International Journal of Computer Applications (0975 - 8887)
Volume 176 - No.11, April 2020

A Review of Common Approaches to Sentiment Analysis and Community Detection

Sarvesh Bhatnagar
NMIMS University
Mumbai, Maharashtra
India

Maitreya Dixit
NMIMS University
Mumbai, Maharashtra
India

Nachiketa Prasad
NMIMS University
Mumbai, Maharashtra
India

ABSTRACT

Sentiment analysis and community detection are two very active fields of research in computer science. They are both intimately linked to the modern phenomenon of social media, and can be very useful for extracting valuable information from a large corpus of social media posts. In this paper, we review the basic concepts of both fields and outline some of the algorithms and approaches that have been successfully applied. Finally, we take a look at the instances where both have been applied together.

General Terms

Natural Language Processing, Clustering

Keywords

Sentiment Analysis, Community Detection

1. INTRODUCTION

Sentiment analysis or opinion mining of text is a field in computer science that spans the fields of natural language processing, machine learning, text mining and other disciplines such as linguistics and the social sciences. It is the task of analysing a piece of text written by a person to identify their opinion, sentiment, attitude or emotion concerning a target entity such as an object, an event, a topic, or a set of people.

This field is very important in the modern world because of the explosion of content being delivered every day on the Internet. In 2017, 2.5 quintillion bytes of data were being created every day, and that number has been growing rapidly due to the proliferation of Internet of Things(IoT) and other domains[1]. From all this data, valuable information may be extracted for use by organisations or private individuals. Organisations, for instance, need to be able to gauge public opinion about their products or services. In the past, they would have to conduct market surveys, or consult focus groups. But today, an abundance of real user-generated data is available on the Internet, some of which may describe their product in the form of reviews or general opinions. In order to understand this data, sentiment analysis is required.

It has also been used in applications like predicting election results, box-office revenues, and the performance of stocks. But the field has a set of limitations and challenges.

This process is not straightforward because sentiments or opinions are inherently subjective, and they are generally expressed in a manner that is not universally the same either. Tone and context both greatly change the correct interpretation of text, and it is very hard to accurately identify them via computation.

Sentiment analysis has generally been investigated at three levels of granularities. These levels are:

(1) **Document level** - An attempt is made to classify a whole document as expressing a positive or negative sentiment. This is often applied to product reviews, for example. It only works well when it may be assumed that only one entity is referred to in the whole document.
(2) **Sentence level** - An attempt is made to classify the sentences within the document as expressing a positive or negative sentiment. Again, it faces the issue that a sentence need not represent a single opinion.
(3) **Aspect level** - It defines opinions as pairs of sentiments and targets that do not conform to language constructs like sentences or clauses. At this level, analysis can be applied to text with the least number of qualifications.

Formally, the sentiment analysis problem may be defined as the task of finding all opinions expressed within a document, where an opinion is defined as a 5-tuple (e_i,a_{ij},oo_{ijkl},h_k,t_l), where

e_i is the entity name.
a_{ij} is an aspect of e_i.
oo_{ijkl} is the orientation or polarity of the opinion about aspect a_{ij} of entity e_i.
h_k is the opinion holder.

2. ALGORITHMS

There are a number of machine learning approaches to sentiment analysis, mainly categorised under supervised learning. Some of the ones which have been successfully applied are -

2.1 Naive Bayesian classification

A Naive Bayesian classifier is based on the Bayesian probability rule, with the assumption that the occurrences of a particular item xi (here, words or n-grams), classified into some class c, do not affect the classification of the other items. The probability of x_i belonging to class c is given by[14],

$$P(x_i|c) = \frac{Count\ of\ x_i\ in\ documents\ of\ Class\ c}{Total\ no.\ of\ words\ in\ documents\ of\ class\ c}$$

The Bayesian rule states that the probability of the whole document d belonging to a class c_i is given by[14],

$$P(c_i|d) = \frac{P(d|c_i) * P(c_i)}{P(d)}$$

If we take the assumption stated above, that each x_i is conditionally independent, then the probability is[14],

$$P(c_i|d) = \frac{(\prod P(x_i|c_j)) * P(c_j)}{P(d)}$$

This can be evaluated and the maximum probability class is the final output.

