oIQa: An Opinion Influence Oriented Question Answering Framework with Applications to Marketing Domain

Dumitru-Clementin Cercel, Cristian Onose, Stefan Trausan-Matu and Florin Pop
Faculty of Automatic Control and Computers
University Politehnica of Bucharest, Romania
clementin.cercel@gmail.com, onose.cristian@gmail.com, stefan.trausan@cs.pub.ro, florin.pop@cs.pub.ro

Abstract

Understanding questions and answers in Question Answering (QA) system is a major challenge in the domain of natural language processing. In this paper, we present a question answering system that influences the human opinions in a conversation. The opinion words are quantified by using a lexicon-based method. We apply Latent Semantic Analysis and the cosine similarity measure between candidate answers and each question to infer the answer of the bot.

1 Introduction

In the past decade, electronic commerce has seen a strong and steady growth causing more and more companies to invest in it. Also, large numbers of users express their opinions about many topics in social media, which has become a means of influencing users to make certain decisions. As such, product or services reviews have become an important part of the purchase journey for customers.

More and more users recognize that reviews influence their buying decision to the point that some consider them to carry as much trust as recommendations from their friends or family.

In this paper, we integrate the influence concept in a traditional question answering system. The hypothesis that underlies our research is that human opinions may change over time as a consequence of the interaction with a chatbot in a question-answering system. A human participating in a question-answering system is influenced over time by the chatbot if he shares beliefs with it for a given amount of the conversation. Thus, influence is analyzed by taking into consideration the changes in human opinions.

The posts in our question-answering system are preprocessed by means of a Natural Language Processing (NLP) tool (De Marneffe et al., 2006) to identify the dependency relations between words. Using dependency relations specific for opinion mining, we extract opinion words.

Opinion lexicons represent important resources for identifying the sentiment of opinion words. In this paper, we adopt an opinion lexicon-based method to identify the sentiment expressed by each opinion word. There are available several opinion lexicons that enable us to classify opinion words into one of the following sentiment categories: positive, negative, or neutral. In this way, we classify the opinion words by using the following four opinions lexicons (Cercel and Trausan-Matu, 2014a): SentiWordNet (Baccianella and Sebastiani, 2010), Micro-WNOp (Cerini et al., 2007), MPQA subjectivity lexicon (Wilson et al., 2005), and Bing Liu’s opinion lexicon (Hu and Liu, 2004).

Using several opinion lexicons is necessary mainly due to the fact that there is no single opinion lexicon that returns the exact sentiment given both the context and the domain in which the opinion words are referenced.

We proposed a model for question-level sentiment analysis, where many opinions that have different sentiments are expressed. In this model, we take into consideration the sentiment strength expressed by opinions.

Finally, we apply Latent Semantic Analysis (Landauer et al., 1998) and the cosine similarity measure between candidate answers and each question to infer the answer of the bot.

This paper is structured as follows. In Section 2, we briefly discuss the existing approaches to the
question answering task. In Section 3, we detail the system proposed by us to solve the problem of opinion influence oriented question answering. The experimental results are shown in Section 4. Finally, Section 5 is dedicated to the conclusions of our research.

2 Related Work

The work presented in this paper is best described as the combination of social influence and two broad NLP areas of interest: opinion mining (also known as sentiment analysis) and question-answering systems. The problem of review corpus based question-answering systems has seen some interest in the past (McAuley and Yang, 2016; Wan and McAuley, 2016), and also opinion mining techniques have been incorporated into such setups to provide more personalized answers to subjective queries (Wan and McAuley, 2016; Balahur et al., 2009a). However, to our knowledge, a question-answering system with the ability to influence the opinion of users is a novel idea.

**QA systems.** Their aim is to automatically provide answers to questions submitted by users in natural language. Given different attributes these systems can be classified in a number of categories (Mishra and Jain, 2016). Our main interest is regarding how the input of the user is handled: (1) Corpus based QAs select candidate answers from a dataset based on different similarity measures; (2) QAs that extract concepts and relations between them from the inputs and formulate an answer.

We focused on corpus based types (i.e. user reviews) because these kinds of datasets are commonly available. An important aspect for this type of QAs is the task of identifying quality candidate answers for a given user query (Anderson et al., 2012; Yuen et al., 2011).

