UCM at TREC-2012: Does negation influence the retrieval of medical reports?

Alberto Díaz, Miguel Ballesteros
Universidad Complutense de Madrid
Spain
{albertodiaz,miballes}@fdi.ucm.es

Jorge Carrillo-de-Albornoz
Universidad Nacional de Educación a Distancia
Spain
jcalbornoz@lsi.uned.es

Laura Plaza
Universidad Autónoma de Madrid
Spain
laura.plaza@uam.es

Abstract

This paper details the UCM participation in the TREC 2012 Medical Records Track. We present several experiments directed to evaluate the effect of detecting negation in the task of retrieving medical reports. In particular, two different algorithms based on syntactic analysis have been applied to detect negations and to infer their scope. These algorithms are then combined with a simple term-frequency approach using Lucene to retrieve the reports that are relevant to a given query. We evaluate whether ignoring the information that is within the scope of negation may result in a higher retrieving performance. However, our experiments reveal that the effect of negation in this task is not significant.

1 Introduction

The goal of the TREC 2012 Medical Records Track is to foster research on providing content-based access to the free-text fields of electronic medical reports. In particular, the task is to retrieve reports from a test collection that are relevant to a given topic. This topic or query consists in a set of words, and specifies a particular disease and/or a particular treatment or intervention. Most reports are associated with a “visit” identifier, visit being seen as an episode of care. Participants systems have to return a list of visits ranked by decreasing relevance, among a collection of more than 100.000 reports.

Our main interest in the present work was to study the effect of negation in the retrieval of medical reports. To this end, we have adapted a simple frequency-term based approach using Lucene, which was initially designed for retrieving medical reports in Spanish, to handle negations and their scopes.

Negation is a complex but essential phenomenon in any language. It turns an affirmative statement into a negative one, thus changing its meaning. In medical reports, negations typically indicate the absence of sings and symptoms, and the negative results of procedures and tests. Therefore, for instance, given the query *Patient with chest pain and fever*, records reporting on patients with *chest pain but no fever* should not be given as much relevance as those reporting on patients who, in addition to chest pain, also presented fever. However, typical approaches not considering the negations occurring within the reports will not be aware of this fact. Therefore, we expect that, by detecting and delimiting the scope of negations, the accuracy of the retrieval task will improve.

We can find several systems that handle the scope of negation in the state of the art. This is a complex problem, because it requires, first, to find the negation cues or signals, and second, to identify the words that are directly (or indirectly) affected by these negation cues. One of the main works that started this trend in natural language processing was published by Morante’s team (2008; 2009), who presented a machine learning approach for detecting negations in biomedical texts which was evaluated on the Bioscope corpus. Other works have dealt with negation in the context of medical reports. Amini et al (2011) used NegEx[1] to identify and remove the negated terms from the queries in a medical reports

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

retrieval task, but found that incorporating information from negations reports marginal improvements. Karimi et al (2011) also used NegEx to treat negations, but apply it over the reports instead of the queries. They create a list of negated terms that are found in the entire collection and replace all negated terms with a single word with a *no* prefix: e.g., if negation is implied for "chronic back pain", all instances of chronic back pain and its variants are replaced with the word "nochronicbackpain". However, they did not evaluate the effect of processing negation over the retrieval results. Similarly, in Limsopatham et al (2011) NegEx is used to detect negation at the sentence level in medical reports, and the terms that have a negative context are prefixed with a special character.

In this paper we present and compare two complex approaches to the treatment of negation in medical reports. Unlike NegEx, which is based on regular expressions, our negation handling systems use a combination of syntactic information and rules to better approximate the scope of negation. The first system makes use of an algorithm that traverses dependency structures and classifies the scope of the negations by using a set of rules that studies linguistic clause boundaries and the outcomes of the algorithm for traversing dependency structures. The second system uses the information from the syntax tree of the sentence in which the negation arises to get a first approximation to the negation scope, which is later refined using a set of post-processing rules that bound or expand such scope. Both systems are used to detect negations in the reports and to remove the text that is found within their scope.

