Identifying Sentiment-based Contradictions

Danny Suarez Vargas, Viviane Moreira

Institute of Informatics
Universidade Federal do Rio Grande do Sul, Brazil
{dsvargas,viviane}@inf.ufrgs.br

Abstract. Contradiction Analysis is a relatively new multidisciplinary and complex area with the main goal of identifying contradictory pieces of text. It can be addressed from the perspectives of different research areas such as Natural Language Processing, Opinion Mining, Information Retrieval, and Information Extraction. This article focuses on the problem of detecting sentiment-based contradictions which occur in the sentences of a given review text. Unlike other types of contradictions, the detection of sentiment-based contradictions can be tackled as a post-processing step in the traditional sentiment analysis task. In this context, we adapted and extended an existing contradiction analysis framework by filtering its results to remove the reviews that are erroneously labeled as contradictory. The filtering method is based on two simple term similarity algorithms which relies on sets of known positive and negative words. An experimental evaluation on real product reviews has shown proportional improvements of up to 30% in classification accuracy and 26% in the precision of contradiction detection by considering a manual selection of positive and negative words. When the sets of positive and negative words are automatically selected, the improvements are of up to 22% in classification accuracy and 37% in the precision of contradiction detection.

Categories and Subject Descriptors: H.3 [**Information Storage and Retrieval**]: Miscellaneous; I.7 [**Document and Text Processing**]: Miscellaneous

Keywords: Sentiment Analysis, Contradiction Analysis

1. INTRODUCTION

Consulting the opinion of others during the decision-making process has always been a common practice in people's lives. The goal is to confront different points of view in the search for the best decision. At present, with more than a third of the world population having access to the Internet [Meeker 2015], this practice has moved to the virtual context, in which people interact with others through opinions. These opinions are usually expressed in the form of product reviews available on the Web. Sentiment Analysis (also known as Opinion Mining) focuses on this context in order to help people manage these reviews and produce or extract useful information from them.

Polarity classification (also called polarity detection or sentiment polarity classification) is one of the most important tasks in the Sentiment Analysis area. It can be viewed as a two or three-class classification problem in which the classes are {positive, negative} and {positive, neutral, negative}, respectively. Furthermore, the classification can be performed in different levels of granularity – document, sentence, clause, or aspect level [Liu 2012].

Existing techniques for addressing the sentiment analysis tasks usually employ three steps: *identification*, *classification*, and *aggregation* [Tsytsarau and Palpanas 2012]. For instance, in the basic polarity classification task at sentence level, the first step consists in the identification of opinionated documents; the second step consists in the polarity classification of all sentences for each document; and the final step aggregates the values obtained for each sentence assigning an overall polarity value

This work was partially supported by CNPq. D. S. Vargas receives a grant from CAPES.

to each document. However, only relying on the aggregation as the third step can lead to the loss of interesting information such as contrastive or contradictory opinions about a topic or aspect as we can see in the examples of reviews shown in Table I. In order to highlight the contradictions, an additional step performing a more fine-grained analysis is needed and this is the focus of this article.

Contradiction Analysis is a novel and complex area that does not have yet a consensual definition. Thus, authors define it according to the context in which they work. This is a multidisciplinary area which involves techniques originated from Natural Language Processing, Opinion Mining, Information Extraction, and Information Retrieval. Analyzing contradictions is a challenging task mainly due to the different ways in which they can appear (mismatching numbers, use of antonyms or negation, contrastive sentences, etc.). The pioneer investigations on Contradiction Analysis tried to find agreements and disagreements over audio files [Hillard et al. 2003; Galley et al. 2004]. Contradiction analysis in text appeared some years later [Harabagiu et al. 2006]. From there, contradiction in texts was defined from different perspectives: Natural Language Processing [de Marneffe et al. 2008], Pattern-Based [Ennals et al. 2010; Ennals et al. 2010], Knowledge-Based [Ritter et al. 2008], and *sentiment-analysis-based approach* [Tsytsarau et al. 2011; Suarez Vargas and Moreira 2015]. The identification of the most important characteristics and a classification of contradictions were provided by de Marneffe et al. [2008]. Contradictions can be classified based on their features (lexical, contrastive, negation, etc.) [de Marneffe et al. 2008], based on the time in which they occur (synchronous and asynchronous) [Tsytsarau and Palpanas 2012; Tsytsarau et al. 2011], and based on the context in which they are analyzed (inter and intra-document) [Tsytsarau et al. 2011]. The only formal definition of a contradiction measure in texts comes from the sentiment-analysis-based approach, given in Tsytsarau et al. [2011].

More specifically, the context of the present work is defined as follows. Our input is a dataset of reviews in which the overall topic is the same, the contradiction analysis is performed at sentence level, and our output is a set of reviews containing contrastive or contradictory reviews. We adapted and extended the three-step contradiction analysis framework proposed by Tsytsarau et al. [2011] through an additional filtering step as shown in Figure 1. The goal of this additional step is to remove the occurrence of false positives (*i.e.,* reviews that were labeled as contrastive or contradictory when in fact they are not). This filtering process relies on word similarity algorithms using sets of positive and negative words. It tries to exploit the vector representation of words obtained from the Word2vec tool [Mikolov et al. 2013], which learns the vector representation of words based on a large text corpus. Our contribution also includes the proposal of two simple similarity-based algorithms to determine the polarities of sentences.

