# Automatic Claim Negation: Why, How and When

**Yonatan Bilu**
IBM Haifa Research Lab
Mount Carmel
Haifa, 31905, Israel
yonatanb@il.ibm.com

**Daniel Hershcovich**
IBM Haifa Research Lab
Mount Carmel
Haifa, 31905, Israel
danielh@il.ibm.com

**Noam Slonim**
IBM Haifa Research Lab
Mount Carmel
Haifa, 31905, Israel
noams@il.ibm.com

## Abstract

The main goal of argumentation mining is to analyze argumentative structures within an argument-rich document, and reason about their composition. Recently, there is also interest in the task of simply detecting claims (sometimes called conclusion) in general documents. In this work we ask how this set of detected claims can be augmented further, by adding to it the negation of each detected claim. This presents two NLP problems: how to automatically negate a claim, and when such a negated claim can plausibly be used. We present first steps into solving both these problems, using a rule-based approach for the former and a statistical one towards the latter.

## 1 Introduction

In Monty Python's famous Argument Clinic Sketch (Chapman and Cleese, 1972), Michael Palin is seeking a good argument, and John Cleese, too lazy to provide a real argument, simply contradicts whatever Mr. Palin is saying.

> MP: An argument isn't just contradiction.
> JC: Well! it *can* be!
> MP: No it can't! An argument is a connected series of statements intended to establish a proposition.
> JC: No it isn't!
> MP: Yes it is! It isn't just contradiction!
> JC: Look, if I *argue* with you, I must take up a contrary position!
> MP: Yes, but it isn't just saying 'no it isn't'.

In this work we aim to explore this last statement from the perspective of an automatic system, aiming to refute an examined claim. Specifically, given a claim, *how* should we contradict it? Is it enough to say "No it isn't", or is a more complicated algorithm required? And *when* can we plausibly use an automatically generated contradiction? When would it be considered a valid counter claim, and when would it seem as an even less comprehensible version of John Cleese? The answers to these questions turn out to be less simple than one might expect at first glance.

The main goal of argumentation mining is to analyze argumentative structures within a document. Typically, documents in which such structures are abundant, such as from the legal domain (Mochales Palau and Moens, 2011; Bach et al., 2013; Ashley and Walker, 2013; Wyner et al., 2010), are analyzed, and compound argumentative structures, or argumentation schemes, are sought (Walton, 2012).

More recently, there is also interest in automatically detecting simple argumentative structures, or the building blocks of such structures, in documents which are not argumentative by nature. For example, in (Aharoni et al., 2014; Levy et al., 2014) it was shown that context-dependent Claims and Evidences (sometimes called Conclusion and Grounds, respectively) are fairly common in Wikipedia articles, and can be detected automatically. In this setting, detection is done within a given context of a pre-specified debatable topic. Then, the objective is to search a given set of documents, and mine Claims and Evidence pertaining to this topic.



*Proceedings of the 2nd Workshop on Argumentation Mining*, pages 84–93,
Denver, Colorado, June 4, 2015. 

One motivation for such context-dependent argumentation mining is that it serves as the first component in a debate-support system. In a second stage, Claims and Evidence can be combined into full fledged Arguments, highlighting to the user the various opinions surrounding the debatable topic.

In order to provide a comprehensive view of these various opinions, it might not be sufficient to rely on the initial set of detected argumentative elements. For example, for practical reasons, an automatic Claim detection system as in (Levy et al., 2014) will present to the user only its top scoring predictions, which will probably represent only a subset of the relevant Claims. Furthermore, the examined corpus might be biased, hence enriched with claims supporting only one side of the debate. Thus, it is of interest to augment the initial set of predictions made by such systems through various means.

Here, motivated by the observation that negating previous arguments has an important function in argumentation (Apothéloz et al., 1993), we suggest a system to augment a given set of relevant Claims by automatically suggesting a meaningful negation per mentioned Claim. More specifically, we require that the automatically suggested negation will be not only grammatically correct, but also plausible to use in a discussion about the given topic. As we discuss and demonstrate, this latter requirement poses a non-trivial challenge. Accordingly, we propose a Machine Learning approach that exploits NLP-based features in order to determine if it is plausible to use the suggested negation. Our results demonstrate the feasibility and practical potential of the suggested approach.

