Negation Cue Detection

Sharona Badloe 2712483
Michiel van Nederpelt 2684581
Desiree Gerritsen 2700295
Jingyue Zhang 2701079

Abstract

This report explores the effects of different systems and features on the task of negation cue detection. A simple rule-based classifier was built as a baseline, and compared with an SVM classifier. The systems were trained on data from the SEM 2012 shared task. A large set of 14 features was extracted from the data, ranging from Part-of-speech tags and lemmas to negation affixes. After an extensive feature ablation, the best features were kept and tested on our test set. The best feature set for negation cue detection on SVM turned out to be: Token, Pre-token, Next-token, Pre-lemma, POS, Negated event, NegAffix and HasNegAffix. The rule-based system performs at a macro average F1-score of 0.911 on the training set and 0.825 on the dev set. The SVM with our best feature selection performed with a macro average F1-score of 0.924 on the dev set. From our experiments, we can conclude that the SVM classifier with the best feature set performed better than the rule-based baseline, although most negations in-between fixed expression (also called I-NEG) were difficult to detect across all systems.

Github repository: `https://github.com/MichielvNederpelt/TMM-SHDM-2021`

1 Task Description

Negation is a common and complex phenomenon in natural language, it is the antithesis of affirmation and can be used to inverse the affirmative polarity in words, phrases, or sentences. Automatic negation detection is a task that has received a lot of attention in the field of natural language processing in recent years, not only because it is a feature of our language, but also because it is closely related to other natural language processing tasks, such as sentiment classification, information extraction, reading comprehension and question answering (Chowdhury, 2012; Abu-Jbara and Radev, 2012; Lapponi et al., 2012).

In the literature, it can be found that negation detection typically consists of three subtasks: negation cue detection, negation scope resolution and negation focus identification. (Chowdhury, 2012; Abu-Jbara and Radev, 2012; Lapponi et al., 2012).

Negation cue detection is finding the negation word, or combination of words, that triggers negation. The occurrence of negation in a sentence is determined by the presence of a negation cue (Morante, 2010). Below, the most common negation cues are illustrated:

- A single word: no, not, never
- A phrase: no longer, by no means
- An affix: un-, im-, -less, -n't
- A set of discontinuous words: neither. . . nor

Negation scope resolution addresses the problem of determining which tokens within a sentence are affected by the negation cue. A scope is a sequence of tokens that can be discontinuous (Morante and

Blanco, 2012). Negated focus identification is defined as the special part in the sentence, which is most prominently or explicitly negated by a negative expression (Zou et al., 2014).

For example, in the sentence below **not** is the negation cue. The negation cue's scope, words whose meaning is altered, is surrounded with square brackets. The focus, the most directly affected part, is underlined. This labeling style will be used throughout the report.

[The potential for transmission of vaccine virus was] **not** *[assessed in trials.]*

This report will solely focus on negation cue detection and it is organized as follows. In section 2, relevant features for negation cue detection will be discussed by means of related work. Furthermore, relevant natural language (pre)processing tools will be outlined. In section 3, we will elaborate about data annotation by means of the data itself and inter-annotator agreement scores. In section 4, we will describe our current dataset. Section 5 will be about preprocessing our corpus and feature extraction. Details about the experiments will be discussed in section 6. The error analysis will be discussed in section 7. In the last section, our conclusions and suggestions for further research will be presented.

2 Related Work: Classification Approaches And Features Used

Over the last two decades the sub-task of negation detection in higher-level tasks such as reading comprehension, information extraction and sentiment analysis has been explored through multiple approaches. Most attempts belong to four distinct categories: Rule-based systems, Machine Learning Classifiers, Conditional Random Field models and more recently Deep Learning through neural network designs.

Instinctively, one could imagine that the detection of negation cues could be relatively easily solved by applying rules and carefully designed heuristics. This exact approach was initially used by the few systems attempting negation detection. However, given the complexity of the human language these attempts were not fruitful. Over the last two decades, as computational power has surged, a large and new variety of systems have been developed and explored. Both Machine- and Deep Learning methods are showing very promising results (Khandelwal and Sawant, 2019).

2.1 Rule-based approaches

Mutalik et al. (2001) pioneered a simple rule-based system in 2001 making use of their algorithm "NegFinder". The approach was based on a lexical scanner using regular expressions to generate a finite state machine which could detect negation cues in natural language. Chapman et al. (2001) made use of NegEx, a simple regular expression algorithm, for negation cue detection. As the algorithm is of higher simplicity and assumes less ambiguity it has been used extensively in further research.

The *SEM 2012 Shared task, although dominated by Conditional Random Field model approaches, saw three teams de Albornoz et al. (2012), Ballesteros et al. (2012) and Basile et al. (2012) making use of a Rule-Based system. While the design of the respective systems was different for all three team approaches, the rule-based methods showed that, with well-defined task-specific rules, systems could attain acceptable performance. The apparent, major, disadvantage is that these rules do not generalize over other data sets or domains.

Peng et al. (2018) introduced the NegBio algorithm which made use of Universal Dependency patterns. The NegBio algorithm was an improvement of the NegEx algorithm introduced by Chapman et al. (2001). NegBio worked extremely well and showed that, for the medical domain, the incorporation of dependency parsing and words in their respective contexts was a major improvement. It should be noted that the use of a continuous bag of words might not be sufficient for further research as the sequential order of words makes a big difference (Khandelwal and Sawant, 2019).

2.2 Machine learning (ML) approaches

Development of Machine Learning classifiers, or systems, for the task of negation detection was first published by Rokach et al. (2008). Moving from rule-based only to the addition of classifiers, the system improved. The model relied on a regular expression matching paradigm to come up with better rules for negation detection (Rokach et al., 2008).

Morante and Daelemans (2009) made the first steps in a memory based Machine Learning approach in 2009 with their IGTREE algorithm. For detecting negation cues, a combination of a memory-based algorithm, SVM (Cortes and Vapnik, 1995) and CRF (Lafferty et al., 2001) classifiers were used resulting in state-of the art results (Morante and Daelemans, 2009).

Ou and Patrick (2015) made a comparison between two rule-based algorithms and a SVM classifier. The SVM classifier outperformed both rule-based methods, showing further need for exploration in Machine Learning Techniques and that the promise was not in furthering rule-based methods (Ou and Patrick, 2015).

2.3 Conditional random field (CRF) approaches

As the promise of improvements by incorporating word order, a third approach for negation cue detection was formulated in the form of Conditional Random Fields. Partly because the first developed systems suggested that training corpora were too small to generalize over natural language, a shared task was formulated in 2012.

The *SEM shared task of 2012 (Morante and Blanco, 2012) saw five teams, with state-of-the-art results for negation cue detection, making use of CRF classifiers (Abu-Jbara and Radev, 2012). Almost all teams performed very well even though the time constraint of the task hampered further research.

Qian et al. (2016) made further use of two CRF classifiers, a semi-Markov CRF and latent-variable CRF, outperforming all other earlier developed systems on the Sherlock dataset, beating Deep Learning as well. Main source of the wowing results was the exploitation of features related to negation cue, long distance dependency and structural information.

2.4 Deep learning approaches

As the usage of Deep Learning methods has been a hot topic and an area of major interest in the entire spectrum of Natural Language Processing, it has also been in the task of negation detection. Qian et al. (2016) pioneered with the usage of Deep Learning architectures. Together with Fancellu et al. (2016) they focused on scope detection, with usage of Convolutional Neural Network and BiLSTM respectively, both groups reported on the best results seen up till then. This was a signal for the entire community for the potential of Deep Learning architectures.

Lazib et al. (2019) used a large variety of Deep Learning architectures on the SFU Review Corpus Dataset. Experimenting with Recurrent Neural Networks, LSTM, BiLSTM, GRU and CRF showing that the BiLSTM outperformed the other architectures. Hereby showing BiLSTM architecture potential over multiple datasets. More recent work has seen an increase in the use of different BiLSTM architectures, improving results even further (Khandelwal and Sawant, 2019).

2.5 *SEM 2012 shared task and features

In this section, a selection of reports made for the *SEM 2012 shared task (Morante and Blanco, 2012) will be discussed in more detail, with attention to the approach and features used in negation cue detection. These reports will be utilized as the experimental basis for our project.

*SEM (2012) is a joint conference on lexical and computational semantics organized by two ACL special interest groups: SIGLEX and SIGSEM. Each year, *SEM hosts a shared task on different aspects of semantic processing. The shared task of the first edition in 2012 was dedicated to resolving the scope and focus of negation. Participants were provided with two annotated datasets. The dataset CD-SCO is a corpus of Conan Doyle stories, annotated with negation cues, negation scope and the event or property that is negated. The dataset PB-FOC is a corpus consisting of 3544 verbal negation instances taken from PropBank with their focus annotated. Only negations marked with MNEG in PropBank were included, and the semantic role containing the most likely focus was selected as annotation.

The team from FBK (Chowdhury, 2012) trained three different CRF classifiers for each of the subtasks: negation cue detection, negation scope identification and negation focus identification. Their system obtained the best F1 score for negation cue detection and ranked third among the participating teams for full negation detection. For negation cue detection, first a list of highly probable negation expressions (NegExpList) was taken from the training data based on frequency. The following negational affixes

were collected from the training data and used to detect negation cue subtokens: 'less', 'un', 'dis', 'im', 'in', 'non' and 'ir'. Additional annotating was done on the training data for negational expressions such as 'by no means'. Then, the CRF classifier was trained on the following features:

- Lemma i: Lemmatized form of token
- Lemma i-1: Lemmatized form of previous token
- POS i: Part-of-speech tag of token
- hasNegPrefix: If tokeni has a negation prefix and is found inside the automatically created vocabulary
- hasNegSuffix: If tokeni has a negation suffix and is found inside the automatically created vocabulary
- MatchesNegExp: If token is on the NegExpList

The team from UMichigan (Abu-Jbara and Radev, 2012) had a similar approach, also using three CRF classifiers for each of the three subtasks: cue, scope and focus. Their system detects negation cues with an F1-score of 90.98%. For each iteration, their CRF model for negation cue detection considers the features of the current token, the two previous tokens, and the two proceeding tokens. Their model also uses n-grams of the tokens and POS-tags. They decided on a five-label classification scheme for negation cues consisting of the following labels: 'O' for non-negations, 'NEG' for single-word negation cues, 'PRE' for cues with a prefix, 'POST' for cues with a suffix and 'MULTINEG' for multi-word negation cues. Regular expression are used to determine which part of the negation cue is the negational affix. Two features, in addition to those mentioned above, have been used in this experiment:

- Part-Of-Speech tag category: Part-of-speech tags reduced into 5 categories: Adjective (ADJ), Verb (VB), Noun (NN), Adverb (ADVB), Pronoun (PRO), and other (OTH)
- Is punctuation mark: This feature takes the value 1 if the token is a punctuation mark and 0 otherwise

The team from UiO2 (Lapponi et al., 2012) used a combination of CRF and SVM classifiers. Negation cue detection was done with the SVM. This team used a slightly different approach then the other two mentioned. In addition to the token-level features described above, this team added character-level n-grams for each token as a feature. They created a lexicon that counts the frequencies of each lemmatized token in the training data, as well as the frequencies of the character n-grams. This is done to detect tokens that possess one of the negation affixes, but are not a negation cue, such as the word 'underlying'. The negation affixes are removed from the token, after which the system looks up in the created lexicon how likely it is for the token to occur without the negation affix. In the case of the token 'underlying', the term without the affix 'derlying' is unlikely to be the start of a word and can therefore be classified as 'not a negation cue'. Their negation cue detection classifier saw good results with an F1-score of 91.31%.

