Mining Comparative Sentences from Social Media Text

Fabíola S. F. Pereira

Faculty of Computer Science
Federal University of Uberlândia (UFU)
Uberlândia, Minas Gerais, Brazil
fabfernandes@comp.ufu.br

Abstract. Comparative opinions represent a way of users express their preferences about two or more entities. In this paper we address the problem of comparative sentences mining focused on social medias. We propose a genetic algorithm able to mine comparative sentences from short sentences based on sequential patterns classification. A comparison among classifiers regarding comparative sentences analysis is also presented. Our results indicate better accuracy for the proposed technique against literature baseline approaches, reaching accuracy levels of 73%.

Keywords: opinion mining, comparative sentences, genetic algorithm, social media mining

1 Introduction

Comparative opinions represent a way of users express their preferences about two or more entities. Mining comparative sentences from texts can be useful in several applications. For instance, a company might be interested in social media rumors of a new product release among consumers. Or, what are the best and worst features of the new product from consumers viewpoint? Nowadays, social medias are great source of this kind of information and mining comparative opinions from them seems to be a very promising direction to unveil valuable knowledge.

Many researches have been done in the field of regular opinion and sentiment classification [3,2]. However, *comparative opinions* represent a different viewpoint of users and an interesting research area. According to [8], a regular opinion about a certain car X is a statement like *"car X is ugly"*. On the other hand, a comparison is like *"car X is much better than car Y"*, or *"car X is larger than car Y"*. Clearly, these sentences have rich information from which we can extract knowledge with specific mining techniques.

In [5] the authors proposed a classification technique for mining comparative sentences based on grammatical sequential patterns. In this paper, based on [5]'s background, our goal is to stress techniques for mining comparative sentences focused on Twitter social data analysis. We argue that social medias *corpora*, as

In: P. Cellier, T. Charnois, A. Hotho, S. Matwin, M.-F. Moens, Y. Toussaint (Eds.): Proceedings of DMNLP, Workshop at ECML/PKDD, Nancy, France, 2014.

a great source of users opinions, must be explored and specific mining algorithms are needed.

The main contributions of this paper are: (1) a publicly available dataset crawled from Twitter. We manually labeled 1,500 tweets as comparative or non-comparative; (2) the genetic algorithm GA-CSR to aggregate to the problem of mining comparative sentences; and (3) a set of experiments comparing our approach with state-of-the-art techniques.

This paper is organized as follows: in Section 2 we introduce the problem of mining comparative sentences, highlighting social medias texts. In Section 3 we discuss techniques proposed in related work and present our proposal. In Section 4 the experimental results are showed. Finally, Section 5 concludes the paper.

2 The Problem of Mining Comparative Sentences

In the context of our study, comparative opinions are opinions that express a relation based on similarities or differences between two or more entities. According to [6], there are four types of comparisons: *non-equal gradable comparison* ("XBox is better than Wii-U"), *equative* ("XBox and Wii-U are equally funny"), *superlative* ("XBox is the best among all video games") and *non-gradable comparison* ("XBox and Wii-U have different features"). The first three types are called *gradable comparative* and are our focus because the sentences allow to establish a preference order among entities being compared.

Definition 1 (Comparative Opinion [6]). *Comparative opinion is a sextuple (E1, E2, A, PE, h, t), where* $E1$ *and* $E2$ *are the entity sets being compared based on their shared aspects A,* $PE \in \{E1, E2\}$ *is the preferred entity set of the opinion holder h, and t is the time when the comparative opinion is expressed. For a superlative comparison, if one entity set is implicit (not given in the text), we can use a special set* U *to denote it. For an equative comparison, we can use the special symbol* $EQUAL$ *as the value for* PE*.*

Example 1. Let us consider the following comparative sentence: "@stephthelamekid tbh wii u games do have better graphics than ps4 and xbox 1 games", posted by user Dinotia_4 in 12/06/2014. The comparative opinion extracted is: ({Wii U games}, {PS4 games, XBox One games}, {graphics}, {Wii U games}, Dinotia_4, 12/06/2014)

One challenge on the problem of comparative sentences is that not all sentences with POS tags JJR, RBR, JJS and RBS (comparative and superlative POS tags) are comparative. For example, *"faster is better."* Moreover, some expressions are comparative, but just can be identified through context, e.g *"PS4 is expensive, but Wii-U is cheap."*

3 Comparative Sentences Mining Techniques

To the best of our knowledge, the most representative technique in literature that addresses the problem of comparative sentences is [5], which is based on sequential pattern mining. In the following we present two new approaches: a naive approach based on *n-grams* classification and a genetic algorithm approach. The technique from [5] is also summarized in this Section.