**Opinion mining** is a research field of great interest because many applications are based on opinion mining, including opinion summarization (Balahur et al., 2009b), opinion propagation (Cercel and Trausan-Matu, 2014b) and sentiment prediction (Kim et al., 2013). We classify the types of granularity studied in sentiment analysis as: the document-level sentiment classification, the sentence-level sentiment classification, and the word-level sentiment classification. Methods of identifying the sentiment expressed by an opinion word can be classified into: manual methods, corpus-based methods, and opinion lexicon-based methods (Liu, 2012).

**Social Influence.** In social influence analysis, several methods have been proposed to measure the influence among users. For example, Tang et al. (2009) studied influence probabilities on different topics in a social network. Also, Liu et al. (2010) studied the modeling of indirect influence in heterogeneous networks, focusing on users’ behaviors such as following, citing, and replying. Apart from the methods presented above, different types of social influence are described in social influence analysis. For instance, conformity is a form of social influence in which people change their opinions or behavior in order to be assimilated into a group (Tang et al., 2013).

3 Approach

The proposed method consists of several modules described below.

3.1 Sentiment Analysis

3.1.1 Opinion Extraction from Questions

The goal of this step is to identify opinion words in a given question. We consider that the target entity to which the opinions refers is syntactically defined by a noun term, also opinion words can be one of the following parts of speech: adjective, adverb, or verb.

The questions are preprocessed by means of the Stanford CoreNLP tool (De Marneffe and Manning, 2008) to identify the dependency relations between words. Each dependency relation is a binary grammatical relation between a governor word (also called head) and a dependent word. A dependency relation is represented in the form: abbreviated relation name (governor, dependent), where the dependency relation exists between the governor word and the dependent word.

Using specific rules for opinion mining, we extract dependency relations which identify opinion words. There rules are given in the form of syntactic dependency relation between a noun term and the opinion word. In Table 1, we have summarized the dependency relations from Stanford CoreNLP used to extract opinion words.

The algorithm for dependency relations extraction (see Algorithm 1) receives as input the entire question and outputs relations in the form of (noun term, opinion word) pairs.
Table 1: The list of dependency relations used for opinion extraction.

| Dependency Relations | Description |
|----------------------|-------------|
| dobj (“direct object”) | The opinion word is a verb or a complement following a copular verb |
| nsubj (“nominal subject”) | The opinion word is a complement following a copular verb |
| amod (“adjectival modifier”) | The opinion word is the adjectival modifier of a noun |
| advmod (“adverbial modifier”) | The opinion word is an adverbial modifier of a verb |
| acom.mod (“adjectival complement”) | The opinion word is the adjectival complement of a verb |
| xcomp (“open clausal complement”) | The opinion word is the complement following a copular verb |
| neg (“negation modifier”) | The opinion word sentiment is shifted by negation word |

We start by running the question through Stanford CoreNLP which determines all dependency relations (A1 : 2). For each dependency relation we identify if it is fitting for discovering opinion words by analyzing the rules in Table 1 (A1 : 3-23).

Algorithm 1: Identifying Stanford Dependency Relations for Opinion Mining

**Input:** q – question  
**Output:** Ω – a set of pairs (noun term, opinion word)

1. Ω ← ∅  
2. R ← Parse(q) // the set {rel(h_k, d_i), jij≥1} of all dependency relations between the words  
3. for each dependency relation rel(h_k, d_i) in R do  
4. if rel_j = “dobj” and checkVerb(h_k) and checkNoun(d_i) then  
5. Ω ← Ω ∪ (h_k, d_i)  
6. end if  
7. if rel_j = “nsubj” and checkAdjective(h_k) and checkNoun(d_i) then  
8. Ω ← Ω ∪ (h_k, d_i)  
9. end if  
10. if rel_j = “amod” and checkAdjective(h_k) and checkNoun(d_i) then  
11. Ω ← Ω ∪ (d_i, h_k)  
12. end if  
13. if rel_j = “advmod” and checkVerb(d_i) and there is rel_k(h_kk, d_i) so that rel_j = “rcmod” and h_kk = d_kk then  
14. Ω ← Ω ∪ (h_k, d_kk)  
15. end if  
16. if rel_j = “acom.mod” and checkVerb(h_k) and there is rel_j(h_kk, d_i) so that rel_j = “rcmod” and h_kk = d_kk then  
17. Ω ← Ω ∪ (d_i, h_kk)  
18. end if  
19. if rel_j = “xcomp” and checkVerb(h_kk) and there is rel_j(h_kk, d_i) so that rel_j = “ccomp” and h_kk = d_kk then  
20. Ω ← Ω ∪ (h_kk, d_i)  
21. end if  
22. end for  
23. end if

For usability purposes, the opinions are modeled by numerical values. To accomplish this, the opinion words, that are extracted from questions, are encoded as follows: -1, 0, or +1, where a value of -1 corresponds to an opinion word with negative sentiment, a value of 0 corresponds to an opinion word with neutral sentiment, and a value of +1 corresponds to an opinion word with positive sentiment.