2.6 Preprocessing the corpus to obtain features

For our project, we will perform an extensive feature ablation study to determine the best features for negation cue detection in our data. We will consider all the features mentioned above in section 2.5, and some additional features. In the three reports on the SEM shared task mentioned above, we noticed that the lemma of the previous token is used as feature, but not the lemma of the next token. We also noticed that the POS-tag of only the current token is considered as feature in those reports, and not the POS-tag of the previous and next tokens. Since no explanation was given as to why these decisions were made, we want to include these features in our ablation study. We hope this will allow us to acquire more information on the (un)informativeness of these additional features.

For convenience and efficiency, we would like to tokenize, lemmatize and POS-tag with one tool throughout our project. We have discussed the use of either SpaCy (SpaCy, 2016) or NLTK (Bird et al., 2009) for obtaining all of these features. However, the specific tools we will choose will depend strongly on the format of our data. Knowing what our data looks like will give us more information on which features are already present, which features still need to be obtained and which tools will work the best for our project. This section will be updated in more detail as soon as the data has been provided.

3 Data annotation

The annotation task consisted of annotating negation cues in ten English documents. As seen in section 2, negation cues could be words, affixes or multiple words that cause certain parts of the sentence to be negated. The main task was to identify the negation cues and mark these. The annotation process was done in Ehost (South et al., 2012), which is an annotation tool that can later on be used to calculate Inter-Annotator Agreement.

There are 25 documents in our batch5 database, and according to our classification there are five categories: posts, newsletters, article summaries, QA and news. We randomly selected at least one document from each category, and a total of 10 documents were selected for annotation. The data consisted of ten English documents that were all related to vaccination, more specifically pro- or anti-vaccination. The documents were provided to us by the Vrije Universiteit Amsterdam.

All authors annotated the same ten documents. It should be noted that all authors are non-native English speakers. Three annotators are native Dutch speakers, and one is a native speaker of Mandarin. The annotators are all Text Mining students at the VU, without annotation experience. The annotators were provided with the same annotation guidelines (Morante et al., 2011) and instructions. All annotators used eHOST to annotate their documents. For all annotated documents, see our github repository. Two annotators used IOS as an operating system, and the other two used Windows. Unfortunately, annotations made on different operating systems cannot be compared. For this reason, AAI scores will be calculated between IOS users and between Windows users.

As mentioned, all annotators were provided with an annotation guideline that was meant to clarify when something is a negation cue. The guideline also provides constructions that might be ambiguous or unclear, to show what to choose in these situations. Some negation cues to consider:

- Words: no, not, never, nothing, without, fail (verb)
- Affixes: un-, im-, -less, ir-. Note to take the meaning of the lexical item into account
- Infixal negation: -less- in breathlessness
- Contracted negation: ‘t, n’t or not in can’t/don’t/cannot
- Multiwords: phrases such as ‘no longer’, ‘by no means’, ‘not for the world’
- Discontinuous multiwords: Neither . . . nor, not. . . not

Keep in mind that not all negation cues are actual negation cues, also known as false negations. Especially in cases where the negation cue is in some confirmation check from the addressee. ‘Isn’t it obvious that the dog is black?’. There are also some other fixed expressions (e.g. ‘nothing of much importance’, ‘none the less’, ‘don’t you think’, ‘no doubt’). For an extensive overview of the guidelines and additional examples, see Morante et al. (2011).

3.1 Inter-annotator agreement analysis

In order to check the reliability of the annotators, annotations and the annotation process, Inter-Annotator Agreement (IAA) scores are calculated. Agreement scores among annotators on the same data show to what extent the annotation process was consistent, or reproducible (Artstein, 2017). In eHOST, the annotated negation cues are saved as spans, meaning their index range in a document. When annotators agree, they have annotated the same span. The score is calculated by summing all the matching annotations and

all the non-matching annotations from both annotators. Then, the matching annotations are divided by all the reported annotations.

3.1.1 IAA scores

Table 1 shows a pairwise comparison between our two annotators operating on Windows, Desiree and Sharona. Both annotators agreed on 224 characters. 477 characters were annotated by Sharona but left out by Desiree. 65 characters were annotated by Desiree, but not by Sharona. This results in an F-score of 45.3%.

Table 2 shows a pairwise comparison between our two annotators operating on IOS, Michiel and Jingyue. Both annotators agreed on 196 characters. 102 characters were annotated by Jingyue but left out by Michiel. 83 characters were annotated by Michiel, but not by Jingyue. This results in an F-score of 67.9%.

Gold standard set	Compared set	TP	FP	FN	Recall	Precision	F-measure
Desiree	Sharona	224	477	65	77.5%	32.0%	45.3%
Sharona	Desiree	224	65	477	32.0%	77.5%	45.3%

Table 1: IAA of Windows operators.

Gold standard set	Compared set	TP	FP	FN	Recall	Precision	F-measure
Jingyue	Michiel	196	83	102	70.3%	65.8%	67.9%
Michiel	Jingyue	196	102	83	65.8%	70.3%	67.9%

Table 2: IAA of IOS operators.

3.1.2 Inter-annotator error analysis

In the following section an error analysis is performed in order to see patterns of agreement and disagreement.

In general, simple negation cues such as negation words (e.g., no, not, nothing) and contracted negation (i.e., n't or 't) went well. Apart from some sloppiness by means of reading over a negation cue or not selecting the entire span, these cues did not really pose a problem. The following categories did pose a problem:

- Stylistic ambiguities: Some annotators selected dashes as part of the negation and some did not. E.g. 'non' versus 'non-'. These errors were very frequent and impacted IAA-scores greatly.
- Semantic ambiguities: Some words can be semantically seen as negation. We found some disagreement on which words are negations between annotators. E.g. some annotated 'a lack of', or 'wrong' as negation, others did not. However not very frequent, this is an error probably influenced by the non-nativity of the annotators.
- False negation ambiguities: According to the guidelines, some negation cues are not supposed to be annotated when they are used in an interrogative or fixed expression. We noticed some disagreement on this in practice.

The errors in the stylistic category can be easily fixed by updating the annotation guidelines. All annotators were in doubt whether or not to annotate dashes as part of the negation cue. As this issue depends on the goal and task of the project, all annotators looked through the guidelines for the answer. However, no information was given on how to deal with dashes in negations. Therefore all annotators had to make their own decision on this matter, resulting in disagreement.

The errors in the second category, semantic ambiguities, are a bit more complex. Some of us annotated terms like 'wrong' and 'lack of' as negations and some did not. The list of negation cues per POS in Appendix A of the guidelines was very helpful to resolve some of these disagreements, but some

ambiguities were still left unsure. The doubt came from the fact that the term 'fails' was mentioned in the guidelines as an example of semantic negation, but not many other correct or incorrect examples were given. Perhaps a bit more of an explanation might decrease some of the errors in this category.

The third category is not so much a guideline issue, since this category was extensively discussed. But even after being explained that some negations do no count as negation cues in the context of interrogations and fixed expressions, how can someone be sure what the pragmatic inference of a sentence is? This depends strongly on the annotators comprehension and familiarity with the language. There are also differences in environment, background and many other factors that can cause two annotators to interpret a sentence or term differently. Therefore, we feel like the errors in this category are the most complex to fix.

3.1.3 Reflection on the annotation task

Based on the error analysis and the language background of the annotators in our group, we summarized the following reflections. First, annotators should read the annotation guideline in more detail, including the Appendix. For uncertain negation cues, the annotation guideline should be checked in time. For the negation cue that is not mentioned in the annotation guideline, such as 'non' versus 'non-', it should be specifically noted and discussed with the annotation guideline writers or annotators. Secondly, the annotator should have the same language background and similar proficiency in the target language, and preferably be a native speaker of the target language. Finally, the choice of annotation documents should include as many different genres as possible; different genres better reflect the consistency among annotators.

4 Dataset description

In this chapter, the information about training and development datasets is explored. Observing and describing datasets in detail is one of the key steps in tackling natural language processing tasks. In this part, we present the format, structure and content of the datasets, digging further into the characteristics of the datasets with basic statistics that reveal whether the dataset is balanced or not.

4.1 General description of the dataset

The datasets that are used were provided by the Vrije Universiteit Amsterdam and *SEM (The Joint Conference on Lexical and Computational Semantics). The corpus is constructed out of Conan Doyle stories. Our data consists of two folders, one contains a training dataset (containing the story 'The hound of the Baskervilles') and a development dataset (containing the story 'The adventure of Wisteria Lodge'). The other folder contains two test datasets (containing the stories 'The Adventure of the Red Circle' and 'The Adventure of the Cardboard Box') (Morante and Blanco, 2012). All datasets are in .txt format. The training and development datasets have the same attribute values:

- Column 1: Name of the story
- Column 2: Sentence number
- Column 3: Token number within the sentence
- Column 4: The token itself
- Column 5: Negation information

In column 5, negation information is labeled in IOB format. If a token does not contain a negation, it is labeled O. If it does contain negations, there are two situations. The first situation: B-NEG for a single negation word without another negation following, so the negation is followed by a non-negation token which is labeled O. The second situation: A negation is labeled B-NEG and the next token is also, or part of, a negation token. In this second situation the B-NEG label is followed by an I-NEG label for phrases and sets of continuous words.

4.2 Basic statistics of our corpora

In order to familiarize with the given datasets, some general information was gathered (see Table 3). It was found that there are 65,451 tokens in our training dataset, and 13,567 tokens in our development set. There are 3,644 sentences in the training set and 787 sentences in the development dataset. In the training set, there were 848 negated sentences (23.3%), whereas the development set has 144 negated sentences (18.3%).