Experiments were performed in order to evaluate the proposed algorithms as well as the quality of our similarity-based filtering method. The similarity algorithms achieved improvements in accuracy ranging from 16 to 19% compared to a widely used baseline (*i.e.,* RNTN–Recursive Neural Tensor Network [Socher et al. 2013]). For contradiction analysis, the use of our additional filtering method brought proportional precision improvements of up to 30%.

This article extends our previous work [Suarez Vargas and Moreira 2016] by designing an automatic method for choosing the k most representative positive and negative words which makes use of logistic regression. Also, here we report on experiments using an additional dataset with varying numbers of positive and negative words.

The remainder of this article is organized as follows: Section 2 describes the problem we address in this work; Section 3 revises the related literature; Section 4 presents our proposed method which included the adaptation and extension of existing work; Section 5 describes the experimental evaluation; finally, Section 6 concludes the article and points out some possibilities of future work.

2. PROBLEM DEFINITION

The detection of sentiment-based contradictions using a contradiction measure was addressed earlier by Tsytsarau et al. [2011]. Here, we adapt the definition of contradiction as well as the contradiction measure to our context as follows.

Intra-document and Inter-document Contradiction. The contradictions that occur within a given text of a single author are called as *intra-document contradictions*, while the contradictions that occur across different texts of one or more authors are called as *inter-document contradictions*.

Sentiment-Based Contradiction. For a given review R, which contains two or more sentences $\{S_1,\ S_2,...,S_n\}$, and their polarity orientation values $\{P_1,\ P_2,...,P_n\}$ where $S_1 \neq S_2 ... \neq S_n$, R is considered a contrastive/contradictory review or contains contrastive/contradictory sentences when the *contradiction Measure* C of R exceeds a certain threshold ρ.

Contradiction Measure C. This measure assigns a contradiction value C to R as follows.

$$C = \frac{nM_2 - M_1^2}{(\vartheta n^2 + M_1^2)} W$$

where n is the cardinality or the number of sentences of R and i refers to the i^{th} sentence in R. $M_1 = \sum_{i=1}^{n} P_i$ and $M_2 = \sum_{i=1}^{n} P_i^2$ are the first and second order moments of the polarity of the values which are based on the mean value μ_s and on variance σ^2 respectively. The small value $\vartheta \neq 0$ is used to limit the level of contradiction when μ^2 is close to zero. W is a weight function which takes into account n of R to calculate C.

$$W = \left(1 + \exp(\frac{1-n}{\beta})\right)^{-1}$$

where β is a scaling factor.

Contradictory versus Contrastive. A given review R consisting of two or more sentences with opposite polarity orientations is considered as having a *contradiction* if the sentences refer to the same topic or attribute, whereas if the divergence in polarities refer to different attributes of the overall topic, the review is considered to have a *contrast*. Table I shows examples of contradiction and contrast. In this work, we are looking for *intra-document synchronous contradictions* or *contrasts* in text from the sentiment analysis approach (sentiment-based contradictions). More specifically, we are looking for reviews that contain contrastive/contradictory sentences using the polarity orientation of the sentences to decide whether a review contains contrastive/contradictory sentences.

3. RELATED WORK

The analysis of contradictions in text was addressed for the first time in Harabagiu et al. [2006]. The authors identified contradictions using lexical, negation, and contrast features as well as a text alignment tool. Later, de Marneffe et al. [2008] contributed with a definition of contradiction for the Natural Language Processing area and described a classification of contradictions based on the features which characterize them.

The literature has diverse definitions for Contradiction Analysis, as each author defined it according to the specific problem that they were trying to solve. For example, Padó et al. [2008], who implemented the contradiction detection system developed by de Marneffe et al. [2008], define it as a

Table I. Contradiction vs Contrast

Sentence 1	Sentence 2	Type of review
"update made it worse"(-)	*"thank you for fixing your app"*(+)	**Contradictory**
"good site and content"(+)	*"bad app hard application to navigate"*(-)	**Contrastive**

textual entailment problem using textual alignment scores, co-referent events, and a logistic regression algorithm to decide whether the two given texts contradict each other. Ennals et al. [2010] address the problem as a search of conflicting topics on the Web through text patterns like "It is not correct that...". In addition, there is also a line of investigation known as controversy research. The aim is to identify whether Web contents deal with controversial topics (such as abortion, religion, same-sex marriage, etc.) and notify the user when the topic that they are searching is controversial [Dori-Hacohen et al. 2015].

Finally, Tsytsarau et al. [2011] define contradiction as a form of *sentiment* diversity and stated that there is a contradiction regarding topic T when there are conflicting opinions on T. This latest work is the baseline of our work, so we describe it in greater detail next. In order to define a novel approach for contradiction detection, the authors proposed concepts of sentiment-based contradiction, aggregated sentiment (mean value), sentiment variance (variance), and a contradiction measure based on these two definitions. Furthermore, contradictions were classified based on the time in which they arise (Synchronous, Asynchronous). A three-step framework for contradiction detection was proposed. The first step of this framework consists in detecting the topic of each sentence of the input data. The second step assigns a sentiment to each sentence-topic pair. Then, contradiction analysis is performed as the final step. An experimental analysis attempted to find contradictions on the topic "internet government control" considering reviews published in a time window of ten days. The authors show plots for the mean, variance, and the contradiction measure over time. On an evaluation with human subjects, the authors found that users were able to identify contradictions faster with their method than when using a visual method proposed by Chen et al. [2006].