3.1 *N-grams* Classification

The technique of document representation through term vector is the most common in the sentiment analysis field and can be used as our baseline. In this approach, each sentence in the corpus is a *document*, *terms* are the most relevant words and we use TF-IDF matrix [6] to represent them. Such matrix is, therefore, submitted to a classifier that builds a model able to identify whether a given sentence is comparative or not.

In this work, just *unigrams* have been considered. We did three pre-processing steps: (1) stop words removal, (2) stemming and (3) 1000 features extraction based on information gain index [10]. As we will present in Section 4, the results obtained with this approach were not expressive, even varying the classification algorithms.

3.2 Sequential Patterns Classification

Sequential patterns classification for comparative sentences mining had been proposed in [5]. Sequential pattern mining (SPM) is an important data mining task [1]. A sub-sequence is called sequential pattern or frequent sequence if it frequently appears in a sequence database, and its frequency is no less than a user-specified minimum support threshold *minsup* [4].

According to [5], a class sequential rule (CSR) is a rule with a sequential pattern on the left and a class label on the right of the rule. Unlike classic sequential pattern mining, which is unsupervised, in this approach sequential rules are mined with fixed classes. This method is thus supervised. For a formal definition, please refer to [5].

After defined the task of mining class sequential rules, then we deploy the algorithm to our problem. However a sentence cannot be handle simply from raw words, as we did on *n-grams* classification approach (Subsec. 3.1). To find sequential POS tags patterns in sentences and, then, build an input dataset of sentences to be classified (supervised learning) as comparative or non-comparative the following steps are needed:

1. **Sentences with pivot keywords.** Many words in English language indicate comparisons, for example *beat*, *exceed*, *outperform* etc. Moreover, those ending with *-est* and *-er* are naturally comparative or superlative adverbs and adjectives. Thus, a set of comparative keywords is considered. The idea is to identity sentences with at least one keyword and use the words that are within the radius of 3 of each keyword in the sentence as a sequence in our data.
2. **Replacing with POS tags.** For each sequence of max length of 7 obtained in previous phase, replace all words with their corresponding POS tags.

3. **Labeling sequences.** For each sequence, we have to label it as *comparative* or *non-comparative*. This is the same label that originated the sequence.
4. **Generating CSR.** In this phase we have to mine sequential patterns. The algorithm PrefixSpan [9] have been used with minimum support 0.1 and minimum confidence 0.6.
5. **Building dataset for classification task.** To translate class sequential rules into input for classification algorithms, the following steps are considered: each CSR is a feature. The classes are *comparative* and *non-comparative*. Each sentence from original corpus is a tuple in dataset. If the sentence matches a given CSR, the value is 1. Otherwise, 0. Each sentence keeps with its class. In this way, we have a well-formed input to a classifier algorithm.
6. **Running the classifier.** In paper [5] the authors use just the Naive Bayes classifier. In our experiments, we also considered the algorithms SVM, Multi-Layer Perceptron (MLP) and Radial Basis Function (RBF).

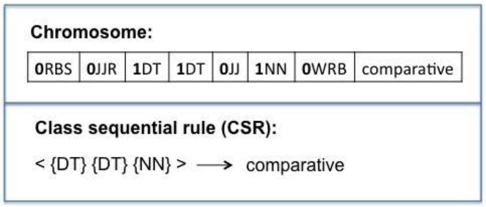
Example 2. In order to illustrate steps 1 to 4, let us consider the sentence: *"this/DT game/NN is/VBZ significantly/RB more/JJR fun/NN with/IN Kinect/NN than/IN without/IN it/PRP."* It has the keyword *more* and the generated CSR is:

$$<\{\text{NN}\}\{\text{VBZ}\}\{\text{RB}\}\{\text{moreJJR}\}\{\text{NN}\}\{\text{IN}\}\{\text{NN}\}> \rightarrow \text{comparative}$$

3.3 Genetic Algorithm

In this paper we propose a genetic algorithm for mining comparative sentences. The idea is to mine class sequential rules (CSR) from [5] (Subsec. 3.2). However, we do not use a classifier, but a genetic algorithm (GA-CSR) to get rules.

Each chromosome represents a CSR. Chromosomes have fixed length of 8 genes, where the first 7 are the sequential patterns with itemsets of length 1 (default gene) and the last one is the class with value *comparative* or *non-comparative* (class gene). Each default gene is an itemset and can assume POS tags domain values. Moreover, for each default gene we have the additional bit '1' or '0' representing whether or not it is part of sequential pattern. The example chromosome coding and its meaning is described in Figure 1. In the following we detail GA-CSR features.


