, been) aux(killed,has)
"Kennedy was/got killed"	auxpass(killed, was/got)

***cc*: coordination**

A coordination is the relation between an element of a conjunct and the coordinating conjunction word of the conjunct.

"Bill is big and honest"	cc(big, and)
"They either ski or snowboard"	cc(ski, or)

***ccomp*: clausal complement**

A clausal complement of a verb, adjective is a dependent clause with an internal subject which functions like an object of the verb, or adjective. Clausal complements for nouns are limited to complement clauses with a subset of nouns like "fact" or "report". We analyze them the same (parallel to the analysis of this class as "content clauses" in (Huddleston and Pullum, 2002)). Such clausal complements are usually finite (though there are occasional remnant English subjunctives).

"He says that you like to swim"	ccomp(says, like)
"I am certain that he did it"	ccomp(certain, did)
"I admire the fact that you are honest"	ccomp(fact, honest)

***complm*: complementizer**

A complementizer of a clausal complement (ccomp) is the word introducing it. It will be the subordinating conjunction "that" or "whether".

"He says that you like to swim"	complm(like, that)

***conj*: conjunct**

A conjunct is the relation between two elements connected by a coordinating conjunction, such as "and", "or", etc. We treat conjunctions asymmetrically: The head of the relation is the first conjunct and other conjunctions depend on it via the conj relation.

"Bill is big and honest"	conj(big, honest)
"They either ski or snowboard"	conj(ski, snowboard)

collapsed dependency representation for conjunct dependencies

Given the following:

"Bell, a company which is based in LA, makes and distributes computer products"

cc(makes, and)

conj(makes, distributes)

The dependencies will be *collapsed* into one single relation:

conj_and(makes, distributes)

***cop*: copula**

A copula is the relation between the complement of a copular verb and the copular verb.

"Bill is big"	cop(big, is)
"Bill is an honest man"	cop(man, is)

***det*: determiner**

A determiner is the relation between the head of an NP and its determiner.

"The man is here"	det(man, the)
"Which book do you prefer?"	det(book, which)

***dobj* : direct object**

The direct object of a VP is the noun phrase which is the (accusative) object of the verb; the direct object of a clause is the direct object of the VP which is the predicate of that clause.

"She gave me a raise"	dobj(gave, raise)
"They win the lottery"	dobj(win, lottery)

***expl*: expletive**

This relation captures an existential "there". The main verb of the clause is the governor.

"There is a ghost in the room"	expl(is, There)

***infmod*: infinitival modifier**

An infinitival modifier of an NP is an infinitive that serves to modify the meaning of the NP.

"Points to establish are..."	infmod (points, establish)
"I dont have anything to say"	infmod(anything, say)

***iobj* : indirect object**

The indirect object of a VP is the noun phrase which is the (dative) object of the verb; the indirect object of a clause is the indirect object of the VP which is the predicate of that clause.

"She gave me a raise"	iobj(gave, me)

***neg*: negation modifier**

The negation modifier is the relation between a negation word and the word it modifies.

"Bill is not a scientist"	neg(scientist, not)
"Bill doesnt drive"	neg(drive, nt)

***nsubj* : nominal subject**

A nominal subject is a noun phrase which is the syntactic subject of a clause. The governor of this relation might not always be a verb: when the verb is a copular verb, the root of the clause is the complement of the copular verb.

"Clinton defeated Dole"	nsubj(defeated, Clinton)
"The baby is cute"	nsubj(cute, baby)

***nsubjpass*: passive nominal subject**

A passive nominal subject is a noun phrase which is the syntactic subject of a passive clause.

"Dole was defeated by Clinton"	nsubjpass(defeated, Dole)

***partmod*: participial modifier**

A participial modifier of an NP or VP is a participial verb form that serves to modify the meaning of the NP or VP.

"Truffles picked during the spring are tasty"	partmod(truffles, picked)
"Bill tried to shoot demonstrating his incompetence"	partmod(shoot, demonstrating)

***pobj*: object of a preposition**

The object of a preposition is the head of a noun phrase following the preposition, or the adverbs "here" and "there". (The preposition in turn may be modifying a noun, verb, etc.) Unlike the Penn Treebank, we here define cases of VBG quasi-prepositions like "including", "concerning", etc. as instances of pobj. (The preposition can be called a FW for "pace", "versus", etc. It can also be called a CC but we don't currently handle that and would need to distinguish from conjoined prepositions.)

"I sat on the chair"	pobj(on, chair)

***preconj*: preconjunct**

A preconjunct is the relation between the head of an NP and a word that is part of a conjunction, an puts emphasis on it (e.g., "either", "both", "neither").

"Both the boys and the girls are here"	preconj(boys, both)

***prep/prepc*: prepositional modifier**

A prepositional modifier of a verb, adjective, or noun is any prepositional phrase that serves to modify the meaning of the verb, adjective, or noun. If the prepositional phrase is a clause, the relation is called prepc when collapsing takes place.

"I saw a cat in a hat"	prep(cat, in)
"I saw a cat with a telescope"	prep(saw, with)
"He is responsible for meals"	prep(responsible, for)

collapsed dependency representation for prepositional dependencies
Given the following:
"Bell, a company which is based in LA, makes and distributes computer products"
prep(based, in)
pobj(in, LA)
The dependencies involving the preposition "in" will be *collapsed* into one single relation:
prep_in(based, LA)

***rcmod*: relative clause modifier** A relative clause modifier of an NP is a relative clause modifying the NP. The relation points from the head noun of the NP to the head of the relative clause, normally a verb.

"I saw the man you love"	rcmod(man, love)
"I saw the book which you bought"	rcmod(book,bought)

***rel*: relative**
A relative of a relative clause is the head word of the WH-phrase introducing it.

"I saw the man whose wife you love"	rel(love, wife)

This analysis is used only for relative words which are not the subject nor the object of the relative clause. Relative words which act as the subject of a relative clause are analyzed as an nsubj, relative words which acts as the object of a relative clause are anlayzed as an dobj.

***xcomp*: open clausal complement**
An open clausal complement (xcomp) of a VP or an ADJP is a clausal complement without its own subject, whose reference is determined by an external subject. These complements are always non-finite. The name xcomp is borrowed from Lexical-Functional Grammar.

"He says that you like to swim"	xcomp(like, swim)
"I am ready to leave"	xcomp(ready, leave)

Appendix B

BioScope

B.1 Background for BioScope Evaluation

The data set

BioScope (Szarvas et al., 2008; Vincze et al., 2008)[1] is a freely available corpus comprised of biological and medical texts. The corpus consists of three sub corpora: biomedical full papers, abstracts from the GENIA corpus (Collier et al., 1999), and clinical (radiology) reports. Every sentence is annotated with negation and speculation triggers, as well as their corresponding linguistic scopes. Table 15 shows the statistics pertaining to negation[2].

	Abstracts	Full Papers	Clinical Reports
# Sentences	14565	3352	7520
# Documents	1273	9	1954
# Negation Cues	1750	378	872
% Sentences with Negation	13.45%	13.76%	6.6%

Table 15: Statistics of the BioScope Corpus.

BioScope negation trigger and scope annotations

In BioScope, the phenomenon of negation is defined as expressing the non-existence of something. An example sentence containing the BioScope gold annotations is presented in (105). Here,*without* is determined by BioScope to be a negation cue (within the <`cue`> tags) and they determine the negation scope (within the <`xscope`> tags) as the prepositional phrase. Thus, the `xscope` span is determined by syntax and in the majority of annotations is extended to the largest syntactic unit to the right of the cue.