The POS-tag distribution of all tokens in both the training and development set (see Table 55, Appendix 6) show a balanced dataset. Despite some small differences (for instance, the order of NNP and RB in the development set), the overall trend seems to be balanced.

In total, 1,003 negation cues were found for the training dataset. It was found that 987 negation cues were labeled as 'B-NEG' and 16 negation cues were labeled as 'I-NEG'. The total number of negation cues for the development set was 179, with 176 being labeled as 'B-NEG' and 3 being labeled as 'I-NEG'. For the training dataset, 136 unique negation cues were found. It was found that 128 unique negation cues were labeled as 'B-NEG', and 9 were labeled as 'I-NEG'. In the development set, it was found that 37 unique cues were labeled as 'B-NEG' and 3 negation cues were labeled as 'I-NEG'. Note that 'no', in both the training and development set, was found in both 'B-NEG' and 'I-NEG' and therefore the total number of unique negation cues is lower. Lastly, the training dataset contained 856 negation words, and 147 negation affixes. In the development dataset, 104 negation words and 75 negation affixes were found. The distribution of POS-tags for negation cues can be found in figure 1. The main POS-tags for negation cues in both training dataset and development set are RB, DT, JJ, NN and IN. In general, the development dataset is smaller compared to the training dataset. Nevertheless, it can be seen that the development dataset somewhat follows the same trend as the training dataset. Unfortunately, some POS-tags (i.e., VBP, CD, VBD) are not present in the development set, but the overall distribution seems balanced. Especially given the fact that the missing POS-tags are also underrepresented in the training dataset.

	Training set	**Development set**
Number of tokens	65,451	13,567
Number of sentences	3,644	787
Number of negated sentences	848	144
% of negated sentences	23,3%	18,3%
Number of negation cues	Total:1003 (B-NEG:987, I-NEG:16)	Total:179 (B-NEG:176, I-NEG:3)
Number of unique negation cues	Total:136 (B-NEG:128, I-NEG:9)	Total:39 (B-NEG:37, I-NEG:3)
Number of negation cues: word vs affix	words: 856, affixes: 147	words: 104, affixes: 75

Table 3: Basic statistics of the training and development dataset.

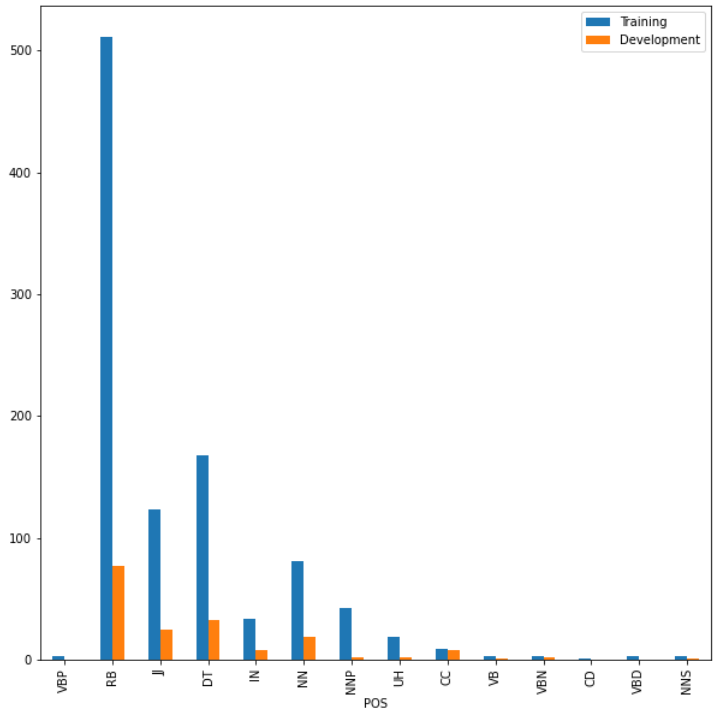

Figure 1: The distribution of POS-tags for negations in the training and development dataset. See Table 6 for the corresponding numbers.

5 Corpus pre-processing and feature extraction

In this chapter, data pre-processing and feature extraction will be explored. We describe the motivation for the choice of NLP toolkit, list the feature set we used in the following experiment and how these features were extracted. Lastly, the research and extraction of complex affixal features will be presented.

5.1 Preprocessing tools

A lot of preprocessing steps have already been made in the data that was provided to us. It has already been tokenized, and sentence borders are marked by whitespaces and sentence numbers. The only further general preprocessing step we have made is lowercasing all the data. Any other preprocessing steps are specific to each feature and will be explained in section 5.2.

We have used two major tools to extract our features: NLTK and SpaCy. During the process of discussing our features, we compared the output of these two tools for POS-tagging and lemmatizing. We found that NLTK gave us questionable results for lemmatizing. For example, it lemmatized 'having' as 'hav'. It also kept the word 'refused' as it was, rather than outputting the lemmatized form 'refuse'. Comparing the output of NLTK with the output of SpaCy, we decided to use SpaCy for this feature.

For POS-tagging, we found that NLTK worked better for our data compared to SpaCy. Our data has already been tokenized, but during the process of POS-tagging, SpaCy split some tokens into more tokens, creating a list of POS-tags rather than a single POS-tag. Since it might be confusing to our classifier to have some tokens with only one POS-tag value, and some tokens with a list of several POS-tag values, we decided to use NLTK for this feature.

5.2 Feature description and motivation

Feature	**Description**	**Feature value**
Token i	The current token	Token
Token i-1	The previous token	Token
Token i+1	The next token	Token
Lemma i	Lemmatized form of token	lemmatized token
Lemma i-1	Lemmatized form of previous token	Lemmatized token
Lemma i+1	Lemmatized form of next token	Lemmatized token
POS i	Part-of-speech tag of token	POS-tag
POS i-1	Part-of-speech tag of previous token	POS-tag
POS i+1	Part-of-speech tag of next token	POS-tag
POS-tag classification	Coarse-grained POS-tags	ADJ, VB, NN, ADVB, PRO or other
Punctuation	If token is a punctuation mark	Binary
MatchesNegExp	If token is on the Negation list	Binary
HasNegAffix	If token has a negation affix and is not found in the positive vocabulary list	Binary
Negated event	For affixal negations: the negated event	Negated part of the affixal negation cue
Negated affix	For affixal negations: the negational affix	Suffixes: less, lessly, lessness and prefixes: un, dis, im, in, non or ir

Table 4: Distribution of POS tags across and within training and development datasets.

Previous token and next token: In order to account for context, the features previous and next token are added. In order to get the previous token, the 'Token' column was shifted downwards. For the next token feature, the list of original tokens was shifted upwards.

Lemma, previous Lemma and next Lemma: Lemmas were found to be an important feature in previous research. In order to extract this feature, tokens were lemmatized by SpaCy. The list of lemmas were added to the dataset. In order to get the previous and next lemma as features, the generated lemma list was used. Shifting the index upwards and downwards provided us with two columns, one for the previous lemma and the other for the next lemma.

POS-tag, previous and next POS-tag: Part-of-speech tag has been pointed out as an important feature in previous papers. NLTK was used to extract the POS-tag of all tokens as a feature. Based on further analysis of the POS-tag of the training set and development set, we found that negative cues are mostly distributed in adverbs, adjectives, and qualifiers (Figure 1). Previous and next POS-tags are also implemented as features, which we assume will be useful to train our classifier to recognize fixed negation expressions and a set of continuous negation cues. In order to get the previous and next POS tags, the generated POS-tag list was used.

POS-tag categorization: a more generalized, less fine-grained, way of presenting the extracted POS-tags to the classifier in six distinct categories. Earlier research showed promising results using the classified POS-tags as a feature. As some POS-tags are more difficult to generalize over the six different categories without the risk of high loss of information, the effects of categorization will be studied further on. Some ambiguous cases like the particle-tag (e.g. 's, not) do not have a clearly defined class but convey a lot of information for negation. Therefore the hypothesis is formed that classifying the particle-tag to "other" might not be the best way and maybe even a seventh class needs to be introduced to shield losing this information. Categories for distribution (now): Adjective (ADJ), Verb (VB), Noun (NN), Adverb (ADVB), Pronoun (PRO), and other (OTH). See Table 14 in the Appendix for categorization.

Punctuation: It was noticed, in the extraction of basic statistics, that punctuation has a high occurrence in our dataset and is composed of points, commas, colons etcetera. As earlier research often showed very promising results using this feature we will incorporate it in our ablation study. So far the utilityof adding this feature is not yet clear (will be added after system design and ablation study). For the extraction of punctuation both Spacy and a built-in python module were used. Spacy punctuation extraction resulted in a list of punctuation values e.g. "hearth-rug" [0, 1, 0] or "1." [0, 1] which might lead to confusing the classifier. However if the built-in python module is used "1." is valued as not containing punctuation. (A choice of method will be made later in the process).

MatchesNegExp: This feature is taken from Chowdhury (2012), where it has been used with good results (see section 2.5). First, the NegExpList is created. This is a list of tokens that are very likely to be a negation cue, taken from the training data based on frequency. The list consists of the following terms: nor, neither, without, nobody, none, nothing, never, not, no, nowhere, and non. (Chowdhury, 2012). This feature checks for each token if it exists on the NegExpList, and outputs a binary value. 1 if the token is on the list, 0 if it is not. Before utilizing the list that the team from FBK (Chowdhury, 2012) made, we gathered all tokens from our data that were annotated as negations, sorted them by frequency and inspected if the NegExpList should be altered in any way for our project. We decided to make no further changes to the list, as we found it was already complete.

5.2.1 Affixal features

With the use of the MatchesNegExp feature, many negational terms such as 'nobody' will be quite easy to spot for the classifier. However, there are also a number of affixal negations present in the data that will be a lot harder to classify. This is because some negatival affixes can also be part of a word that is not a negation, such as the word 'until'. The following features have been added to our system to help the classifier distinguish between true negational affixes, and apparent negational affixes.

hasNegAffix: To implement this feature, a list of positive vocabulary was collected from the training data. This list consists out of every token that was annotated with the label 'O', and is therefore not seen as a negation cue according to our gold data. Then, the following list of negational affixes that occur in the training data was taken from Lapponi et al. (2012): un, dis, ir, im, in, less, lessly, lessness. The last two affixes are included as an internal representation for the infix 'less', as seen in Lapponi et al. (2012). This feature assigns each token a binary value based on two conditions:

- Does it have a negational affix
- Has it been excluded from the positive vocabulary

If the token has one of the aforementioned affixes, and does not occur on the positive vocabulary, the token can be seen as an affixal negation cue and gets a value of 1. In all other scenarios the token gets a

value of 0. E.g. When our system encounters the token 'until', it will recognize the potential negational affix 'un'. It will then check if the token occurs on the positive vocabulary. Since 'until' is not annotated as negation, it will be present on the positive list. Therefore it is not a negation cue, and the affix 'un' is not seen as a negational affix in this case. The token 'until' then gets a value of 0, because it is not an affixal negation cue.

