Unsupervised Classification of Opinions

Itu Vlad Vasile, Rodica Potolea and Mihaela Dinsoreanu
Department of Computer Science, Technical University of Cluj-Napoca, 26-28 Gh. Baritiu, Cluj-Napoca, Romania

Keywords: Unsupervised Learning, Opinion Mining, NLP, Domain Independent Learning, Implementation.

Abstract: Opinion mining is gaining more interest thanks to the ever growing data available on the internet. This work proposes an unsupervised approach that clusters opinions in fine grain ranges. The approach is able to generate its own seed words for better applicability to the context and eliminating user input. Furthermore, we devise a computation strategy for the influence of valence shifters and negations on opinion words. The method is general enough to perform well while reducing subjectivity to a minimum.

1 INTRODUCTION

Sentiment analysis, is the branch of natural language processing that tracks the mood of the public about a particular item. The web hosts ever growing free information about virtually everything that can be used to mine opinions about certain products. Opinion mining systems need to be accurate while applicable on any domain to which an input document may pertain to. This is usually achieved by unsupervised approaches, which eliminate the need for training data sets and increase throughput by reducing human input.

Usually, unsupervised systems perform worse than supervised ones, while semi-supervised (or weakly-supervised) approaches tend to be in-between them. Semi-supervised systems most often make use of seed words as a starting point of an expanding knowledge base. Seed words are usually available off-the-shelf or need little human input to be compiled. One weakness is using the incorrect seed words relative to domain. Therefore, a system which generates its own seed words is highly desirable, especially if this generation is domain-aware. This would cross the domain-dependency barrier while eliminating the need for user fine-tuning.

In this paper we focus on identifying and classifying opinion-bearing words in customer reviews of different products. The input documents are written in natural language (English) and do not contain any additional information, such as the star rating. To achieve this goal, our system follows a domain-independent, unsupervised approach to classifying opinions using as little input as possible while achieving consistent classifications. Our approach continues the work proposed originally in (Suciu et al, 2014) by improving the classification part. The main objective was to create an efficient unsupervised polarity assignation system, which generates its own seed words, considers negations and valence shifters and performs well on cross-domain corpora. We propose a parameterized system which can be fine-tuned based on certain conditions. Using this approach, we obtained a performance level similar to other unsupervised systems, while reducing subjectivity to a bare minimum.

2 RELATED WORK

The different approaches towards assigning polarity values to opinion-bearing words, use mostly external dependency polarity lexicons to retrieve polarities.

(Bakliwal et al, 2012)'s solution uses the Named Entity Recognizer technique (Hu and Liu, 2004) to identify objects and link all the adjectives and adverbs to them to create the object modifiers. They use SentiWordNet (SWN) (Esuli, 2010) to determine the polarity of each modifier and classify them as positive, negative or neutral. The strategy proposed in (Bhattacharyya, 2010) identifies subjective/objective sentences in a given input document based on a pre-compiled list of subjectivity-bearing words and the orientation strength of the main verb as retrieved from SWN. A more refined classification into four categories: positive, negative, neutral and conflict is proposed in (Hangya V. et al, 2014). They employ a method based on Distance-weighted Bag-of-Words

Vasile, I., Potolea, R. and Dinsoreanu, M.
Unsupervised Classification of Opinions.
DOI: 10.5220/0006069903600366
In *Proceedings of the 8th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2016) - Volume 1: KDIR*, pages 360-366
ISBN: 978-989-758-203-5

features to determine opinions and their related features. SWN is also used to create a summation of polarities in the input document and, thus, determine the category to which it belongs. Early approaches that take into consideration intensifiers and diminishers, such as (Zhang, 2012), consider their influence towards broad categories of words conveying different intensities such as *extreme, high, moderate* and *standard*. Modifiers have different influence based on the category of the word they modify. The *force* and *focus* metrics are computed to reflect the opinion intensity with respect to negation and other modifiers. The limitations consist in the actual need to classify words into intensity categories and the subjectivity that comes with both this classification and the non-numeric influence of modifiers. In (Cellier, 2014) an unsupervised approach is presented, which assigns polarities retrieved from SWN to opinion words and their modifiers. They propose a formula to amplify the opinion word polarity with the modifier polarity iteratively. The shortcoming is that by not spreading the polarities throughout the input document, the results will likely contain inconsistent values.