[1] http://www.inf.u-szeged.hu/rgai/bioscope

[2] Refer to (Morante and Daelemans, 2009) for more detailed statistics of the BioScope Corpus.

(105) Mildly hyper inflated lungs <xscope><cue>*without*</cue> *focal opacity*</xscope>.

In the evaluation presented in Section 5.1, the two gold annotations related to negation are considered: the `cue` annotations in BioScope correspond directly to the trigger annotations in NEGATOR. The BioScope `xscope` annotations indicate the extent of its (continuous) narrow scope of a given negation trigger, equivalent, but not identical to the corresponding NEGATOR scope. The annotation convention for BioScope is to include the trigger in their narrow scope. The NEGATOR scope annotations do not consider this convention, rather assuming that the trigger is outside the scope. Thus, a *true positive* instance differs in exactly the text of the NEGATOR trigger/`xcue` as illustrated in Example (106):

(106) (a) The B cell NFAT complex, however, was ***not*** *functional, since it* ***failed*** *to activate transcription from an NFAT-driven chloramphenicol acetyl- transferase (CAT) construct.*

(b)

explicit negation

BioScope	***offset***	***scope***
cue	*549-552*	*not*
xscope	*549-563*	*not functional*
NEGATOR	***offset***	***scope***
explicitNeg	*549-552*	*not*
explicitNegScope	*553-563*	*functional*

(c)

implicit negation

BioScope	***offset***	***scope***
cue	*575-581*	*failed*
xscope	*575-677*	*failed to activate...*
NEGATOR	***offset***	***scope***
implicitNeg	*575-581*	*failed*
implicitNegScope	*582-677*	*to activate...*

B.1.1 Evaluation Setup

Each sentence is parsed with the Charniak Parser (Charniak and Johnson, 2005) and the dependency relations are extracted using the Dependency Module (de Marneffe et al., 2006). For trigger detection: the NEGATOR *implicit and explicit*[3] negation trigger word lists were used as input to the NEGATOR Trigger Detection Module (please refer to section (3.2) for preprocessing requirements for running the trigger module). For *implicit and explicit* negation scope detection: the NEGATOR *narrow* heuristics were employed in the NEGATOR Scope Detection module (please refer to section (4.3) for the preprocessing requirements for running the scope module).

[3]BioScope does not annotate *affixal* negation, therefore this list was not used.

B.2 BioScope Error Analysis

This section presents a fine-grained view of the results for the evaluation of the NEGATOR trigger-scope modules on the BioScope corpus (see section 5.1). It also describes the different `partial` and `omit` classes pertaining to the evaluation.

Table 16 presents the results of running the NEGATOR scope detection module and NEGATOR trigger lists categorized by negation trigger type on BioScope. The `false negative` category presented in Table 16 is further subdivided into:

1. `partials`: BioScope and NEGATOR annotations have significant overlap but diverge on the exact extent of the scope span.

2. `omits`: NEGATOR does not detect a negation `trigger` for an existing BioScope `cue`, or NEGATOR does not detect a negation `scope` for an existing BioScope `xscope`.

BioScope Abstracts

		false negatives							
	tp	**partial**	**omits**	**fp**	**Gold**	NEGATOR	**p**	**r**	**f**
implicit	188	56	13	1744	257	1988	.10	.73	.18
explicit	1159	263	71	169	1493	1591	.87	.78	.82
Totals	1347	319	84	1913	1750	3579	.41	.77	.54

BioScope Full Texts

		false negatives							
	tp	**partial**	**omits**	**fp**	**Gold**	NEGATOR	**p**	**r**	**f**
implicit	30	15	1	225	46	270	.12	.65	.20
explicit	229	93	10	73	332	395	.76	.69	.72
Totals	259	108	11	298	378	665	.47	.69	.56

BioScope Clinical Reports

		false negatives							
	tp	**partial**	**omits**	**fp**	**Gold**	NEGATOR	**p**	**r**	**f**
implicit	5	18	5	68	28	91	.07	.17	.14
explicit	712	101	31	32	844	845	.96	.84	.89
Totals	717	119	36	100	872	936	.88	.82	.85

Table 16: NEGATOR Scope detection on BioScope Corpus using NEGATOR trigger lists

We observe from Table 16 that the majority of `false negative` instances are not because of `omit` errors, rather are classified as `partial` errors. The different partial error classes and the omit error classes pertaining to this evaluation are described in greater detail in the next two sections.

B.2.1 `partial` error classes

Table 17 breaks down the `partial` errors into eight different categories organized by trigger type for all data sets. Examples from the resultant classes follow in the next subsection.

	BioScope Abstracts			BioScope Full Papers			Clinical Reports		
	Implicit	**Explicit**	**Total**	**Implicit**	**Explicit**	**Total**	**Implicit**	**Explicit**	**Total**
NEGATOR s_offset < Bio s_offset NEGATOR e_offset = Bio e_offset	0	79	79	0	5	5	0	11	11
NEGATOR s_offset < Bio s_offset, NEGATOR e_offset > Bio e_offset	0	3	3	0	6	6	0	1	1
NEGATOR s_offset < Bio s_offset, NEGATOR e_offset <Bio e_offset	4	3	7	0	1	1	3	1	4
NEGATOR s_offset = Bio s_offset, NEGATOR e_offset < Bio e_offset	18	62	80	5	13	18	1	67	68
NEGATOR s_offset = Bio s_offset, NEGATOR e_offset > Bio e_offset	15	44	59	6	31	37	14	11	25
NEGATOR s_offset > Bio s_offset, NEGATOR e_offset = Bio e_offset	16	48	64	3	25	28	0	8	8
NEGATOR s_offset > Bio s_offset, NEGATOR e_offset < Bio e_offset	0	10	10	0	3	3	0	0	0
NEGATOR s_offset > Bio s_offset, NEGATOR e_offset > Bio e_offset	3	14	17	1	9	10	0	2	2
Total Partials	**56**	**263**	**319**	**15**	**93**	**108**	**18**	**101**	**119**

Table 17: Partial Results of NEGATOR Scope Detection on BioScope Corpus.

B.2.1.1 NEGATOR start offset starts before BioScope

The first class: (NEGATOR `s_offset < Bio s_offset`,NEGATOR `e_offset = Bio e_offset`) occurs with the negation trigger *no* when the heuristic triggered by the *det* dependency relation is used. Specifically, NEGATOR will include the main verb linked by the *dobj* dependency relation to the direct object as a second discontinuous scope span. As a result, both BioScope and NEGATOR agree on the right scope span (to the right of the negation trigger), but NEGATOR includes a second left scope span which BioScope does not, as illustrated in Example (107).

(107) *(a) Ascorbate and AZT also had no effect on NF-kappa B activation following TNF-alpha- or PMA-induced stimulation of U1 promonocytic cells.*

(b) `det(effect,no)`, `dobj(had,effect)`

explicit negation

BioScope	***offset***	***scope***
cue	*975-977*	*no*
xscope	*975-1082*	*no effect on NF-kappa B activation following TNF-alpha- or PMA-induced stimulation of U1 promonocytic cells*

(c)

NEGATOR*:*	***offset***	***scope***
explicitNeg	*975-977*	*no*
explicitNegScope	*971-974*	*had*
explicitNegScope	*976-1082*	*effect on NF-kappa B activation following TNF-alpha- or PMA-induced stimulation of U1 promonocytic cells*

This class is responsible for 25% of the total `partial` errors in the Abstracts dataset. However, for the other data sets it is less significant.