## 2 Related work

Negation detection has received much attention in NLP. This includes the detection of negated clauses and subclauses, and of negation expressions and of their scope. Methods employed in this detection include conditional random fields (CRFs) (Councill et al., 2010), regular expressions (Chapman et al., 2013), and syntactic rules (Lotan et al., 2013; Mutalik et al., 2001). Negation detection is critical for medical applications, for example, in order to classify whether the text contains the existence or absence of a condition (Chapman et al., 2013). It was also shown to improve the results in sentiment analysis (Wiegand et al., 2010; Zhu et al., 2014), as negation alters the sentiment of the text in its scope. Despite these results, it is not trivial, in general, to infer the meaning of a negated utterance, or what is in fact negated—it depends on the focus of the original sentence (Blanco and Moldovan, 2011). By contrast, here we deal with negating typically short claims (12 words long on average), where focus and scope are usually relatively easy to infer.

Several works have tackled the task of surface realization—the transformation from a logical representation to human-readable text—providing systems such as SimpleNLG (Gatt and Reiter, 2009). However, these earlier works do not provide a principled method of negating existing sentences given as free text statements.

To our knowledge, there has been just one work on generating the negations of existing sentences (Ahmed and Lin, 2014). Ahmed and Lin use a set of syntactic rules to phrase all possible negations of a given sentence, according to the possible scopes of negation. Their focus is rather different from ours. First, they are interested in sentences in general, rather than Claims – which, in our corpus, tend to have a typically simple structure. Second, their interest is mainly in finding the scopes where negation can be applied, and applying it using simple rules. Here we consider only one scope, and explore the fluency and plausibility of the resulting statement and its argumentative value. Finally, Ahmed and Lin exemplify their technique on a small set of sentences, whereas here the statistical analysis and learning are done on much larger data.

## 3 Problem definition and associated challenges

Similarly to (Aharoni et al., 2014; Levy et al., 2014) we define the following two concepts:

- **Topic** – a short phrase that frames the discussion.
- **Context Dependent Claim (CDC)** – a general, concise statement that directly supports or contests the given Topic.

For brevity, henceforth we refer to a CDC as simply a Claim. Given such a Claim, our goal is to automatically generate its *Claim Negation*, defined here as

*a statement that asserts the opposite of the original Claim, and can be plausibly used while discussing the Topic*.

For example, given the Claim *affirmative action is effective*, its Claim Negation could be stated as follows: *affirmative action is* **not** *effective*. However, for many Claims, the situation is somewhat more complex. Specifically, we identify four levels of complexity when trying to automatically generate Claim Negations.

- *Grammar*—as with any task in which text is automatically generated or modified, one has to make sure the new text is grammatically correct. For example, a naïve system which simply inserts the word "does not" before the verb "have", might transform the Claim:

  > *As a standard embryo does have a highly valuable future, killing it is seriously wrong.*

  into the grammatically incorrect statement:

  > *As a standard embryo does* ***does not*** *have a highly valuable future, killing it is seriously wrong.*

  As will be discussed in the sequal, such errors are rare, and, by and large, are a result of errors in the negation algorithm, which in retrospect could have been easily fixed.

- *Clarity*—an automatically generated negation might be grammatically correct, but unclear and incoherent. For example, an automatic system trying to negate the Claim:

  > *School should be made to fit the child, rather than the other way around.*

  may suggest the following statement, which is grammatically correct, yet unintelligible:

  > *School should* ***not*** *be made to fit the child, rather than the other way around.*

- *Opposition*—a naïve negation might be grammatically correct, and even clear, but still not expressing the opposite of the original Claim. For example, given the Claim:

  > *Children who fail to engage in regular physical activity are at greater risk of obesity.*

  the following suggested negation is not stating its opposite, hence is not a valid Claim Negation (the scope is wrong):

  > *Children who* ***do not*** *fail to engage in regular physical activity are at greater risk of obesity*

- *Usability*—finally, a suggested negation that satisfies the above three criteria, may still not be plausible to use while discussing the Topic. Consider the following two Claims:

  > *Affirmative action has* ***undesirable*** *side-effects in addition to failing to achieve its goals.*
  >
  > *The selection process* **should not** *be based on some arbitrary or irrelevant criterion.*

  and their corresponding candidate negations:

  > *Affirmative action has* **desirable** *side-effects in addition to failing to achieve its goals.*
  >
  > *The selection process* **should** *be based on some arbitrary or irrelevant criterion.*

  Both suggested negations pass the previous three criteria, but nonetheless it is hard to imagine someone stating them in earnest.