Chromosome:

0RBS	**0**JJR	**1**DT	**1**DT	**0**JJ	**1**NN	**0**WRB	comparative

Class sequential rule (CSR):

< {DT} {DT} {NN} > ⟶ comparative

Fig. 1: Chromossome coding

- **Fitness.** The rules fitness ($Fitness$) in the population is calculated based on a function containing two terms, namely Specificity (Sp) and Sensitivity (Se), where $Sp = TN/(TN + FP)$, $Se = TP/(TP + FN)$ and $Fitness = Se * Sp$. The variables TP, TN, FP and FN correspond to true positives, true negatives, false positives and false negatives, respectively.
- **Population creator.** The population is randomly generated.
- **Selection and crossover.** Two best chromosomes are selected by applying roulette wheel selection method and two point crossover method is applied over them to generate new children chromosomes. The class gene is not considered.
- **Mutation.** The mutation process changes the value of an attribute to another random value selected from the same domain. It can occurs in any gene type and does not consider the flag bit of each gene.
- **Insertion and removal operator.** Insertion and removal operators control the size of a rule. Insertion operator activates the gene by setting its flag bit and removal operator deactivates a gene by resetting the flag bit with a varying probability P_i and P_r, respectively.
- **Survivor selection.** GA-CSR uses fitness-based selection where individuals are selected from the set composed by parents and offspring. The top T_p fitness individuals are selected, where T_p is the population size.

In the end, we have a set of class sequential rules and just those greater than minimum support and confidence are considered for test and model validation.

4 Experimental Settings

In this section we report our experiments. We aim to compare best classification accuracies. Section 4.1 describes the datasets used to train and test the models. It also presents our experiments set-up and parameter setting. Finally, we expose our results in terms of success rate of the classifiers in Section 4.2.

4.1 Datasets and Parameterization

We tested our algorithms over two datasets: Amazon product reviews and Twitter texts. Our goal is to show how text mining social medias is different because of specific features of text length and language.

The Amazon product review is about mp3 players and was obtained from [7]. Twitter dataset contains tweets about PlayStation 4 and XBox video games and we collect them from Twitter API[1]. Both datasets were manually labeled. In Table 1 we detail the datasets features.

Our test set is composed by 9 runs for each dataset. For the *n-grams* approach, we used 4 classifiers: SVM (SVM-Unigram), NB (NB-Unigram), MLP (MLP-Unigram) and RBF (RBF-Unigram). In the CSR approach we also use 4 classifiers: SVM-CSR, NB-CSR, MLP-CSR and RBF-CSR. Finally, we run our proposed genetic algorithm GA-CSR. In Figure 2 we present detailed parameters used for each approach.

[1] https://dev.twitter.com/rest/public

	DB-Amazon	DB-Twitter
# sentences	1000	1500
# comparative sentences	97 (9.7%)	199 (13.26%)
Texts dates	2003-2007	Dec 2014
Topic	mp3 players	XBox and PS4

Table 1: *Datasets* used for tests

	Unigrams	
Train/Test	10-fold cross-validation	
Pre-process	stop words, stemm, infogain	
# Features	1000	
	MLP-Unigram	RBF-Unigram
Momentum	0.8	-
Learning rate	0.6	-
# neurons in hidden layer	15	2
# hidden layers	1	1

(a) Parameters for *n-grams* approach

	CSR	
Train/Test	10-fold cross-validation	
Radius	3	
minsup	0.1	
minconf	0.6	
Sequential pat. algorithm	PrefixSpan	
	MLP-RCPS	RBF-RCPS
Momentum	0.8	-
Learning rate	1.0	-
# neurons in hidden layer	7	7
# hidden layers	1	1

(b) Parameters for CSR approach

	GA-CSR
Train/Test	10-fold cross-validation
Learning rate (T_m)	0.8
Crossover rate (T_c)	0.8
Insertion rate (P_i)	0.3
Removal rate (P_r)	0.3
minsup	0.1
minconf	0.6
# generations	100
Population size (T_p)	50
Fitness	$Se * Sp$

(c) Parameters GA-CSR approach

Fig. 2: Parameterization

4.2 Experimental Results

The first test set was performed over DB-Amazon dataset (Figure 3). We can observe a poor performance for *n-grams* approach. As expected, it is a simple baseline that does not take into account elaborated features of our mining problem. Varying classifiers algorithms does not impact on results that reach a maximum accuracy of 68.6% for RBF neural network.