3.1.2 Opinion Word-Level Sentiment Identification

After the step of extracting opinion words from questions, we proceed to determine the sentiment expressed by each opinion word. For the task of identifying the sentiment of opinion words, we have decided upon four opinion lexicons that we have used in our experiments as follows:

- **SentiWordNet** (Baccianella and Sebastiani, 2010) is an opinion lexicon in which the synsets, which are part of the WordNet lexical database (Fellbaum, 1998), are defined by three sentiment scores: positive score, negative score and neutral score. Each sentiment score is a numerical value from the interval [0, 1] so that the sum of the three sentiment scores for each synset is 1. SentiWordNet contains over 155,000 words from 117,659 synsets.

- **Micro-WNOp** (Cerini et al., 2007) is composed of a subset of WordNet synsets more precisely 1,105 synsets, which represent 1,960 distinct words. Every Micro-WNOp synset is annotated with two numerical scores: one score indicates the grade of positivity of its component words, and the other one indicates the grade of negativity. Both scores have values between 0 and 1, and their sum is less than or equal to 1.
• MPQA Subjectivity Lexicon (Wilson et al., 2005) as single-word clues. The sentiment expressed by each clue was manually annotated in one of the following categories: positive, negative, both (positive and negative), or neutral.

• Bing Liu Lexicon (Hu and Liu, 2004) is a list of 6,785 words divided into two sets according to their sentiment: a set of words with positive sentiment and another set of words with negative sentiment.

Since the lexicons have various ways of representing the sentiment score, it becomes necessary to use specific algorithms for each opinion lexicon in order to map the output to numerical values, as outlined below.

**SentiWordNet:** The outline of the algorithm based on SentiWordNet for determining the sentiment of an input opinion word is shown in Algorithm 1. Initially, the algorithm establishes if the input opinion word, that corresponds to a given part of speech, is not part of SentiWordNet. If this is the case then it will return the sentiment score of 0 (A2 : 1-3), otherwise all the sense numbers of the input word will be arranged in ascending order (A2 : 4). For each of the sorted senses we compute the difference between the positive score and the negative score returned by SentiWordNet (A2 : 6). If the difference is zero, and the positive score is non-zero, the algorithm continues with the next iteration (A2 : 7-9). If the difference is zero, and the positive score is also zero, then the output sentiment is neutral (A2 : 10-12). If the difference is greater than zero, then the sentiment is positive (A2 : 13-15). If the difference is less than zero, then the sentiment is negative (A2 : 16-18). Finally, if the input opinion word has the positive score equal to the negative score for all its senses, then the sentiment is considered positive (A2 : 20).

**Algorithm 2 (A2): Opinion Word Sentiment Identification using SentiWordNet**

| Input: | w – opinion word; |
|--------|--------------------|
| Input: | pos – part of speech for the opinion word w; |
| Output: | a value \( \epsilon \{-1, 0, +1\} \) – indicates the sentiment of the opinion word w, where: -1 denotes a negative sentiment, 0 denotes a neutral sentiment, and +1 denotes a positive sentiment; |
| 1: if | !FindOpinionWord(w, pos) then |
| 2: | return 0 |
| 3: | end if |

4: \( S \leftarrow \text{GetSenses}(w, \text{pos}) \)
5: for each \( s \in S \) do
6: \( \text{diff} \leftarrow \text{GetPositiveScore}(w, s) - \text{GetNegativeScore}(w, s) \)
7: if \( \text{diff} = 0 \) and \( \text{GetPositiveScore}(w, s) \neq 0 \) then
8: continue
9: end if
10: if \( \text{diff} = 0 \) and \( \text{GetPositiveScore}(w, s) = 0 \) then
11: return 0
12: end if
13: if \( \text{diff} > 0 \) then
14: return +1
15: end if
16: if \( \text{diff} < 0 \) then
17: return -1
18: end if
19: end for
20: return +1

**Micro-WNOp:** Micro-WNOp is a human annotated lexicon that can be broken into three parts. In the initial stage five human evaluators work together, while for the other parts they are split into groups that work independently from one another. Each opinion word is given a positive sentiment score and a negative sentiment score regardless of the way the groups are formed. Because the groups work separately, there are differences between the scores. To solve this issue, we consider that the sentiment of an opinion word is given by the first evaluator’s annotation. The algorithm based on Micro-WNOp resembles the algorithm that uses SentiWordNet which is shown in Algorithm 2.