Finally, it is interesting to note that for some Claims, a corresponding Claim Negation does not necessarily exist. Consider the following two Claims:

> *People continue to die at a high rate due in large part to lack of sufficient aid.*
>
> *Rather than 'punish' the banks and others truly responsible for the crisis, the government is instead 'punishing' regular people for the 'crimes' of others.*

While one can think of many ways to try and refute these Claims, it is less clear how one states the exact opposite of either of them.

## 4 Automatic claim negation algorithm

In this section we describe our technical approach to automatically generating a Claim Negation. We start with a description of some preliminary analysis. Motivated by this analysis, we defined a two-stage approach. In the first stage, described in section 4.2, given a Claim, a simple rule–based algorithm is applied to generate its candidate negation. In the second stage, described in section 4.3, an automatic classification scheme is used to assess the plausibility of using the suggested negation (i.e. whether or not it passes the Usability criterion).

### 4.1 Preliminary analysis

The purpose of the preliminary analysis was to better understand where most of the challenge lies. Is it difficult to suggest a grammatically correct negation? Or perhaps the main difficulty is in automatically determining if the suggested negation is plausible to use? Furthermore, how often one should expect a Claim Negation to actually exist—clearly a prerequisite for the system to correctly produce one?

Towards that end we asked a team of five annotators to manually analyze the first 200 Claims in the dataset published in (Aharoni et al., 2014). Each annotator was asked to examine each Claim, and to determine the *difficulty* of generating a negation of that Claim. Specifically, the annotator was asked to label the negation difficulty as "Type 1" (namely, "simple"), if it can be derived from the original Claim by one of the following alterations:

1. Adding the word "no" or "not".
2. Removing the word "no" or "not".
3. Adding the phrase "does not" or "do not", and possibly changing the conjugation of an adjacent verb.

The annotator was asked to define the negation difficulty as "Type 2" (namely, "complex") if she could think of a negation, but not of one derived through the simple modifications mentioned above. If the annotator could not easily phrase a clear negation to the examined Claim, she was asked to define the negation difficulty as "Type 3" (namely, "none available").

Given the annotation results, the negation difficulty of an examined Claim was defined as the majority vote of the five annotators. By this scheme, 128 Claims were annotated as having a *simple* negation, 37 as having a *complex* negation, and 25 with *none available*. For an additional 10 Claims the vote was a 2-2-1 split. This was rather encouraging, suggesting that for about 75% of the Claims that can be negated, simple rules may suffice.

In addition, each annotator was asked to determine if it is plausible to use the envisioned negation in a debate. For only 47 out of the 200 Claims examined, the majority of the labelers determined that the negation would be usable. These results suggested that the main challenge for automatic Claim negation would lie in determining usability rather than in generating a grammatically correct negation, which led us to the approach described in the sequel.

### 4.2 Claim negation: How?

The first stage of our algorithm receives a Claim as input, and uses a simple rule–based machinery to generate its candidate negation, aiming for it to be a Negated Claim. Specifically, the algorithm runs as follows:

1. Tokenize the Claim and label the tokens for parts-of-speech using the Stanford Core NLP pipeline (Manning et al., 2014).
2. Find the first token labeled as one of the following: a modal verb, a verb in the present tense, or a verb ending in "n't". We denote this token as $T_1$.
3. If $T_1$ is followed or preceded by one of several negation strings (e.g., "no", "not"'), remove this negation and finish.
4. If $T_1$ ends in "n't", remove this suffix and finish (so, e.g., "can't" would be transformed to "can").
5. If $T_1$ is a modal verb, is a form of the verb "to be" (e.g. "is" or "are"). or is followed by a gerund, then:

(a) If $T_1$ is followed by a word composed of a negation prefix (e.g. "un", "non") and a WordNet (Miller, 1995) adjective (e.g., "unworkable"), remove the negation prefix and finish.

(b) Otherwise, insert the word "not" after $T_1$, and finish.