Negated event: This feature displays the negated event of any token that is classified as an affixal negation cue by the feature hasNegAffix. The negational affix is removed from the token, leaving behind the negated event. The negated event of the token 'unknown' is 'known'. If the token is not an affixal negation cue, the token gets a dash (-) to signify that this feature does not apply to the token.

Negational affix: This feature displays the negational affix of any token that is classified as an affixal negation cue by the feature hasNegAffix. The negational affix is extracted from the token. The negational affix of the token 'unknown' is 'un'. If the token is not an affixal negation cue, the token gets a dash (-) to signify that this feature does not apply to the token.

6 Experiments

As was described in the related work section, the task of negation detection has seen multiple approaches, some more fruitful than others. Many other NLP task solutions have found their origin in a rule-based approach, and so has negation detection. However, great results have been seen for other methods as well. For our experimental set-up, we have chosen to compare two approaches in negation cue detection: rule-based, and machine learning. In order to sufficiently compare the two approaches, an extensive feature ablation will be performed on our machine learning approach. As an additional experiment, both of our approaches will be tested on a similar development dataset, and on a new dataset (BioScope). This is done in order to see how well our systems generalize across various datasets.

In section 6.1 we will include a short description of our classification task. In section 6.2, we will explain the systems and our experimental set-up in more detail. Section 6.3 contains a detailed description of the inner mechanisms of our rule-based approach and our motivation for using this system. In section 6.4 we dive into our SVM system. In section 6.5, we evaluate our results. This section also contains an extensive feature ablation and a brief parameter optimization. In section 6.6 we conclude our experiments.

6.1 Classification task description

For our report, the classification task is to predict the correct label given a token and its features. The system should classify a negation cue as 'B-NEG' or 'I-NEG', and other (non negation) words as 'O'. Throughout our project, two types of negations will be detected.

- Lexical negations: words that have negation as its main purpose and meaning, e.g. 'nobody' and 'not'.
- Affixal negations: words with an affix added to it that negates the meaning of the word without the affix, e.g. 'infrequent.'

6.2 System description

For this task, two systems will be trained in order to see which system performs best for our task. We tried to do a stepwise increase in system complexity based on the level of results in previous research. Our system design is progressive in this order: rule-based to machine learning. The first system is the baseline. The goal is to predict a negation label solely based on a set of conditions, also known as a rule-based system. The next system will be a machine learning system, Support Vector Machine (Boser et al., 1992). SVM classifiers are user friendly, generate quick results and are easy to work with when performing a feature ablation (Enger et al., 2017). Previous research showed that SVM classifiers perform well on simple tasks such as negation cue detection, or sentiment analysis (Enger et al., 2017). After training and testing on our development set, an extensive feature ablation study will be performed in order to find the best feature combination.

As an additional experiment, we will be feeding another dataset into our systems to see how well they generalize over various datasets. The data that will be used is the BioScope corpus. The BioScope corpus consists of medical and biological texts annotated for negation, speculation and their linguistic scope. The corpus is publicly available for research purposes, and was provided to us in this course.

6.3 Baseline system

To sufficiently compare our Machine Learning, we have created a simple rule-based system as a baseline. By implementing this baseline into our experiment, we hope to gain more information on the usefulness of Machine Learning for negation cue detection.

Just like our SVM, our rule-based system will detect lexical and affixal negation cues. We have created a lexicon of negation cues, and the system takes a lexicon look-up approach for classification. It takes as input a tokenized corpus in the form of a list, and assigns each token one of the following classes: B-NEG, I-NEG, or O. The output is a list of all the assigned classes in the same order as the tokens. This list of classified labels can then be used together with the list of gold labels for evaluation.

Our rule-based system is created as an extension of work done by Chantal van Son, Emiel van Miltenburg and Roser Morante for ExProm 2016 (Blanco et al., 2016). ExProm 2016 is a workshop which focuses on Machine Learning approaches for different types of semantic cue detections such as contradiction detection, uncertainty cue detection and negation cue detection. As a way to bring light to the many complications of negation cue detection, both within and outside of Machine Learning, they created the start of a negation lexicon from WordNet for others to extend according to their task.

The ExProm 2016 team started by extracting all words from WordNet that have a direct antonym. The words were put into a Table together with their antonym to create word-pairs of direct opposites. Many of these word-pairs contain affixal negations, for example, the direct antonym of 'unbiased' is 'biased'. But antonym pairs like 'relative' and 'absolute' are also present in their lexicon. For our task, these words do not count as negation and therefore need to be filtered out.

To extract all affixal negations from the list of antonym word pairs, we iterated over each word in all the pairs. If a word has a negational affix, and the word without the affix matches its antonym, then it counts as an affixal negation cue and will be added to our lexicon. We then added the words in our NegExpList (see section 2.5) to all the affixal negations we extracted from WordNet to create a large lexicon of 1761 lexical and affixal negation cues.

Our rule-based system then operates on the following simple conditions:

For each token in the input file, if the token occurs on our created lexicon, assign it a B-NEG. If not, assign the label O. To include the I-NEG class, all fixed multi-word negation expressions found in the training data have been hardcoded into the system. The system can then classify these sequences of recurring words with the appropriate classes.

6.4 SVM system

For negation cue detection, Support Vector Machine classifiers (SVM) as a machine learning system is shown to be a sufficient system that is easy to use and quick (Enger et al., 2017). An SVM system can deal with multiple features and it can classify data into multiple classes. Given the training data, the system finds the best hyper plane that can separate the data most effectively into its classes. The system takes as data input vectorized tokens with features. The system is trained on the vectorized tokens together with features and the corresponding label.

The classifier that was used in this report is a linear SVM from sklearn (Pedregosa et al., 2011), with default parameters (see Table 15 for the parameters, values and descriptions). See section 6.5.3 for the parameter optimization. After training, the trained classifier can be used to generate predictions for the testing data. In our case, this is the development set. The system is evaluated by comparing the gold labels of the development set and the predicted labels generated by the SVM, by taking into account precision, recall and F1-scores per class and the macro average scores. In order to evaluate the system, confusion matrices were created. See our github repository for the scripts.

6.5 Evaluation

Below, one can find evaluation metrics for each system. The evaluation is done based on confusion matrices of the predicted and the gold labels. The evaluation metrics show the overall performance of each system. In order to compare the systems, macro average scores are compared. Macro-averaged scores are averages of the F1-scores obtained for each class. Which makes it better to evaluate the model and how it handles imbalance (Boin et al., 2020). In our datasets, the 'O' class is much more represented compared to 'B-NEG' and 'I-NEG'.

6.5.1 Evaluation metrics

As shown in Table 5, the rule-based baseline systems performed extremely well. In the training dataset, the baseline successfully detected negation cues and got an F1 score higher than 0.8 for each class. In the development dataset, the outcome of the baseline is not as good as it is in the training set, but most scores are higher than 0.8 except the score of 'I-NEG', this may be due to the size of the data set and data bias. When comparing the baseline system to the SVM system, the SVM scores better in each category, almost all reaching above 0.9, except 'I-NEG'. In general, the SVM system performs better than the rule-based system. It should be noted, however, that the SVM system does not detect 'I-NEG' as well as the rule-based approach does on the training data.

	Baseline (training data)			**Baseline (development data)**			**SVM (all features)**		
	P	R	F1	P	R	F1	P	R	F1
B-NEG	0.905	0.883	0.894	0.841	0.784	0.812	0.961	0.972	0.966
I-NEG	0.737	1.000	0.842	0.667	0.667	0.667	1.000	0.667	0.800
O	0.998	0.998	0.998	0.997	0.998	0.998	1.000	0.999	1.000
Accuracy			0.997			0.995			0.999
Macro avg	0.877	0.961	0.911	0.835	0.816	0.825	0.987	0.879	0.922
Weighted avg	0.997	0.997	0.997	0.995	0.995	0.995	0.999	0.999	0.999

Table 5: Evaluation metrics of the SEM dataset

The Table below shows the performance of the rule-based system and SVM classifier tested on a different dataset, being BioScope. It shows a decline in performance of the SVM classifier on the BioScope dataset. All scores dropped from around 0.900 to a score around 0.655. Especially 'I-NEG' labels are difficult to detect in the BioScope dataset. This might be due to annotation differences between both datasets.

	Baseline			**SVM (all features)**		
	P	R	F1	P	R	F1
B-NEG	0.898	0.960	0.928	0.904	0.883	0.894
I-NEG	1.000	0.167	0.286	0.727	1.000	0.842
O	0.999	0.998	0.998	0.998	0.998	0.998
Accuracy			0.997			0.997
Macro avg	0.966	0.708	0.737	0.877	0.961	0.911
Weighted avg	0.997	0.997	0.997	0.997	0.997	0.997

Table 6: Evaluation metrics of the BioScope dataset

6.5.2 Feature ablation

In order to find the best features, our SVM takes the feature 'token' as a reference. The main approach is to add each individual feature together with the token to see what the impact is of each feature on the evaluation scores (step 1). As one can see in bold, the features 'Pre-token', 'Next-token', 'POS',

'Negated event', 'NegAffix' and ' Lemma' all contribute to a better performing system. Features such as 'Pre-POS', 'Next-Pos', 'Pre-Lemma', 'Next-Lemma', 'POS classified', 'punctuation' and 'HasNegAffix' did not individually contribute to a better system. All the features that were beneficial to the system were taken together in step 2 to evaluate the performance. It is shown that with this set of features, the performance is almost as good as taking all features (see SVM evaluation metrics, Table 5). For step 3, taking it one step further, we explored which features could be added to the set of selected features to improve the system even more. Systematically, each individual remaining feature was added to the new set. For step 4, three features that were shown to be equally performative or better (Punctuation python, MatchesNegExp, HasNegAffix) were taken together with the selected features of step 2. Unfortunately, performance was better with only adding 'HasNegAffix' as a feature.