Our work uses the formula proposed in (Cellier, 2014) and adapts it to our solution based on experimental results. We focus on obtaining high precision-recall values while maintaining a cross-domain character and totally eliminating the need of having a pre-compiled list of seed words.

3 THE PROPOSED TECHNIQUE

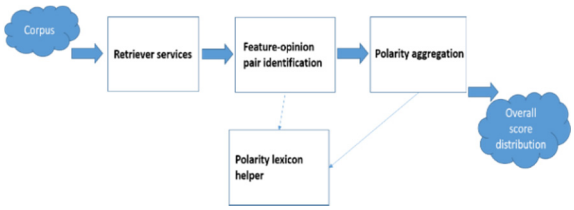

Figure 1: Pipelined modules of the system.

Our previous work (Suciu et al, 2014) describes an unsupervised domain-independent approach at mining opinions by using a weakly supervised seed word oriented methodology called Double Propagation and fine-tuning it to work with just 2 seed words: *good* and *bad*. However, to aggregate polarities using the same methodology we need a big set of seed words. The current strategy aims to completely reduce the need for seed words for the polarity assignment problem, thus fully achieving domain-independency. Furthermore, we aim to reduce the need for an acceptability range due to subjectivity constraints to a barely minimum while obtaining good performance. This is going to be achieved by fine tuning the previous methods to use all adjectives and adverbs in the parsed text as seed words and considering polarity shifters and negations. The system is composed of a pipeline of three modules described briefly below and depicted in Figure 1. We only make changes at the Polarity aggregator level, specifically on the *Seed word polarity initializer* and the *Polarity propagator* modules. The *Seed word polarity initializer* will use a different source for seed words as mentioned above and will consider polarity shifters and negations. The *Polarity propagator* module will be modified to reflect the use of polarity shifters and negations.

The Retriever module generates the *syntactic trees* from a given input corpus representing text files written in free-form in English. This preprocessing module involves: tokenization, lemmatization, part-of-speech tagging and syntactic parsing. To assign polarities to all opinion bearing words in a given text, we first need to identify them. The *Feature-opinion pair identification* component (second module in Figure 1) extracts feature-opinion pairs using the Double Propagation algorithm (Qiu, 2011). A pair contains an *opinion-bearing word* and the feature or target it directs its opinion to.

The third component, *Polarity Aggregator*, assigns polarities to the extracted opinion words and features. Polarities are assigned first to all adjectives and adverbs considered as seed words in an initialization step and then propagated text-wide by using the double propagation algorithm. This module takes the needed polarities from SWN. The possible values for polarities belong to the $[-1,1]$ interval, where -1 denotes a highly negative polarity (*utterly horrible*) and 1 represents a highly positive polarity (*excellent, perfect*). Values in the middle of the interval are associated with neutral opinions, e.g. *green car*.

Our previous work takes the following rough steps: (i) Initialize each seed word from the identified set of seed words with the associated score as extracted from the lexical resource. Performed by the *Seed word polarity initializer* sub module (Figure 1); (ii) Propagate the seed words' scores into the text by using the double propagation algorithm. This is done by the *Polarity propagator* sub module in Figure 1.

To ensure a coherent output, both an opinion word and its determined target must have the same polarity. Moreover, scores must be propagated only between the same opinion words or the same targets. Assuming there are no multiple occurring targets in

the input document, to assign polarities to different opinion bearing words, we need a number of seed words greater or equal than the number of opinion words in the input document. In case the input document contains multiple occurring targets with different opinion words, then the strategy propagates scores from target to target. In this case, it would need just one seed word that matches one of the opinion words determining that specific target. Unfortunately, this is not a reliable approach as we may not have any seed word – opinion word match. This would result in a large number of unassigned tuples.