6. Otherwise, insert the words "does not" or "do not" (according to plurality) before $T_1$, and replace $T_1$ with its lemmatized form.

Note that the algorithm may fail in step 2, if no appropriate token exists. This happened in five of the 200 Claims we initially considered, so for the remainder of this paper we ignore this problem.

### 4.3 Claim negation: When?

The second stage of our algorithm, receives as input the output of the first stage – namely, a candidate negation, and aims to determine its Usability, i.e., whether or not it is plausible to use the suggested negation in a debate about the Topic. To this end, we used a Logistic Regression classifier. Specifically, we developed a set of 19 features, and, accordingly, each candidate negation was transformed into an 19-dimensional feature vector. The classifier was trained and tested based on these representations. Importantly, to avoid overfitting, the features were designed and developed by examining only the initial results of the algorithm on the set of 200 Claims exploited in our preliminary analysis (section 4.1), and all of them were used in later experiments.

The features eventually included in our algorithm were as follows, and are discussed in greater detail below:

1. Counts: Number of words in the Claim.

2. Tokens: Whether or not the Claim contains the following tokens: "and", "or", "than", "," (one feature per token).

3. PoS Tags: Whether or not the Claim contains the following PoS Tags: "VBZ", "VBP", "MD" (one feature per PoS tag).

4. Sentiment: Number of words with positive sentiment and number of words with negative sentiment, taken from (Hu and Liu, 2004) (two features).

5. Algorithm step: Which step in the rule-based algorithm of section 4.2 yielded the negation (8 features; some steps are divided in 2).

6. Frequency in real world corpora – of the altered phrase in the suggested negation, compared to that of the original phrase, in the original Claim.

The motivation for selecting the first five types of features is that it is probably more challenging to automatically generate valid Claim Negations to complex and comparative Claims. In addition, removing an existing negation may behave differently from inserting a negation.

The relative frequency feature is motivated by examples like the non-usable negation mentioned in section 3:

> *Affirmative action has desirable side-effects.*

The relatively low frequency of the phrase "desirable side effects" compared to that of the original phrase "undesirable side effects" may be indicative to the implausibility of using the former. For example, in the Wikipedia dump we examined, the former appears just five times and the latter 120 times.

More specifically, the frequency feature, denoted $f$, was computed as follows. We tokenized both the original Claim and the suggested negation, yielding two sequences of tokens, denoted $\{c_1, c_2, \ldots, c_{k_1}\}$ and $\{n_1, n_2, \ldots, n_{k_2}\}$. We then found the last token up to which both sequences agree, and denoted its position $i$. Thus, in these notations, $c_i$ and $n_i$ are the same token (e.g., "has" in the aforementioned example), while $c_{i+1}$ differs from $n_{i+1}$ (e.g., "undesirable" versus "desirable" in the same example). We then considered the following sequences of 5 tokens - $\{c_i, \ldots, c_{i+4}\}$ and $\{n_i, \ldots, n_{i+4}\}$, and their respective frequency in Google *n*-grams (Michel et al., 2011) for $n = 5$, denoted $f_c$ and $f_n$, respectively. If both sequences were not found (or if either sentence had less than $i + 4$ tokens), we repeated the process for sequences of 4 tokens, and if needed, for sequences of 3 tokens, until one of the sequences was present at least once in the corresponding Google *n*-grams corpus. Finally, we defined $f = (f_n + 1)/(f_c + 1)$. Thus, if the sequence

obtained in the suggested negation ("has desirable side effects" in our example) was rare in Google $n$-grams compared to the corresponding sequence of tokens in the original Claim ("has undesirable side effects" in our example) then $f$ was correspondingly receiving a relatively low value.

## 5 Experimental results

### 5.1 Experimental setup

We started with a data set of 1,240 Claims, collected in the context of various debatable topics, using the same protocol described in (Aharoni et al., 2014). Given this data, the algorithm described in section 4.2 was used to generate 1,240 pairs of the form (Claim, candidate negation). Each pair was annotated by 5 annotators, out of a set of 11 annotators. Specifically, each annotator was asked to assess the candidate negation according to the 4 criteria mentioned in section 3 – i.e., whether the candidate negation is grammatically correct; clear; states the opposite of the original Claim; and usable in a debate to rebut the Claim. Taking the majority over the 5 annotators determined the candidate negation's label. Thus, a candidate negation was considered "usable" if at least three annotators determined it was such. We note that each pair of annotators either considered no pairs of (Claim, candidate negation) in common, or at least 250. This was important when measuring agreement (section 5.2), ensuring a reasonable sample size.