Among the differences between Tsytsarau's work and ours, is the fact that while they look for contradictions that occur across different documents (inter-document), we look for contradictions that occur inside a single document (intra-document). The other difference is that, instead of only relying on the contradiction measure to detect contradictions, we consider an additional filtering process which is detailed in Section 4.

In our previous work [Suarez Vargas and Moreira 2015], we presented a sentiment-based framework for contradiction detection. In that work, we did not consider any formal contradiction measure nor the current proposed similarity and filtering algorithms.

4. FRAMEWORK TO DETECT SENTIMENT-BASED CONTRADICTIONS

In this section, we describe the process of adapting and extending the original framework by Tsytsarau et al. [2011]. Figure 1 shows the extended framework for sentiment-based contradiction analysis. The extension is the filtering step, which is detailed further in Figure 2.

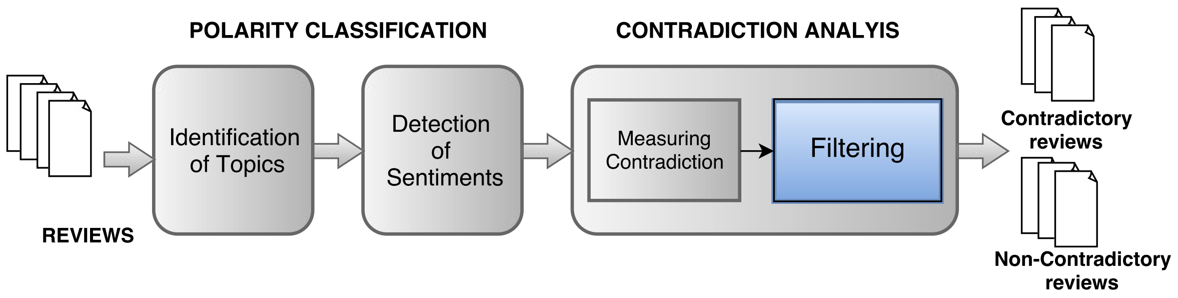

Fig. 1. Extended sentiment-based contradiction analysis framework

4.1 Adapting the Framework

The original framework, as introduced in Section 3, is a three-step method over which we perform some modifications in order to adapt it to our context.

Identification of Topics. Since we are looking for intra-document contradictions and considering that the input reviews are about a single overall topic, the step in which topics are identified is not necessary in our context.

Detection of Sentiments. The goal of this step is to assign a sentiment value (*i.e.,* positive, negative and neutral) to each sentence-topic pair. Since we are dealing with a single topic, we only need to perform the assignment of sentiment values to each sentence, which can be achieved with polarity classification. In order to perform polarity classification, we used RNTN [Socher et al. 2013] which is detailed in Section 5.

Measuring Contradictions. In the original framework, this step aims at finding the contradictory opinions across documents based on the contradiction measure C. We also perform this step by considering the adapted version of the measure as presented in Section 2. At this point, we have the sentences classified as positive or negative. Then, based on these classified sentences, the contradiction value C was calculated for each review. So, we selected the reviews with the highest C value, labeling them as contradictory.

After adapting the original framework, we extend it by adding a filtering step (see Figure 2) that aims to remove the reviews erroneously labeled as contradictory. This method is based on word similarity, more specifically, on the cosine between the vector representation of two groups of words (group of k-positive/negative words and the words of the given input sentence). The way to determine the k-positive and k-negative words, the vector representation of words, the similarity algorithms as well as the process of filtering errors are detailed next.

k-positive and k-negative Words. In this step, we select the k-most representative positive and negative words. This selection can be manually, automatically, or semi-automatically performed. The manual selection requires domain knowledge [Liu et al. 2004]. The automatic selection can be performed, for example, by relying on results of clustering algorithms or on the results of regression models [Sangani and Ananthanarayanan 2013], while the semi-automatic selection combines manual and automatic selection methods.

In order to select the k most positive and k most negative words, we performed a logistic regression. Similar to Sangani and Ananthanarayanan [2013], we hypothesize that some words are indicative of a high or low star rating. Then, the unigram representation of text $\phi(x)$ and the star rating $y(x)$ of each review are used iteratively to find these indicative words. More formally, we define our sigmoid function using $\phi(x)$, and a weight vector θ as:

$$h_\theta(\phi(x)) = g(\theta.\phi(x)) = \frac{1}{1+\exp(-\theta.\phi(x))}$$

where $h_\theta(x)$ is the prediction of the normalized rating of x. The weight vector θ is learnt using stochastic gradient descent for a given learning rate α as

$$\theta := \theta + \alpha(y^i - h(\phi(x^i)))\phi(x^i)$$

The stochastic gradient descent is performed iteratively over each review of the dataset T until:

$$\mid E_{\theta_{i+1},T} - E_{\theta_i,T} \mid < \epsilon$$

where $E_{\theta_i,T}$ is the error over T and is defined as follows:

$$E_{\theta_i,T} = \frac{1}{\mid T \mid}\sum_{j\in T}(y^j - h_\theta(\phi(x^j)))^2$$

After performing the logistic regression, the weight vector θ organized the unigrams according to their importance for score prediction. We take the k unigrams with the highest values and the k unigrams with the lowest values. Then, these unigrams are considered as the k most positive and k most negative words. However, at this point, some of the selected words may be variations of the same base word (*e.g.,* variations in number, gender, or even misspellings). For example, the words *awesome* and *awsome* could both be among the most positive words, but they refer to the same concept. To unify the many variants, we grouped these words based on approximate string matching, by using Levenshtein Distance. Then, for each group the total weight is calculated by adding the weight of each word and a word is selected as the *representative* for that concept. Finally, the groups of words are reorganized based on their total weight and we can select the k different representative words.

