8 AN AUTOMATIC IDENTIFICATION OF NEGATION IN DESIGN DOCUMENTS

Sanghee Kim[a] **and Ken Wallace**[b]

Engineering Design Centre, University of Cambridge, U.K. Tel: +44-(0)1223-748245.
E-mail: [a]*shk32@eng.cam.ac.uk,* [b]*kmw@eng.cam.ac.uk*

Designers use negation to describe: (1) design problems; (2) contradictory findings; and (3) unlikely root causes. An example negation is *no signs of engine failures due to fuel exhaustion.* Designers should be able to search for negated information in the same way as searching for confirmed information. For example, the following query is useful for designers carrying out a root cause analysis *Find engine failures that were not caused by fuel exhaustion.* Current search engines provide only limited support for negation finding. Using a search symbol such as '−' or 'not', users can direct what keywords should not be parts of the search. Whereas such symbols are helpful to filter out irrelevant documents, it is difficult to retrieve satisfactory answers to some queries. One reason is that current systems interpret such symbols mathematically rather than linguistically. An automatic method for identifying negation and converting it into an easily retrieved format is therefore needed. This research presents the results of developing a simple, yet powerful, automatic method. Its efficiency is attributed to the use of recently available advanced linguistic analysis software. A test with 1511 reports from an aerospace company demonstrates 88% precision and 86% recall.

Keywords: Negation Expressions in Engineering, Problem Reports, Natural Language Processing.

1. INTRODUCTION

Narrative texts are commonly used as a means of communicating, informing and sharing information. Many organizations maintain a large number of electronic texts and extracting useful information from them is the goal of every organization that would like to take advantage of the experience encapsulated in those texts. A keyword-based search is commonly used for computer-supported retrieval systems. It is proven to be efficient and useful when retrieving a list of documents whose contents are assumed to consist of similar terms to the keywords. However, keyword searching provides only limited support when locating and extracting a specific piece of information rather than retrieving a whole document. Extracting specific information from documents requires the comparison of information at a fine-grained level for which string-based indexing is not suitable.

Negated information is one example for which current keyword-based searches cannot always retrieve satisfactory answers. Users can select which keywords should not be the parts of the answers by using a '−' or 'not' search operator. Such operators are useful to filter out irrelevant documents when keywords have multiple meanings across different domains. For example, using Google, the query of *Rembrandt* brings documents relating to: (1) the Dutch artist; and (2) hotels named after him. However, if the query is re-submitted as *Rembrandt –hotel*, the Web pages containing the 'hotel' keyword, are removed. However, current systems interpret such operators mathematically rather than linguistically, and thus some queries are not answered correctly.[1] Assume that a designer is looking for *engine failures due to fuel exhaustion*. Using a keyword search, all four sentences (a), (b), (c) and (d) below will be retrieved. Whereas (a) is a correct answer to the query, (b), (c) and (d) are negative and incorrect answers. Retrieving information by only looking up the existence of the keywords can therefore be misleading.

(a) Evidence indicated that the engines had stopped because of fuel exhaustion.
(b) There was no evidence to indicate the engines had stopped because of fuel exhaustion.
(c) The investigation denied any speculations that engines quit because of fuel exhaustion.
(d) They did not consider that engines quit because of fuel exhaustion.

In addition, users cannot search explicitly ruled-out or double-negated information. Such information is important. For example, when looking up failure modes for a specific design problem, a designer might need to filter out some possible sources of failures. By explicitly specifying the sources that could be excluded, the designer can quickly and correctly identify the right ones.

Negated information in this research refers to information describing: (1) problems, e.g. *The engine does not function*; (2) contradiction, e.g. *Engine functions but is not easy to use*; and (3) unlikely root causes, e.g. *There was no evidence to indicate the engines had stopped because of fuel exhaustion.*

This research presents the results of developing an automatic method that identifies sentences containing negation and then extracts negation phrases from those sentences. A challenge is to identify correctly only negated information, i.e. negation scope. For example, in sentence (b) above, the negation signal *no* applies to *evidence* but not to *fuel exhaustion*. In particular, if the texts were written and shared by individuals, different expressions can be used to indicate similar negations. However, according to studies,[2–4] domain specific texts, e.g. patients healthcare records or engineering problem reports, are less ambiguous, less stylistic and more restricted compared to open-domain texts. Domain-specific texts also show recurrent expression patterns and common vocabularies leading to the need for only a few extraction rules.