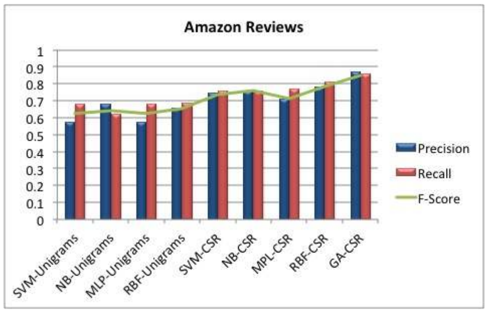


Fig. 3: Experimental results over DB-Amazon

Regarding CSR approach from [5], the results were similar to original paper. The difference is that in [5] just Naive Bayes classifier had been used. In our experiments we also considered other classification algorithms. The neural network RBF-CSR reached the best accuracy of 81.13%. Finally, our proposed genetic algorithm reached 85.23% of accuracy indicating the best approach for DB-Amazon dataset.

The second test set ran over DB-Twitter (Figure 4). Graphics curves maintained the trend, however the average accuracy decreased around 10%. This can be explained due to the large amount of noise in Twitter texts. Moreover, sentences grammatical errors potentially harm the grammatical pattern approaches.

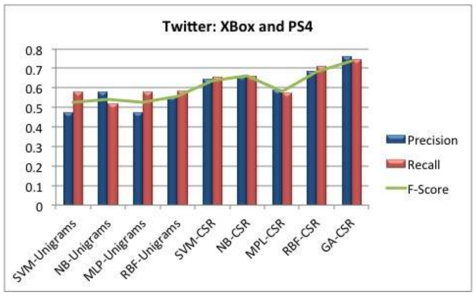


Fig. 4: Experimental results over DB-Twitter

5 Conclusion

In this paper we addressed the problem of mining comparative sentences. We carried out an experiment using 1,500 short sentences from Twitter.com, equally divided into two domain categories: *comparative* and *non-comparative* sentences. The results showed that the higher success rate was obtained with our genetic algorithm approach (73%). As our sample is relatively small, we used cross-validation (10-fold) to avoid overfitting and increase the accuracy of the success rate of the classifiers.

To ensure reproducibility of our results, in conjunction with the publication of this paper, we have released the full genetic algorithm GA-CSR code and Twitter data in the format used by our algorithm[2].

As future work, once mined comparative sentences, our focus will be on mining user preferences. We consider that comparative sentences are good source of users opinions, enabling the development of reasoning user preferences models from social data.

References

1. Agrawal, R., Srikant, R.: Mining sequential patterns. In: Proceedings of the Eleventh International Conference on Data Engineering. pp. 3–14. ICDE '95 (1995)
2. Arias, M., Arratia, A., Xuriguera, R.: Forecasting with twitter data. ACM Trans. Intell. Syst. Technol. 5(1), 8:1–8:24 (2014)
3. Ceron, A., Curini, L., Iacus, S.M.: Using sentiment analysis to monitor electoral campaigns: Method matters-evidence from the united states and italy. Soc. Sci. Comput. Rev. 33(1), 3–20 (2015)
4. Fournier-Viger, P., Wu, C.W., Tseng, V.: Mining maximal sequential patterns without candidate maintenance. In: Advanced Data Mining and Applications, vol. 8346, pp. 169–180 (2013)
5. Jindal, N., Liu, B.: Identifying comparative sentences in text documents. In: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 244–251. SIGIR '06 (2006)
6. Liu, B.: Sentiment Analysis and Opinion Mining. Morgan Claypool Pub. (2012)
7. McAuley, J., Leskovec, J.: Hidden factors and hidden topics: Understanding rating dimensions with review text. In: Proceedings of the 7th ACM Conference on Recommender Systems. pp. 165–172. RecSys '13 (2013)
8. Pang, B., Lee, L.: Opinion mining and sentiment analysis. Foundations and Trends in Information Retrieval 2, 1–135 (2008)
9. Pei, J., Han, J., Mortazavi-Asl, B., Wang, J., Pinto, H., Chen, Q., Dayal, U., Hsu, M.C.: Mining sequential patterns by pattern-growth: The prefixspan approach. IEEE Trans. on Knowl. and Data Eng. 16(11), 1424–1440 (Nov 2004)
10. Sharma, A., Dey, S.: An artificial neural network based approach for sentiment analysis of opinionated text. In: Proc. of the 2012 ACM Research in Applied Computation Symposium. pp. 37–42 (2012)

[2] http://lsi.facom.ufu.br/~fabiola/comparative-mining