So using just two words (*good* and *bad*) as seed words for propagating scores is not always feasible for correctly assigning polarities using the double propagation algorithm. For a weakly supervised approach, we would need a list of seed words that covers all opinion bearing words in the text. But how to compile such a list and guarantee it would eventually match every opinion word occurring in the input document? One technique would be to have a large list of seed words available for matching any of the opinion words. Unfortunately, having such an exhaustive list would introduce a lot of noise (as more seed words will match one target and influence its score). Such a technique would end up having the one target with several different polarities in the same corpus. This is a source of inconsistencies and the only feasible way of smoothing-out spikes is to perform multiple assignment runs. As shown in (Suciu et al, 2014) the results are pretty noisy as good performance (over 70% precision) is obtained only with a considerable margin of subjectivity (± 0.3).

To overcome such issues, the best approach would be one that covers all opinion words in the extracted tuples, but not more. The current work proposes the use of the opinion words in the input document as seed words and not more (that is, the seed word list is particular for each input document and is represented by the opinion words in the input, extracted in a preprocessing step). Such an approach considers the following steps: (i) Create a list of modifiers (based on SWN), (ii) Retrieve seed words (all adjectives and adverbs extracted from the input document), (iii) Handle modifiers (including negation), (iv) Assign scores to tuples (based on seed words) and (v) Propagate scores.

Create a List of Modifiers

Our polarity lexicon of choice is SWN due to its popularity and high quality contents. The list of modifiers is created by the *Seed word polarity initializer* submodule (Figure 1) by searching and extracting all *intensifiers* from SWN together with their negative and positive scores.

Retrieve Seed Words

The *Retriever* module of this system creates its output as a list of syntactic trees. Each syntactic tree is built from the words making a phrase and their part of speech. To create the list of seed words we have to simply retrieve all the extracted adjectives and adverbs in the input document. At this phase it is also important to associate all modifiers to each seed word occurring in the input document. For instance, if *good* has in the input two modifiers, i.e. *very very*, we have to associate them to the opinion word (thus considering as opinion a triple *very_very_good*, vs. *good*). For this, we use the syntactic tree structure, which represents a sentence and parse the text (right to left, starting from the opinion word) to check if it is a modifier. If so, the algorithm appends it to the word and continues until the first non-modifier word. A special case of modifier is the negation. While it does not necessarily decrease the overall score of a construct *(not + (optional_modifier) + word)*, it does inverse it (by the multiplication with (-1)). Lemmatization takes care of the *'t* negations and transforms them into *not*. This approach works for most cases but it fails if a non-modifier word intervenes. Consider the construct *not so good*. Because of the non-modifier to the left of the opinion word, the negation is never reached.

Handle Modifiers

After the list of modifiers and the list of seed words are created, they are validated and merged.

Opinion Words Containing Modifiers

The process of computing the score of a modified opinion word (OW) is an iterative one starting from the scores of the OW and the modifier(s). We propose two approaches to compute the construct score, one is a *formula* proposed in (Cellier et al, 2014) adjusted based on the experimental results (1) and the other one is a simple iterative *addition* (2):

$$\text{constructScore} = \text{sign} * (|\text{constructScore}| + (1 - |\text{constructScore}|) * \text{modifierScore}) \quad (1)$$

$$\text{constructScore} = \text{sign} * (\text{modifierScore} + \text{actualWordScore}) \quad (2)$$

The computation of the construct score results in two new additions to the seed words list: the whole construct of words *(OW + modifiers)* and only the first modifier. Each of them has the same score, the previously computed one. The reason for adding just

the first modifier is that most often there is just one modifier and if there are more, chances are that the other ones are either negation or already added modifiers.

OW Without Modifiers

If a seed word is unmodified, the strategy extracts it and assigns its polarity from the polarity lexicon.

Assign Scores to Tuples

This step performs an initial assignment of scores to all opinion words and their targets that can be matched (the same scores as the scores of their corresponding seed words). Our approach ensures that all tuples are matched (as the list of seed words is the same with the list of OW). To accurately perform this step, we propose the use of the following rules: (i) Preserve the exact modifier rule: Only assign scores from seeds with modifiers to tuples having the same modifier. This rule prevents inconsistent polarities between different modifiers as *very* conveys a different strength than *extremely*. (ii) Handle unmodified OW rule: Only assign polarities from unmodified seeds to unmodified tuples. By applying a modifier to an opinion word, its polarity is changed. Because targets and their opinion words have the same polarity, we cannot assign polarities from modified seed words to unmodified tuples because their meaning will change.