4.2 Extending the Framework

This section explains how the framework was extended by the addition of a filtering step.

Vector Representation of Words. This step is responsible for providing a vector representation of words. This vector representation plays an important role in the effectiveness of our proposed algorithms. We used the high dimensional word vector representation provided by Word2Vec tool [Mikolov et al. 2013] which is detailed in Section 5.

Similarity Algorithm. From the vector representation of words, we used the well-known cosine similarity which measures the similarity of two vectors by relying on the cosine of their angle. Furthermore, we formally define the similarity between words as follows. Given a word w and a set of words V=$\{v_1$,v_2,...,$v_k\}$, the similarity function SW assigns a value $d \in$ [-1,1] to w based on the cosine similarity algorithm ϕ of the vector representation of word w regarding the vector representation of each v_i with $i \in$ {1,2,...,k}

$$SW\left(w, V\right) = \phi\left(CosSimil\left(w, v_i\right)\right)$$

We propose two algorithms ϕ, described in Algorithm 1 and 2 to measure the similarity between two sets of words. These algorithms are used to calculate the similarity between the set of k-positive/negative words and the set of words of a given sentence S. These values indicate the positive and negative orientation of a given sentence S. The difference between the two proposed algorithms is the way that they compute the similarity values. In our experiments, we used the maximum and mean values.

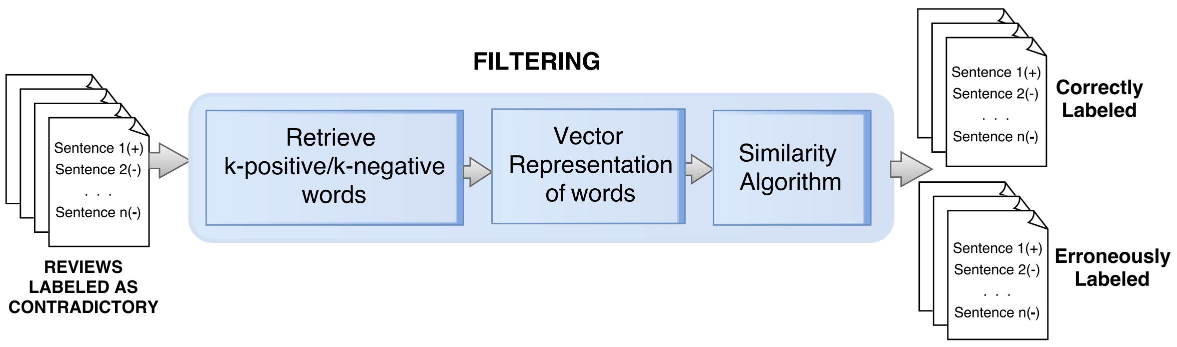

Fig. 2. Similarity-based filtering method

Algorithm 1: Measuring the mean-similarity between two sets of words

```
input  : A set of $n$-words $A$ and a set of $m$-words $B$
output: A real value $resp$ that represents the similarity between $A$ and $B$
1  $first\_array \leftarrow []$;
2  for $i \leftarrow 0$ to $n-1$ do
3      $second\_array \leftarrow []$;
4      for $j \leftarrow 0$ to $m-1$ do
5          $simil \leftarrow cos\_distance(Word2Vec(A[i]), Word2Vec(B[j]))$ ;
6          if $(simil >= -1)$ then
7              $second\_array.add(simil)$;
8          end
9      end
10     $second\_array\_without\_outliers \leftarrow remove\_outliers(second\_array)$;
11     $mean\_value \leftarrow mean(second\_array\_without\_outliers)$;
12     $first\_array.add(mean\_value)$;
13 end
14 $resp \leftarrow mean(first\_array)$;
```

Algorithm 2: Measuring the max-similarity between two sets of words

```
input  : A set of $n$-words $A$ and a set of $m$-words $B$
output: A real value $resp$ that represents the similarity between $A$ and $B$
1  $first\_array \leftarrow []$;
2  for $i \leftarrow 0$ to $n-1$ do
3      $second\_array \leftarrow []$;
4      $max\_simil \leftarrow -2$;
5      for $j \leftarrow 0$ to $m-1$ do
6          $simil \leftarrow cos\_distance(A[i], B[j])$ ;
7          if $(simil > max\_simil)$ then
8              $max\_simil \leftarrow simil$;
9          end
10     end
11     if $(max\_simil >= -1)$ then
12         $first\_array.add(max\_simil)$;
13     end
14 end
15 $resp \leftarrow mean(first\_array)$;
```

Each of our proposed similarity algorithms can be combined with an existing classifier from the literature in order to implement our polarity orientation algorithm. In our work, we chose the state-of-the-art in polarity classification at sentence level for movie reviews, the RNTN classifier which is detailed in Section 5. Thus, we combine the results of RNTN with the results of each of our similarity algorithms to determine the polarity orientation of sentences, as described in Algorithm 3. To classify a given sentence S, the required inputs are: the two real values obtained with the previous algorithms, the sentiment orientation $sent_{classifier}$ assigned by the state of the art classifier, and a threshold t. In our experiments, the value of t was chosen empirically. We carried out experiments varying the threshold values t (from 0 to 1) and selected the best value based on the polarity classification results. The output is a label indicating the polarity of the sentence.