Next, a logistic-regression classifier was trained and tested based on the features described in section 4.3, in a 10-fold cross validation framework, using the "usable" (yes/no) annotation as the class label. That is, the data set was divided into 10 chunks of consecutive Claims. At each of the 10 iterations, a logistic-regression classifier was trained on 9 of the chunks, and predicted whether or not each of the candidate negations in the remaining chunk should be considered "usable". There is a caveat here - on the one hand each fold should be of the same size, while on the other hand including claims from the same topic in both train and test set may conceivably create a bias (if deciding successful negation is somehow topic-dependant). As a compromise we ordered the claims according to topic. This way folds are of the same size, and at most two topics are split between the train and test sets.

| | Grammar | Clarity | Opp. | Use |
|---|---|---|---|---|
| Frac. pass | 0.96 | 0.85 | 0.79 | 0.50 |
| Mean agree | 0.90 | 0.84 | 0.84 | 0.72 |

Table 1: Fraction of negated claims which passed each criteria according to majority vote, and mean pairwise agreement among annotators. Pairwise agreement is defined as the fraction of candidate negations for which the two annotators give the same "yes/no" label.

The weights assigned to each train sample were the product of two numbers - a normalizing factor and a confidence score. The normalization factor is assigned so that the total weight for positive samples is the same as that of negative samples. Namely, if $k$ out of $n$ samples are positive, then the normalization factor for positive samples is $(n-k)/k$ (and 1 for negative samples). The confidence score was defined as the size of the majority which determined the label, divided by 5. So 0.6 in the case of a 3-2 split, 0.8 in the case of a 4-1 split and 1.0 in the case of a unanimous vote.

The complete data-set, including the Claims, candidate negations, and associated annotations, are available upon request for research purposes.

### 5.2 Results

The first stage – rule-based part – of the Claim negation algorithm performed quite well on the first three levels of this task, being labeled as correct on 96% of the Claims for Grammar, and on about 80% of them for Clarity and Opposition. On the other hand, the generated negations were deemed usable for only 50% of the instances (Table 1).

It is interesting to note that this number is still twice as much as would be expected from our initial study, where only 23.5% of the Claims were annotated as having a usable negation. This may be due to the sample size, or differences in the phrasing of the guidelines—one difference is that in the initial task we asked whether a given Claim has a usable negation, whereas in this task we explicitly stated a candidate negation. The second difference is that in the initial guidelines we asked whether the negation was useful in a debate, and here we asked whether it is useful for refuting the original Claim. We made this second change because we felt that the failing to

| Frac. in | Grammar | Clarity | Opp. | Use |
|---|---|---|---|---|
| Grammar | 1.00 | 0.88 | 0.82 | 0.52 |
| Clarity | 0.99 | 1.00 | 0.91 | 0.59 |
| Opp. | 0.99 | 0.98 | 1.00 | 0.63 |
| Use | 1.00 | 1.00 | 1.00 | 1.00 |

Table 2: Fraction of claims which pass both criteria from those which pass the one listed on the left column. If $n_1$ claims pass criterion $i$, and $n_2$ pass both $i$ and $j$, the $(i, j)$ entry in the table is $n2/n1$.

| Kappa | Mean | Std | Min. | Max. |
|---|---|---|---|---|
| Annot. | 0.43 | 0.09 | 0.23 | 0.63 |
| Class. | 0.29 | 0.10 | 0.21 | 0.36 |

Table 3: Pairwise Cohen's kappa statistics among annotators (first line), and comparing annotators to classifier (second line).

explicitly mention the context of rebuttal in the initial phrasing may indeed have led the annotators to be too strict in their evaluation.

Next, we observe that the suggested negations that pass each criterion form almost a perfect hierarchy with respect to containment (Table 2). All suggested negations that pass the Usability criterion also pass the Clarity criterion and the Opposition criterion. Suggested negations that pass the Clarity criterion and those that pass the Opposition criterion are almost the same ones (intersection covers 91.6% and 97.6% of the original sets, respectively), and both sets almost always pass the Grammar criterion (98.9% and 99.1%, respectively).