2. RELATED WORKS

Recently, a growing interest has been shown in retrieving information at a fine-grained level for advanced information processing. This includes sentence-basis retrieval that aims to locate and extract sentences as answers rather than documents for a given query. Sentence classification is the automatic classification of sentences into pre-defined sentence types.[5] Example applications are semantic orientations and negation identification. Semantic orientation looks for the evaluative character of a word in order to extract opinions, feelings, and attitudes expressed in a text.[6] The orientation is classified as positive if it contains praise or recommendation. Negative orientation indicates criticism or non-recommendation. A combination of cue phrases, e.g. *excellent* or *low fees*, and linguistic features is commonly used. Those cue phrases can be created either manually or using machine learning techniques. On average, the accuracy is observed to be around 70%.

In medical domains, a main focus is to identify negated medical concepts such as diseases or infections. In doing so, a number of medical terminology collections are often used.[3] Example collections are Systematized Nomenclature of Medicine Clinical Terms (SNOMED) and Unified Medical Language System (UMLS). Two example systems are NegEx[2] and Elkin *et al.*[7] NegEx[2] uses a small number of pre-defined negation signals and limits the scope of the negation phrases. The signals are the words that indicate the existence of negation. It requires a sentence to be indexed with medical concepts in advance and detects negations only in the indexed medical concepts. A total of 35 negation signals were manually developed. It has two extraction patterns: (1) ⟨negation signal⟩ * ⟨medical concepts⟩; and (2) ⟨medical concepts ⟩ * ⟨negation signal⟩. The asterisk allows a maximum five words positioned between the negation signal and the concepts. Elkin *et al.*[7] compared the automatically identified negations with the negations assigned by a human expert. They developed a negation ontology consisting of negation terms, their variants and associated identification rules. 41 clinical documents were used for the comparison. A test showed 97% accuracy, i.e. 1662 negations out of 1710 were agreed by the expert.

Both methods have two limitations: (1) it is difficult to identify negated concepts if they are more than a few words away from the negation signal; and (2) only concepts matched with the medical terminologies can be identified. Huang and Lowe[8] addressed these limitations by developing a system that extended the identification rules with syntactical and lexical constraints. They excluded 'partial negations', e.g. *probably not*, and negations within a word as in the case of negative prefix or suffix. They also focused on identifying biomedical noun phrases instead of pre-defined medical concepts. The rules were manually created based on 30 radiology reports and validated with 470 reports. The method achieved 92.6% sensitivity (recall) with 98.6% precision.

There exist different approaches for rule discovery in Information Extraction (IE).[9] Wrapper induction is used to process online Web pages by identifying boundaries and is based on delimiter-based rules. On the other hand, for free texts, extraction rules are based on the structure of the texts and syntactic and/or semantic constraints. Consequently, in order to apply the extraction rules, the input texts have to be parsed syntactically and semantically. One of the challenges in doing this is to reduce the cost of developing extraction rules for new tasks by automating some or all the processes of rule discovery. With advances in Natural Language Processing (NLP), it is known that such automated processes can be developed with less human effort and resources.

3. DEVELOPMENT

The main objective of this research is to develop a simple method that automatically identifies negation in design reports. The method uses the recently developed Stanford NLP parser that generates typed dependency grammars, i.e. grammatical relations between words.[10] Example relations are *nsubj* that defines a subject relation or *aux* defining a non-main verb. Such a dependency parse makes it easier to identify a predicate-argument structure that is useful for advanced indexing and retrieval. That is, these relations are useful to classify a given sentence into a negation type and to structure that sentence into easily accessible formats. The Stanford NLP parser is a statistical parser that tries to produce the most likely analysis of a new sentence by measuring its similarity to existing grammars.