B.2.1.2 Mismatch in start and end scope offset boundary

Classes 2 and 3 are related cases in that they are triggered by the same heuristic as the first class, with the negation trigger *no*. However, in these cases either the right NEGATOR scope span is longer than BioScope's or shorter. The Full Papers data set contains the most errors resulting from Class 2 (NEGATOR `s_offset` $<$ `Bio s_offset`,NEGATOR `e_offset` $>$ `Bio e_offset`). Example (108) illustrates such a case where the right scope span of NEGATOR is longer. The scope span is determined to be the phrasal node (a noun phrase), which dominates both the direct object and the prepositional object. However, the constituent within parentheses is included within the resulting noun phrase, which is not included in the BioScope `xscope` span.

(108) *(a) In contrast, a complete loss of Ser signaling had no effect on bristle density (Figure 1K).*

(b)
```
det(effect,no)
prep_on(effect,density)
dobj(had,effect)
```

explicit negation

BioScope	***offset***	***scope***
cue	*14667-14669*	*no*
xscope	*14667-14695*	*effect on bristle density*

(c)

NEGATOR*:*	***offset***	***scope***
explicitNeg	*14667-14669*	*no*
explicitNegScope	*14663-14666*	*had*
explicitNegScope	*14670-14707*	*effect on bristle density (Figure 1K).*

B.2.1.3 BioScope end offset ends after NEGATOR

Class 4: (NEGATOR `s_offset = Bio s_offset`,NEGATOR `e_offset < Bio e_offset`) accounts for 25% of the total number of `partial` errors in the Abstracts data set. It is also quite significant for the Clinical Reports where it accounts for 57% of total `partial` errors. We observe that in the Abstracts data set this error class is trigged in the majority of cases by explicit negation triggers, however it is responsible for the majority of `partial` errors within the implicit negation category as well. The explicit negation trigger *no* is the major culprit to initiate this error in the Clinical Reports data set. The following three examples illustrate specific `partial` errors present in this class.

1: Example (109) illustrates a case where NEGATOR fails to determine the correct span. There is a *part_mod* (participial modifier) relation between (scarring, identified), which NEGATOR does not account for, and will only find the noun phrase which dominates both *scarring* and *no* to be the scope span.

(109) *(a)* *There is no hydronephrosis or cortical scarring identified on the current exam.*

(b) `det(scarring,no)`

(c)

explicit negation

BioScope	***offset***	***scope***
cue	*138-140*	*no*
xscope	*138-207*	*no hydronephrosis or cortical scarring identified on the current exam*

NEGATOR*:*	***offset***	***scope***
explicitNeg	*138-140*	*no*
explicitNegScope	*141-176*	*hydronephrosis or cortical scarring*

2: Not including a prepositional phrase that BioScope does is another major source of error here, as illustrated in (110).The error is not that the Charniak parsed sentence does not have the *prep_of* relation correctly identified, rather it does not find the *det* relation between (hydronephrosis, no), rather it finds the *dep(hydronephrosis, no)*. This is the most generic dependency relation, and is used when another relation cannot be found. NEGATOR does have a default heuristic to catch the *dep* case, but it is not refined: i.e. it will not include cases for finding attached prepositional phrase in order to extend the scope span.

(110) (a) *There is no hydronephrosis or loss of corticomedullary junction.*

(b) `det(hydronephrosis,no)`
`prep_of(hydronephrosis,junction)`

(c)

explicit negation

BioScope	***offset***	***scope***
cue	*55-57*	*no*
xscope	*55-109*	*no hydronephrosis or loss of corticomedullary junction*
NEGATOR*:*	***offset***	***scope***
explicitNeg	*55-57*	*no*
explicitNegScope	*58-80*	*hydronephrosis or loss*

3: As previously discussed this `partial` class incurs the majority of implicit negation errors in the Abstracts data set. The implicit negation trigger *absence* incurs many such errors. As illustrated in (111), NEGATOR triggers the correct heuristics however the candidate scope is the noun phrase constituent: *the presence or absence of ascorbate* and does not include the terms after. This issue regarding items which should be conjunctions by using commas, is a reoccurring issue which incurs errors, in that these terms are not included in necessarily included in the same parent constituent.

(111) (a) *. . . we carried out gel shift analysis on nuclear extracts prepared under different conditions of cell stimulation in the presence or absence of ascorbate, N-acetylcysteine (NAC), or zidovudine (AZT).*

(b) `prep_of(presence,ascorbate)`
`conj_or(presence,absence)`

(c)

implicit negation

BioScope	***offset***	***scope***
cue	*568-575*	*absence*
xscope	*568-633*	*absence of ascorbate, N-acetylcysteine (NAC), or zidovudine (AZT)*
NEGATOR*:*	***offset***	***scope***
implicitNeg	*568-575*	*absence*
implicitNegScope	*576-588*	*of ascorbate*

B.2.1.4 NEGATOR end offset ends after BioScope

Error class 5: (NEGATOR `s_offset = Bio s_offset`,NEGATOR `e_offset > Bio e_offset`) is a major culprit for `partial` errors in the Full Abstracts data set. The explicit negation trigger *not* incurs many of these errors. The following two examples illustrate errors in this class.

1: As illustrated in Example (112), NEGATOR will identify the *neg* dependency relation between (predict, not), and subsequently the verb phrase headed by the main verb

(predict) will be the scope span. However, there is the relative clause headed by *which* contained in the verb phrase. BioScope determines that the relative clause should not be included in the scope, however NEGATOR does not apply this particular pruning strategy.

(112) (a) *. . . Expression and Disorder modules do not predict any protein pairs (positive or negative) above a posterior odds ratio of 1 , which is expected as the highest likelihood ratios they achieve are lower than 400 (see Figure 1A).*

(b) `neg (predict,not)`

explicit negation

BioScope	***offset***	***scope***
cue	*24801-24804*	*not*
xscope	*24801-24887*	*predict any protein pairs (positive or negative) above a posterior odds ratio of 1*

(c)

NEGATOR*:*	***offset***	***scope***
explicitNeg	*24801-24804*	*not*
explicitNegScope	*24805-24988*	*predict any protein pairs (positive or negative) above a posterior odds ratio of 1, which is expected as the highest likelihood ratios they achieve are lower than 400 (see Figure 1A).*

2: This fifth `partial` error class is also responsible for 78% of implicit negation errors in the Clinical Reports data set. Here, the same negation trigger *negative* always incurs the error. Again we observe that since the clinical reports data set has many sentences with the same syntactic pattern. Consequently, the errors in this class all occur due to one reoccurring syntactic pattern which NEGATOR interprets differently than BioScope. Example (113) demonstrates such a case: NEGATOR will invoke the heuristic for the *amod* dependency relation and subsequently the noun modified by the adjective *negative* is determined to be the negation scope. However, BioScope will consistently in this case only mark *negative* as the scope of negation and not the modified noun.