Propagate Scores

Score propagation uses the same algorithm as the second module, called the Double Propagation algorithm (Liu, 2012) and adapted to our rules (Suciu et al, 2014). This is an algorithm proven to work very well with seed words. The same propagation algorithm is used for score propagation as well. Consider we have a seed word *perfect* with a polarity of 0.8. By propagating it throughout the corpus, we find the tuple *perfect sound*. As we already know that *perfect* has a polarity of 0.8, both the opinion word of the tuple, *perfect*, and the target *sound* are assigned a polarity of 0.8, thus increasing the knowledge base. After further steps, we get to the target *sound* which has an already known polarity. We can thus propagate it and find a match for *nicest sound*. Suppose both these words are unassigned. Then, the 0.8 polarity is assigned to both the newly found *sound* target and to the *nicest* opinion word. So we now know that *nicest* also has a value of 0.8 and use this info to propagate its polarity furthermore. We call this mechanism the opinion word/target polarity propagation.

To ensure consistency, the same unmodified target should have the same polarity throughout a corpus. This makes sense as we are interested in aspect-based opinion mining and we have to consider the overall opinion for a certain target. To accomplish this, we established the following propagation rules: (i) When a modified tuple is propagating its score and it finds the same unmodified tuple, both keep their polarities, (ii) When a modified tuple matches another modified tuple that does not have the same modifier, both keep their polarities, (iii) When a modified tuple matches another modified tuple that has the same modifier, their scores are averaged; (iv) When a non-modified tuple matches a modified tuple, both keep their polarities; (v) When a non-modified tuple matches a non-modified tuple, their scores are averages; (vi) When there is a negation present, simply flip the polarity by -1. After going through all the steps, the assigned tuples are passed to the *Polarity summarization* sub module (Figure 1) to produce a meaningful output. This includes an average polarity for the entire corpus, a polarity distribution chart and the possibility to find the average polarity for a given target. There are two different evaluations of the system: one for the Feature-opinion pair identification module and one for the Polarity Aggregator module. The Feature-opinion pair identification module uses manual annotations described in more detail in (Suciu et al, 2014), while the last module uses a custom adaptation of the data set. The two dataset files used for evaluating the Polarity Aggregator module are created by manually annotating opinion words and using a tool created by us specifically for this task. The automated annotator is a simple application that recognizes already applied annotations and uses SWN to retrieve polarities which it appends to the already annotated words. It also annotates modifier words extracted from SWN in the same manner as presented in Section 3. The steps for the annotation process are as follows: (i) Manually annotate opinion words; (ii) Using an automated annotator, automatically annotate modifiers, without scores as the modifier score is not useful at this point, (iii) Using an automated annotator, automatically add polarities extracted from SWN to all manually annotated opinion words. The extracted polarities apply just to the opinion word and any modifier is ignored. At this point, no modifier rule can be applied because we would end up evaluating the same rule used for automatic annotations and also for assigning polarities, which would yield 100% results; (iv) Manually analyze the given polarities and adjust if needed by searching for the best contextual match in SWN. Considering modifiers at this point also yields more subjectively-accurate results.

The evaluation algorithm is the same as described in (Suciu et al, 2014). To account for the annotator and the reviewer's subjectivities, we propose setting a certain range of acceptability. If polarity p falls in the interval $[-range, +range]$, we consider it as being correctly assigned and increase the TP counter. There are two ways of specifying such a range: fixed and dynamic. A fixed threshold was originally employed in (Suciu et al, 2014), but we now consider it to be too permissive for accurate results. Thus, we propose and compare the use of a fixed and a dynamic range, while mentioning that the fixed range is now narrower, decreased from 0.3 to 0.1, while 0.2 creates a baseline. The dynamic range we propose is meant to become more restrictive as we approach the extremities of the assignment interval $[-1,1]$. This is entirely on par with the human thinking as extreme words such as *excellent* or *awesome* tend to convey the same meaning for different persons while neutral words seem yield different opinions. This translates into a ratio from a given number and the remainder to the end of the interval. To simplify things, the ratio is a percentage:

$$\text{Range} = (\text{percentage} / 100) * (1.0 - \text{score}), \quad \text{score} > 0 \tag{3}$$

$$\text{Range} = (\text{percentage} / 100) * (-1.0 - \text{score}), \quad \text{score} > 0 \tag{4}$$

Regardless of the range type, scores that cross the positivity line are not considered as being. For example, take a static range of 0.1, an assigned polarity of -0.06 and an annotated polarity of +0.0.3. This is an erroneous assignment because we are dealing with a positive word that was incorrectly classified as negative.

