Sentiment Analysis of Korean Teenagers' Language Based on Sentiment Dictionary Construction

Jason Kim[1], Min Kyoung Kim[1], Yeoeun Park[1], Eomji Kim[1], Junhee Lee[1], Dongho Kim[1(✉)], and Seonho Kim[2]

[1] Dongguk University, Seoul 04620, Korea
dongho.kim@dgu.edu
[2] University of Southern California, Los Angeles, CA 90007, USA

Abstract. This paper is to introduce Korean teenagers' sentiment analysis through composing the Korean sentiment dictionary for teenagers' language which is using KoNLPy and an algorithm for text sentiment analysis based on N-Gram concept which is used to calculate sentiment score from sentences. We gathered the text data (sentences) by web crawling several Korean web sites and using the Korean sentiment dictionary and algorithm described above, we conducted an experiment to analyze sentiment of the public.

Keywords: Text sentiment analysis · Korean sentiment dictionary · Informal language · Teenager · Web crawling · KoNLPy

1 Introduction

In modern society, people tend to get information by searching comments on articles, reviews and criticisms. For this reason, 'opinion mining', which is a technology that extracts and discriminates writer's sentiment and opinion from text on the internet, is important and can be used in various categories. By text sentiment analyzing, enterprise can make efficient marketing strategies based on people's sentiment. It also can reduce huge amount of cost in investigating public opinion on social issues. Besides, there are endless applications to this technology.

It is important to analyze teenagers' text sentiment among other social groups. There are two reasons of that. It, first of all, was easy to collect teenagers' text data since they tend to share opinion and information through the internet. Secondly, there has been no research on teenage word sentiment analysis. Since the young generation is using their own 'coined word', the older generation has difficulty understanding their informal language that is abbreviated, compounded and twisted. Such language barrier between the generations induces communication disconnection in society.

This research suggests a method of analyzing the sentiment of teenagers' language by crawling comments data on TV Programs related websites. Thus, it will be possible to alleviate language barriers among various social groups and making them possible to communicate smoothly with teenager's informal language and analyzing the sentiment of it.

J. C. Hung et al. (Eds.): FC 2018, LNEE 542, pp. 541–550, 2019.
https://doi.org/10.1007/978-981-13-3648-5_63

1.1 Related Work

Sentiment analysis is a way to guess and classify sentiment of visual content and textual content into positive and negative. It is based on words that contain polarity for emotion in sentences. A dictionary of words containing polarity for emotion is called sentiment dictionary. Therefore, the process of sentiment analyzing the positive and the negative sentence depends heavily on the sentiment dictionary. A typical example of how to construct sentiment dictionaries is the SentiWordNet which evaluates the positive/negative/neutral polarity of words using PMI. However, the word polarity value of the sentiment dictionary may be different depending on the do-main, because the same word may indicate positive meaning in one sentence but negative in another sentence according to a specific domain.

Research on Korean sentiment dictionary construction consists of studying the polarity of words using collective intelligence [1], constructing sentiment dictionary by translating SentiWordNet into Korean and applying it to Twitter [2], sentiment dictionary construction using machine learning by collecting sentences including seven classes about emotions such as anger, confusion, and depression [3]. In the previous research, sentiment analysis was applied to Twitter or Naver Movie review based on the sentiment dictionary, but it was considered only in the sentence composed of single morpheme of 1-Gram method. For example, 'not beautiful' indicates the meaning of negation, but in the preceding research, it cannot be analyzed negative or positive because it is positive polarity word 'beautiful'. In order to overcome this problem, this study has solved sentiment analysis of negative sentences or other complex sentences by using 'Contrary' which reverses the polarity value of sentences in sentiment dictionary.

2 Methods

2.1 Project Overview

Our goal is to detect Korean teenagers' sentiment from comments on the web. Since it is practically impossible to construct all the sentiment words, we decided to concentrate on making teenagers' informal sentiment words dictionary, which does not exist today. Teenagers usually transform, combine and reduce the existing words. The words are emerging and declining by trendy issues thus it is hard for older generation to fully understand what teenagers are talking. We thought that it could not only contribute to break down the language barrier between teenagers and older generations but also make decision to grasp teenagers' opinion which is strongly related to public opinion.