	Features	**Precision**	**Recall**	**F1-score**
	Token (reference)	0.640	0.623	0.631
Step 1	Token - **Pre-token**	0.971	0.732	0.796
	Token - **Next-token**	0.986	0.719	0.795
	Token - Lemma	0.640	0.623	0.631
	Token - **Pre-lemma**	0.971	0.723	0.796
	Token - Next-lemma	0.650	0.608	0.628
	Token - **POS**	0.973	0.734	0.798
	Token - Pre-POS	0.640	0.623	0.631
	Token - Next-POS	0.640	0.619	0.629
	Token - POS classified	0.640	0.623	0.631
	Token - Punctuation python	0.640	0.623	0.631
	Token - MatchesNegExp	0.640	0.623	0.631
	Token - HasNegAffix	0.640	0.623	0.631
	Token - **Negated event**	0.643	0.663	0.653
	Token - **NegAffix**	0.643	0.663	0.653
Step 2	Selected features (Token, Pre-token, Next-token, Pre-lemma, POS, Negated event, NegAffix)	0.990	0.876	0.922
Step 3	Selected features - Lemma	0.985	0.877	0.920
	Selected features - Next-lemma	0.990	0.872	0.920
	Selected features - Pre-POS	0.990	0.874	0.921
	Selected features - Next-POS	0.990	0.872	0.920
	Selected features - POS classified	0.990	0.874	0.921
	Selected features - Punctuation python	0.990	0.876	0.922
	Selected features - MatchesNegExp	0.990	0.876	0.922
	Selected features - **HasNegAffix**	0.990	0.881	0.925
Step 4	Selected features, Punctuation python, MatchesNegExp, HasNegAffix	0.989	0.881	0.924

Table 7: Feature ablation for features in the SVM classifier. All scores are macro average scores.

To conclude, the best features (in combination with an SVM classifier) to detect negation cues are: Token, Pre-token, Next-token, Pre-lemma, POS, Negated event, NegAffix and HasNegAffix

These features together contribute to an SVM classifier that outperforms the rule-based system. Additionally, it shows which features are unnecessary, and therefore can be eliminated

6.5.3 Parameter Optimization

Using Sklearn Grid search, we could identify the best parameters given our dataset and features. See our github repository for the accompanying code. The best parameters are:

- C: 1.0

- Loss: 'squared-hinge'
- tol: 0.0001

The best parameters are the same as the default setting of Linear SVC. Changing the parameter to 'hinge' gave the same results as with 'squared-hinge'. Changing the parameters (as can be seen in Table 8) resulted in a slight improvement on some of the outcomes for the system with all features. When applied to our best performing set of features, there is a decline in performance.

	SVM with all features	SVM with selected features
Loss = 'Hinge'	Precision: 0.987 , Recall: 0.879 , F1: 0.922	Precision: 0.990 , Recall: 0.768 , F1: 0.824
C = 0.5	Precision: 0.990 , Recall: 0.879 , F1: 0.924	Precision: 0.990 , Recall: 0.770 , F1: 0.825
tol = 0.001	Precision: 0.987 , Recall: 0.879 , F1: 0.922	Precision: 0.989 , Recall: 0.881 , F1: 0.924

Table 8: Alternative parameter choices and the impact on the macro average score.

6.6 Experiment on the test set

After developing and finetuning the SVM classifier, the classifier is run on two test data sets. This will provide an unbiased evaluation where the data is independent of the training set. As mentioned before, the test set is also from the SEM shared task team, containing the stories 'The Adventure of the Red Circle' and 'The Adventure of the Cardboard Box'. Beforehand, the test set was untouched and not looked into. For the test set, the same features were extracted in order to give the system the same input as before. The classifier consisted of the best features identified in section 6.5.2. Below in Table 9, the evaluation metrics are shown. As a reference, the evaluation on the development set is also shown. It can be seen that the classifier does not work well on the test set. Especially in classifying I-NEG labels, the system does not perform at all. The 'B-NEG' and 'O' scores are pretty similar to the development set. This would suggest that the low macro average score is primarily due to low I-NEG scores.

	SVM best features (development data)			**SVM best features (test data, circle)**			**SVM best features (test data, cardboard)**		
	P	R	F1	P	R	F1	P	R	F1
B-NEG	0.972	0.977	0.975	0.957	0.985	0.971	0.963	0.970	0.967
I-NEG	1.000	0.667	0.800	0.000	0.000	0.000	0.000	0.000	0.000
O	1.000	1.000	1.000	0.999	0.999	0.999	1.000	1.000	1.000
Accuracy			0.999			0.999			0.999
Macro avg	0.990	0.881	0.925	0.652	0.662	0.657	0.654	0.657	0.655
Weighted avg	0.999	0.999	0.999	0.998	0.997	0.998	0.999	0.999	0.999

Table 9: Evaluation metrics of the SEM test set

6.7 Conclusions of experiments

During the process of the experiment, we encountered some problems. First, the rule-based baseline system itself has a relatively strong performance. Beating such a strong baseline is a challenge. Based on the experimental results, it can be seen that picking the right features and using more advanced machine learning algorithms can indeed improve negation cue detection even further. Second, an extremely unbalanced dataset is a big challenge. In the development dataset, there are only three 'I-NEG' labels. This has the potential to skew our results. When testing on the SEM development set, we have generally seen high results for all three classes. But we see that especially when making predictions on the BioScope data, the imbalance in data leads to a decline in the 'I-NEG' class. Thirdly, our experiment with the BioScope data brought to light that our systems are not entirely generalizable at this point in time. This is largely due to the fact that the two datasets have been annotated according to different annotation guidelines, and therefore have different labels for the same occurrences. Especially for the rule-based

	B-NEG	I-NEG	O
Errors	Lexical: 4 Affixal: 0 Total: 4	Lexical: 1 Affixal: 0 Total: 1	Total: 5
Matches	Lexical: 139 Affixal: 33 Total: 172	Lexical: 2 Affixal: 0 Total: 2	Total: 13383

Table 10: Error and matching statistics of the SEM development set.

system, our experiment with a different dataset caused a significant drop in performance. The conditions for the rule-based system do not perform well on data that they have not been specifically fine-tuned on.

From our experiments, we can conclude that our machine learning approach is able to outperform the rule-based approach with the following set of features: Token, Pre-token, Next-token, Pre-lemma, Part Of Speech token, Negated event, NegAffix and HasNegAffix. We also see that the SVM generalizes better over different types of data.

7 Error Analysis

In this section, an error analysis will be performed on the development dataset. In order to do so, two error categories were chosen based on Lapponi et al. (2012), being *false positives* and *false negatives*. False positives indicate errors that are due to predicting an event for a non-factual context, which means predicting negation cues which are not actual negations based on the context. False negatives are errors which are due to not predicting an event given the context. In this case, given the context the system wrongly classified a token as a negation.

7.1 Overview of errors on development set

As can be seen in Table 10 and 11, there are ten errors identified which is 0.07% of the entire dataset (10/13567). All errors are lexical, and no affixal errors were found (see Table 10). Five errors are false positives where the system identifies the token as a negation instead of an 'O'. Given the context, the system should have identified the tokens as part of a fixed expression and thus a false negation. Three cases could be identified as being part of a fixed expression (i.e., none the less) where part of the fixed expression is a false negation cue. The system does not seem to capture the context. Two other cases are also non negations given the context (i.e., 'never' and 'no'), where our system cannot distinguish between a negation and a false negation.

7.2 Sources and patterns of errors

The development set was checked in order to find matching cases, where the system did perform on the same given examples. It was found that the development set only contained three 'none the less' instances. All 'none' tokens of these three instances were predicted incorrectly as 'B-NEG' instead of 'O'. With regards to the 'no' tokens, 40 'no' tokens were found in the development set. Of these tokens, five were annotated as 'O'. Only one is predicted incorrectly by our classifier. The token 'never' is found twelve times in the development set itself. Only one 'never' is annotated as 'O', which is also classified incorrectly by our system. The other eleven instances were predicted correctly by the system.

Other mistakes were identified by means of false negatives, where the system does not recognize a negation cue as a negation. It assigns an 'O', whereas it should have assigned an 'I-NEG' or 'B-NEG'. Regarding the first error of 'by' in 'by no means', the system incorrectly predicted 'by' as 'O'. In the development set itself, there was only one instance of 'by no means' which was incorrectly predicted. In the training set, there were however three instances of 'by no means' which were all annotated as negations. 'By' is a word that occurs often in language and in the training set (190 instances of 'by' on its own) with 187 instances with 'O' as label, which could bias the system into a certain direction.

The next false negative is the token 'nothing' (see Table 11, index 7). When looking at the development set, 16 instances of 'nothing' can be found, which are all annotated as negations. So only one instance is predicted incorrectly. The token 'not' was incorrectly assigned an 'O' by our system in one case, whereas 2 other cases were annotated and predicted correctly as 'O'. A total of 45 instances of 'not' could be found in the entire development set of which 42 were negations. Next, 'more' in the context of 'no more' should have been predicted as a negation. In the entire development dataset, a total of 12 'no more' could be found, where 11 were correctly predicted as 'O'. So our system only incorrectly predicted one instance. The very last example is 'save', which occurs five times in the development set. Only one time, 'save' is annotated as a negation. At first, it was assumed that this was an error, but the guidelines showed the exact example with 'save' as negation cue.

Even though few errors are made, it is still interesting to see that most errors are connected to fixed expressions. In order to find where the mistakes might come from, the training set was explored. For some errors it was found that these fixed expressions were not represented in the training data (i.e., none the less). Other errors with regards to fixed expressions were due to the fact that parts of a fixed expression ('by' in 'by no means') are individually represented as 'O'. In combination with under-representation of examples of fixed expressions, the systems assigns 'O' for the individual token. The same is true the other way around, over-representation of 'nothing' as a negation results in a system that will not assign 'O' to that token.

Somewhat the same patterns can be found in the errors of the test set (see Table 17 in the appendix), but there are some differences. Affixal errors were found in the test set, which could be possible due to the fact that these tokens were not in the positive negation list used for the HasNegAffix feature. Therefore, the system could not have recognized these tokens as a negation. The other errors all involve fixed expressions that either are negations or non-negations depending on context, such as 'absolutely nothing'. Other errors were due to lack of examples of the training data, such as 'far from' which is not represented in the training data, and therefore the system cannot recognize this word combination as negation. Due to these fixed expressions, not all 'I-NEG' classes could be identified and therefore the overall results were lower compared to the development set.

To conclude, minor mistakes were made by our system. The errors that were made on the dev set all concerned lexical negations, both false positives and false negatives. Most errors were found in false negations in fixed expressions, such as 'none the less' or 'by no means'. This indicates that our system does not always take context into account correctly. This is mostly due to the representation of tokens in the training set. Fixed expressions are often underrepresented in the training data. Also, a lot of tokens that often belong to a fixed expression, also can be used on their own. On their own, they are often annotated as 'O'. Due to under-representations and overshadowing of individual tokens, the system has difficulty assigning the right class to fixed expressions.