4 RESULTS

The data set we used for testing consists of two files from the ones originally proposed in (Hu and Liu, 2004). They contain user reviews for two different products. Each of them has approximately 200 phrases and a combined average of about 3900 words. They contain an average of 17 words per sentence and 19 respectively, out of which 312 and 219 respectively are opinion bearing words. The total number of opinion words is 531. The average number of opinion words per sentence is 2 for each file.

They are annotated using the method described above and will be referred to as Dataset 1 and Dataset 2 from now on. There are only 2 seed words used for the entire system: *good* and *bad*. The actual evaluation represents an extra optional step in the Polarity aggregator module and is called upon the end of the modules' execution. The system ignores the annotations on the evaluation input documents in the first step and performs normally, just as it would with any other file. The annotations are only taken into consideration in the extra evaluation step to ensure the execution is not influenced in any way.

For each experiment we measure the precision and the recall metrics. For each range we are also monitoring the effect of the 2 polarity modifier computation methods, namely (1) and (2). We aimed to evaluate the score assignment problem with support for modifiers and negations by using a fixed range of acceptability (Section 3). There are only two fixed ranges taken into consideration: 0.1 and 0.2. Our aim was to decrease this value by as much as reasonably possible. The system conveys good performance even under a restrictive subjectivity range of 0.1. Ultimately, polarities tend to converge to the interval margin instead of being normalized. It is due to this convergence of polarities when using (1) that the use of fixed subjectivity ranges is not a good fit. There is a lack of consistency between the converging polarities, which ultimately yield more precise results, and the fixed range that accepts the same magnitude of errors for all values in the interval.

To overcome this issue, we propose the use of a dynamic range which gets more restrictive as polarities approach the end of the interval. Its behavior resembles (1) which makes it a good fit for a precise evaluation. Suppose we have an assigned polarity of 0.1. By using a 10% value for the percentage variable, it computes a range of 0.09. This means that the assigned polarity of 0.1 will be considered valid if the annotated polarity is in the range of 0.1 ± 0.09. Thus, assuming the annotated polarity is 0.18, it would have a pass. Similarly, for a highly positive word like *perfect*, which has a polarity of 0.8 in SWN, it needs to fit in the range of $[0.78,0.92]$ assuming a 10% value of the percentage. Figures 2 and 3 depict the results obtained in the experiments performed with the dynamic threshold for each dataset. The X-axis represents the percentages used to compute the dynamic threshold. We can notice that the non-convergent (2) performs worse than (1) because it does not get more precise towards the interval margin. The 1% value suggests the power of this system as most the ranges are very small and do not account for any subjectivity. Thus, we can safely state that the system displays a reduced amount subjectivism. However, the most reasonable compromise is to employ the use of a 15% value for computing the dynamic threshold, where the ranges

of subjectivity are decent and the performances are very good. These results also demonstrate that a simple addition of modifiers conveys comparable results thus encouraging the cross-domain character of this system due to the use of non-amplified context-independent polarities.

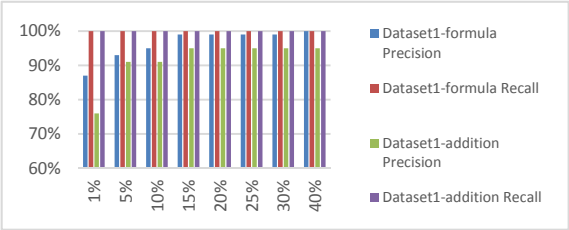

Figure 2: Dynamic range with/without addition for Dataset1.