We had to find out, because of importance of teenagers' opinion, the area that has many data related with teenagers. We found out TV program, especially the entertainment show program is a hot issue among them and chose it as a data set. By choosing TV show program as the data set, it was easy to accumulate data by crawling website.

When we construct informal sentiment dictionary and apply it to our own sentiment analyzing algorithm, many TV production companies and broadcaster can realize the public opinion, especially teenager's opinion toward new TV program or existing

program, so that they could make decision by gaining the faster feedback from the public (Fig. 1).

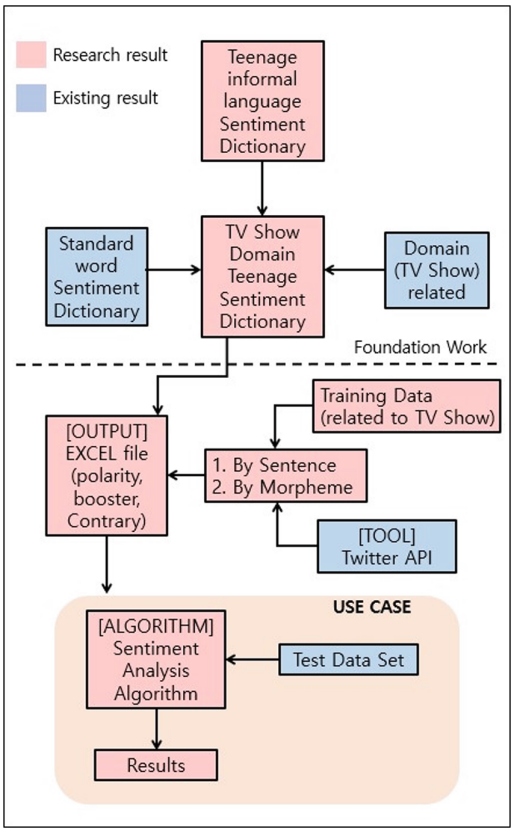

Fig. 1. System configuration diagram

2.2 Procedure and Analysis Plan

We defined the algorithm to extract the polarity value from the sentence through the N-Gram concept introduced above. In addition, through web crawling, we constructed data and built a sentiment dictionary with analyzed morphemes using KoNLPy morpheme analyzer (Fig. 2).

2.3 Data Collections

Web Crawling

We had to collect not only sentences that include informal sentiment words, but also standard language sentences related to TV program. So, we collected the former from

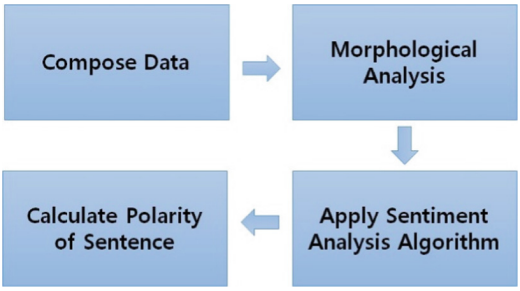

Fig. 2. Experiment procedure

websites such as naver.com, dcinside.com, imbc.com and Twitter SNS by web crawling. These comments data will be used in KoNLPy sentiment extraction performance test, Korean teenagers' sentiment dictionary composition and algorithm performance test described below.

2.4 Morphological Analysis (Natural Language Processing)

For morphological analysis, we decided to use KoNPLy which is an open API for Korean Natural Language Processing. KoNLPy includes 5 classes, Hannanum, Kkma, Komoran, Twitter and Mecab.

On the Internet, there are many sentences that do not keep spacing or spelling so that we set the following four criteria and select a morpheme analyzer for our research.

First, even if there is a spacing error, the morpheme must be analyzed correctly. Data such as Internet posts and comments have many non-spaced sentences. As a result, we tested each sentence that did not have a space between each morpheme analyzer. As a result, it was confirmed that the subjects of Kkma, Twitter, and Mecab were less sensitive to the spacing error.

Second, the word must be able to be normalized. Through the normalization function, words such as typo can be converted into one regular word. From this function, it is able to reduce sentiment dictionary composing time and access time. Only Twitter API provides normalization function.