**Bing Liu Lexicon:** The summary of the algorithm based on Bing Liu’s opinion lexicon for determining the sentiment of an input opinion word is shown in Algorithm 3. For the Bing Liu’s opinion lexicon, we assign a sentiment score of +1 to each opinion word in the positive list (A3 : 4-6), and a sentiment score of -1 to each opinion word in the negative list (A3 : 7-9). Also, if the opinion word is not found in the lexicon the algorithm outputs a sentiment score of 0 (A3 : 1-2).

**Algorithm 3 (A3): Opinion Word Sentiment Identification using Bing Liu’s opinion lexicon**

| Input: | w – opinion word; |
|--------|--------------------|
| Output: | a value \( \epsilon \{-1, +1\} \) – indicates the sentiment of the opinion word w, where: -1 denotes a negative sentiment and +1 denotes a positive sentiment; |
| 1: if | !FindOpinionWord(w) then |
| 2: | return 0 |
| 3: else |
MPQA Subjectivity Lexicon: The algorithm, that uses the MPQA subjectivity lexicon, for determining the sentiment of an input opinion word is described in Algorithm 4. For the MPQA subjectivity lexicon, the algorithm associates the sentiment scores of -1, 0, or +1 to the lexicons’ outputted categorical values of negative, neutral or positive, respectively (A4 : 5-16). What’s more in this lexicon, opinion words can be annotated with both positive and negative classes. In this case, we assign the sentiment score of +1 to the opinion words. Finally, if the input word is not found in the lexicon, then the returned sentiment score is 0 (A4 : 1-2).

Algorithm 4 (A4): Opinion Word Sentiment Identification using the MPQA subjectivity lexicon

Input: w – opinion word; pos – part of speech for the opinion word w;
Output: a value e [-1, 0, +1] – indicates the sentiment of the opinion word w, where: -1 denotes a negative sentiment, 0 denotes a neutral sentiment, and +1 denotes a positive sentiment;
1: if FindOpinionWord(w, pos) then
2: polarity ← GetPolarity(w)
3: if polarity = “positive” then
4: return +1
5: end if
6: if polarity = “negative” then
7: return -1
8: end if
9: if polarity = “neutral” then
10: return 0
11: end if
12: if polarity = “both” then
13: return +1
14: end if
15: end if

3.1.3 Question-Level Sentiment Identification

Question-level sentiment identification determines whether a subjective question expresses a positive, negative, or neutral sentiment. We determine the sentiment of a question by taking into account the sentiment strength of its opinion words. Let q be a question in the conversation. The sentiment score for the question q is given by the following equation:

\[ score(q) = \sum_{w \in q \cap w_{pos}} score(w) \times \lambda_i + \sum_{w \in q \cap \neg w_{pos}} score(w) \times \lambda_i \]

where:

- \[ score(w) \] = the sentiment score of the opinion word w and is calculated by using the opinion lexicons as described above;
- \[ S_j^i \] = the list of adjectives in the superlative;
- \[ S_j^2 \] = the list of adjectives in the comparative of superiority;
- \[ S_j^3 \] = the list of adjectives in the comparative of inferiority;
- \[ S_R^i \] = the list of adjectives in other degree;
- \[ S_R^j \] = the list of adverbs in the superlative;
- \[ S_R^3 \] = the list of adverbs in the comparative of superiority;
- \[ S_R^4 \] = the list of adverbs in the comparative of inferiority;
- \[ S_v^i \] = the list of verbs;
and \(|w|\) denotes the size of list.

The variables \( \lambda_1, \lambda_2, \lambda_3, \) and \( \lambda_4 \) take the values 0.9, 0.6, -0.6, and 0.3, respectively. The question q is considered to express a positive sentiment if \( score(q) \in (0, +1) \), a negative sentiment if \( score(q) \in [-1, 0) \), and a positive sentiment if \( score(q) = 0 \).