A strategy for Naive Bayesian (NB) classification was developed by Gamallo et al. The advantage of using NB is that the training and operation process is time-efficient, while providing a reasonable level of accuracy. But the individual features of text are generally not independent of each other, violating the NB assumption. In spite of that, a high level of accuracy may be achieved.

The classifier was trained on a sentiment-labelled corpus of tweets from Twitter using only positive and negative tweets, as defined by the presence of terms from a polarity lexicon (containing 10,850 entries from various sources). The base features used are lemmas extracted from the text using lemmatization, specifically, nouns, verbs, adjectives and adverbs. In addition, the following part-of-speech patterns were extracted as n-grams (multiwords)

- NOUN - ADJECTIVE
- NOUN - NOUN
- ADJECTIVE - NOUN
- NOUN - PREPOSITION - NOUN
- VERB - NOUN
- VERB - PREPOSITION - NOUN

At the preprocessing stage, URLs, usernames and hashtags were removed from the text, unnecessary replicated characters were removed, and emoticons were replaced with equivalent polarity expressions. To handle negations in the text, the preprocessor looks for syntactically linked polarity words (nouns, verbs, or adjectives) upto two words after the negation, and reverses their polarity. The F-score (harmonic mean of precision and sensitivity) of this classifier was 0.63[5].

Another strategy for NB classification was put forward by Narayanan et al. They used a public dataset of movie reviews from IMDb compiled by Andrew Maas et al(25000 labelled movie reviews each for training and testing), where each review is classified as negative or positive. To handle unknown words in the training set, they used Laplacian smoothing, where the conditional probability of each such word x_i is calculated by[14],

$$P(x_i|c_j) = \frac{Count(x_i) + k}{(k+1) * (No.\ of\ Words\ in\ Class\ c_j)}$$

Here, k is set to 1. To handle negations, they attached a negation state variable to each word, so that if it is set, the class of the word is reversed. It is set when the word follows a negation such as not, but is reset where punctuation's or double negations are encountered.

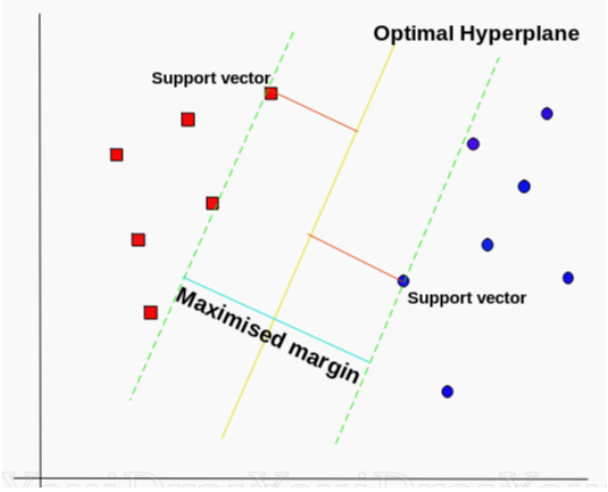

Fig. 1. Classification using Support Vector Machine

Finally, they added the use of n-grams for more meaningful items xi and calculated the mutual information of the presence of x_i and the obtained class. The items selected were the top 32000 items by mutual information.

The obtained classification accuracy was 88.8% on the testing set, with time complexity of O(n + V lg V) for the training process and O(n) for the testing, where n is the number of words in the documents (linear) and V is the size of the vocabulary[14].

2.2 Support Vector Machine classification

Support Vector Machines (SVMs) are machine learning models that attempt to construct hyperplanes in n-dimensional space that individually separate a set of n-featured items into two classes. The hyperplane showing the greatest separation between items of the classes is taken as the result, and it can be used to classify new items based on the side of the hyperplane on which they lie. A support vector is a point that is maximally close to the hyperplane, and influences its orientation. Figure 1 is an example of such a classification[4].

It is possible to use SVMs to classify non-linearly separable items by using a kernel function that maps to a higher dimension where they are separable. However, in most textual applications, only a linear kernel is required as the number of features in text is generally quite large.