3.1. Negation in Design Reports

The analysis of a small set of design reports was carried out in order to understand how negation was expressed and what semantics were conveyed. These reports were obtained from an engineering company and were short descriptions of the problems faced in the later stages of design. A specific database was used to store and retrieve the reports. The main users of these reports are the project managers who make decisions on which problems are important to solve, as well as the designers and team leaders who are asked to solve the problems. In addition, individual engineers might search the reports to see if similar problems have arisen before and, if so, what solutions were proposed. The first author examined the reports manually, highlighted sentences containing negations, and grouped them according to negation signals. Table 1 summarizes the analysis. The negation signals, e.g. *not*, are shown in bold.

As this research aims to allow extracted negation to be searched directly by designers, it is necessary to structure the extracted negation into easily accessible and retrieval formats. In doing so, a negated sentence is indexed with three arguments: (1) negation signal; (2) negated terms; and (3) optional features. The negation signal indicates the existence of negation. The negated terms are negated concepts. Optional features are used to locate the negated terms and to help validate the identified negations. For example, *no further increase* is differentiated from *no strong increase* since two optional features *further increase* and *strong increase* have different lexical meanings. Multiple negations in one sentence are processed multiple times. Negations are limited to within a single sentence, i.e. no negations crossing sentence boundaries are identified.

Sentence (f) in Table 1 is an example of double-negation and (g) is a false negation that should be excluded by an automatic identification method. That is, *without any further* does not indicate negation. The example texts were extracted from the Australian Transport Safety Bureau (ATSB) database.[11]

3.2. The Proposed Method

The proposed method takes design reports as inputs, indexes them with pre-defined negation signals and determines whether or not the phrases containing those signals are negated. In doing so, it needs

Table 1. Examples of negation and the proposed structures for extraction.

Example negation	Extraction Structure
(a) They did **not** consider that engines quit because of fuel exhaustion.	Negated terms: that engines quit Optional features: did consider
(b) There was no **evidence** to indicate the engines had stopped because of fuel exhaustion.	Negated terms: the engines had stopped Optional features: evidence to indicate
(c) The system **lacks** effective and timely detection of the cabin altitude alert probably leading to a more serious occurrence.	Negated terms: effective and timely detection of the cabin altitude alert Optional features: The system
(d) The pilot was **unable** to conduct an examination of the fuel selector system fitted to NMQ before the aircraft was returned to service.	Negated terms: the pilot Optional features: was to conduct an examination of the fuel selector system
(e) **No** injuries were reported as a result of the accident.	Negated terms: injuries Optional features: were reported
(f) The fuel tanks were **impossible** to fit on the aircraft model 680 **without** modification.	Double negation
(g) Once the pilots had corrected the error, the subsequent approach was conducted **without any further** incident.	False negation

rules for identifying negation signals and their negation scopes. The rules were created from examples that are selected by the first author manually. The examples are then analysed by the Stanford NLP parser and this helps derive efficient extraction rules by generalizing common patterns among the examples. The following steps describe the method in more details:

Step 1: Corpus preparation

Design reports were originally obtained as a single Microsoft Excel file. A Perl script was used to identify individual reports from the file and to store them as plain text files. A Perl library developed by Lingua Project was used to read the text files and to split each report into sentences.[12]

Step 2: Manual annotation

This step examined the extracted sentences and annotated negations manually. A graphical user interface using the Java programming language was developed to help the annotation, which included the identification of sentences containing negations and the extraction of three arguments from the identified sentences.

Step 3: Pattern creation

Each annotated example was analysed using the Stanford Parser. A total of 48 dependency grammar relations are currently defined in the Stanford Parser. The parser offers various output formats: at the simplest level, a Part-Of-Speech (POS) Tag is used and, at more complex level, a collapsed dependency parse is used. In this research, two outputs are generated for each sentence: (1) POS tag; and (2) a dependency grammar relation. The following shows the parsing results for the sentence *They did not consider that engines quit because of fuel exhaustion*.

POS: They/PRP did/VBD not/RB consider/VB that/IN engines/NNS quit/VBD because/RB of/IN fuel/NN exhaustion/NN./.