The results obtained show, however, that the effect of negation in this task is not significant, and we think the reasons are (1) the fact that only a few number of terms in the reports are affected by negations and (2) most of these negated terms do not appear in the queries.

The paper is organized as follows. In Section 2 we present the two algorithms that we propose for inferring the scope of negation. In section 3 we describe the different indexing and retrieval approaches. In Section 4 we discuss the evaluation results. Finally, in Section 5 we draw conclusions and suggest future work.

2 Inferring the Scope of Negation

The next subsections describe the two systems for inferring the scope of negations.

2.1 Rule-Based System based on Dependency Structures

The first system (`system1`) consists of two main independent processes:

1. An algorithm that extracts the words that are affected by negation cues (or negative operators), such as *not* or *no*.

2. An algorithm that infers the whole scope of negations by using the output of the first one.

This system was developed in two steps. The very first version (Ballesteros et al., 2012b) was implemented with the intention of annotating negation scopes in the Bioscope corpus (Vincze et al., 2008). The second one (Ballesteros et al., 2012a) was developed in order to participate in the *SEM Shared Task 2012 (Morante and Blanco, 2012) and its aim was to infer the scope of negations in a collection of Conan Doyle stories (Morante and Daelemans, 2012). This second implementation is the one that we use for the present work.

In order to detect negations, the system includes a static lexicon of negation cues, which is used through the process. We show an excerpt in Table 1. The whole static lexicon contains 152 different negation cues. It is possible to find a discussion about the cues considered in the study presented about the Conan Doyle corpus by Morante and Daelemans (2012).

not	no	neither..nor
none	nowhere	n't
rather than	cannot	nowhere
nothing	windless	without

Table 1: Excerpt of the lexicon

The first algorithm uses the output of a dependency parser, Minipar (Lin, 1998). The algorithm searches within the dependency structures the negation cues included in the lexicon, and then it applies some rules in order to extract the words that are affected by the cue detected (or cues, if there is more than one). The system iterates through all the nodes included in the dependency structures and returns a list of words affected by each negation cue. The rules are the following:

- If the node is a negation cue, and it is a verb, such as *cannot*, it is marked as a negative operator.
- If the node is a negation cue and it is not a verb, the algorithm marks the verb that is directly related with the negation cue as a negative operator.
- If the node is not a negation cue but it depends directly on any of the nodes that has previously marked as negative operator, the system includes it in the list of nodes affected by negative operators.
- Otherwise, the system just starts processing the next node.

The second algorithm is the one that annotates the scope of negations by using the output of the first one. It operates as follows:

- The system opens a new scope when it finds a negation cue detected by the affected wordforms detection algorithm. The system goes backward and opens the scope when it finds the subject involved.
- The system closes a scope when there are no more wordforms to be added in the scope that is being processed:
  - There are no more words in the output of the first algorithm.
  - It finds words that indicate another statement, such as *but* or *because*.
  - End of sentence.

2.2 Rule-Based System based on Constituency Parsing

The second system (`system2`) uses the information from a phrase structure syntax tree of the sentence in which the negation arises to give a first approximation to the negation scope, which is later refined using a set of post-processing rules that bound or expand such scope. A brief description of the method is given next (more details may be found in (Carrillo-de-Albornoz et al., 2012)).

The system starts by detecting the negation cues. To this end, we use a list of predefined negation cues. The list has been extracted from different previous works and may be found in (Carrillo-de-Albornoz et al., 2012). It includes common spelling errors such as the omission of apostrophes. We also deal with false negations, such as *not just*, *not only* or *not to mention*.

The system next delimits the scope of negation. To this aim, for each sentence where a negation cue has been detected, we generate the syntax tree using the Stanford Parser.[2] We next find in this tree the first common ancestor that encloses the negation token and the word immediately after it, and assume all descendant leaf nodes to the right of the negation token to be affected by it. This process may be seen in Figure 1, where the syntax tree for the sentence: *The patient* ***denies*** *any acute pain* is shown.