(113) (a) *Negative examination. UTI.*

(b) `amod (examination, Negative)`

implicit negation

BioScope	***offset***	***scope***
cue	*9-17*	*Negative*
(c) *xscope*	*9-17*	*Negative*
NEGATOR*:*	***offset***	***scope***
implicitNeg	*9-17*	*Negative*
implicitNegScope	*18-29*	*examination*

B.2.1.5 BioScope start offset starts before NEGATOR

The last three `partial` classes all share the common property that the BioScope annotation starts to the left of the negation trigger. BioScope will annotate to the left of the negation trigger in very specific instances. The first case is when the negated verb is linked to the passive subject constituent via the *nsubjpass* dependency relation. NEGATOR has a heuristic for this case and performs very well in detecting the passive subject. However, there are other less consistent instances where BioScope will start the scope span to the right of the trigger. NEGATOR by default does not deal with many of these cases.

Of these last three classes the one with the majority of errors is class 6:(NEGATOR `s_offset > Bio s_offset`,NEGATOR `e_offset = Bio e_offset`) for all three data sets. In fact, 20% of total `partial` errors occur here for the Abstracts, and 25% for the Full Papers set. The majority of instances are incurred by the explicit negation trigger *not*, but proportionally quite a few implicit negation instances as well (i.e. unable). The following three examples illustrate errors from this class.

1: In Example (114), BioScope decides to not only determine the scope of negation to be the verb phrase, rather they also unconventionally determine that the subject constituent should be also part of the scope span. NEGATOR does not, and therefore the resultant scope span does not fully match.

(114) (a) *. . . because in five patients, normal TCRzeta levels were present although kappaB binding was not inducible.*

(b) `neg(inducible,not)`

explicit negation

BioScope	***offset***	***scope***
cue	*3230-3233*	*not*
(c) *xscope*	*3211-3243*	*kappaB binding was not inducible*
NEGATOR*:*	***offset***	***scope***
explicitNeg	*3230-3233*	*not*
explicitNegScope	*3234-3243*	*inducible*

2: Another inconsistent error is illustrated in Example (115), where BioScope unconventionally determines that the *was* term should also be within the negation scope and NEGATOR does not.

(115) *(a)* *. . . and was not accounted for by the advanced age of the study cohort.*

(b) `neg(accounted,not)`

explicit negation

BioScope	***offset***	***scope***
cue	*2792-2795*	*not*
xscope	*2788-2849*	*was not accounted for by the advanced age of the study cohort*

(c)

NEGATOR*:*	***offset***	***scope***
explicitNeg	*2792-2795*	*not*
explicitNegScope	*2796-2849*	*accounted for by the advanced age of the study cohort*

3: Finally, the implicit negation trigger *unable* also incurs numerous errors within this error class. As shown in Example (116), BioScope determines that with the trigger *unable*, not only should the adjective phrase be in the negation scope, but the subject constituent as well. Again, here NEGATOR will only annotate the adjective phrase.

(116) *(a)* *VDR DNA-binding mutants were unable to either bind to this element in vitro or repress in vivo. . .*

(b) `xcomp(unable,bind)`

implicit negation

BioScope	***offset***	***scope***
cue	*1209-1215*	*unable*
xscope	*1180-1274*	*VDR DNA-binding mutants were unable to either bind to this element in vitro or repress in vivo*

(c)

NEGATOR*:*	***offset***	***scope***
implicitNeg	*1209-1215*	*unable*
implicitNegScope	*1216-1274*	*to either bind to this element in vitro or repress in vivo*

B.2.2 `omit` error classes

This section of the Appendix describes the different `omit` errors pertaining to the evaluation of NEGATOR on BioScope (see section 5.1). Table 18 shows the distribution of `omits` errors for each data set organized by the negation trigger that would require a scope. The GOLD column for each data set shows the distribution of that trigger as a gold `cue`. The NEGATOR

column shows the number of NEGATOR triggers that match and have at least a partially overlapping scope annotation with the corresponding BioScope `xscope` annotation.

Negation Trigger Missed	Abstracts			Full Papers			Clinical Reports		
	NEGATOR	GOLD	% Missing	NEGATOR	GOLD	% Missing	NEGATOR	GOLD	% Missing
not	1020	1070	4.7%	217	218	.45%	60	62	3.2%
no	201	212	5.2%	49	50	2%	650	675	3.7%
without	81	83	2.4%	23	26	11.5%	96	97	1%
nor	41	44	6.8%	2	2	0	1	1	0
neither	40	42	4.8%	6	6	0	1	1	0
either	0	2	100%	0	0	0	0	1	100%
instead of	3	3	0%	4	4	0	0	0	0
rather than	19	20	5%	8	13	38.4%	1	1	0
favored over	0	0	0	0	0	0	0	2	100%
ruled out	0	0	0	0	0	0	0	1	100%
with the notable exception of	0	1	100%	0	0	0	0	0	0
absence	52	57	8.7%	6	6	0	1	1	0
absent	11	13	15.3%	3	3	0	0	0	0
excluding	0	0	0	1	1	0	1	1	0
failure	7	8	12.5%	1	2	50%	0	0	0
loss	0	1	100%	0	0	0	0	0	0
lack	53	54	1.9%	15	15	0	4	4	0
lacking	27	29	6.9%	5	5	0	0	0	0
negative	1	1	0	0	0	0	17	21	19%

Table 18: Scope errors of NEGATOR on BioScope Corpus.

The `omit` errors resulting from the negation trigger *not* are most frequent. There are two different error classes that result from the *not* trigger being determined by NEGATOR to have no scope in all three data sets:

B.2.2.1 Elliptical sentences with *not*

In the case of elliptic sentences, BioScope will annotate the negation trigger as both the `cue` and the `xscope` as illustrated in Example (117b): since the verbal phrase (the scope of *not*) in the sentence (117a) is not repeated. In contrast, NEGATOR will only mark the trigger and no scope annotation.

(117) *(a)* *... whereas those of I kappa B alpha/MAD-3 mRNA did not .*

explicit negation

	BioScope	***offset***	***scope***
	cue	*1778-1781*	*not*
(b)	*xscope*	*1778-1781*	*not*
	NEGATOR	***offset***	***scope***
	explicitNeg	*1778-1781*	*not*
	explicitNegScope -		

B.2.2.2 The *but ... not* pattern

The second case is an error with the *... but not ...* construction. As illustrated in (118a), one would expect negator to detect the *conj_negcc* relation between the first and second *CD*

terms. However very often the Charniak Parser will not identify any dependency relation, and consequently no scope will be identified.

(118) (a) *The tumour associated cell surface antigen A6H is costimulatory for human CD4+ but not CD8+ T cells .*

explicit negation

BioScope	***offset***	***scope***
cue	*101-104*	*not*
xscope	*101-117*	*not CD8+ T cells*

(b)

NEGATOR ***offset***	***scope***
explicitNeg	*101-104*	*not*
explicitNegScope	-	

B.2.2.3 Errors resulting from conjuncts

For the Abstracts data set, the second most common error occurs with the *no* trigger. As illustrated in Example (119): the presence and position of the *or* in the sentence compels the dependency module to create the *conj_or* dependency relation between (little, no). Consequently, no dependency relation is created between (effect ,no). Instead, *effect* is identified as an adverb modifier of *little*. NEGATOR does not account for this type of construction and therefore no scope is identified.