Third, it must be able to add words to the user dictionary. The users can add informal words that are not registered in dictionary, so that morphological analysis can be more accurate. In this case, all five classes were supported (Fig. 3).

	Stemming Function	Normalize Function	Add User Words
Twitter	O	O	O
Komoran	X	X	O
Kkma	O	X	O
Hannanum	X	X	O
Mecab	O	X	O

Fig. 3. Property of each KoNLPy classes

For a little more certainty, we also had an experiment to find out the best API for our project. the experiment was conducted with 200 sample sentences into each API.

We tested with Twitter, Komoran, Kkma, and Hannanum classes to see how many morphemes were extracted with 200 sentences. The result as shown in the chart be-low, we could extract around 1400 morphemes before excluding duplication morphemes per each API and gain around 500 morphemes by removing duplication morphemes.

Twitter and Kkma morpheme analyzers were selected in consideration of the characteristics of each class to experiment how much sentiment words were extracted from the morpheme. As a result, Kkma could extract 10% sentiment words from morpheme and 14% sentiment words for Twitter. Therefore, we decided to use the Twitter morpheme analyzer, which offers many functions and has the largest sentiment word extraction amount (Fig. 4).

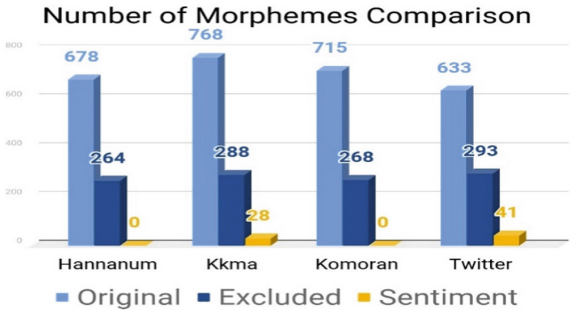

Fig. 4. KoNLPy experiment result

2.5 Sentiment Dictionary

In this section, we describe how to construct sentiment dictionary by handling and scoring data set. The overall process of constructing sentiment dictionary is as follows (Fig. 5).

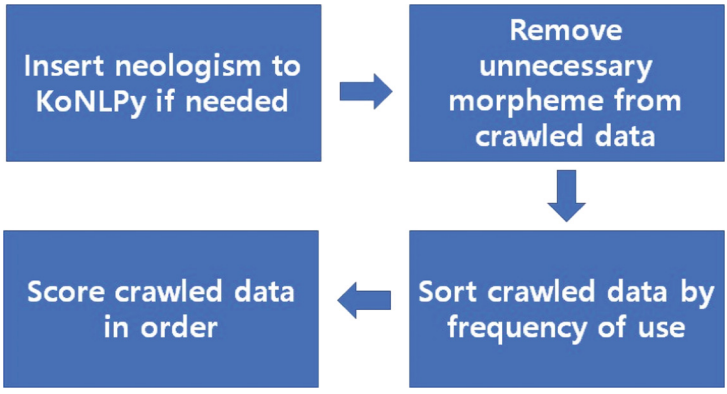

Fig. 5. Sentiment dictionary process flow

Classification of Word Type

There are three kinds of words: Sentiment word, Booster word, Contrary word. The type of words was refined by referring to English sentiment dictionary, SentiWordNet. The polarity value of each word, therefore, can be classified as follows.

Sentiment Word

A polarity value of the sentiment word is [−1, 1], the negative word is closer to −1, and the positive word has a value close to 1.

Booster Word

A Booster word is a word that emphasizes the polarity of the sentence and does not have a polarity value. Therefore, in the case of the booster word, multiply the polarity of the sentence by the weight value w1 (1.2) [4].

Contrary Word

A contrary word, it is divided into two cases, to distinguish the characteristics of the contrary word, the words were separated by using values outside the range of the polarity value [−1, 1] of the sentiment word. In the case of contrasting the polarity of the sentence before the word, the value is set to −100, and when the polarity of the sentence after the word is contrasted, the value is set to 100. Contrary word.