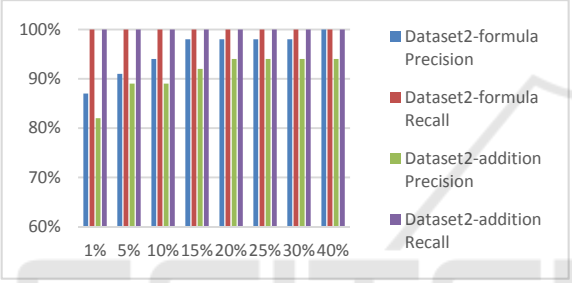

Figure 3: Dynamic range with/without addition for Dataset2.

Comparing the static and the dynamic thresholds, we can clearly see that the results are very similar at the same range (0.3 for 15% dynamic threshold) while at the extremities the dynamic thresholding shows a convergence tendency versus the more linear behaviour of the static range. This fact resembles free-speech where people tend to agree on highly positive or highly negative words. Considering this, we tend to believe that dynamic threshold with 15% percentage value applied with (1) manifests the best results while leaving just enough room to interpretation. It is worth mentioning that the Recall metric has a 100% value all throughout the testing due to the use of all the opinion bearing words as seed words and the assignation of polarities to each of them. This translates into matching all tuples, but not more, and being able to provide an accurate polarity. The results are promising given the unsupervised character of the system.

Table 1: Compares our results with related work.

	Precision (avg)	Recall (avg)
Hu and Liu, 2004	0.64	0.69
Zagibalov, 2008	0.9	0.89
Our approach	0.92	1.0

5 CONCLUSIONS

The strategy proposed in this paper devises an unsupervised cross-domain approach for classifying opinion-bearing words in natural-language written product reviews. The main challenge was to granularly classify opinion-bearing words while reducing the need for user-compiled lists of seed words, thus gaining a context-independent character. Our solution starts by identifying opinion words and their targets by comparing tuples against predefined grammatical rules and making use of a pre-compiled external resource to generate its own seed words and assign polarities in complex constructs. By taking into consideration and correcting polarity assignments based on the presence of modifiers or negations, the classifications are complete, consistent and correct. By fine-tuning the effect of modifiers on opinion words we managed to minimize the need for a subjectivity range to a bare minimum while maintaining good performances. Furthermore, we proved that an iterative converging formula is a good approach towards computing the influence of valence shifters on opinion words. This work constitutes a starting point for current seed-word-based semi-supervised approaches that may find ways to generate their own seed words based on certain context-related traits.

REFERENCES

Hu, Mingqing and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of SIGKDD'04*, pages 168-177.

D. Suciu, V. Itu, A. Cosma, M. Dinsoreanu, R. Potolea, Learning good opinions from just two words is not bad. In *6th International Conference on Knowledge Discovery and Information Retrieval KDIR 2014*.

G. Qiu, B. Liu, J. Bu, C. Chen, Opinion Word Expansion and Target Extraction through Double Propagation. In *Computational Linguistics*, March 2011, Vol. 37, No. 1: 9.27.

P. Cellier, T. Charnois, A. Hotho, S. Matwin, M-F. Moens, Y. Toussaint, Interactions between Data Mining and Natural Language Processing. In *Proceedings of 1st International Workshop, DMNLP 2014.*

D. Bhattacharyya, S. Biswas, T. Kim, A review on Natural Language Processing in Opinion Mining. In

International Journal of Smart Home, Vol. 4, No. 2, April, 2010.

V. Hangya, G. Berend, I. Varga, R. Farkas, SZTE-NLP: Aspect Level Opinion Mining Exploiting Syntactic Cues. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), pages 610-614, Dublin, Ireland, August 23-24, 2014.*

A. Bakliwal, P. Arora, V. Varma, Entity Centric Opinion Mining from Blogs. In *Proceedings of the 2nd Workshop on Sentiment Analysis where AI meets Psychology (SAAIP 2012), pages 53-64, COLING 2012, Mumbai, December 2012.*

L. Zhang, S. Ferrari, P. Enjalbert, Opinion analysis: the effect of negation on polarity and intensity. In *Proceedings of KONVENS 2012, Vienna, September 21, 2012.*

T. Zagibalov. a. J. Carroll, "Automatic seed word selection for unsupervised sentiment classification of Chinese text," in *In Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, 2008.

A. Esuli, F. Sebastiani, SentiWordNet: A Publicly Available Lexical Resource for Opinion Mining, 2010.