A solution could be to add token n-grams as a feature, with a window size of 2 tokens. Another feature that might be helpful in combination with the n-grams is a lexicon of fixed negation expressions. The lexicon could contain one list for all fixed expressions that are negations, and one list that contains all fixed expressions that are false negations. These lists in combination with a broader n-gram window could benefit the system.

8 Conclusions

In the following chapter, we will summarize the main points of our experiment, reflect on the results and provide suggestions for further research.

8.1 Summary

This report explores several approaches for negation cue detection, and determines the best approach based on three experiments: a comparison of algorithms, a feature ablation study and an experiment with a different data set. For the comparison of algorithms, a machine learning approach (SVM) and a

Index	Error	Context	Gold	Prediction	Error category
1	none	'.... whom I have absolutely no sympathy, but **none** the less, having heard your name–"	O	B-NEG	False positive
2	none	As impassive as ever to the casual observer, there were **none** the less a subdued eagerness ...	O	B-NEG	False positive
3	none	..., it has **none** the less presented surprising difficulties ...	O	B-NEG	False positive
4	no	I desire you to spare no expense and **no** pains to get at the truth."	O	B-NEG	False positive
5	never	A nobler man **never** lived upon earth.	O	B-NEG	False positive
6	by	"Remarkable, **by** no means impossible," ...	B-NEG	O	False negative
7	nothing	Apparently the tenants had brought little or **nothing** with them, ...	B-NEG	O	False negative
8	save	**Save** for this one excursion, he spent his days in long and often solitary walks, or ...	B-NEG	O	False negative
9	not	We could **not** doubt that justice,...	B-NEG	O	False negative
10	more	From that day they were seen no **more** in England.	I-NEG	O	False negative

Table 11: Error analysis of the development set.

self-created rule-based system have been chosen. In order to sufficiently compare the two approaches, first a feature ablation and parameter optimization had to be done on the SVM to find the best features for our task. Then we compared our rule-based system with the best feature set on SVM. Lastly, we ran both of our algorithms on a data set that was less comparable to the original train and dev sets, to see how generalizable our systems are over various types of data. Both of our systems detect three classes: B-NEG, I-NEG and O. And two types of negation cues: lexical and affixal.

The rule-based system has been run on the training set, and on the dev set. It performs at a macro average F1-score of 0.911 on the training set and 0.825 on the dev set. The SVM with all features has a macro average F1-score of 0.922. The SVM with our best feature selection performed with a macro average F1-score of 0.924. The SVM classifier has the highest performance throughout our entire experiment with the following selection of best features: Token, Pre-token, Next-token, Pre-lemma, POS, Negated event, NegAffix and HasNegAffix.

8.2 Discussions

Over all systems, the I-NEG class performs the worst, and the O class has the highest performance. This has a clear explanation in the imbalance of the data set, the O's were highly overrepresented whereas some data sets only contained three instances of I-NEG. This resulted mostly in a very low recall for this class, even though the precision was often quite high or even perfect. Therefore, to further improve negation cue detection across our systems, specific attention needs to be put on improving the recall of the I-NEG class.

We have also seen, from our experiment with the BioScope dataset, that our systems do not generalize well over different types of data at this point. The SVM classifier outperforms the rule-based system in this regard, but it still performs considerably lower when trained on the SEM dataset. We think this is largely due to the fact that the BioScope data is annotated according to different annotation guidelines compared to the SEM datasets. Ideally, consistency is sought when annotating in order to generalize over datasets from different sources. Therefore, we are advocates of datasets being annotated using

one annotation guideline. This would also mean that the annotation guideline to be used needs to be complete. Taking into account section 3, it was shown that even annotators using the same guideline can result in low Inter-Annotator-Agreement scores, due to the fact that given examples are not sufficient. It is strongly recommended to extend the guidelines of Morante et al. (2011) to avoid doubt about ambiguous phrases in texts.

Errors that were made by our system were mostly due to fixed expressions which the system could not identify as the correct label. This was the case for parts of fixed expressions being negations, as well as fixed expressions containing a negation without being an actual negation. This is due to the fact that there are a lack of examples regarding fixed expressions in the training set. Also, the overrepresentation of the 'O' class biased the system into thinking that parts of fixed expressions should also be labeled as 'O' instead of a negation.

8.3 Future Research

Here we advise on future research possibilities and provide potential solutions for some of the problems we have encountered.

8.3.1 Systems

Previous research showed that the use of algorithms such as CRF and BERT can be beneficial for the entire task of negation detection and scope resolution (Khandelwal and Sawant, 2020). We suggest adding a CRF and/or and Deep Learning algorithm to the experiment for a more complete comparison of approaches. That way, a more definitive statement can be made about which approach is better for this data, and for our specific task of negation cue detection.

8.3.2 Preprocessing

As stated in the data description, observing, preprocessing, and describing datasets in detail is one of the key steps in tackling natural language processing tasks. The training data used in our experiment was overrepresented by "O" gold labels and underrepresented by I-NEG labels. A critical revision of the training data, e.g. removing some "O" classified tokens or trying to increase the "I-NEG" labels in training, might very well improve overall performance.

Another recommendation would be to use data from different sources, and from different timelines as a training set. In the current report, the system was trained, developed and tested on stories written by Conan Doyle. An author from the early 1900's, which probably has different language use compared to other datasets.

Aside from that, one might want to explore different preprocessing tools to test the effect on the performance on the datasets. For example, it might be that Stanza has better results for part-of-speech tagging than the used Spacy tool. A complete study on preprocessing tool performance was beyond the scope of this report but better performing preprocessing tools might improve overall results.

There was also a small challenge with preprocessing the BioScope data. The data had commas and full stops trailing the tokens. These posed an issue with feature extraction. We removed them because the annotated labels just considered the token on that line, and not the punctuation. This might have had an influence on the system although we cannot state so for certain. We suggest a more thorough approach to resolve this potential issue.

8.3.3 Features

Because most of our errors stem from wrong classification of tokens in the I-NEG class, the features mentioned here are all focused on potential improvement of that class. We believe that the usage of features that capture context would further increase our system performance. Potential features might include, but are not limited to; syntactic dependencies, token bigrams and a lexicon of fixed negation expressions.

Features that capture syntactic dependencies might very well help with identifying fixed word expressions to increase the I-NEG class scores. They could also help the system to learn more exceptions, as some tokens are only a negation cue in certain contexts. As a last suggestion, the use of token n-grams,

together with a lexicon of fixed negation expressions, could be very helpful for the system. This would give the system the ability to recognize certain sequences of tokens as a fixed negation expression. This might increase the identification of multi-word negation cues in the I-NEG class.

Appendix 1: distribution of work for assignment 1

Sharona Work done:

- Researched and discussed negation features with Michiel.
- Collected papers on features and tools and shared with the group.
- Wrote features section of report.
- Checked and discussed everyones work with the group.
- Added group report draft to overleaf.

Michiel Describe work done.

- Met, researched and discussed negation features with Sharona.
- Collected papers on related work, features and tools and shared with the group
- Highlighted all provided and extra papers and shared with group.
- Wrote related work section of report.
- Checked and discussed everyone's work with the group.
- Checked report before handing in.

Desiree Work done:

- Wrote task description (together with Jingyue).
- Read articles about negation detection and features.
- Revised/checked the Related Work section.
- Added references to the task description and to a part of the related work section.

Jingyue Work done.

- Wrote task description (together with Desiree).
- Searched for papers on negation cue and features.
- Discussed our task with the group.
- Checked report before handing in.

Appendix 2: distribution of work for assignment 2

Sharona Work done:

- Studied the guidelines and annotated 10 documents.
- Discussed error analysis with my annotator pair: Desiree.
- wrote error analysis section of report (together with Desiree).
- Added group report draft to overleaf.
- Created Tables in the right format.

Michiel Work done:

- Read the annotation guidelines.
- Annotated 10 documents.
- Generated IAA report for Jingyue and myself (IOS).
- Discussed with Jingyue which Table to use. Choose mine as Desiree generated on Windows.
- Compared the annotation results with Jingyue and analysed the results.

Desiree Work done:

- Read the annotation guidelines (Morante et al., 2011).
- Annotated 10 documents.
- Wrote the annotation task description.
- Generated the IAA report for annotations between me and Sharona.
- Generated the IAA report for annotations between Jingyue and Michiel.
- Met with Sharona to discuss the results and defined patterns of agreement and disagreement.

Jingyue Work done:

- Read the annotation guidelines.
- Annotated 10 documents.
- Tried to calculate IAA but failed to generate report, Desiree helped me in this part.
- Added some sentences on annotated corpus description.
- Wrote the reflection on the annotation task.
- Compared the annotation results with Micheal's and analysed the results.

Appendix 3: distribution of work for assignment 3

Sharona Work done:

- Made a feature split and assigned to group
- Collected information on affixal features
- Extracted the following features: previous token, next token, matchesNegExp, hasNegaffix, Negated event and Negational affix.
- Described the features: matchesNegExp, hasNegaffix, Negated event and Negational affix.
- Wrote preprocessing tools section.
- Compiled and commented feature_extraction.py

Michiel Work done:

- Found and highlighted related work for data description
- Coded two features: punctuation, POS-classification.
- Linguistic research for POS classification and conversion.
- Generated Table for POS classification and put them in overleaf.
- Generated Table for features with basic, short, description.
- discussed as much as possible with group to make report and division.

Desiree Work done:

- Wrote the data description together with Jingyue.
- Coded three features: lemma, previous lemma and next lemma.
- Calculated the basic statistics in python.
- Generated graphs and put them in overleaf.
- Put all the Tables in overleaf.
- Checked work of the group members.

Jingyue Work done:

- Wrote the data description together with Desiree.
- Wrote intro for chapter 4 and 5.
- Extracted and described three features: POS tag, previous and next POS tag.
- Calculated part of the basic statistics in python.
- Put chapter5 in overleaf.

Appendix 4: distribution of work for assignment 4

Sharona Work done:

- Created and described rule-based lexicon
- Extracted features from bioscope corpus and uploaded the .py file
- Compared and wrote down results between SEM and bioscope data
- Analysed and concluded experiments between systems
- Updated readme and cleaned up github
- Checked work of group members.

Michiel Work done:

- Motivated classifier approach and narrative
- Wrote introduction chapter 6
- Wrote system description
- Tried to find anything to make BERT work. Did not succeed
- Wrote description for BERT but was taken out as BERT did not perform
- Put most of chapter 6 in overleaf
- Added references to text throughout
- Checked work of the group members.