2.6 Sentiment Analysis Algorithm

The algorithm presented it this session calculates sentiment values of sentences and returns them as an output. The values used in the calculations are all derived from the sentiment dictionary constructed earlier. And as introduced above, we de-rive calculations by associating morphemes based on N-grams. In this case, to improve accuracy, two emotion morphemes are related to each other by using the 2-Gram method introduced in Ewha Women's University [4].

One word has the case number of three words: 'contrary', 'booster', 'sentiment'. Since we are using 2-Gram method, the combination of the two words, there are total nine cases. At this time, we considered only five cases where we think that calculation is necessary as follows.

1. Contrary word + Sentiment word
2. Sentiment word + Contrary word
3. Booster word + Sentiment word
4. Sentiment word + Booster word
5. Booster word + ⋯ + Booster word

For the above five cases, the polarity can be calculated by two calculation methods.

Combination A) For 1, 2, 3, and 4 cases, multiply each sentiment word by the booster word or the polarity value of the contrary word.
Combination B) In case of 5, the polarity value of the booster word is multiplied by the number of accumulated words.

Based on this, sentiment analysis algorithm implemented in the following order.

1. Break sentences into morphemes.
2. Matches the sentiment dictionary and makes the matching words into list form.
3. List the two values from the beginning to the end are associated with each other to calculate the value.

Below is the pseudocode that shows the progress steps described above (Fig. 6). Following expression is an example of the calculation process.

$$\text{Contrary: 1 Sentiment Word: 1, Booster Word: } 1.2 = -1 * 1 * 1.2 = -1.2 \quad (1)$$

```
Sub analyzeSentiment(textList, dicList)

     Foreach (text in textList)
      datas = re.split("[\n.!?~]", str(text))
      total_polarity = ["", 0.0]
      # [NEGATICE/POSITIVE/NEUTRAL, polarity]

      Foreach (data in datas)
          pos_data = Twitter().pos(data)
          contrary_word = [False, 0.0]
          booster_word = [False, 0]

          Foreach (pos in pos_data)
              isExist = False
              Foreach (i in range(0, len(dicList)))
                  If pos[0] Is dicList[i][0] Then
                       isExist = True
                       If boosterWord Is True Then break
                       Elif contraryWord Is True Then break
                       Else Then
                            booster_word[0] = False
                            contrary_word[0] = False
                       If processing_polarity Is True
                       Then break

                       total_polarity[1] += dicList[i][2]
                       contrary_word[1] = dicList[i][2]
             If isExist Is False And booster_word[0]
              Is True Then
                  total_polarity[1]
                  += W1 * booster_word[1]
                  booster_word[0] = False
                  break

      check_polarity(total_polarity)
End Sub
```

Fig. 6. Sentiment analysis pseudocode (input: sentence, crawled data/output: score, calculated sentiment score)

```
# Check the word is contrary word
Sub check_contrary_word(dicList, idx, booster_word, con-
trary_word, total_polarity)

    If dicList[idx][1] Is "H_Contrary" Then
    contrary word[0] = True
    booster word[0] = False
    return True

    Elif dicList[idx][1] Is "T_Contrary" Then
    total_polarity[1] -=contrary_word[1]
    return True

    Else Then return False
End Sub

Sub processing_polarity(dicList, idx, booster_word, con-
trary_word, total_polarity) :

      If booster word[0] Is False
      And booster word[1] Is Not 0 Then
           total_polarity[1]
           += W1*booster_word[1] *dicList[idx][2]
           booster_word[1] = 0
           return True

      # contrary word process
      If contrary_word[0] Then
            total_polarity[1] -= dicList[idx][2]
    contrary word[1] = False
    return True

      return False
End Sub
```

Fig. 6. (*continued*)

3 Experiment

3.1 Experiment Overview

To test the accuracy of our sentiment analyzing algorithm and sentiment dictionary, we had an experiment.

First, we gathered 2622 test sentences from Naver TV comments, which is an adequate web site to collect text data that includes teenager's sentiment toward TV

programs. Second, we input the text data and ran the algorithm. From the output of the algorithm, we could get the sentiment of each sentence. Third, team members judged the results whether it is correct or wrong. Since we cannot define the exact polarity of the sentiment, we chose binary polarity (positive/negative) as a criteria of sentiment, when constructing sentiment dictionary. Likewise, our team members judged the polarity of the sentences and calculated the accurate rate of the results.