Determining whether or not a suggested negation is usable is inherently subjective, and as seen in Table 3, agreement between annotators achieved mean pairwise Cohen's kappa coefficients of 0.43 (this is considered fair to good agreement (Landis and Kock, 1977; Fleiss, 1981)). This is similar to what was obtained in (Aharoni et al., 2014) for a similar task: Claim and Evidence confirmation—the task in which one is presented with a Claim or Evidence candidate and needs to determine whether or not it is indeed one. There the reported mean kappas are 0.39 for Claims and 0.4 for Evidence.

Nonetheless, taking majority vote as labels and training a logistic-regression classifier, prediction accuracy was 0.66%, notably higher than expected at random. Similarly, among the top scoring predictions of each fold, some 80% were indeed annotated as usable (Figure 1). That is, for each fold the 124 suggested negations on which the classifier was tested were sorted according to the classifier's score. Then, for each $k = 1, \ldots, 124$, the fraction of Usable negations among the top $k$ was computed, and averaged across folds. Specifically, on average 86% of the suggested negations for $k = 5$ passed the Usability criterion, 83% for $k = 10$, and 78% for $k = 20$.

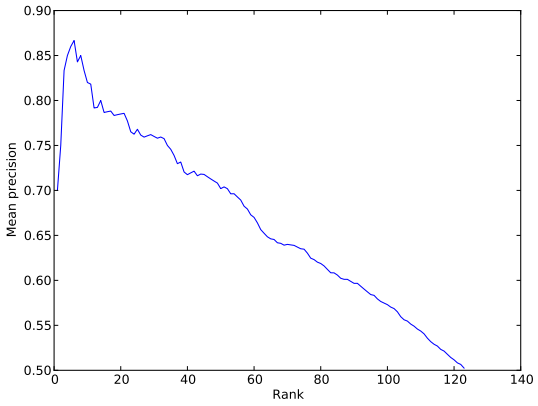


Figure 1: Mean precision (over 10 folds) of the top-ranked candidates, according to classifier score.

Another way to appreciate the classifier's predictions on this subjective task is comparing the annotators' agreement with these predictions to the agreement among the annotators themselves. As seen in (Table 3), while agreement with the former is lower than the latter, the difference in kappa coefficients is less than two standard deviations, and mean agreement with the classifier's predictions is within the range of pairwise agreements displayed among annotators.

It is interesting to understand the relative importance of each of the 19 features, as expressed by the coefficient in the logistic-regression model (when trained on the entire set). The Counts and Sentiment features are normalized, so all values are roughly of the same scale. For the indicators of tokens and for the PoS tag "VBZ", coefficients are less than $10^{-4}$. Indeed, the same results are obtained with and without them (not shown). Other features have roughly the same magnitude of coefficients, with the highest obtained for the Counts feature (1.9), and lowest for the negative Sentiment feature (0.12).

A different way to examine this is via the correlation between feature values and labels. Towards this end we computed the Spearman Correlation between the vector $l$ of $0-1$ labels, and the vector $f_i$ of feature values for each feature $i$. Namely, $l_c$ is 1 if the negation suggested for Claim $c$ was determined "usable" by the majority of the annotators, and 0 otherwise; $f_{ic}$ is the value of feature $i$ computed for Claim $c$. The highest Spearman Correlation so attained (in absolute value) is with the Counts feature (-0.35) and the PoS "MD" Indicator (-0.21). The *n*-gram frequency feature comes in third, with correlation coefficient of 0.07.

Note that since the suggested criteria form a hierarchy, getting good predictions for lower levels of the hierarchy may already yield non-trivial results for higher levels. For example, suppose we had a perfect classifier for Clarity, and we would use it to predict Usability. Predicting that a negation that fails the Clarity criterion is also not usable would always be correct – since all negations which fail Clarity also fail Usability. Conversely, predicting that a negation that passes the Clarity criterion is also usable would be correct 59% of the time (as per Table 2). Since 85% of the automatically suggested negations pass the Clarity criterion, overall accuracy for such a hypothetical classifier would be $0.85 \cdot 0.59 + 0.15 \cdot 1.0 = 0.65$, similar to what is obtained here. Indeed, since many of the features we develop aim to capture the complexity of the Claim, they are probably relevant for classifying success at lower levels of the hierarchy as well. In other words, much of the classifier's success may be attributed to capturing Clarity, rather than Usability. We defer further investigation of these ideas to future work.