Filtering Errors. This step performs the filtering process of reviews that were erroneously labeled as contradictory. The input of this step consists of all reviews that were labeled as contradictory in the Measuring Contradiction step, the sentences of these reviews labeled as positive or negative by RNTN, and two real values for each sentence which represent the positive and negative orientation. Based on these inputs, the algorithms that perform the filtering process are presented next in Algorithm 4. If the absolute value of the variable *diff* is equal to 0 or 1, it means that the review R contains the same (or a very similar) number of positive and negative sentences about a product. So, based on the fact that the review is about a single topic, we can assume that R is contrastive or contradictory.

Algorithm 3: Determining the polarity orientation of a given sentence

```
input  : A sentence $S$, the real value $SP$ w.r.t. the $k$-positive words, the real value $SN$ w.r.t. the $k$-negative
         words, the sentiment orientation $sent_{classifier}$ assigned by the state of the art classifier, and a threshold $t$
output: the polarity orientation $sent$ of $S$ based on $SP$, $SN$, and $t$
diff ← $SP - SN$;
if $\|diff\| > t$ then
    if $diff > 0$ then
        $sent \leftarrow positive$;
    else
        $sent \leftarrow negative$;
    end
else
    $sent \leftarrow sent_{classifier}$;
end
```

Algorithm 4: Determining if a given review should be filtered

```
input  : Array of $n$ sentences that represent the current review, array $SP$ with the similarity values with the
         $k$-positive words, array $SN$ with the similarity values with the $k$-negative words, and a threshold $t$
output: review $R$ labeled as contradictory or not
for $i \leftarrow 0$ to $n - 1$ do
    $sentence \leftarrow sentences[i]$;
    $sentim \leftarrow sentiment\_orientation(sentence\_a, SP[i], SN[i], t)$;
    if $sentim == positive$ then
        $array\_positives.add(sentence)$;
    else
        $array\_negatives.add(sentence)$;
    end
end
$diff = length(array\_positives) - length(array\_negatives)$;
if $\|diff\| == 0$ or $\|diff\| == 1$ then
    The review is contradictory
else
    The review is not contradictory (it should be filtered out)
end
```

5. EXPERIMENTS

In this Section, we describe the experiments performed to evaluate our proposed algorithms.

5.1 Experimental Setup

Datasets. Two datasets were used in our experiments, we refer to them as *main* and *external.* Both are composed of users' reviews about Android applications collected from the Google Play Store. This source was selected as it is expected to have a considerable number of contrastive sentences in its reviews. Each review contains information on reviewer ID, creation time, rating (from 1 to 5), and review text.

The *main dataset* was collected by Sangani and Ananthanarayanan [2013] and is used for testing. There are 31,500 reviews split into seven groups. Each group contains 4,500 reviews in English about a different Android application. For the experiments with this dataset, we only used the *review text* and its *rating.* The positive reviews (4 and 5 stars) are the most frequent.

The *external dataset* was collected specifically to perform the logistic regression with the purpose of selecting the k-positive and k-negative words. The raw dataset, crawled from Google Play Store, contains 17,566 reviews about 12 different applications. From this dataset, we only used the *review tittle* and *rating.* The rationale is that while the review text may contain both positive and negative words (and thus be ambiguous), the title tends to summarize the overall sentiment conveyed by the review text.

The **Word2Vec Tool** [Mikolov et al. 2013] provides the implementation of two model architectures: Continuous Bag-of-words model and Continuous skip-gram model. These models are used for computing continuous vector representations of words learned by neural networks. The first model allows predicting the current word based on its surrounding words (words that appear before and after the current word) and the second model allows predicting the surrounding words based on the given current word. In our experiments, we used an available Word2Vec binary file which was generated based on a Wikipedia dump for the English language[1].

RNTN Classifier. Recursive Neural Tensor Network [Socher et al. 2013] is a model that aims at capturing the compositional effects of longer phrases in the task of sentiment detection. RNTN performs better than other neural networks that ignore word order and establishes the state of the art in the polarity classification task at sentence level by using the Stanford sentiment treebank. The Stanford sentiment treebank is a large and labeled compositional resource which consists in fine-grained sentiment labels for 215,154 phrases in the parse trees of 11,855 sentences about movie reviews. In our experiments, we used the RNTN classifier (RNTN pre-trained on the Stanford sentiment treebank) in order to perform polarity classification.

Pre-processing and RNTN Classification. A pre-processing step was performed in order to remove incomplete reviews such as those that did not contain star ratings. This step reduced the number of reviews from 31500 to 31482. Then, polarity classification was performed using RNTN. This classification takes the text of the reviews as input, splits them into sentences and assigns one of five possible values to each sentence (1,2,3,4,5). This values can be organized in three groups ((1,2), (3), (4,5)) that represent the negative, neutral and positive orientation, respectively. In this step, the number of reviews is reduced from 31482 to 30228. The main reason for this reduction is the presence of non-English words or single-emoticons sentences which cannot be classified.

Measuring Contradictions. Thus, for each review, we calculate its contradiction value C with the small value ϑ fixed in $0,0005$ (as this was the reference value used bu Tsytsarau et al. [2011]). After that, we performed the selection of the reviews with the highest C value. C ranges from 0.00 to $2.98e-06$. It takes the minimum value when all sentences of a given review have the same polarity value, and assumes the maximum value when the sentences of a given review have the same number of positive and negative sentences.