(119) *ML-9 had little or* <xscope><cue>*[no] effect on the morphology of U937 cells* </cue></xscope>*,...*

The most common error in the Clinical Reports data set also occurs with the *no* trigger. As the syntactic patterns in the sentences in the Clinical Reports are quite repetitive: when one pattern is missed this may incur many errors, as is the case here. The pattern involves *...with no...* , as illustrated in Example (120). In this case NEGATOR would expect the scope heuristic triggered by the *det* relation to be invoked. However, in the case of the Charniak parsed sentence the *det* relation between (no, fever) is not created, rather *fever* is determined to be a noun compound modifier of *no* and the *nn* dependency relation is created. NEGATOR has no such heuristic and consequently no scope span is created.

(120) (a) *This is a 9 year old patient with one episode of urinary tract infection and hematuria with no fever.*

explicit negation

BioScope	***offset***	***scope***
cue	*223-225*	*no*
xscope	*223-231*	*no fever*

(b)

NEGATOR	***offset***	***scope***
explicitNeg	*223-225*	*no*
explicitNegScope	-	

B.2.2.4 Issue with the trigger *negative*

The negation scope for the trigger word *negative* in the Clinical Reports data set tends to either incur `partial` errors (see `partial` error analysis section) or as seen here it incurs an omit error. However, here it is not due to errors in determining the scope, rather NEGATOR detects no scope for the negation trigger and BioScope will determine the negation trigger itself to be the scope. This case is illustrated in Example (121) where NEGATOR determines *negative* to be a trigger but finds no scope and BioScope annotates both the `cue` and `xscope` annotations to be *negative*.

(121) *(a)* *. . . Otherwise negative.*

explicit negation

BioScope	***offset***	***scope***
cue	*75-83*	*negative*
xscope	*75-83*	*negative*

(b)

NEGATOR	***offset***	***scope***
explicitNeg	*75-83*	*negative*
explicitNegScope	-	

B.2.2.5 Missing triggers from NEGATOR trigger lists

There are also `omit` errors due to the BioScope cues not being present in the NEGATOR trigger lists. The following `cues` *either, favored over, ruled out and with the notable exception of* incur such errors.

(122) *(a)* *Right upper lobe linear density is favored to represent subsegmental atelectasis over early infiltrate.*

explicit negation

BioScope	***offset***	***scope***
cue	*44-51*	*favored*
cue	*90-94*	*over*
xscope	*9-111*	*Right upper lobe linear density is favored to represent subsegmental atelectasis over early infiltrate*

(b)

NEGATOR	***offset***	***scope***
explicitNeg	-	
explicitNegScope	-	

The other `omit` errors are due to parser inconsistencies, or due to the relevant dependency relations not being detected. As observed from the relevant GOLD columns in Table 18, in the cases of *missing* implicit negation scope for triggers such as *lack, lacking, absence , failure* or even for the explicit the negation trigger *rather than*: these triggers when they

occur, they occur quite frequently (i.e. *absence* occurs 57 times in the Abstracts subcorpus). Consequently, proportionally very few are *missing* and are at least partially correct NEGATOR scope annotations. Given the relatively few `omit` errors overall, that NEGATOR's heuristics seem to be quite effective within a domain that it was not tailored to.

Appendix C

Conan Doyle Extended Analysis

C.1 Background for Conan Doyle Evaluation

The data set

The Conan Doyle corpus is comprised of the following stories of Sir Arthur Conan Doyle annotated with negation triggers and scope : a training set of 3644 sentences drawn from *The Hound of the Baskervilles*, a development set of 787 sentences taken from *Wisteria Lodge* and a held-out test set of 1089 sentences from *The Cardboard Box* and *The Red Circle*. The Corpus statistics for the training and development sets are shown in Table 19.

	Training Set	Development Set
# Sentences	3644	787
# Sentences w/ Negation	848	144
# Cues	984	173
# Scopes	887	168

Table 19: Corpus statistics: (Morante and Blanco, 2012).

The Conan Doyle test set is provided in the format depicted in Figure 12. Each row represents a single token in a sentence. In addition to information concerning negation cues, scopes and events, the data is tokenized, lemmatized, PoS-tagged and parsed (Morante and Blanco, 2012). Negation is represented in the three rightmost columns in Figure 12: cues are found in the first one, scope tokens relevant to the cue in the second column and negated events in the third. In the case of multiple scopes in one sentence, each additional cue originates its own triplet of columns, so that the length of each row depends on the number of scopes in the sentence.

Chapter	Sentence#	Token#	Token	Lemma	Pos	Constituent	- - - - - Negation - - - - -		
baskervilles02	101	0	He	He	PRP	(S(NP*)	_	He	_
baskervilles02	101	1	never	never	RB	(ADVP*)	never	_	_
baskervilles02	101	2	returned	return	VBD	(VP*)	_	returned	returned
baskervilles02	101	3	.	.	.	*)	_	_	_

Figure 12: A sample sentence from the Conan Doyle Corpus.

Evaluation Setup

The Conan Doyle corpus is provided in a format that also includes all the required pre-processing information: sentences, tokens, lemmas, PoS-tags for each token and gold parse trees. In this evaluation, we used the pre-existing sentence and token annotations from the gold standard. Each sentence is parsed with the Charniak Parser (Charniak and Johnson, 2005) and the dependency relations are extracted using the Dependency Module (de Marneffe et al., 2006). For trigger detection: the NEGATOR *implicit, explicit and selfNeg* negation trigger word lists were used as input to the NEGATOR Trigger Detection Module. For *implicit, explicit, and selfNeg* negation scope detection: the NEGATOR *wide* heuristics were employed in the NEGATOR Scope Detection module.

C.2 Conan Doyle Error Analysis

This section presents a fine-grained view of the results for the evaluation of the negator trigger-scope modules on the entire Conan Doyle Corpus (see section 5.1). It also describes the different `partial` and `omit` classes pertaining to the evaluation.

Table 20 shows the extended results for negation scope detection using the NEGATOR lists on the entire Conan Doyle corpus. The results are grouped by trigger type and the `false negative` instances have been subcategorized into `omits` and `partials`.

Conan Doyle Training Set

					false negatives				
	gold	NEGATOR	**tp**	**fp**	**partials**	**omits**	**precision**	**recall**	**f-score**
Explicit	721	789	530	69	190	1	.88	.73	.80
Impicit	37	174	28	139	7	2	.17	.76	.28
Affixal	129	156	65	44	47	17	.60	.50	.55
Totals	887	1119	623	252	244	20	.71	.70	.70

Conan Doyle Dev Set

					false negatives				
	gold	NEGATOR	**tp**	**fp**	**partials**	**omits**	**precision**	**recall**	**f-score**
Explicit	134	149	83	15	51	0	.85	.62	.72
Impicit	5	23	2	19	2	1	.10	.40	.16
Affixal	29	30	13	4	13	3	.76	.45	.57
Totals	168	202	98	38	66	4	.72	.58	.64

Conan Doyle Test Set

					false negatives				
	gold	NEGATOR	**tp**	**fp**	**partials**	**omits**	**precision**	**recall**	**f-score**
Explicit	213	238	147	27	64	2	.84	.69	.76
Impicit	1	31	0	30	1	0	-	-	-
Affixal	35	45	18	11	16	1	.62	.51	.56
Totals	249	314	165	68	81	3	.71	.66	.68

Table 20: NEGATOR Scope Detection on Conan Doyle Corpus grouped by trigger type.