3.2 Experiment Consideration

Sentences With Neutrality

It was necessary to consider the fact that the sentences consist of sentences that convey objective information or words that do not contain sentiment. At this time, the polarity of the sentiment was determined by the tester's judgment, and it was confirmed that it matches with the resultant value.

Sentence With a Double Negative

We have thought about how to deal with sentences with double injustice. However, in the domain of the program, we decided not to deal with complicated sentence structure such as double negation because it does not show much. Instead, it was decided by the tester's judgment whether the resultant value matches the polarity of sentiment and decided to consider it when supplementing the program later.

Sentences With Ambiguity

We consider the case that the polarity of a sentence cannot be understood without grasping the context like 'This is crazy'. We first gathered some of the words and analyzed them in some cases. In the program's domain, words such as 'crazy' continue to be used in a positive sense, thus giving positive polarity. In future programs, we will consider using 2-way and 3-way corpus dictionaries to handle these cases.

3.3 Experiment Results

The experiment proceeded as follows.

1. Crawled new dataset so as not to refer duplicated dataset used composing sentiment dictionary.
2. Extracted sentiment value through sentiment analysis algorithm.
3. Divided the accuracy of the polarity value into True or False.

A total of 2622 sentences were tested through above procedure. As a result, we were able to confirm that it had a total accuracy of 75.05% for test dataset.

4 Conclusion

There are some expectations by using the research results. First, we can easily analyze the opinions of young people about already existing programs or upcoming new programs. It makes teenagers possible to actively communicate with the broadcasting system and provide rapid feedback to them. Through the communication with the viewer, the broadcasting environment can be improved to the direction desired by the viewer.

As mentioned above, there were no exact linguistic criteria on scoring the polarity of sentiment words in our research. So, we had to choose binary polarity (±) when making SentiWord dictionary. Although there was such limitation on our research, the result was reliable and was somewhat accurate. We believe our research can be upgraded and complemented by some future works. First, we can improve the accuracy of text sentiment analyzing by adopting new technology and improving sentiment analysis algorithm. We can also change the way of accumulating data so that we can collect higher quality of text data. Besides, we can update the teenager sentiment dictionary by searching and adding newly generated words. So, we hope that our research will be utilized and referenced in wide variety of social issues to alleviate the language barrier between the younger and the older generation.

Acknowledgements. This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the National Program for Excellence in SW supervised by the IITP (Institute for Information and communications Technology Promotion) (2016-0-00017).

References

1. An, J., Kim, H.W.: Building a Korean sentiment lexicon using collective intelligence. J. Intell. Inf. Syst. **21**(2), 49–67 (2015)
2. Suck, S., Yun, H.: Implementation of the Sentiment Analysis Algorithm with Korean Twitter Data (2016)
3. Lee, C., Choi, D., Kim, S., Kang, J.: Classification and analysis of emotion in Korean microblog texts. KIISE: Databases **40**(3), 159–167 (2013)
4. Song, J.: The comparative study on the performance of ?N-Gram by change of the number of words based on characteristic of Korean language (2017)
5. Pang, B., Lee, L.: A sentimental education: sentiment analysis using subjectivity summarization based on minimum cuts. In: Proceedings of the Association for Computational Linguistics, pp. 271–278 (2004)
6. Chen, T., Yu, F., Chen, J., Cui, Y., Chen, Y.-Y., Chang, S.-F.: Object based visual sentiment concept analysis and application. In: ACM MM, pp. 367–376 (Nov 2014)
7. Borth, D., Ji, R., Chen, T., Breuel, T., Chang, S.-F.: Large-scale visual sentiment ontology and detectors using adjective noun pairs. In: ACM MM, pp. 223–232, ACM (2013)
8. Chen, T., Yu, F.X., Chen, J., Cui, Y., Chen, Y.-Y., Chang, S.-F.: Object-based visual sentiment concept analysis and application. In: ACM Multimedia (2014)