Data Annotation. For our two experiments, we selected the reviews which have the maximum C value (2.98e-06), which resulted in 840 reviews, all with two sentences each. Furthermore, we manually annotated the sentences in order to allow for subsequent analysis. The first annotation consists in labeling each of the 1680 selected sentences as positive, negative, or neutral. The second annotation consists in labeling each of the 840 reviews as contradictory or not by relying on the polarity of the first annotation. If a given sentence S_1 in review R has the polarity orientation assigned by RNTN different from their manually assigned polarity orientation, the review is considered as erroneously labeled as contradictory.

Selection of k-positive and k-negative Words. This step can be performed manually, semi-automatically, and automatically as we mentioned in Section 4.1. In this work, we tested the manual and the automatic selection of words.

The first selection was performed manually by picking the most significant words from a list of 30 words assembled by Sangani and Ananthanarayanan [2013]. In our experiments, the value of k was 19. The manually selected words were:

[1]`https://dumps.wikimedia.org/enwiki/`.

Negative-Words (N) = {"update", "open", "sucks", "phone", "uninstall", "ads", "play" , "bad", "poor", "crap", "crashes", "useless", "uninstalled", "force", "terrible", "horrible", "uninstalling", "waste", "annoying"}
Positive-Words (P) = {"love", "great", "good", "awesome", "best", "excellent", "nice", "game", "cool", "fast", "easy", "fun", "amazing", "addictive", "perfect", "super", "helpful", "fantastic", "better"}.

For the automatic selection, we used logistic regression over the external dataset as explained in Section 4.1. The title and the rating of the reviews were considered here. First, in the preprocessing step, we removed stopwords and incomplete reviews (*i.e.,* reviews without the title or the rating), and tokenized the titles. For the stopword removal, we used the list available with NLTK (Natural Language Processing with Python) [Bird et al. 2009]. After that, the number of reviews was reduced to 4,655. Then, we generated the positive and negative groups of reviews. The first group consists of reviews with rating values of 4 and 5, while the second group consist of reviews with rating values of 1 and 2. The reviews with rating value 3 were not considered in the present experiment.

Since the groups (positive and negative) had different sizes, we equalized them by discarding randomly chosen reviews from the largest group. At this point, each group contained 1431 reviews. From these reviews, we considered the set of distinct words, *i.e.,* the lexicon, which contained 1479 unique words. Then, each title is represented as 1479-dimensional vector by using the presence (1) or absence (0) of words. Furthermore, the weight vector θ is also a 1479-dimensional vector of zeros, while the class of each tittle text is 1 for the positive group and -1 for the negative group. Then, using the vector representation of each review tittle, the weight vector θ, and the class of each review, the logistic regression for $\alpha = 0.001$, and $\epsilon = 0.00000057$ is performed. After that, the words with the Levenshtein Similarity $\geq 80\%$ were grouped together, their weight values were added up and the representative word (*i.e.,* the word that will represent the group of similar words) was selected. For example, in the set {*good*, *gooood*, *goodie*}, the form "*good*" could be used as the representative word. Next, the set of representative words is sorted by their weight and the top-k and the bottom-k words were selected. In our experiments, we used k ={30, 60, 90}.

Filtering Method. In order to determine which part of the reviews should be filtered, we performed the steps detailed next. We calculated the similarity of the 1680 selected sentences regarding each group of selected words by using the vector representations provided by Word2Vec in our similarity Algorithms 1 and 2 (max_similarity, and mean_similarity). Furthermore, we calculated the polarity orientation of each sentence based on our Algorithm 3 with the parameter value t fixed set to 0.05 and the k-positive/negative selected words. Finally, we used Algorithm 4 to determine which reviews should be filtered.

Polarity Classification Experiment. Polarity classification is performed over the 1680 sentences by using our polarity_orientation Algorithm 4 with the parameter value t fixed at 0. The experiments at this task were organized in two groups. For the first group the k-positive and k-negative words were manually selected. For the second group, these words were automatically selected using the procedure described in Section 4.1. Furthermore, for both groups, we used the two similarity algorithms(max_similarity, and mean_similarity) by taking as the parameter each of the 1680 selected sentences.

Contradiction Detection Experiment. Here, the experiments were also separated into two groups. While the first group uses manual selection of positive/negative words, the second group employs automatic selection. In addition, we compared contradiction detection using the adapted framework without our filtering method (relying only on the C value) and with our filtering method.

Evaluation Metrics. The evaluation was performed for the two groups of experiments: Polarity Classification and Contradiction Detection. For Polarity Classification, we calculated precision, recall, accuracy, and F1 values for the RNTN classifier. For Contradiction Detection with the Filtering

Method, we calculated the accuracy of the adapted framework with and without our filtering method. We were not able to calculate the recall for the filtering method as we do not know the total number of reviews with contradictions.

5.2 Results

In this Section, we present the results of the different experimental settings. The evaluation was performed on the main dataset.

5.2.1 *Polarity Classification.* The results for Polarity Classification are summarized in Table II, where the best scores are in bold. The proportional improvement of each method compared to the baseline is shown between brackets.

Using manually selected positive and negative words. The polarity-orientation algorithm (Algorithm 3) employing the two similarity measures (Algorithms 1 and 2) was compared with the RNTN classifier (baseline). The results showed that with both the max and the mean, there are gains in recall, accuracy, and F1. These gains were much larger than the loss in precision that the methods brought. This was a consequence of a large reduction on the number of false negatives but with a (smaller) increase on the number of false positives. Comparing the two proposed similarity measures, we observed a slight difference in favor of Algorithm 2 (max_similarity). A Wilcoxon signed-rank test on the accuracy of each method has shown that both improvements are statistically significant, yielding p-values < 0.0001. The same test applied to our two proposed versions showed that Alg. 2 is significantly superior to Alg. 1 (p-value $= 0.0016$). We attribute the gains to the effective vector representation of the words achieved by Word2Vec, which is based on a very large corpus ($\approx$50Gb).