3.2 Answer Selection

The process of answer generation sums up to choosing the answer that is the closest to the question from a semantic point of view. Besides semantic similarity, we also impose the constraint that the answer-level sentiment score has to be positive or neutral. We accomplish this by using Latent Semantic Analysis (LSA) and the cosine similarity measure between candidate answers and each question. Candidate answers are selected from the Stanford Sentiment Treebank dataset (Socher et al., 2013).

3.2.1 Latent Semantic Analysis

LSA is a method used in natural language processing to measure the semantic similarity between two texts. The idea behind this method is that similar texts comprise words with the same or approximately the same meaning.

LSA is a corpus based method that requires a series of document \( D = \{d_1, d_2, ..., d_n\} \) for the
training step. Let $W = \{w_1, w_2, \ldots, w_m\}$ be a set consisting of all the distinct terms in the corpus $D$. Let the number of documents in the corpus be $D$ and the number of terms in the set $W$ be $n$ and $m$, respectively. The LSA method starts by creating a term-document matrix $A = \{a_{ij}\}$ of dimension $m \times n$, where the column $j^{th}$ in the matrix $A$ corresponds to the document $d_i$ in the corpus $D$, and the row $i^{th}$ in the matrix $A$ corresponds to the term $w_i$ in the set $W$. The entry $a_{ij}$ is defined as follows (B. Liu, 2011, p. 243):

- $U = [U_1, U_2, \ldots, U_r]$ is an orthonormal matrix of dimension $m \times r$, i.e. $U^T U = I_r$ (identity matrix), and the value $r$ is the rank of the matrix $A$. The columns of the matrix $U$, which are referred to as left singular vectors, are eigenvectors corresponding to the non-negative eigenvalues of the matrix $A^T A$.
- $\Sigma = \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_r)$ is a diagonal matrix of dimension $r \times r$. The diagonal elements $\sigma_1, \sigma_2, \ldots, \sigma_r$ are referred to as singular values, and the following relation between them is true: $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$. These singular values are the square roots of the non-negative eigenvalues of $A^T A$.
- $V = [V_1, V_2, \ldots, V_n]$ is an orthonormal matrix of dimension $n \times r$, i.e. $V^T V = I_n$. The columns of the matrix $V$, which are referred to as right singular vectors, are eigenvectors corresponding to the non-negative eigenvalues of the matrix $A^T A$.

4 Experimental Results

The user interface of the application is presented in Figure 1. The interface contains three major discernible areas that provide visual representation of the sentiments expressed by both human and bot.

The left zone is designed to allow interaction between human and bot, users can either input different opinions about subjects or query the system for information. Questions and answers are sorted according to a timestamp maintained by the chat area.

The evolution of sentiments for both bot and human are visible in the center and left area of the interface (the plotted data includes results from all the lexicons discussed in the previous section).

As the conversation progresses, the main focus is to have a high increase of the influence score. As such, alongside the presented components, the application contains an area where one can view the influence score exerted by the chat bot.

The metric used to measure the influence score is based on the sentiments calculated for the questions. We take into consideration only positive and negative sentiment questions. The influence score is calculated as the ratio of the number of positive sentiment questions and the sum between the number of positive sentiment questions and number of negative sentiment question:

\[
\frac{\# \text{ of positive questions}}{\# \text{ of positive questions} + \# \text{ of negative questions}}
\]

5 Conclusions

Communication constitutes the core element of human interactions and recently also between humans and computers. However, communicating means not only giving or receiving information, but also the ability to influence one another's beliefs.

In this paper, we presented a question answering system in order to integrate a module to analyze opinion influence. A human is influenced by a bot in a conversation if he changes his initial opinion. More precisely, the user inputs more and more questions with positive sentiments over time. The results show that this is possible.
Figure 1: Interface for the oIQa framework.

Acknowledgments
This work has been funded by University Politehnica of Bucharest, through the “Excellence Research Grants” Program, UPB – GEX. Identifier: UPB–EXCELENTA–2016, Contract number 11/30.09.2016.