Dependency grammar relation: nsubj(consider-4, They-1)
aux(consider-4, did-2)
neg(consider-4, not-3)
complm(quit-7, that-5)
nsubj(quit-7, engines-6)
ccomp(consider-4, quit-7)
nn(exhaustion-11, fuel-10)
prep_because_of(quit-7, exhaustion-11)

The Stanford Parser first identifies a set of tokens in each sentence. Tokens are words lying between two spaces including a full stop. A total number of 11 tokens are identified in the example above. A POS tag identifies not what a word is, but how it is used. It is useful to extract the meanings of words since the same word can be used as a verb or a noun in a single sentence or in different sentences. For example, *They/PRP* indicates that *They* is classified as a personal pronoun, i.e. *PRP*. Each POS-tagged keyword is compared with WordNet[13] definitions to achieve term normalization, e.g. revealed → reveal.

Dependency grammar relations provide direct relations between words. For example, the *nsubj* relation captures a nominal subject which is a noun phrase, e.g. *nsubj(consider-4, They-1)* in *They did not consider*. The *neg* is the relation between a negation word and the word it modifies, e.g., *neg(consider-4, not-3)*, in *did not consider*. In most cases, the modified word, e.g. *consider*, is a verb and it is classified by optional features. In some cases, the word is either a noun or adjective, e.g. *neg(available-6, not-3)* in *not available*.

A total of 117 negations were annotated from 322 reports which had 1159 sentences. The total numbers for each negation signal type were: not adverb = 75, no determiner = 26, impossible adjective = 2, without preposition = 7, and fail verb = 7. The most common expression is a *not adverb* and this accounts for 64% of the total number of annotated examples. The expressions identified by *not adverb*, *no determiner*, or *without preposition* account for 92% of the total. This implies that high identification accuracy can be achieved if the method could efficiently create extraction patterns for those three signals. This confirms the study by Mutalik *el al.*[14] that the words *no, not, denied* and *without* made up 92.5% of negation.

Extraction rules were created by analysing annotated negation examples. The analysis revealed that negations had relatively fixed linguistic patterns. These recurrent patterns made it easy to create negation extraction rules. The annotated examples were classified based on the POS tag types of modified words. For example, in the following sentence, *They did not consider that engines quit because of fuel exhaustion*, the negation signal was identified by the *neg* grammar relation, i.e. *neg(consider-4, not-3)* and the modified word, i.e. *consider* is POS tagged as *verb*. The reason of using such classifications was to reduce the number of rules by identifying common grammar patterns among the examples, even though different signals were used.

Four POS tag types, i.e. *verb, noun, preposition*, and *adjective*, were identified. For the *verb* modified words, initially a total of 13 patterns were identified and by taking random pairs of two patterns, generalizing each pair, and then selecting the best generalization as the new pattern, a total of 6 extraction rules were created. The following shows one rule:

Rule One: [{aux|auxpass|advmod}][neg-not]{aux}[{noun phrase|clause}]
negation signal: not adverb
negation terms: noun phrase or clause
optional features: verb phrase

This rule applies to the negation identified by the *not adverb* signal, and extracts the noun phrase or clause as a negation term and the modified verb phrase as an optional feature. It has four conditions:

(1) the preceding dependency relation should be either *aux, auxpass* or *advmod*. *aux* defines *a non-main verb of the clause, auxpass* defines *a non-main verb of the clause which contains the passive information*, and *advmod* defines *RB* or *ADVP that modifies the meaning of the word*;
(2) the negation signal should be *not neg* relation; and/or
(3) the next relation should be *aux*, and
(4) the next relation should be either noun phrase or clause

For example, the following sentence *They did not consider that engines quit because of fuel exhaustion* is matched with this rule. It has the smallest number of matching conditions but covers the largest number of examples. A total of 12 extraction rules were derived from the dataset.

4. TEST

This test only presents the results of identifying negations using the created extraction rules. The results of structuring negations according to the three arguments, i.e. negation signal, negation terms, and optional features, are excluded.