## 6 Conclusions and future work

We presented an algorithm that, given a Claim, automatically generates a possible negation, and further determines the plausibility of using this negation to refute the original Claim. Our results highlight the main challenges in generating a meaningful Claim Negation, and demonstrate the feasibility of the proposed approach.

Automatic Claim Negation can augment the results of automatic Claim Detection systems (Levy et al., 2014) and thus enhance the performance of debate support systems. In particular, this could be useful in a setting where the context includes, in addition to a debatable topic, some initial Claims regarding it. For example, in the legal domain, where some Claims have already been made, and one is interested in substantiating them further, or in refuting them. The most basic refutation of a Claim is simply claiming its negation. Of course, for this to be useful, the system would also need to detect Evidence supporting the automatically generated Claim Negation.

The algorithm we presented here for automatic Claim negation is rather naïve. While the results may suggest that the main challenge is in the "when" rather than the "how", improving the first stage – rule based part – of the algorithm is certainly an important step in achieving better automatic negation system. For example, the algorithm negates modal verbs by adding "not" after them (see section 4.2). However, after "must" or "may", this is often erroneous, as in:

> *Countries must be prepared to allow Open borders for people fleeing conflict.*

or

> *National security may simply serve as a pretext for suppressing unfavorable political and social views.*

This may be the reason why the indicator of modal verbs was found to be negatively correlated with the "usable" label, and suggests that more subtle rules, which take negation scope into account, may carry important potential. A database of modal verbs, such as the one in (Pakray et al., 2012), may be helpful for this purpose.

When the algorithm introduces negation, rather than removes it, it always negates the verb. As pointed out in (Ahmed and Lin, 2014), this is the easy case. While this also turns out to be the most common negation scope when it comes to Claims, one could probably improve the negation algorithm by considering other scopes, as done in (Ahmed and Lin, 2014). Determining which of these scopes is the one which actually states the intuitive contradiction of the original claim may be an interesting task in itself, and may make use of corpus-frequency features like the n-gram one described here

As for improving the decision for when a suggested negation is usable, one should keep in mind that while Claims are at the heart of an argument, they usually require supporting Evidence for the argument to be whole. Hence, the existence or absence of supporting Evidence for the suggested negation (or opposing Evidence to the original Claim) may be a strong indicator regarding the suggested negation usability.

Finally, automatic Claim negation may be seen as a special (and relatively simple) case of augmenting a set of Claims via *automatic Claim generation*. That is, rather than building the text from atomic elements, as is usually done in Natural Language Generation, this paradigm suggests to generate new Claims by modifying existing ones. Examples of this are Claim rephrasing towards a specific goal (e.g., making them more assertive or more persuasive), and combining existing Claims into novel ones (e.g., combine the Claim *X causes Y* and *Y causes Z* into *X causes Z*). We believe that any Claim generation task would benefit from the four-tier analysis we suggested here, namely - Grammar, Clarity, Goal attainment (Opposition in the case of Claim Negation), and Usability. In this sense, the present work can be seen as a first step towards constructing more general automatic Claim generation systems.

## Acknowledgements

We thank Ido Dagan and Ruty Rinot for helpful comments and suggestions.

## References

Ehud Aharoni, Anatoly Polnarov, Tamar Lavee, Daniel Hershcovich, Ran Levy, Ruty Rinott, Dan Gutfreund, Noam Slonim. A Benchmark Dataset for Automatic Detection of Claims and Evidence in the Context of Controversial Topics 2014. *Workshop on Argumentation Mining, ACL*

Afroza Ahmed and King Ip Lin. Negative Sentence Generation by Using Semantic Heuristics. 2014. *The 52nd Annual ACM Southeast Conference (ACMSE 2014), Kennesaw, GA, USA*.

Denis Apothéloz, Pierre-Yves Brandt and Gustav Quiroz. The function of negation in argumentation. 1993. *Journal of Pragmatics*, 19 23-38. North-Holland.