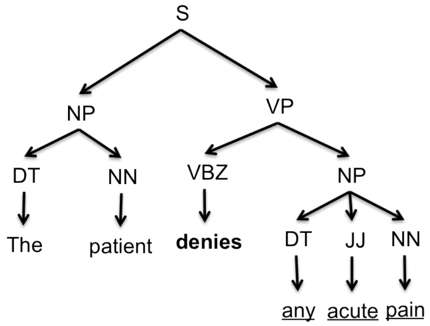

Figure 1: Syntax tree of the sentence: *The patient denies any acute pain*. The negation cue is shown in bold. The negation scope is underlined.

This general processing is, however, improved with three rules that expand or restrict the scope of negation, as necessary. The first rule expands the negation scope in order to include the subject of the sentence within it. In this way, for instance, the subject "the patient" in the sentence *The patient denies any acute pain* is included within the scope of *denies*. The second rule concerns some types of subordinate sentences and is used to restrict the scope to the main clause. Finally, the third rule expands the negation scope in order to include prepositional

[2] http://nlp.stanford.edu/software/lex-parser.shtml

phrases after the negated event (see (Carrillo-de-Albornoz et al., 2012) for more details about these rules).

3 Indexing and Retrieval Approaches

We use the Lucene framework[3] to construct different types of indexes and searches. The indexes differ in whether they are based on reports or visits and in whether they take negations into account or not. The searches differ in the index used and in the way that the report scores are combined to obtain a final score per visit.

3.1 Preprocessing

We first parse the medical reports and extract from the XML the different sections in the documents. In particular, we extract the *checksum* tag that identifies the visit, and the *report_text* tag that contains the core text of the medical report. A filter is also applied to eliminated the text associated with de-identification information, such as NAMES or DATES. For some experiments, documents without visit id (8023 reports) were discarded. Besides, 17.195 new documents have been generated, each one representing a different visit, that contain the text in the *report_text* tags of all reports associated to the visit.

3.2 Processing Negation

The two algorithms described in Section 2 are applied to all the reports in the collection in order to detect negations and infer their scope. More precisely, for each report and each algorithm a new document is created where all the terms within a negation scope have been removed. For instance, if a document presents the sentence *Patient presents respiratory distress but he does not refer chest pain*, then we only include in the new document the portion of the sentence that is affirmed (i.e., *Patient presents respiratory distress*).

Table 2 shows the negation cues detected by `system1` (see Section 2.1) that occur more than 500 times in the collection. It may be seen that, as expected, the most frequent negation cue is *no*, which appears more than 325000 times in the collection, followed by *not* and *without*.

Table 3, in turn, shows some statistics on the number of negated terms that are found in the collection by this algorithm.

Cue	#	Cue	#
no	325292	nor	1049
not	131921	independent	1045
without	39355	nothing	911
unable	6523	uncertain	896
never	2824	none	871
irregular	1759	prevent	784
refused	1231	unlikely	767
unknown	1121	unusual	665

Table 2: Number of appearances (#) for negation cues that occur 500 times or more in the collection.

Reports with negations	83249
Negated terms	3845651
Negated terms per report	46,19

Table 3: Some statistics on the number of negated terms in the collection.

Concerning the efficiency of the algorithms, the time needed by `system1` to infer the negation scopes is approximately 100 hours, while `system2` takes around 250 hours to perform the same task.

We have also used NegEx in a similar way to remove all the terms detected within a negation scope. The time needed by NegEx is approximately 3.5 hours.

3.3 Indexing

Different types of indexes have been created depending on if they are based on reports or visits, and if they take negations into account or not.

Regarding the indexing unit, we generate two types of indexes: *report-based* and *visit-based*. The first type uses the report as the indexing unit, while the second one uses the visit as such unit. For the first category of indexes, we create a Lucene document for each report in the collection, using the *path*, the *visit_id* and the *report_text* as fields. For the second type of indexes, a Lucene document object is created for each visit document, using the *visit_id* and the text associated to the visit. In Section 3.1

[3] http://lucene.apache.org/

we explained how these visit documents were generated.

