Identifying Sentiment-Based Contradictions

Danny Suarez Vargas, Viviane Moreira

[1]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – 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 paper 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. 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.*

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 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 some topic or aspect as we can see in the examples of reviews shown in Table 1. In order to highlight the contradictions, an additional step performing a more fine-grained analysis, is necessary and this is the focus of this paper.

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. Analysing 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. 2010a, Ennals et al. 2010b], 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 is a three step method performed from a similarity-based approach. 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%.

2. Problem Definition

The detection of sentiment-based contradictions based on a contradiction measure was addressed earlier by Tsytsarau *et al.* [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. $M_1 = \sum_{i=1}^{n} P_i$ and $M_2 = \sum_{i=1}^{n} P_i^2$ are the first and second order moments of the polarity 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, it 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 1 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.

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**

Table 1. Contradiction vs Contrast

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 of 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 *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. 2010a] 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 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 and Allan 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 topics for each sentence of the input data. The second step assigns a sentiment to each sentence-topic pair. Then, contradiction analysis is performed in 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 later 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].

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 to find 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 the similarity of words, more specifically, on the cosine between the vector representation of two groups of words (group of k-positive/negative words and words of a given input sentence). The way to retrieve 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.

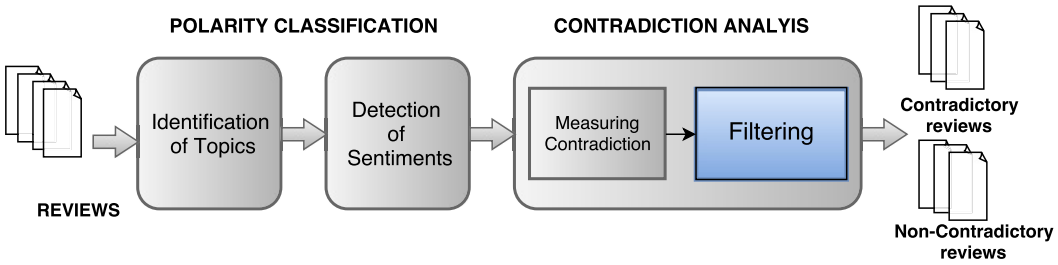

Figure 1. Extended sentiment-based contradiction analysis framework

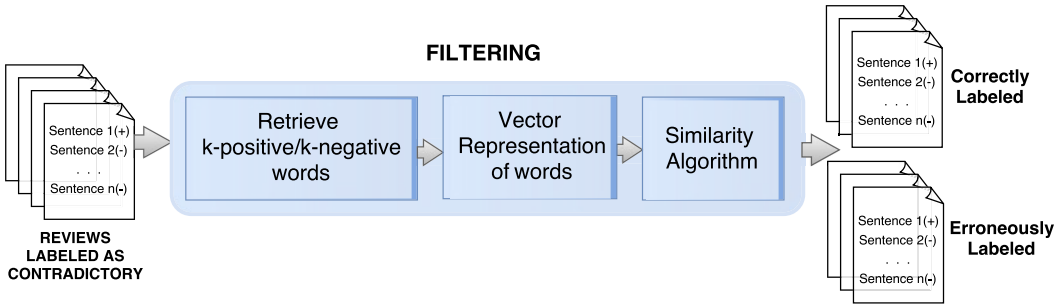

Figure 2. Similarity-based filtering method

4.2. Extending the Framework

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. So, we decide to use the high dimensional word vectors 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 $i \in \{1,2,...,k\}$

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

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.