Kevin D. Ashley and Vern R. Walker. From Information Retrieval (IR) to Argument Retrieval (AR) for Legal Cases: Report on a Baseline Study. 2013.

Ngo Xuan Bach, Nguyen Le Minh, Tran Thi Oanh, and Akira Shimazu. A Two-Phase Framework for Learning Logical Structures of Paragraphs in Legal Articles. 2013. *In ACM Transactions on Asian Language Information Processing (TALIP)*. 12(1):3

Eduardo Blanco, and Dan I. Moldovan. Some Issues on Detecting Negation from Text. 2011. *FLAIRS Conference*.

W.W. Chapman, D. Hilert, S. Velupillai, et al. Extending the NegEx Lexicon for Multiple Languages. 2013. *Studies in health technology and informatics*, 192:677-6813.

Graham Chapman and John Cleese. The Argument Clinic. 1972. *Monty Python's Flying Circus*, 29:12.

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

J.L. Fleiss. Statistical methods for rates and proportions (2nd ed.) 1981.

A Gatt and E Reiter (2009). SimpleNLG: A realisation engine for practical applications. 2009. *Proceedings of ENLG-2009*.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. 2004. *In Knowledge Discovery and Data Mining*: 168-177.

J.R. Landis and G.G. Kock. The measurement of observer agreement for categorical data". 1977. *Biometrics* 33 (1): 159174.

Landis, J.R.; Koch, G.G. (1977). "The measurement of observer agreement for categorical data". Biometrics 33 (1): 159174.

Ran Levy, Yonatan Bilu, Daniel Hershcovich, Ehud Aharoni and Noam Slonim. Context Dependent Claim Detection 2014. *In The 25th International Conference on Computational Linguistics*

Amnon Lotan, Asher Stern, and Ido Dagan. 2013. TruthTeller: Annotating predicate truth. 2013. *In Proceedings of the Annual Meeting of the North American Chapter of the ACL, pages 752757, Atlanta, Georgia*.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. 2014. *In Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations* 55-60.

Jean-Baptiste Michel, Yuan Kui Shen, Aviva Presser Aiden, Adrian Veres, Matthew K. Gray, William Brockman, The Google Books Team, Joseph P. Pickett, Dale Hoiberg, Dan Clancy, Peter Norvig, Jon

Orwant, Steven Pinker, Martin A. Nowak, and Erez Lieberman Aiden. Quantitative Analysis of Culture Using Millions of Digitized Books. 2011. *Science* 331(6014):176-182

George A. Miller. WordNet: A Lexical Database for English. *COMMUNICATIONS OF THE ACM*, 38:39-41.

Mochales Palau, Raquel and Moens, Marie-Francine. Argumentation mining. 2011. *In Artificial Intelligence and Law*, 19(1): 1-22.

Douglas Walton, Argument Mining by Applying Argumentation Schemes 2012. *In Studies in Logic* 4(1):38-64

Michael Wiegand, Alexandra Balahur, Benjamin Roth, Dietrich Klakow, and Andrs Montoyo. A survey on the role of negation in sentiment analysis. 2010. *In Proceedings of the Workshop on Negation and Speculation in Natural Language Processing, pages 6068, Uppsala*.

Adam Wyner, Raquel Mochales-Palau, Marie-Francine Moens, and David Milward. Approaches to text mining arguments from legal cases. 2010. *In Semantic processing of legal texts* 60-79.

Xiaodan Zhu, Hongyu Guo, Svetlana Kiritchenko, Saif Mohammad. An Empirical Study on the Effect of Negation Words on Sentiment. 2014. *The 52th Annual Meeting of the Association for Computational Linguistics (ACL-2014). Baltimore, USA*.

Partha Pakray, Pinaki Bhaskar, Somnath Banerjee, Sivaji Bandyopadhyay, Alexander F. Gelbukh. An Automatic System for Modality and Negation Detection. 2012. *In proceedings of CLEF (Online Working Notes/Labs/Workshop)*.

Mutalik PG, Deshpande A, Nadkarni PM. Use of general-purpose negation detection to augment concept indexing of medical documents: a quantitative study using the UMLS. 2001. *J Am Med Inform Assoc. 2001 Nov-Dec;8(6):598-609*.