Table 20 shows that the majority of instances found in the *false negative* category for all datasets are due to *partial* matches and not full misses. In fact 92% in the Training set, 94% in the Development set, and 96% in the Test set of *false negatives* are partial matches. The different partial error classes and the omit error classes pertaining to this evaluation are described in greater detail in the next two sections.

C.2.1 `partial` error classes

The results in Table 20 show that the percentage of `partial` matches out of total gold scope spans is 27%, 40% and 33% for the training, development and test sets respectively. Table 21[1] further breaks down the `partial` category from Table 20. The individual results are

[1] N = NEGATOR, CD = Conan Doyle.

grouped by negation type. The columns in Table 21 are organized in the following manner: since there are at least 2 discontinuous gold spans for given negation trigger, the `partial` category has been divided according to the accuracy of the following NEGATOR scope spans:
LHS: the scope span extending to the left of the negation trigger, and
RHS: the scope span extending to the right of the negation trigger.
Subsequently, there are three distinct classes:

1: N $\neq$ CD, N $\equiv$ CD: The LHS scope span is not a match, the RHS scope span is an exact match.

2: N $\equiv$ CD, N $\neq$ CD: The LHS scope span is an exact match, the RHS scope span not a match.

3: N $\neq$ CD, N $\neq$ CD: Both the LHS and RHS scope spans are not correct matches.

Training Set				
	LHS, RHS N $\neq$ CD, N $\equiv$ CD	**LHS, RHS** N $\equiv$ CD, N $\neq$ CD	**LHS, RHS** N $\neq$ CD, N $\neq$ CD	**Total Partial**
Explicit	100	83	7	190
Implicit	5	1	1	7
Affixal	25	19	3	47
	130	103	11	244
Dev Set				
	LHS, RHS N $\neq$ CD, N $\equiv$ CD	**LHS, RHS** N $\equiv$ CD, N $\neq$ CD	**LHS, RHS** N $\neq$ CD, N $\neq$ CD	**Total Partial**
Explicit	18	25	8	51
Implicit	1	0	1	2
Affixal	11	2	0	13
	30	27	9	66
Test Set				
	LHS, RHS N $\neq$ CD, N $\equiv$ CD	**LHS, RHS** N $\equiv$ CD, N $\neq$ CD	**LHS, RHS** N $\neq$ CD, N $\neq$ CD	**Total Partial**
Explicit	29	30	5	64
Implicit	1	0	0	1
Affixal	12	4	0	16
	42	34	5	81

Table 21: Partial Results of NEGATOR Scope Detection on Conan Doyle Corpus.

C.2.1.1 Mismatch in scope span extending to the right of trigger

Table 21 shows that in the Training Set and for almost half of the Test set, the majority of `partial` cases for all negation types results in the first class: small N $\neq$ CD, N $\equiv$ CD. This is encouraging as that indicates that in the majority of cases the RHS span (to the right of the negation trigger) is an exact match with the gold standard. Consequently, the errors in this class occur with the determination of the LHS span. The following five examples illustrate specific `partial` errors present in this class.

1: In all data sets, over half of the explicit negation instances in this class of a partial category error occur with the negation trigger *not*. A common cause for mismatch is illustrated in Example (123). This error occurs due to the gold standard starting the LHS span from the beginning of the subordinate clause (*why...*) whereas NEGATOR will only start the LHS span from the subject constituent (in this case *I*). Similar issues occur with other subordinate clauses that start with terms like: (*had I ...*, *which I ...*).

(123) (a) "...there is no reason why I should not be perfectly frank . "

explicit negation

Conan	***offset***	***scope***
cue	*17490-17493*	*not*
scope	*17477-17489*	*why I should*
scope	*17494-17512*	*be perfectly frank*

(b)

NEGATOR	***offset***	***scope***
expNeg	*7490-17493*	*not*
expNegScope	*17481-17489*	*I should*
expNegScope	*17494-17512*	*be perfectly frank*

2: Another cause for this mismatch with the negation trigger *not*, is when the subject is not correctly identified is illustrated in Example (124). This error occurs when the subject constituent is to be found in the first coordinate clause, and the negation trigger is located in the second coordinate clause. NEGATOR has a *wide* scope heuristic for this situation where, one will first identify the coordinate term via the *conj* relation, and once found, the first coordinate term will be identified to have the *nsubj* relation and the subject will be found. However, as seen in Example (124), the *conj* relation not correctly identified, due a Stanford Dependency Module error - and consequently the relevant NEGATOR heuristic is never triggered.

(124) (a) "...the marks which you saw were on the path and not on the grass ?."

explicit negation

Conan	***offset***	***scope***
cue	*1807-1810*	*not*
scope	*1762-1771*	*the marks*
scope	*1786-1790*	*were*
scope	*1811-1823*	*on the grass*

(b)

NEGATOR	***offset***	***scope***
expNeg	*1807-1810*	*not*
expNegScope	*1811-1823*	*on the grass*

3: The explicit negation trigger *no* is the second most frequent culprit in this class of error for all datasets. A common cause for mismatch in this case is where NEGATOR

will mark the entire LHS span, by identifying the parent node which contains the two nodes in the *expl*dependency relation, and then pruning out all constituent including and proceeding the negation trigger. However, the gold standard will annotate a similar constituent, but will also remove from the scope span lexical items like adverbs (i.e. certainly). Example (125) illustrates such a case, where the adverb *certainly* is pruned out of the gold LHS span:

(125) (a) "There was certainly no physical injury of any kind."

explicit negation

	Conan	***offset***	***scope***
	cue	*21531- 21533*	*no*
	scope	*21511-21520*	*There was*
(b)	*scope*	*21534- 21561*	*physical injury of any kind*
	NEGATOR	***offset***	***scope***
	expNeg	*21531-21533*	*no*
	expNegScope	*21511-21530*	*There was certainly*
	expNegScope	*21534- 21561*	*physical injury of any kind*

4: In contrast, there are errors in this class with the explicit negation trigger *n't/not* where NEGATOR will not include the adverb (i.e. just), where the gold standard chooses a wider scope span for the LHS as illustrated in (126). In this example NEGATOR will annotate the subject constituent and the auxiliary verb, however the gold standard will mark the entire span:

(126) (a) " I just do n't attempt to explain it ."

explicit negation

	Conan	***offset***	***scope***
	cue	*4164-4167*	*n't*
	scope	*4154-4163*	*I just do*
(b)	*scope*	*4168- 4169*	*attempt to explain it*
	NEGATOR	***offset***	***scope***
	expNeg	*4164-4167*	*n't*
	expNegScope	*4154-4155*	*I*
	expNegScope	*4161-4163*	*do*
	expNegScope	*4168-4189*	*attempt to explain it*

5: A final case illustrated for explicit negation triggers in this error class is where the cause for mis-match is when NEGATOR does identify a subject constituent but the constituent found is not the one that the gold standard allocates in its determination of the LHS span of the trigger. Again, this type of error occurs frequently with coordinate clauses. As illustrated in Example (127), NEGATOR does identify the *conj* relation (wanted, ask) and consequently the *nsubj* relation between (wanted, he) is

also not identified. Consequently, NEGATOR will determine *he* to be the LHS scope span. However the gold standard marks the subject external to the clausal complement constituent as being the LHS. Their annotation makes more sense, and again this is possibly an error due to the Stanford Dependency module regarding the determination of the *conj* relation.