References
Ashton Anderson, Daniel Huttenlocher, Jon Kleinberg, and Jure Leskovec. 2012. Discovering value from community activity on focused question answering sites: a case study of stack overflow. In Proceedings of the 18th International Conference on Knowledge Discovery and Data Mining, pages 850–858.
Alexandra Balahur, Ester Boldrini, Andres Montoyo, and Patricio Martinez-Barco. 2009a. Opinion and generic question answering systems: a performance analysis. In Proceedings of the ACL-IJCNLP 2009 conference short papers, pages 157–160.
Alexandra Balahur, Mijail Alexandrov Kabadjov, Josef Steinberger, Ralf Steinberger, and Andrés Montoyo. 2009b. Summarizing Opinions in Blog Threads. In Proceedings of PACLIC, pages 606-613.
Dumitru C. Cercel, and Stefan Trausan-Matu. 2014a. Opinion Influence Analysis in Online Forum Threads. In Proceedings of Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), pages 228-235.
Dumitru C. Cercel and Stefan Trausan-Matu. 2014b. User-Level Opinion Propagation Analysis in Discussion Forum Threads. In Proceedings of Artificial Intelligence: Methodology, Systems, and Applications, pages 25-36.
Sabrina Cerini, Valentina Compagnoni, Alice Demontis, Maicol Formentelli, and G. Gandini. 2007. Micro-WNOP: A gold standard for the evaluation of automatically compiled lexical resources for opinion mining. In A. Sans (Ed.), Language resources and linguistic theory: Typology, second language acquisition, English linguistics: Franco Angeli Editor.
Andrea Esuli, Stefano Baccianella, and Fabrizio Sebastiani. 2010. SentiWordNet 3.0: An Enhanced Lexical Resource for Sentiment Analysis and Opinion Mining. In Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10), Valletta, Malta.
Christiane Fellbaum. 1998. WordNet: An Electronic Lexical Database. Cambridge, MA: MIT Press.
Minqing Hu, and Bing Liu. 2004. Mining and summarizing customer reviews. In Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining, Seattle, WA, USA, pages 168-177.
Jihie Kim, Jae-Bong Yoo, Ho Lim, Huida Qiu, Zornitsa Kozareva, and Aram Galstyan. 2013. Sentiment Prediction Using Collaborative Filtering. In Proceedings of ICWSM.
Thomas K. Landauer, Peter W Foltz, and Darrell Laham. 1998. An introduction to latent semantic analysis. Discourse processes, 25(2-3), 259-284.

Bing Liu. 2011. Web Data Mining Exploring Hyperlinks, Contents and Usage Data: Springer-Verlag.

Lu Liu, Jie Tang, Jiawei Han, Meng Jiang, and Shiqiang Yang. 2010. Mining topic-level influence in heterogeneous networks. In Proceedings of the 19th ACM international conference on Information and knowledge management, Toronto, ON, Canada, pages 199-208.

Marie-Catherine De Marneffe, Bill MacCartney, and Christopher D Manning. 2006. Generating typed dependency parses from phrase structure parses. In Proceedings of LREC, pages 449-454.

Marie-Catherine De Marneffe, and Christopher Manning. 2008. Stanford typed dependencies manual.

Julian McAuley and Alex Yang. 2016. Addressing complex and subjective product-related queries with customer reviews. In Proceedings of the 25th International Conference on World Wide Web, pages 625–635.

Amit Mishra and Sanjay K. Jain. 2016. A survey on question answering systems with Classification. Journal of King Saud University-Computer and Information Sciences. 28(3): 345–361.

Richard Socher, Alex Perelygin, Jean Y. Wu, Jason Chuang Christopher D. Manning, Andrew Y. Ng and Christopher Potts. 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).

Jie Tang, Sen Wu, and Jimeng Sun. 2013. Confluence: conformity influence in large social networks. In Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining, Chicago, Illinois, USA, pages 347-355.

Jie Tang, Jimeng Sun, Chi Wang, and Zi Yang. 2009. Social influence analysis in large-scale networks. In Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, Paris, France, pages 807-816.

Mengting Wan and Julian McAuley. 2016. Modeling ambiguity, subjectivity, and diverging viewpoints in opinion question answering systems. In Proceedings of the International Conference on Data Mining (ICDM), pages 489–498.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing, Vancouver, British Columbia, Canada, pages 347-354.

Man-C. Yuen, Irwin King, and Kwong -S. Leung. 2011. A survey of crowdsourcing systems. In Proceedings of Privacy, Security, Risk and Trust (PASSAT) and 2011 IEEE Third International Conference on Social Computing (SocialCom), 2011 IEEE Third International Conference on, pages. 766–773.