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
$first\_array \leftarrow []$;
for $i \leftarrow 0$ **to** $n-1$ **do**
  $second\_array \leftarrow []$;
  for $j \leftarrow 0$ **to** $m-1$ **do**
    $simil \leftarrow cos\_distance(Word2Vec(A[i]), Word2Vec(B[j]))$;
    if $(simil >= -1)$ **then**
      $second\_array.add(simil)$;
    end
  end
  $second\_array\_without\_outliers \leftarrow remove\_outliers(second\_array)$;
  $mean\_value \leftarrow mean(second\_array\_without\_outliers)$;
  $first\_array.add(mean\_value)$;
end
$resp \leftarrow mean(first\_array)$;

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
$first\_array \leftarrow []$;
for $i \leftarrow 0$ **to** $n-1$ **do**
  $second\_array \leftarrow []$;
  $max\_simil \leftarrow -2$;
  for $j \leftarrow 0$ **to** $m-1$ **do**
    $simil \leftarrow cos\_distance(A[i], B[j])$;
    if $(simil > max\_simil)$ **then**
      $max\_simil \leftarrow simil$;
    end
  end
  if $(max\_simil >= -1)$ **then**
    $first\_array.add(max\_simil)$;
  end
end
$resp \leftarrow mean(first\_array)$;

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 value t. 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 below.

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$\leftarrow SP - SN$;
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

5.1. Experimental Setup

Dataset. Our dataset is composed of users' reviews about Android applications. The reviews were collected from the Google Play Store [Sangani and Ananthanarayanan 2013]. This type of dataset was selected as it is expected to have a considerable number of contrastive sentences in its reviews. The 31500 reviews are split into seven groups according to the application they refer to. Each of them contains 3500 reviews in English about a different Android application. Each review, contains information on reviewer ID, creation time, rating (from 1 to 5), and review text. For the experiments, we only used the review text and its rating. The positive reviews (4 and 5 stars are the most frequent).

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 to capture 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

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

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 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 $0,0005$. After that, we perform 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 Negative Words. In this step, word selection was done by manually 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[2]. *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"}.

Filtering Method. In order to determine which part of the reviews should be filtered, we performed the steps detailed below. 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 in 0.05 and the k-positive/negative selected words. Finally, we use Algorithm 4 to determine which reviews should be filtered.

[2]We have experimented with another set of words taken from a different set of reviews from the same site. The results have shown a small variation of 0.1 percentage points.

Table 2. Results on the classification task.

	Precision	**Recall**	**Accuracy**	**F1**
RNTN(Baseline)	**0.87**	0.60	0.63	0.71
Mean_Similarity (Alg. 1)	0.81 (-7%)	0.94 (+57%)	0.78 (+24%)	0.87 (+23%)
Max_Similarity (Alg. 2)	0.83 (-5%)	**0.95** (+58%)	**0.82** (+30%)	**0.88** (+24%)

Polarity Classification Experiment. The polarity classification is performed over the 1680 sentences by using our polarity_orientation Algorithm 4 with the parameter value t fixed at 0. We performed two experiments, one for each of our two similarity algorithms(max_similarity, and mean_similarity) which take as the parameter each sentence of the 1860 selected sentences and the k-positive/negative selected words (Word2Vec is used in our algorithms in order to retrieve the vector representation of words).

Contradiction Detection Experiment. This experiment consists in two parts. The first part is the contradiction detection by using the adapted framework without our filtering method (relying only on the C value) and by considering 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 the Contradiction Detection with Filtering Method, we calculated the accuracy of the adapted framework without our filtering method and the accuracy of the framework with 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

Results for the polarity_orientation algorithm using the mean_similarity algorithm and the max_similarity algorithm are shown in Table 2, and the results for Contradiction Detection are shown in Table3. The best results are shown in bold.

Polarity Classification. The polarity-orientation algorithm (Algorithm 3) using the two similarity measures (Algorithms 1 and 2) was compared with the RNTN classifier. The results showed that with both the max and the mean, there are gains in recall, accuracy, and F1. The proportional improvement is shown between brackets in Table 2. 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 show 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 (≈50Gb).