Using automatically selected positive and negative words. The polarity-orientation algorithm (Algorithm 3) using the two similarity measures (Algorithms 1 and 2) was also compared to the baseline. The results showed that only with the max there are gains in recall, accuracy, and F1. Furthermore, we can observe that the best results are obtained when the number of positive and negative words (k) is 60. Finally, we ran a Wilcoxon signed-rank test on the accuracies of each Algorithms 1 and 2 (using both manual and automatic selection) compared to the accuracy of the baseline. The results of the test confirm that both achieve a significant improvement (p-values < 0.0001).

5.2.2 *Contradiction Detection.* For detecting contradictions, we employed our filtering Algorithm (Alg. 4) with the two variations of the polarity-orientation algorithm. The results are summarized in Table III, where the best scores are in bold. The proportional improvement in precision of each method compared to the baseline is shown between brackets. In this experiment, since we did not have annotations for all pairs of sentences in the dataset (and only for the top scoring in the contradiction measure), it was not possible to calculate recall and F-measure.

Table II. Results for the polarity classification task.

	Selection	**k**	**Precision**	**Recall**	**Accuracy**	**F-measure**
RNTN (Baseline)	–	–	0.87 (+0%)	0.6	0.63	0.71
Mean_(Alg. 1)	**Manual**	19	0.81 (-7%)	**0.94** (+57%)	**0.78 (+24%)**	**0.87** (+23%)
Mean_(Alg. 1)	Automatic	30	0.86 (-1%)	0.32 (-47%)	0.44 (-30%)	0.46 (-35%)
Mean_(Alg. 1)	Automatic	60	0.88 (+1%)	0.41 (-32%)	0.51 (-19%)	0.56 (-21%)
Mean_(Alg. 1)	Automatic	90	**0.90** (+3%)	0.35 (-42%)	0.48 (-24%)	0.50 (-30%)
Max_(Alg. 2)	**Manual**	19	0.83 (-5%)	**0.95** (+58%)	**0.82 (+30%)**	**0.88** (+24%)
Max_(Alg. 2)	Automatic	30	0.80 (-8%)	0.91 (+52%)	0.76 (+21%)	0.85 (+20%)
Max_(Alg. 2)	Automatic	60	0.80 (-8%)	0.92 (+53%)	0.77 (+22%)	0.86 (+21%)
Max_(Alg. 2)	Automatic	90	**0.88** (+1%)	0.80 (+33%)	0.76 (+21%)	0.84 (+18%)

Table III. Results for the Contradiction Detection task.

	Selection	**k**	**Precision**
Without_Filtering (Baseline)	–	–	0.19
Filtering_mean (Alg. 1)	Manual	19	**0.21 (+11%)**
Filtering_mean (Alg. 1)	Automatic	30	0.20 (+5%)
Filtering_mean (Alg. 1)	Automatic	60	0.20 (+5%)
Filtering_mean (Alg. 1)	Automatic	90	0.20 (+5%)
Filtering_max (Alg. 2)	Manual	19	0.24 (+26%)
Filtering_max (Alg. 2)	**Automatic**	**30**	**0.26 (+37%)**
Filtering_max (Alg. 2)	**Automatic**	**60**	**0.26 (+37%)**
Filtering_max (Alg. 2)	Automatic	90	0.25 (+32%)

With the manual selection of positive/negative words, both variations achieved improvements in precision. However, the biggest advantage was yielded by max_similarity (Alg. 2). On the other hand, with the automatic selection of positive/negative words, only max_similarity (Alg. 2) achieved significant improvement in precision.

We believe that the mean_algorithm performed poorly in cases in which words that were not significant reduced the mean value impacting negatively on the classification results. The improvements in this task are directly dependent on the results achieved in the classification task.

5.2.3 *Discussion.* Here we discuss some of the main findings and limitations of this work.

Manual vs. Automatic Selection. Looking at the results of the proposed method for automatically selecting the positive/negative words, we can see that with Alg. 2, its F-measure is very close to the F-measure of the manual method (only two percentage points lower). In terms of precision, it even outperforms the baseline (for $k = 90$). In terms of recall, however, the results are worse. When the mean similarity algorithm (Alg. 1) is considered, the results of the automatic selection are worse. This decrease arises from the presence of noise on the selected k positive/negative words and mainly because the way that the polarity is calculated (average). Also, recall that the reviews used as basis for the automatic selection of words come from the external dataset (which is not the same dataset used for testing). Given that the results are close, the automatic method could be used as an alternative to the manual selection.

Error Analysis. The polarity-orientation algorithm (Alg. 3) using Alg. 1 with both manual and automatic selection of positive/negative words suffers with sentences that start with an overall (positive/negative) evaluation followed by some (negative/positive) evaluation such as *"great app but it's lacking the feature to play audio while taking notes in bookmark"*. In this type of sentences, the overall sentiment is lost when it is averaged with the other additional evaluations.

Limitations. Since the proposed algorithms are based on the similarity of isolated words without considering the proximity among words, we do not cover cases such as the existence of negated terms nor compounds.