We submitted several runs for evaluation, using the different strategies for generating indexes explained above: some include all the reports in the test collection (indexR11) and others only include the reports with a visit_id (indexR12). A further run was sent that uses the visit as index unit (indexV1). For these experiments, we use the StandardAnalyzer (version 3.6) in Lucene for analyzing the text.

In later experiments, we use the PorterStemFilter and the StopAnalyzer with the default stoplist. We generate indexes using the report as indexing unit. Concerning the treatment of negation, different indexes were created using both (1) the complete documents as given in the evaluation collection (indexR2), and (2) the documents containing only the text that is outside of the scopes of the negations, as produced by the two systems described in Section 2 (indexRN1, indexRN2), and by NegEx (indexRN3).

3.4 Retrieval

Searches are performed using the basic functionality of Lucene, using the indexes generated in the previous step. In order to combine the scores returned by Lucene for each report into a single score per visit, two different strategies has been implemented:

1. *Best score*: the score for a visit is computed as the best score obtained for any report associated to the visit.

2. *Sum of scores*: the score for a visit is computed as the sum of the scores obtained for all reports associated to the visit.

Different experiments have been also performed taking into account different numbers of *hitsPerPage*.

4 Results

In this section we present the evaluation results. We first show the results of the initial set of experiments (Table 4). In these experiments, we tested different combinations of the *hitsPerPage* parameter in Lucene, different strategies for combination of report scores and different indexes. The four best runs in this table were submitted to the TREC 2012 organization for evaluation.

Table 4 shows that our best run (ucm3) is that which uses reports to generate the index, the best score strategy for combining scores and 1000 hits per page. The remaining runs produce slightly worse results. In particular, we found that using the visit as the indexing unit does not improve the retrieval results.

Concerning comparison with other participants in the track, it may be seen from Table 5 that our best system produces results close to the median.

Table 6 shows the results of other post-submission experiments. For these experiments we try again different strategies for combination of report scores and different indexes, and compare the results with those obtained using negation processing systems.

Table 6 shows that the retrieval results are only slightly better than those of the best initial run. With respect to the effect of negation, the results show that eliminating the terms in the scope of negation only produce minor improvements. We think the reasons for these results are (1) the fact that only a few number of terms in the reports are affected by negations ($\approx$ 46 terms per report) and (2) most of these negated terms do not appear in the queries. The differences between the parsing based algorithms and NegEx are not significant.

5 Conclusions and Future Work

In this paper we presented our participation in the TREC 2012 Medical Records Track. Our aim in this paper was to evaluate the effect of detecting negation in the task of retrieving medical reports. The results have shown that processing negation improves retrieval performance, but this improvement is not significant. Besides, the execution of the parsing based negation processing algorithms takes a lot of time, which makes us question if it is appropriate to use them to deal with negations in the task at hand.

In the future, we aim to study the effect of a query expansion method, by including expanded terms that could appear in negated contexts.

We also want to explore the effect of negation in other related tasks, such as the extraction of specific information from medical reports (i.e., symptoms, treatments or procedures), where the role of negation

index	hitsPerPage	reports scores combination	submitted as	R-prec	P@10	infAP	infNDCG
indexR11	100	best score	-	0.2318	0.4340	0.1274	0.3334
indexR11	100	sum of scores	ucm1	0.2308	0.4532	0.1335	0.3460
indexR11	1000	best score	ucm4	0.2761	0.4340	0.1483	0.3849
indexR11	1000	sum of scores	ucm3	0.2859	0.4383	0.1548	0.4058
indexR12	1000	best score	ucm5	0.2722	0.4149	0.1455	0.3747
indexV1	1000	-	-	0.2549	0.3766	0.1413	0.3512

Table 4: UCM initial experiments.