(127) (a) "...he offered me two guineas if I would do exactly what he wanted all day and ask no questions."

explicit negation

Conan	***offset***	***scope***
cue	*16991-16993*	*no*
scope	*16941-16948*	*I would*
scope	*16987-16990*	*ask*
scope	*16994-17003*	*questions*

(b)

NEGATOR	***offset***	***scope***
expNeg	*16991-16993*	*no*
expNegScope	*16965-16967*	*he*
expNegScope	*16987-16990*	*ask*
expNegScope	*16994-17003*	*questions*

C.2.1.2 Affixal Negation: mismatch in left scope span

The `partial` category: (N $\neq$ CD, N $\equiv$ CD) contains the majority of partial category errors for instances of *affixal* negation. The errors in this class occur with the determination of the RHS span where the LHS span is correct. The following five examples illustrate specific `partial` errors present in this class.

1: A common cause for mismatch in all data sets is illustrated in Example (128). In this example, the gold standard marks the adverb *most* as part of the LHS scope span where NEGATOR only marks the determiner *a*.

(128) (a) The woman who approached me was certainly that , and of a most uncommon type.

affixal negation

Conan	***offset***	***scope***
cue	*17783-17785*	*un*
scope	*17776-17782*	*a most*
scope	*17785-17796*	*common type*

(b)

NEGATOR	***offset***	***scope***
selfNeg	*17783-17785*	*un*
selfNegScope	*17776-17777*	*a*
selfNegScope	*17785-17796*	*common type*

2: However, as illustrated in a second cause for mismatch in Example (129), the gold standard in this case will not mark the adverb as part of the scope span, while NEGATOR does:

(129) (a) *It is surely inconceivable that he could have held out upon the moor during all that time.*

affixal negation

Conan	***offset***	***scope***
cue	*2687- 2689*	*in*
scope	*2674- 2679*	*It is*
scope	*2689-2673*	*conceivable that he could have held out upon the moor during all that time*

(b)

NEGATOR	***offset***	***scope***
selfNeg	*2687-2689*	*in*
selfNegScope	*2674-2686*	*It is surely*
selfNegScope	*2689-2673*	*conceivable that he could have held out upon the moor during all that time*

3: A third common cause for mismatch is illustrated in Example (130) where the gold standard does not annotate any scope span on the LHS (except for the root term) and NEGATOR does:

(130) (a) *The light shone steadily as if he were standing motionless.*

affixal negation

Conan	***offset***	***scope***
cue	*14059- 14063*	*less*
scope	*14053-14059*	*motion*

(b)

NEGATOR	***offset***	***scope***
selfNeg	*14059-14063*	*less*
selfNegScope	*14036-14059*	*he were standing motion*

4: Another cause for mismatch, is when NEGATOR will take a more narrow approach and only annotate the relevant subject, where the gold standard will annotate all the relevant constituents in the clause relevant to the negation trigger. Such a case is illustrated in Example (131):

(131) (a) *. . . which will probably swallow up the remainder of his fortune and so draw his sting and leave him harmless for the future.*

(b)

affixal negation

Conan	***offset***	***scope***
cue	*9757- 9761*	*less*
scope	*9657-9667*	*which will*
scope	*9743-9757*	*leave him harm*
scope	*9762-9776*	*for the future*
NEGATOR	***offset***	***scope***
selfNeg	*9757-9761*	*less*
selfNegScope	*9749-9757*	*him harm*
selfNegScope	*9762-9776*	*for the future*

5: A final example for affixal negation errors `partial` errors in this class is when no NEGATOR scope heuristic is triggered. Example (132) illustrates such a case. NEGATOR identifies the LHS scope span correctly, but does not find any relevant heuristic linking the affixal negation trigger term with the *in my ways out West* part of the clause.

(132) (a) . . . and it may be that I have got a little careless in my ways out West .

(b)

affixal negation

Conan	***offset***	***scope***
cue	*12305-12309*	*less*
scope	*12281-12305*	*I have got a little care*
scope	*12310-12329*	*in my ways out West*
NEGATOR	***offset***	***scope***
selfNeg	*12305-12309*	*less*
selfNegScope	*12281-12305*	*I have got a little care*

C.2.1.3 Implicit Negation: mismatch in left scope span

The `partial` category: (N $\neq$ CD, N $\equiv$ CD) also contains `partial` errors for instances of *implicit* negation. The errors in this class occur with the determination of the RHS span where the LHS span is correct. The following two examples illustrate specific `partial` errors present in this class for implicit negation.

1: An example for an implicit negation `partial` error in this class occurs with the trigger *neglect*. The cause for error is similar to a case present with explicit negation where the gold standard will mark the LHS starting from the beginning of the subordinate clause and will mark the relevant constituents. In contrast NEGATOR takes a narrower approach and will only include the subject as the LHS scope span. This case is shown in Example (133):

(133) (a) It may have been that Barrymore had some private signal which we had neglected to give,. . .

(b)

implicit negation		
Conan	***offset***	***scope***
cue	*31708-31717*	*neglected*
scope	*31695-31703*	*which we*
scope	*31718-31725*	*to give*
NEGATOR	***offset***	***scope***
impNeg	*31708-31717*	*neglected*
impNegScope	*31701-31707*	*we had*
impNegScope	*31718-31725*	*to give*

2: Another example of an implicit negation `partial` error in this class occurs with the trigger *prevent* as illustrated in Example (134). The cause for this error is that NEGATOR determines the RHS negation scope to be the verb phrase whose main verb is *prevent* (without the trigger), in contrast the gold standard takes a wider approach and marks the RHS span all the way to the end of the sentence.

(134) *(a)* *. . . to prevent it from being visible , save in the direction of Baskerville Hall .*

(b)

implicit negation		
Conan	***offset***	***scope***
cue	*30454-30461*	*prevent*
scope	*30462-30527*	*it from being visible , save in the direction of Baskerville Hall*
NEGATOR	***offset***	***scope***
impNeg	*30454-30461*	*prevent*
impNegScope	*30462-30483*	*it from being visible*

C.2.1.4 Explicit Negation: mismatch in left scope span

The following five *explicit* negation examples demonstrates errors with the `partial` category: (N $\neq$ CD, N $\equiv$ CD) The errors in this class occur with the determination of the RHS span where the LHS span is correct.

1: As seen in Table 21, that in the Development Set the majority of `partial` instances for explicit negation results in the second class: N $\equiv$ CD, N $\neq$ CD. The fact that the Development set performs less well in determining correct RHS scopes contributes to the overall lower Recall in the Development set in comparison to the other data sets (see Table 20). A reason for this is that the sentences in the Development set are long, and contain multiple clauses. In contrast, the sentences in the other data sets tend to be simpler syntactically. Example (135), illustrates such a case where NEGATOR allocates a too wide scope on the RHS, because it annotates the entire verb phrase

whose main verb is *return*, whereas the gold standard will annotate only until the beginning of the subordinate clause (. . . it was probable. . .).

(135) (a) *. . . and if Garcia did not return by a certain hour it was probable that his own life had been sacrificed.*

explicit negation

Conan	***offset***	***scope***
cue	*15008-15011*	*not*
scope	*14997-15007*	*Garcia did*
scope	*15012-15036*	*return by a certain hour*

(b)

NEGATOR	***offset***	***scope***
expNeg	*15008-15011*	*not*
expNegScope	*14997-15007*	*Garcia did*
expNegScope	*15012-15090*	*return by a certain hour it was probable that his own life had been sacrificed.*

2: In contrast to the aforementioned error case, another cause for mismatch in the RHS scope spans is where NEGATOR allocates a narrower scope span than the gold standard. Example (136) illustrates such a case. Here, NEGATOR will annotate the parent noun phrase of *no* and *move* as the RHS scope span whereas the gold standard will annotate the entire verb phrase whose main verb is *make*.