Chikersal et al. created a system that combines a rule based classifier with an SVM to classify the sentiment of tweets from Twitter as positive, negative or neutral[3].
In the pre-processing stage, all references to usernames or URLs are changed to known constant terms. An existing part-of-speech tagger called CMU is used to extract the relevant items from the tweet. Only the tweets without any emoticons are sent to the SVM classifier, the rest are sent to the rule based classifier.

The SVM classifier cannot be given the tagged tweets directly as input, so a feature vector is generated for each tweet, which contains the following[3],

- Word n-grams of size not more than 3 from the tweet, vectorised using TF-IDF.
- Character n-grams of size not more than 3 from the tweet, vectorised using TF-IDF.
- Part-of-speech tag ratios including the count of adjectives, adverbs, nouns, verbs and interjections divided by the total number of tags.
- Booleans indicating whether the tweet contains usernames, URLs, hashtags and is part of a discourse.
- A set of polarity measures calculated from the count of positives and negatives in the tweet from a corresponding set of n-gram lexicons.
- Negations as a part of corresponding word n-grams to invert polarity.

The SVM classifier uses a linear kernel and L1-regularisation. When all these features were used, the classifier returned a total correct classification rate of 71.5%, with an f-score of 0.662, without considering the rule-based classifier[3].

Huq et al. also attempted classification of tweets from Twitter and they evaluated classifiers based on both SVMs and k-nearest neighbours algorithm. The feature set used includes[7],

- Words, except all stop words, are included as feature weights calculated as the ratio of the count of the word divided by the total count of words. This ratio for weight is calculated for each feature.
- n-grams, where n is limited to 5. High weight n-grams that occur rarely are significant for classification.
- Patterns, which are ordered sequences of high-frequency words with slots for content (low or medium frequency) words. They must state and end with HFWs and include 2-6 HFWs, 1-5 slots for CWs.
- Punctuations, including exclamation marks, question marks, quotes and word capitalization. The weight is calculated by dividing the count of the punctuation by the average of the maximal weight of previous feature types.
- Key-based, where a lexicon is used to assign polarity-based weights to each feature.

The SVM classifier implemented using these features achieved a maximum of 77.97% accuracy, with an f-score of 0.80 for the same.

2.3 Deep learning-based classification

A lot of recent research on sentiment analysis has focused on the application of deep learning techniques. Their operation may be described as the application of a hierarchy of concepts, where each concept is informed by a set of simpler concepts. Hence, a deep set of layers is formed.

Most commonly, this hierarchy takes the form of an artificial neural network, where the individual concepts are artificial neurons. These neurons are simple constructs that perform a weighted linear combination of a set of inputs. A mathematical function is applied on the result, and the value obtained is the output of the neuron. It is modelled after an actual human neuron. By adjusting the weights, we can modify when an output is obtained.

The Figure 2 shows two layers of neurons, where the outputs of one layer are given as inputs to another[13]. A number of variants

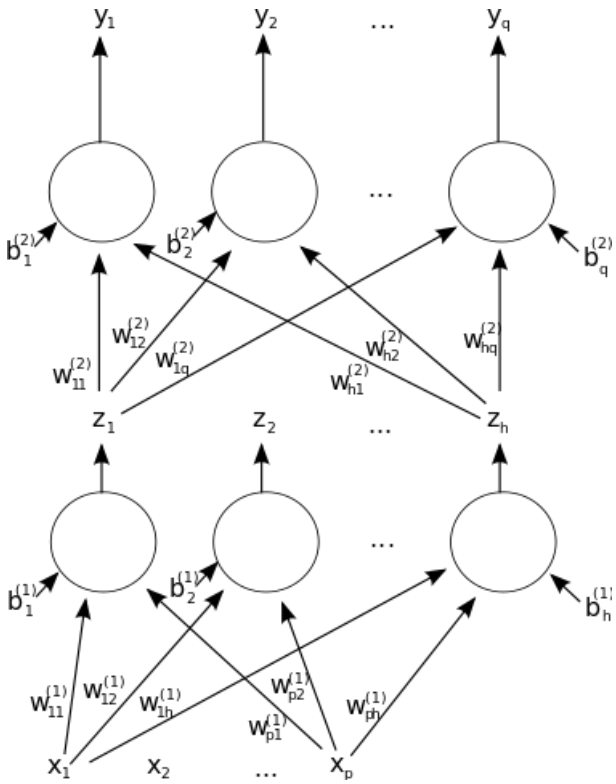

Fig. 2. Two layered neuron

of this type of network are in use. A very important variant is a convolutional neural network, where some of the non-terminal layers perform a convolution operation on the inputs to give the output.