Desiree Work done:

- Created the SVM code.
- Described the SVM classifier of section 6.4.1.
- Performed feature ablation + coding + description.
- Performed parameter optimization.
- Tested SVM classifier on BioScope data.
- Checked work of the group members.

Jingyue Work done:

- Tried coding CRF, did not work out.
- Ran the SVM and baseline systems.
- Compared the performance of SVM with baselines'.
- Tried coding Bert, did not work out.
- Wrote part of conclusion of experiments.
- Put draft in overleaf
- Checked work of the group members.

Appendix 5: distribution of work for assignment 5

Sharona Work done:

- Extracted errors and matching tokens from our experiments for analysis
- Extracted statistics on the errors and matches and made a table format to put them in
- Implemented the code feedback
- Worked on error analysis with group
- Wrote part of the summary, conclusion and discussion sections
- Finished writing the abstract

Michiel Work done:

- Processed all comments of previous assignments.
- Wrote future work section.
- Took care of cohesiveness of report.

Desiree Work done:

- Explored the errors for error analysis.
- Wrote the error analysis description.
- Tested the SVM on the test set, and added the results plus description to section 6.
- Helped with writing the discussion.
- Put tables in overleaf + parts of the texts.
- Processed comments from assignment 4.

Jingyue Work done:

- Explored the errors for error analysis
- Created tables for error analysis
- Proofread report

Appendix 6: Tables and Figures

POS-tag	Training set	Development set
NN (noun)	7,732	1,680
IN (preposition)	7,016	1,370
DT (determiner)	5,788	1,198
PRP (pronoun; personal)	5,708	1,094
VBD (verb; past tense)	3,423	780
JJ (adjective; numeral)	3,361	758
RB (adverb)	3,017	545
NNP (noun; proper; singular)	2,331	592
VB (verb; base form)	2,236	416
CC (conjunction)	2,029	425
PRP$ (pronoun; possessive)	1,653	373
NNS (noun; common plural)	1,515	327
VBN (verb; past participle)	1,420	361
VBP (verb; present tense)	1,310	246
TO (to; preposition or infinitive)	1,272	257
MD (modal auxiliary)	1,104	200
VBZ (verb; present tense)	1,097	204
VBG (verb; present participle)	743	104
WDT (WH-determiner)	407	69
WP (WH-pronoun)	373	75
CD (numeral)	366	82
WRB (WH-adverb)	361	64
RP (particle)	298	68
EX (existential)	242	44
POS (genitive marker)	155	31
JJR (adjective; comparative)	135	16
RBR (adverb; comparative)	90	11
UH (interjection)	84	13
PDT (predeterminer)	83	16
JJS (adjective; superlative)	72	11
RBS (adverb; superlative)	42	10
FW (foreign word)	10	0
NNPS (proper noun; plural)	9	10
WP$ (possessive WH-pronoun)	6	8

Table 12: Distribution of POS tags across and within training and development datasets.

POS-tag	Training set	Development set
VBP	3	0
RB	511	77
JJ	123	25
DT	168	33
IN	34	8
NN	81	19
NNP	42	2
UH	19	2
CC	9	8
VB	3	1
VBN	3	2
CD	1	0
VBD	3	0
NNS	3	1

Table 13: Distribution of POS tags for negations in the training and development datasets.

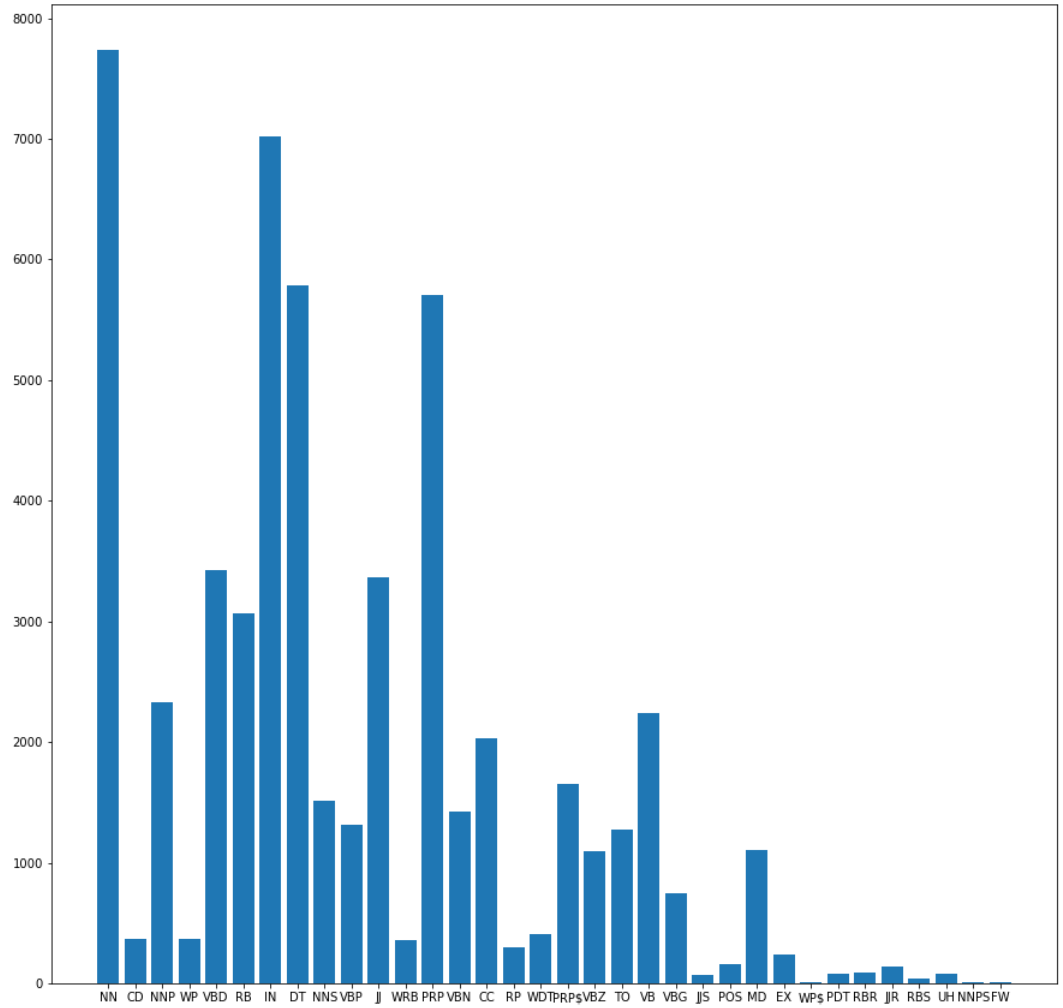

Figure 2: The distribution of POS-tags for the training dataset.

POS-tag	**Converted class**
WRB: Wh-adverb	ADVB
WP: WH-pronoun	PRO
WDT: WH-determiner	OTH
VBZ: verb, present tense, 3rd person singular	VB
VBP: verb, present tense, not 3rd person singular	VB
VBN: verb, past participle	VB
VBG: verb, present participle or gerund	VB
VBD: verb, past tense	VB
UH: interjection	OTH
TO: ”to” as preposition or infinitive marker	OTH
RP: particle	ADVB (subject to change)
RBS: adverb, superlative	ADVB
RBR: adverb, comparative	ADVB
RB: adverb	ADVB
PRP(dollar): pronoun, possessive	PRO
PRP: pronoun, personal	PRO
POS: genitive marker	PRO
PDT: pre-determiner	OTH
NNS: noun, common, plural	NN
NNP: noun, proper, singular	NN
NN: noun, common, singular or mass	NN
MD: modal auxiliary	VB
LS: list item marker	OTH
JJS: adjective, superlative	ADJ
JJR: adjective, comparative	ADJ
JJ: adjective or numeral, ordinal	ADJ
IN: preposition or conjunction, subordinating	ADJ
EX: existential there	OTH
DT: determiner	OTH
CD: numeral, cardinal	OTH
CC: conjunction, coordinating	ADVB (subject to change)

Table 14: Course-grained classification of NLTK POS tags.

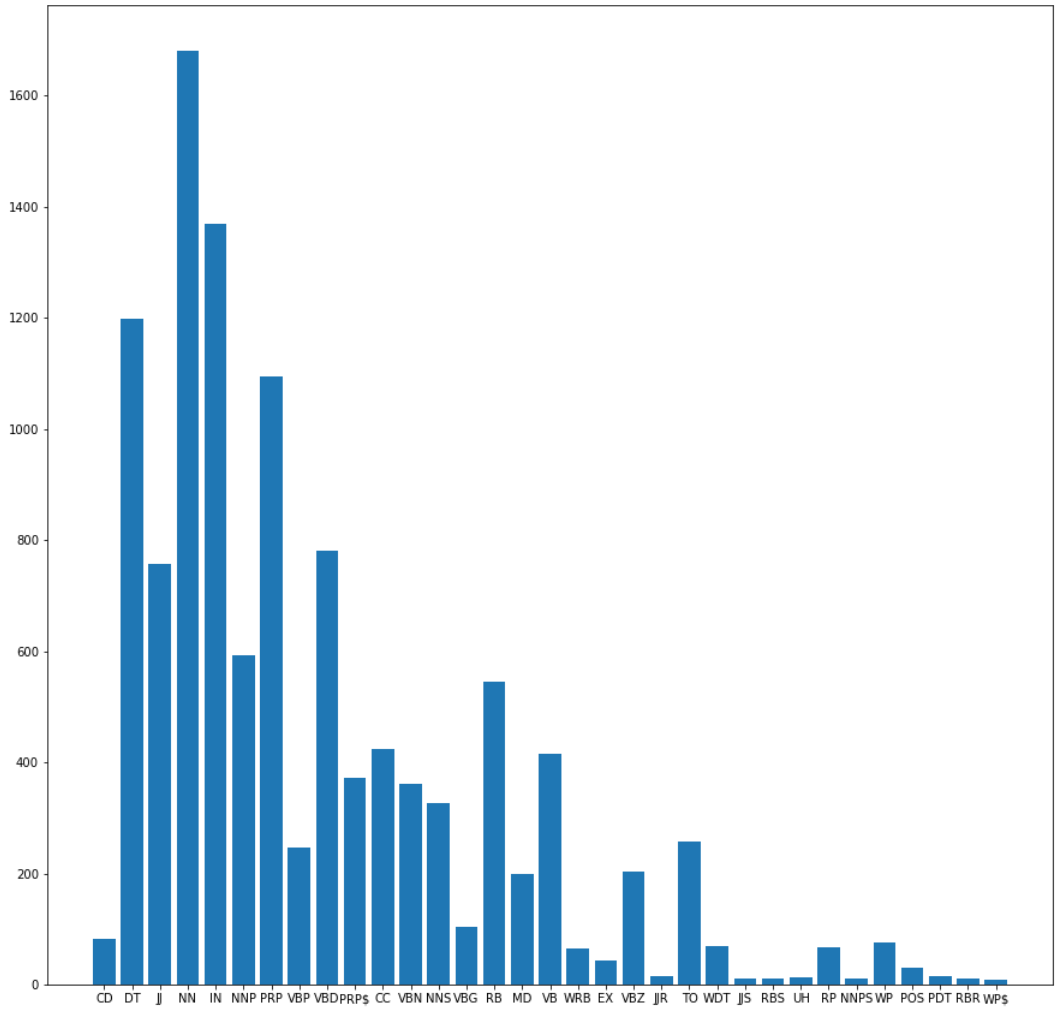

Figure 3: The distribution of POS-tags for the development dataset.