	R-prec	P@10	infAP	infNDCG
Best	0.5761	0.8574	0.4697	0.7970
Median	0.2961	0.4702	0.1689	0.4244
Worst	0.0048	0.0213	0.0014	0.0179

Table 5: Average results for the run submitted by the different participants in the track.

index	hitsPerPage	reports scores combination	R-prec	P@10	infAP	infNDCG
indexR2	1000	best score	0.2817	0.4468	0.1608	0.4031
indexR2	1000	sum of scores	0.2931	0.4404	0.1545	0.4130
indexR2N1	1000	best score	0.2915	0.4447	0.1629	0.4084
indexR2N1	1000	sum of scores	0.2963	0.4362	0.1553	0.4155
indexR2N2	1000	best score	0.2893	0.4362	0.1625	0.4102
indexR2N2	1000	sum of scores	0.2991	0.4298	0.1560	0.4168
indexR2N3	1000	best score	0.2978	0.4532	0.1679	0.4150
indexR2N3	1000	sum of scores	0.3060	0.4447	0.1584	0.4194

Table 6: UCM results for post-submission negation experiments.

seems to be, a priori, more relevant.

Acknowledgments

This research is funded by the Spanish Ministry of Education and Science (TIN2009-14659-C03-01 Project). This research was also supported by the European Union (FP7-ICT-2011-7 - Language technologies - nr 288024 (LiMoSINe).)

References

Iman Amini, Mark Sanderson, David Martinez, and Xiaodong Li. 2011. Search for clinical records: Rmit at trec 2011 medical track. In *Proceedings of the twentieth Text REtrieval Conference (TREC 2011)*.

Miguel Ballesteros, Alberto Díaz, Virginia Francisco, Pablo Gervás, Jorge Carrillo de Albornoz, and Laura Plaza. 2012a. Ucm-2: a rule-based approach to infer the scope of negation via dependency parsing. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics (*SEM 2012), Montreal, Canada.*

Miguel Ballesteros, Virginia Francisco, Alberto Díaz, Jesús Herrera, and Pablo Gervás. 2012b. Inferring the scope of negation in biomedical documents. In *Proceedings of the 13th International Conference on Intelligent Text Processing and Computational Linguistics (CICLING 2012)*, New Delhi. Springer.

Jorge Carrillo-de-Albornoz, Laura Plaza, Alberto Díaz, and Miguel Ballesteros. 2012. Ucm-i: A rule-based syntactic approach for resolving the scope of negation. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics (*SEM 2012), Montreal, Canada.*

Sarvnaz Karimi, David Martinez, Sumukh Ghodke, Lumin Zhang, Hanna Suominen, and Lawrence Cavedon. 2011. Search for medical records: Nicta at trec 2011 medical track. In *Proceedings of the twentieth Text REtrieval Conference (TREC 2011)*.

Nut Limsopatham, Craig Macdonald, Iadh Ounis, Graham McDonald, and Matt-Mouley Bouamrane. 2011. University of glasgow at medical records track 2011: Experiments with terrier. In *Proceedings of the twentieth Text REtrieval Conference (TREC 2011)*.

Dekang Lin. 1998. Dependency-based evaluation of MINIPAR. In *Proceedings of the Workshop on the Evaluation of Parsing Systems*, Granada.

Roser Morante and Eduardo 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), Montreal, Canada.*

Roser Morante and Walter 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.

Roser Morante and Walter Daelemans. 2012. Conandoyle-neg: Annotation of negation in conan doyle stories. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC). Istanbul, Turkey*.

Roser Morante, Anthony Liekens, and Walter Daelemans. 2008. Learning the scope of negation in biomedical texts. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 715–724, Stroudsburg, PA, USA. Association for Computational Linguistics.

Veronika Vincze, Gyorgy Szarvas, Richard Farkas, Gyorgy Mora, and Janos Csirik. 2008. The BioScope corpus: biomedical texts annotated for uncertainty, negation and their scopes. *BMC Bioinformatics*, 9(Suppl 11):S9+.