A total of 1511 reports were used to test the proposed method. These reports were written for different product types, but the contents were similar to the training examples in Section 3. From these 8388 sentences and 775 negations were identified using the method. New negation signals, e.g. lack, were added. These were identified from the literature. The total numbers for each negation signal type were: not adverb = 449, no determiner = 138, without preposition = 62, impossible adjective = 8, unable adjective = 16, failed verb = 78, and lack noun = 24. The number of negations identified using *not adverb, no determiner*, and *without preposition* accounts for 84% of the total. Focussing on the first two signals, Figure 1 shows the distribution of modified words estimated from the number of examples matched. The most commonly used word was *possible*. Using *not adverb*, a total of 34 unique words was matched with over half of the examples. A total of 211 unique words were identified and 130 words occurred only once. As such, Figure 1 shows a very long tail, i.e. there are a few very common words and a large number of less frequent ones. This means that by focusing on the common words, it is relatively easy and quick to cover over 50% of the examples, but it is a challenge to cover more than that since it requires handling increasingly rare ones.

Precision and recall were used to measure the performance of the method. Recall is defined as the percentage of the test negations that were classified as negation, whether these were correct or not. Precision is defined as the percentage of the recalled negations that were correctly matched. Table 2 shows the test results. A total of 803 negations were identified from the manual analysis, i.e. Actual num. For example, using the *adverb* signal, 92% recall and 94% precision were observed. On average,

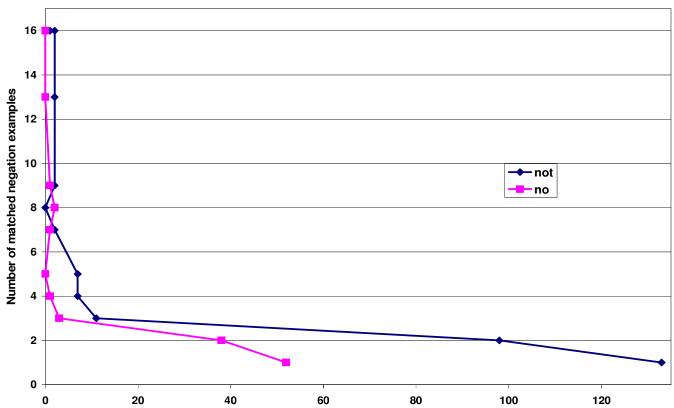

Figure 1. The distribution of modified words according to the number of matched examples.

Table 2. Test results.

Type of negation signal	Actual number	Extracted number	Incorrect number	Missed number	Recall	Precision
Adverb (e.g. not)	460	449	25	36	92%	94%
Determiner (e.g. no)	143	138	6	11	92%	96%
Noun (e.g. lack)	24	24	3	3	88%	88%
Adjective (e.g. impossible)	27	24	4	7	74%	83%
Preposition (e.g. without)	51	62	12	1	98%	81%
Verb (e.g. refused)	98	78	10	30	69%	87%
Total	**803**	**775**	**60**	**88**	**86%**	**88%**

Table 3. Error analysis.

Error types	Num
Ill-formed sentences	43
Incorrect identification of sentence boundaries	25
Incorrect parse results	15
Incomplete negation signals	35
Double negation	15
False negation	15
Total	**148**

86% recall and 88% precision were observed. These were macro-averaged, i.e. calculated for each negation signal and then averaged. In particular, the extraction rules for *adjective* and *verb* signals need improvement.

The proposed method was able to identify correct negation scopes using syntactic parsing results and grammar relations. For example, in the sentence of *There is currently no holding function for Controller-Pilot Data Link Communications (CPDLC)*, *no* negation signal applies only to *holding function* but not *CPDLC*. The lowest precision (81%) was observed with *without preposition*. Whereas it was easy to identify negations using the signal, there exist a large number of double negations that should have been excluded. Table 3 shows details of errors made by the method.

In order to check if any potential negation was not identified by the method, the remaining unmatched sentences were examined manually. A total of 30 negations were not identified since the rules did not cover negation signals such as *refused* or *unlikely*. A large number of sentences were grammatically incorrect. Some sentences were incomplete and contained misspellings. These ill-formed sentences were the most important error sources. It was also difficult to identify double negation since the rules did not have lists of antonyms for the adjective signal, e.g. *common* versus *uncommon*. Hence, the double negations such as *not uncommon* and *no design deleted* were not correctly interpreted.