(136) (a) *. . . and that he would lie low and make no move so long as he thought he was in any danger .*

explicit negation

Conan	***offset***	***scope***
cue	*25179-25181*	*no*
scope	*25153-25161*	*he would*
scope	*25174-25178*	*make*
scope	*25182-25229*	*move so long as he thought he was in any danger*

(b)

NEGATOR	***offset***	***scope***
expNeg	*25179-25181*	*no*
expNegScope	*25153-25161*	*he would*
expNegScope	*25174-25178*	*make*
expNegScope	*25182-25186*	*move*

3: A third related error case in this partial errors class occurs with the negation trigger *not*. It is well established by now that NEGATOR's scope heuristics rely on the output of the parse tree. Consequently, there are cases where NEGATOR will allocate a wider RHS negation scope span to a negation trigger then the gold standard, due to possible

errors in the manner in which the constituents are allocated by the parser. These errors often occur with coordinate clauses. As illustrated in Example(137), NEGATOR will determine the RHS negation scope span to be the verb phrase whose main verb is *is* (with the negation trigger removed). The resultant scope span is too wide due to the coordinate phrase being determined to be part of this verb phrase constituent.

(137) (a) I suppose the whole thing is not a vision and a touch of nerves ? ” .

explicit negation

Conan	***offset***	***scope***
cue	*2597-2600*	*not*
scope	*2578-2596*	*the whole thing is*
(b) scope	*2601-2609*	*a vision*
NEGATOR	***offset***	***scope***
expNeg	*2597-2600*	*not*
expNegScope	*2578-2596*	*the whole thing is*
expNegscope	*2601-2631*	*a vision and a touch of nerves*

4: There are a few sentences in the Conan Doyle corpus where the RHS gold scope span is composed of a few discontinuous spans. In these cases NEGATOR usually has at least one of the RHS discontinuous scope spans , but will miss the other ones as in Example (138):

(138) (a) I could not call you in , Mr. Holmes , without disclosing these facts to the world , and I have already given my reasons for not wishing to do so .

explicit negation

Conan	***offset***	***scope***
cue	*3344-3347*	*not*
scope	*3336-3343*	*I could*
scope	*3348-3359*	*call you in*
(b) scope	*3375-3418*	*without disclosing these facts to the world*
NEGATOR	***offset***	***scope***
expNeg	*3344-3347*	*not*
expNegScope	*3336-3343*	*I could*
expNegScope	*3348-3359*	*call you in*

5: A final cause for error in this partial category is there are few occurrences where there is no NEGATOR scope heuristic triggered for the RHS span. As seen in example (139), NEGATOR will identify the expletive constituent by the *expl* dependency relation between *nothing* and *There*, however NEGATOR does not have a heuristic triggered by the *rcmod* dependency relation with the governor being *nothing*. Consequently, there is no RHS scope span allocated by NEGATOR for this sentence.

(139) (a) *There is nothing upon which we can apply for a warrant .*

explicit negation

Conan	***offset***	***scope***
cue	*21667-21674*	*nothing*
scope	*21658-21666*	*There is*
(b) *scope*	*21675-21712*	*upon which we can apply for a warrant*
NEGATOR	***offset***	***scope***
expNeg	*21667-21674*	*nothing*
expNegScope	*21658-21666*	*There is*

The third partial category class shown in Table 21 contains partial matches where both the RHS and LHS NEGATOR scope spans are incorrect. The causes for these errors are in essence a combination of errors from the first and second partial error classes but occurring with one negation trigger instance. There are proportionally very few of this last partial error class for all data sets, and all negation types.

C.2.2 `omit` error classes

The `omit` category in Table 20 indicates the actual true misses. These errors are either due to a missing trigger or because NEGATOR allocates the correct trigger but fails to allocate a scope span.

C.2.2.1 Affixal Negation: missing scope spans

In both the Training and Development Sets, the majority of `omit` errors occur with instances of *affixal* negation. In the Training set, eleven out of seventeen of these instances is because NEGATOR does not identify the trigger (is not within the *SelfNeg* word list). The remaining six errors are due to NEGATOR not identifying a relevant scope heuristic for the negation trigger. Example (140) illustrates such a case where the gold standard finds three discontinuous scope spans for the affixal trigger. However, NEGATOR does not find any relevant scope spans (except for the default narrow scope for *selfNeg* triggers).

(140) (a) *“He was a strong-minded man , sir , shrewd , practical , and as unimaginative as I am myself .”*

(b)

affixal negation

Conan	***offset***	***scope***
cue	*978-980*	*un*
scope	*915-921*	*He was*
scope	*975-977*	*as*
scope	*981-1006*	*imaginative as I am myself*
NEGATOR	***offset***	***scope***
selfNeg	*978-980*	*un*
selfNegScope	*981-991*	*imaginative*

C.2.2.2 Explicit Negation: missing scope spans

The `omit` errors for the *explicit* negation category in the Training and Test sets are because NEGATOR identifies the triggers correctly but does not find a relevant heuristic to allocate a scope span. The following two examples demonstrate two such errors.

1: As illustrated in Example (141), NEGATOR identifies the explicit negation trigger *rather than* correctly, however fails to annotate a scope span. NEGATOR does have a heuristic for this trigger: the *prepc_than* dependency relation should be created. However in this sentence, the Stanford Dependency module fails to identify this dependency relation and in turn NEGATOR does not trigger the appropriate heuristic.

(141) (a) *"...and why was he waiting for him in the yew alley rather than in his own house ? "*

(b)

explicit negation

Conan	***offset***	***scope***
cue	*16077-16088*	*rather than*
scope	*16041-16059*	*he was waiting for him*
scope	*16100-16105*	*in his own house*
NEGATOR	***offset***	***scope***
expNeg	*16077-16088*	*rather than*
expNegScope	-	-

2: The negation trigger *nothing* is the culprit for the `omit` errors in the Test set. Example (142) illustrates one case where NEGATOR expects the dependency relation *dobj(know,nothing)*, however the Stanford Dependency module creates the dependency relation *iobj(know, nothing)* - nothing is considered to be the indirect object and not the direct object. Consequently, NEGATOR does not trigger any scope heuristics as none of the correct conditions are met.

(142) (a) *"What is the use of asking me questions when I tell you I know nothing whatever about it ?"*

(b)

explicit negation

Conan	***offset***	***scope***
cue	*11457-11464*	*nothing*
scope	*11450-11456*	*I know*
scope	*11465-11473*	*whatever about it*
NEGATOR	***offset***	***scope***
expNeg	*11457-11464*	*nothing*
expNegScope	-	-

The implicit negation *omit* error in both the training and development set is due to the missing trigger *save* (not part of the *implicitNeg* trigger list). Even though this type of error does exist (where the Stanford Dependency module does not create a dependency relation that NEGATOR should potentially identify), it is observed that the cases are extremely few. As was the case with BioScope, NEGATOR's heuristics based on the relevant dependency graphs are not only stable but also perform well across different text genres.