Contradiction Detection. For detecting contradictions, we employed our filtering Algorithm (Alg. 4) with the two variations of the polarity-orientation algorithm. Both variations achieved an improvement in precision, however, the biggest advantage was yielded by max_similarity (Alg. 1). We believe that the mean_algorithm suffered with cases in

Table 3. Results on the Contradiction Detection task.

	Precision
Without_Filtering(Baseline)	0.19
Filtering_mean (Alg. 1)	0.21 (+10%)
Filtering_max (Alg. 2)	**0.24 (+26%)**

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.

Error Analysis. The polarity-orientation algorithm (Alg. 3) using Alg. 1 suffers with sentences that start with an overall (positive/negative) evaluation followed by some (negative/positive) evaluations 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 with other words, we do not cover some cases such as the existence of negation terms nor phrasal words.

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 have shown that our method can improve the detection of sentiment-based contradictions.

As future work, we plan to design an automatic method for choosing the k most representative words. This could be implemented using logistic regression or clustering. We can also explore other ways to compare sets of words. For example, instead of comparing the words of a sentence with two independent sets (k-positive, k-negative), we can be sure of the the existence of an antonymy relationship between the two sets.

Acknowledgments: This work was partially supported by CNPq/Brazil. D. S. Vargas received a grant from CAPES.

References

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

de Marneffe, M., Rafferty, A. N., and Manning, C. D. (2008). Finding contradictions in text. In *ACL 2008, Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics, June 15-20, 2008, Columbus, Ohio, USA*, pages 1039–1047.

Dori-Hacohen, S. and Allan, J. (2015). *Automated Controversy Detection on the Web*, pages 423–434.

Ennals, R., Byler, D., Agosta, J. M., and Rosario, B. (2010a). What is disputed on the web? In *Proceedings of the 4th workshop on Information credibility*, pages 67–74. ACM.

Ennals, R., Trushkowsky, B., and Agosta, J. M. (2010b). Highlighting disputed claims on the web. In *Proceedings of the 19th international conference on World wide web*, pages 341–350.

Galley, M., McKeown, K., Hirschberg, J., and Shriberg, E. (2004). Identifying agreement and disagreement in conversational speech: Use of bayesian networks to model pragmatic dependencies. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 669.

Harabagiu, S., Hickl, A., and Lacatusu, F. (2006). Negation, contrast and contradiction in text processing. In *AAAI*, volume 6, pages 755–762.

Hillard, D., Ostendorf, M., and Shriberg, E. (2003). Detection of agreement vs. disagreement in meetings: Training with unlabeled data. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology: companion volume of the Proceedings of HLT-NAACL 2003–short papers-Volume 2*, pages 34–36.

Liu, B. (2012). Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167.

Liu, B., Li, X., Lee, W. S., and Yu, P. S. (2004). Text classification by labeling words. In *AAAI*, volume 4, pages 425–430.

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

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Padó, S., de Marneffe, M.-C., MacCartney, B., Rafferty, A. N., Yeh, E., and Manning, C. D. (2008). Deciding entailment and contradiction with stochastic and edit distance-based alignment. In *Proceedings of the 1st Text Analysis Conference (TAC'08)*.

Ritter, A., Downey, D., Soderland, S., and Etzioni, O. (2008). 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*, pages 11–20.

Sangani, C. and Ananthanarayanan, S. (2013). Sentiment analysis of app store reviews. Technical report, Stanford University.

Socher, R., Perelygin, A., Wu, J. Y., Chuang, J., Manning, C. D., Ng, A. Y., and Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on empirical methods in natural language processing (EMNLP)*, volume 1631, page 1642.

Suarez Vargas, D. and Moreira, V. (2015). Detecting contrastive sentences for sentiment analysis. In *Proceedings of the Brazilian Symposium on Databases*.

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

Tsytsarau, M., Palpanas, T., and Denecke, K. (2011). Scalable detection of sentiment-based contradictions. In *First international workshop on knowledge diversity on the Web, Colocated with WWW 2011*.