Jianqiang et al. applied a convolutional neural network to determine the sentiment polarities of a corpus of tweets from Twitter. The preprocessing steps applied include[8],

(1) Removal of all non-ASCII and non-English characters.

(2) Removal of all URLs, numbers and stopwords. It is observed that they generally do not contain sentiment information.

(3) Replacement of negative references with expanded forms. For instance, wont is replaced with will not.

(4) Replacement of acronyms, slang and emoticons with their full or original text form using an online dictionary.

(5) Tokenization with Tweet-NLP, a freely available tool meant for tweets.

The tweets are represented as sets of features, which include the following[8],

(1) Word n-grams, specifically unigrams and bigrams, from the tweet.

(2) Twitter-specific features, such as the number of hashtags and emoticons.

(3) Word sentiment polarity score, which is the sum of the sentiment word polarity score of each word in the tweet. Polarities obtained from the AFINN and SentiWordNet lexicons.

(4) Word representations, specifically, word-level embeddings generated by unsupervised learning using the Global Vectors model for word representation.

The word vectors from the GloVe model are concatenated with the other listed features to form the final feature vector for a tweet.

The deep convolutional neural network model applied on the feature vectors achieved a maximum average classification accuracy of 87.62%, which was achieved on the STSTd dataset. An SVM approach on the same feature vectors and dataset achieved 81.61% accuracy[8].

3. ALGORITHMS FOR COMMUNITY DETECTION

Community detection algorithms are employed at various fields such as social sciences, biology, bibliometrics, etc[9]. Consequently,There are various types of community detection algorithms, such as community detection algorithms based on subgroup cohesiveness and mutuality, model based community detection, vertex clustering, divisive and quality optimisation based community detection algorithms and community detection algorithms which aim at finding overlapping communities in the network.[16, 11]. In this paper we are restricting our scope to community detection methods such as Louvain method, label propagation method and Markov clustering. Furthermore we will be discussing how these algorithms for community detection are being used along with sentiment analysis.

3.1 Louvain Method

Louvain method[2] is a heuristic method based on modularity optimization. It is a type of optimization method which aims to maximize modularity of the formed communities. Modularity is a measure of division of networks into modules, high modularity indicates dense intra-connection and sparse inter-connection and is given by :

$$Q = \frac{1}{2m}\sum_{ij}(A_{ij} - \frac{k_i k_j}{2m})\delta(c_i, c_j)$$

where A_{ij} represents the weight of the edge between i and j, $K_i = \sum_i(A_{ij})$ is the sum of the weights of the edges attached to the vertex i, c_i is the community to which vertex i is assigned, the δ function returns 1 if both nodes are in the same community otherwise 0, and $m = (1/2)\sum_{ij}(A_{ij})$. Modularity Q ranges from -1 to 1.

Louvain algorithm consists of two phases, which are repeated iteratively. The first phase concerns itself of assignment of community to each node and for each node i, we consider its neighbours j and check the modularity gain if i is removed from its initial community and is placed in j. If the modularity is increased and is maximum among the neighbours of i, we perform the operation otherwise leave the node as it is. This process is applied for all the nodes and first phase stops when local maxima is reached, i.e when there is no further gain in modularity. Modularity gain ΔQ is calculated as follows:

$$A = \frac{\sum_{in} + k_{in}}{2m} - \left(\frac{\sum_{tot} + k_i}{2m}\right)^2$$

$$B = \frac{\sum_{in}}{2m} - \left(\frac{\sum_{tot}}{2m}\right)^2 - \left(\frac{k_i}{2m}\right)^2$$

$$\Delta Q = A - B$$

In the above equation, $\sum_{in}$ is the sum of weights inside C, $\sum_{tot}$ is the sum of weights of links incident to nodes in C, K_i is the sum of weights if links incident to nodes i, $K_{i,in}$ is the sum of weights from i to nodes in C and m is the sum of weights of all links in network.