Parameter	Value	Description
Penalty	'l1', 'l2' (default='l2')	This parameter specifies the norm used in the penalization
Loss	'hinge', 'squared hinge' (default='squared hinge')	Specifies the loss function
Dual	True or False (default=True)	Select the algorithm to solve dual or primal optimization
Tol	Float (default=1e-4)	Tolerance for stopping criteria
C	Float (default=1.0)	Regularization parameter
Multi class	'ovr', 'crammer singer' (default='ovr')	Determines the multi-class strategy
fit intercept	True or False (default=True)	Whether to calculate the intercept for this model
intercept scaling	Float (default=1.0)	A parameter which lessens the effect of regularization
Class weight	Dict or auto (optional parameter)	Optional parameter that specifies what weights to assign to classes
Verbose	Int (default=0)	Enables more elaborate/verbose output
Random state	Int, RandomState instance or None (default=None)	Controls for random number generations for shuffling the data
max iter	Int (default=1000)	The maximum number of iterations to be run

Table 15: LinearSVC parameters (Pedregosa et al., 2011)

	B-NEG	I-NEG	ALL NEGS	O
Error statistics development set				
Errors	Lexical: 4 Affixal: 0 Total: 4	Lexical: 1 Affixal: 0 Total: 1	Lexical: 5 Affixal: 0 Total: 5	Total: 5
Matches	Lexical: 139 Affixal: 33 Total: 172	Lexical: 2 Affixal: 0 Total: 2	Lexical: 141 Affixal: 33 Total: 174	Total: 13383
Error statistics test set - cardboard				
Errors	Lexical: 2 Affixal: 2 Total: 4	Lexical: 1 Affixal: 0 Total: 1	Lexical: 3 Affixal: 2 Total: 5	Total: 5
Matches	Lexical: 110 Affixal: 20 Total: 130	Lexical: 0 Affixal: 0 Total: 0	Lexical: 110 Affixal: 20 Total: 130	Total: 10044
Error statistics test set - circle				
Errors	Lexical: 2 Affixal: 0 Total: 2	Lexical: 4 Affixal: 0 Total: 4	Lexical: 6 Affixal: 0 Total: 6	Total: 6
Matches	Lexical:119 Affixal: 14 Total: 133	Lexical: 0 Affixal: 0 Total: 0	Lexical: 119 Affixal: 14 Total: 133	Total: 8887

Table 16: Error statistics on SEM development set and SEM test sets.

Index	Error	Context	Gold	Prediction	Error category
Test set - cardboard					
1	incredulity	. . . the same thing you expressed **incredulity**.	B-NEG	O	False negative
2	Undoubtedly	. . . , he would **undoubtedly** have done old address.	B-NEG	O	False negative
3	not	"Why **not**?" says she.	B-NEG	O	False negative
4	far	But I was still **far** from satisfied.	B-NEG	O	False negative
5	from	But I was still far **from** satisfied.	I-NEG	O	False negative
6	nothing	. . . , half-pound honeydew box, with **nothing** distinctive. . .	O	B-NEG	False positive
7	without	I'm never **without** one or the other before me.	O	B-NEG	False positive
8	never	. . . if that woman had **never** darkened our door.	O	B-NEG	False positive
9	without	'Can't you be happy for five minutes **without** Mary, Jim? . . .	O	B-NEG	False positive
10	nothing	. . . , and we had ceaseless rows about **nothing**.	O	B-NEG	False positive
Test set - circle					
1	absolutely	Do you say nothing has come out of that room– **Absolutely** nothing?	B-NEG	O	False negative
2	nothing	Do you say nothing has come out of that room– Absolutely **nothing**?	I-NEG	O	False negative
3	more	'If not, I 'll have no **more** to do with you.'	I-NEG	O	False negative
4	more	. . . from the house which he was never **more** to enter.	I-NEG	O	False negative
5	more	" I'll have no **more** of it!	I-NEG	O	False negative
6	n't	If we wait a little, Watson, i do**n't** doubt that the affair . . .	B-NEG	O	False negative
7	n't	'Danger, 'Is**n't** it?	O	B-NEG	False positive
8	not	Is it **not** so? "	O	B-NEG	False positive
9	not	Why **not** write?	O	B-NEG	False positive
10	not	. . . to punish those whom they feared or hated by injuring **not** only their own persons but those whom they loved,...	O	B-NEG	False positive
11	not	**Not** only was his body that of a giant but everything . . .	O	B-NEG	False positive
12	prevent	. . . should be made of them as would **prevent** any other victim from rebelling.	O	B-NEG	False positive

Table 17: Error analysis on two SEM test sets.

References

Amjad Abu-Jbara and Dragomir Radev. 2012. Umichigan: A conditional random field model for resolving the scope of negation. 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)*, pages 328–334.

Ron Artstein. 2017. Inter-annotator agreement. In *Handbook of linguistic annotation*, pages 297–313. Springer.

Miguel Ballesteros, Alberto Díaz, Virginia Francisco, Pablo Gervás, Jorge Carrillo De Albornoz, and Laura Plaza. 2012. Ucm-2: a rule-based approach to infer the scope of negation via dependency parsing. 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)*, pages 288–293.

Valerio Basile, Bos Johan, Evang Kilian, and Venhuizen Noortje. 2012. Ugroningen: Negation detection with discourse representation structures. In *First Joint Conference on Lexical and Computational Semantics (* SEM)*, pages 301–309. Association for Computational Linguistics.

Steven Bird, Ewan Klein, and Edward Loper. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit*. " O'Reilly Media, Inc.".

Eduardo Blanco, Roser Morante, and Roser Saurí, editors. 2016. *Proceedings of the Workshop on Extra-Propositional Aspects of Meaning in Computational Linguistics (ExProM)*, Osaka, Japan, December. The COLING 2016 Organizing Committee.

Jean-Baptiste Boin, Nat Roth, Jigar Doshi, Pablo Llueca, and Nicolas Borensztein. 2020. Multi-class segmentation under severe class imbalance: A case study in roof damage assessment. *arXiv preprint arXiv:2010.07151*.

Bernhard E Boser, Isabelle M Guyon, and Vladimir N Vapnik. 1992. A training algorithm for optimal margin classifiers. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 144–152.

Wendy W Chapman, Will Bridewell, Paul Hanbury, Gregory F Cooper, and Bruce G Buchanan. 2001. A simple algorithm for identifying negated findings and diseases in discharge summaries. *Journal of biomedical informatics*, 34(5):301–310.

Md Faisal Mahbub Chowdhury. 2012. Fbk: Exploiting phrasal and contextual clues for negation scope detection. 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)*, pages 340–346.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning*, 20(3):273–297.

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 ** 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)*, pages 282–287.

Martine Enger, Erik Velldal, and Lilja Øvrelid. 2017. An open-source tool for negation detection: a maximum-margin approach. In *proceedings of the workshop computational semantics beyond events and roles*, pages 64–69.

Federico Fancellu, Adam Lopez, and Bonnie Webber. 2016. Neural networks for negation scope detection. In *Proceedings of the 54th annual meeting of the Association for Computational Linguistics (volume 1: long papers)*, pages 495–504.

Aditya Khandelwal and Suraj Sawant. 2019. Negbert: A transfer learning approach for negation detection and scope resolution. *arXiv preprint arXiv:1911.04211*.

Aditya Khandelwal and Suraj Sawant. 2020. Negbert: A transfer learning approach for negation detection and scope resolution.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

Emanuele Lapponi, Erik Velldal, Lilja Øvrelid, and Jonathon Read. 2012. Uio 2: sequence-labeling negation using dependency features. 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)*, pages 319–327.

Lydia Lazib, Yanyan Zhao, Bing Qin, and Ting Liu. 2019. Negation scope detection with recurrent neural networks models in review texts. *International Journal of High Performance Computing and Networking*, 13(2):211–221.

Roser Morante and Eduardo Blanco. 2012. * sem 2012 shared task: Resolving the scope and focus of negation. 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)*, pages 265–274.

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-2009)*, pages 21–29.

Roser Morante, Sarah Schrauwen, and Walter Daelemans. 2011. Annotation of negation cues and their scope: Guidelines v1. *Computational linguistics and psycholinguistics technical report series, CTRS-003*, pages 1–42.

Roser Morante. 2010. Descriptive analysis of negation cues in biomedical texts. In *LREC*.

Pradeep G Mutalik, Aniruddha Deshpande, and Prakash 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*, 8(6):598–609.

Ying Ou and Jon Patrick. 2015. Automatic negation detection in narrative pathology reports. *Artificial intelligence in medicine*, 64(1):41–50.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Yifan Peng, Xiaosong Wang, Le Lu, Mohammadhadi Bagheri, Ronald Summers, and Zhiyong Lu. 2018. Negbio: a high-performance tool for negation and uncertainty detection in radiology reports. *AMIA Summits on Translational Science Proceedings*, 2018:188.

Zhong Qian, Peifeng Li, Qiaoming Zhu, Guodong Zhou, Zhunchen Luo, and Wei Luo. 2016. Speculation and negation scope detection via convolutional neural networks. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 815–825.

Lior Rokach, Roni Romano, and Oded Maimon. 2008. Negation recognition in medical narrative reports. *Information Retrieval*, 11(6):499–538.

*SEM. 2012. `http://ixa2.si.ehu.eus/starsem/index.php.html`.

Brett South, Shuying Shen, Jianwei Leng, Tyler Forbush, Scott DuVall, and Wendy Chapman. 2012. A prototype tool set to support machine-assisted annotation. In *BioNLP: Proceedings of the 2012 Workshop on Biomedical Natural Language Processing*, pages 130–139.

SpaCy. 2016. `https://www.nltk.org`.

Bowei Zou, Guodong Zhou, and Qiaoming Zhu. 2014. Negation focus identification with contextual discourse information. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 522–530.