6. CONCLUSION

In this work, we extended and adapted a framework for contradiction detection in the text of online reviews. We proposed and evaluated two simple similarity metrics which serve as the bases for a polarity-assignment algorithm. Also, we proposed a filtering algorithm to improve classification performance. An experimental evaluation on real reviews has shown that our method can improve the detection of sentiment-based contradictions.

In addition, we proposed and tested an automatic method for choosing the k most representative positive and negative words which are needed for polarity detection. This modification showed that our max_similarity algorithm (Alg. 2) improves the baseline even when the selection of the positive/negative words is automatically performed on an external dataset. On the other hand, the results obtained with the mean_similarity algorithm (Alg. 1) were worse as, in some cases, the overall sentiment is lost when it is averaged. Finally, an important observation is that the results for contradiction detection improve in comparison to the baseline even with Alg 1.

As future work, we can explore other ways of comparing sets of words. For example, instead of comparing the words of a sentence with two independent sets (k-positive, k-negative), we could check for the existence of an antonymy relationship between the two sets.

REFERENCES

Bird, S., Klein, E., and Loper, E. Learning to Classify Text. In J. Steele (Ed.), *Natural Language Processing with Python.* O'Reilly Media, Inc., Sebastopol, CA, USA, pp. 221–257, 2009.

Chen, C., Ibekwe-SanJuan, F., SanJuan, E., and Weaver, C. Visual analysis of conflicting opinions. In *IEEE Symposium On Visual Analytics Science And Technology.* Baltimore, MD, USA, pp. 59–66, 2006.

de Marneffe, M., Rafferty, A. N., and Manning, C. D. Finding contradictions in text. In *ACL, Proceedings of the Annual Meeting of the Association for Computational Linguistics.* Columbus, OH, USA, pp. 1039–1047, 2008.

Dori-Hacohen, S., Allan, J., Kazai, G., Rauber, A., and Fuhr, N. Sentiment and Opinion. In A. Hanbury (Ed.), *Automated Controversy Detection on the Web.* Springer International Publishing, Berlin, pp. 423–434, 2015.

Ennals, R., Byler, D., Agosta, J. M., and Rosario, B. What is disputed on the web? In *Proceedings of the Workshop on Information Credibility.* Raleigh, NC, USA, pp. 67–74, 2010.

Ennals, R., Trushkowsky, B., and Agosta, J. M. Highlighting disputed claims on the web. In *Proceedings of the International Conference on World Wide Web.* Raleigh, NC, USA, pp. 341–350, 2010.

Galley, M., McKeown, K., Hirschberg, J., and Shriberg, E. Identifying agreement and disagreement in conversational speech: Use of bayesian networks to model pragmatic dependencies. In *Proceedings of the Annual Meeting on Association for Computational Linguistics.* Barcelona, Spain, pp. 669–676, 2004.

Harabagiu, S., Hickl, A., and Lacatusu, F. Negation, contrast and contradiction in text processing. In *Proceedings of the National Conference on Artificial Intelligence.* Boston, MA, USA, pp. 755–762, 2006.

Hillard, D., Ostendorf, M., and Shriberg, E. Detection of agreement vs. disagreement in meetings: Training with unlabeled data. In *Proceedings of the Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology: Companion Volume of the Proceedings of HLT-NAACL –short Papers.* Edmonton, Canada, pp. 34–36, 2003.

Liu, B. Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies* 5 (1): 1–167, 2012.

Liu, B., Li, X., Lee, W. S., and Yu, P. S. Text classification by labeling words. In *Proceedings of the National Conference on Artifical Intelligence.* San Jose, California, pp. 425–430, 2004.

Meeker, M. Internet trends-code conference. *Glokalde* 1 (3): –, 2015.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. Efficient estimation of word representations in vector space. *CoRR - Computing Research Repository - arXiv.org* vol. abs/1301.3781, pp. –, 2013.

Padó, S., de Marneffe, M., MacCartney, B., Rafferty, A. N., Yeh, E., and Manning, C. D. Deciding entailment and contradiction with stochastic and edit distance-based alignment. In *Proceedings of the Text Analysis Conference, TAC.* Gaithersburg,MD, USA, pp. –, 2008.

Ritter, A., Downey, D., Soderland, S., and Etzioni, O. It's a contradiction—no, it's not: A case study using functional relations. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing.* Honolulu, HA, USA, pp. 11–20, 2008.

Sangani, C. and Ananthanarayanan, S. Sentiment analysis of app store reviews. http://cs229.stanford.edu/proj2013, 2013.

Socher, R., Perelygin, A., Wu, J. Y., Chuang, J., Manning, C. D., Ng, A. Y., and Potts, C. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing.* Seattle, WA, USA, pp. 1631–1642, 2013.

Suarez Vargas, D. and Moreira, V. P. Detecting contrastive sentences for sentiment analysis. In *Proceedings of the Brazilian Symposium on Databases.* Quitandinha, PetrÃşpolis, BR, pp. –, 2015.

Suarez Vargas, D. and Moreira, V. P. Identifying sentiment-based contradictions. In *Proceedings of the Brazilian Symposium on Databases.* Salvador, Bahia, BR, pp. 76–87, 2016.

Tsytsarau, M. and Palpanas, T. Survey on mining subjective data on the web. *Data Mining and Knowledge Discovery* 24 (3): 478–514, 2012.

Tsytsarau, M., Palpanas, T., and Denecke, K. Scalable detection of sentiment-based contradictions. *DiversiWeb, WWW* 2011 (1): 9–16, 2011.