Although the Stanford Parser is a highly optimized tool, some parsing errors were made. One reason is the rare and unpredictable behaviour of certain words in engineering texts that are new to the parser. For example, *pipes* is mostly used as a plural of a *pipe* noun word in engineering, but the parser often classifies *pipes* as a present-tense third person singular verb. The extraction rules also did not cover false negations. For example, the sentence *No other change is required for the design of the Sector 3 console* was classified as negation, since the rules did not include *no other change* as a possible false negation signal. A total of 148 errors, i.e. 17%, were made by the method developed by this research.

5. CONCLUSIONS AND FUTURE WORK

This research has presented a simple yet effective method for identifying and extracting negation from design documents. Extraction rules were based on a small number of training examples. It took only a few days to develop the method and to test it with new examples. Its efficiency is attributed to the use of advanced linguistic analysis software which has recently become available. A total of 12 extraction rules were created. A test showed over 86% recall and 88% precision. One of the shortcomings of the method is the possibility of its limited coverage when applying it to texts in new domains. In particular, if the new reports have different vocabularies and styles from the training examples, it is likely that the rules will not cover a number of new examples. Further testing is planned. In addition, the method needs evaluating for its efficiency in extracting the three arguments from the negations and using them to answer designers' queries.

One challenge of using an automatic extraction of specific types of information at a fine-grained level is that the interpretation of extracted information can differ among individuals. For example, when attempting to extract automatically sentences that contain 'causes and effects', there is only around 65% agreement between experts as to whether or not the identified sentences represent causality. The results for extracting negated information look more encouraging with agreement promising to be in excess of 80%.

ACKNOWLEDGMENTS

This work was funded by the University Technology Partnership for Design, which is a collaboration between Rolls-Royce and the Universities of Cambridge, Sheffield and Southampton.

REFERENCES

[1] Gavriel, M. (2005). Capturing Negation in Question Answering Systems. MSc Thesis, The University of Edinburgh.
[2] Chapman, W., Bridewell, W., Hanbury, P., Cooper G. F. and Buchanan, B. G. (2001). A Simple Algorithm for Identifying Negated Findings and Diseases in Discharge Summaries. Journal of Biomedical Informatics, 34:301-310.
[3] Gindle, S. (2006). Negation Detection in Automated Medical Application — A Survey. Technical Report, TU Vienna, Asgaard-TR-2006-1.
[4] Grishman, R. (1999). Adaptive information extraction and sublanguage analysis. Proceedings of the IJCAI Conference.
[5] Hackey, B. and Grover, C. (2005). Sequence modeling for sentence classification in a legal Summarisation System. *In ACM symposium on Applied computing*, U.S.A. 292–296.
[6] Turney, P. D. (2002). Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews, Proceedings of the 40th Association of the Computational Linguistics (ACL), U.S.A., 417–424.
[7] Elkin, P. L., Brown, S. H., Bauer, B. A., Husser, C. S., Carruth, W., Bergstrom, L. R. and Wahner-Roedler, D. L. (2005). A controlled trial of automated classification of negation from clinical notes. BMC Medical Informatics and Decision Making, 5:13.
[8] Huang, Y. and Lowe, H. J. (2007). A Novel Hybrid Approach to Automated Negation Detection in Clinical Radiology Reports. J. Am Med Inform Assoc., 14:304-311.
[9] Muslea, I. (1999). Extraction Patterns for Information Extraction Tasks: A Survey. Proceedings of the AAAI 99 Workshop on Machine Learning for Information Extraction.
[10] Marneffe, M. C., MacCartney, B. and Manning, C. D. (2006). Generating Typed Dependency Parses from Phrase Structure Parses, 5th International Conference on Language Resources and Evaluation (LREC 2006).
[11] ATSB, Australian Transport Safety Bureau, http://www.atsb.gov.au
[12] Sleator, D. and Temperley (1991). Parsing English with a Link Grammar, Carnegie Mellon University Computer Science, technical report.
[13] Miller, G. A., Beckwith, R. W., Fellbaum, C., Gross, D. and Miller, K. (1993). Introduction to wordnet: An on-line lexical database. In Int. Journal of Lexicography, 3(4):235–312.
[14] Mutalik, P. G., Deshpande, A. and Nadkarni, P. M. (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.
[15] Hobbs, J. R., Information Extraction from Biomedical Text. Journal of Biomedical Informatics, 35(4):260–264.