The second phase of the algorithm consists of building a new network whose nodes are now the communities found during the first phase. The edge weight between communities is simply the sum of all edges between communities. The passes are then iterated again until there are no more changes and a maximum of modularity is attained.

3.2 Label Propagation

In label propagation[19], each node is initialised with unique label and at every iteration of the algorithm each node adopts a label that a maximum number of its neighbours have. As the labels propagate, densely connected group of nodes forms a consensus on their labels, this in turn leads to communities being formed. At the end of the algorithm, the nodes with the same labels are grouped together into a single community.

Label propagation algorithm does not use any optimisation criterion unlike Louvian method, instead it uses the underlying network structure to form communities. This helps in simplifying the algorithm.

In label propagation algorithm, each node attempts to find the best neighbouring community to join, which would lead that node to have maximum number of neighbours being in the same community. Although ideally, label propagation algorithm should stop when there are no changes being made by the algorithm, there can be cases wherein a node have equal maximum number of neighbours in more than one community. Hence, we perform iterative process until every node in the network has a label to which the maximum number of its neighbours belong.

Therefore, the communities found by label propagation algorithm requires each node to have at least as many neighbours within its community as it has with each of the other community. This definition is similar to the definition of strong communities proposed by Radicchi et al. [18].

Steps for label propagation algorithm are as follows:

(1) Initialize all nodes with labels.

(2) Set $t = 1$

(3) Arrange nodes in the network in a random order and set it to X.

(4) for each $x \epsilon X$ chosen in that specific order let $C_x(t) = f(C_{x_{i1}}(t), ...C_{x_{im}}(t), C_{x_{i(m+1)}}(t-1), ..., C_{x_{ik}}(t-1))$. Note: f returns the label occurring with highest frequency among the neighbours and ties are broken uniformly and randomly.

(5) If every node has a label that the maximum number of their neighbours have, then stop the algorithm. else set $t = t + 1$ and goto 3.

Since label propagation algorithm starts with each node having a unique label, the initial groups formed are very small but as the algorithm proceeds, the momentum increases and larger groups starts to be apparent. However, when a consensus group reaches the border of another consensus group, competition for the membership of bordering nodes takes place and the algorithm converges when a global consensus is reached.

4. ADVANCEMENTS IN COMMUNITY DETECTION AND SENTIMENT ANALYSIS

The use of community detection algorithm has been employed successfully in various fields. Symeon et al.[16] surveys community detection algorithms with respect to social media. In the paper they discuss various community detection algorithms with the emphasis on performance of the algorithm with respect to social media analysis. The performance measure taken under considerations are computational complexity of various algorithms and their memory requirements. Symeon et al.[16] further discusses various applications using community detection with respect to social media analysis including Topic detection in collaborative tagging systems[15, 17], User profiling[6], Photo clustering[12], Event detection[20], etc.

Vishnu et al.[22] proposes a recommendation system for reddit which is able to take a user and give suggestions as to other related communities based on general user behavior. They start by creating a large weighted graph with edges between subreddits and then after finding communities within the graph they give a general recommendation based on the community a user belongs. They use louvain method[2] for community detection which is used for dimensionality reduction which further results in improvement over recommendation system proposed by Karypis[10]. Also an interesting finding was that user based recommendation was found to outperform item based recommendations, contrary to what Karypis[10] predicted which was able to be achieved due to dimensionality reduction using community detection with a 70.8% accuracy over 53.6% accuracy.

News media have been frequently criticized for failing to display a wide range of viewpoints, owing to it Jonathan et al[21] proposes a way to diversify news comments through the use of community detection algorithm on a network of voting data to identify common sentiment group in news discussion thread. They take a note that many previous research considers entities for forming community but in the process they miss a vital information in the network data, which is of voting. Voting is one of the more common interactions in most of the forums, including Reddit, Facebook and Twitter, also voting helps in identifying the common sentiment between users as in weather they have the same attitude towards a topic or they defer. Thus, they form a network wherein nodes are users who have interacted with the story and edges representing the level of agreement between users. The graph is then fed to community detection algorithm and then systems ability to categorise the commenters based on sentiment is compared to humans ability, consequently they found that the average agreement of the system (62.09%) is within a single standard deviation (8.73) of the participants average (66.37%).

5. CONCLUSION

We started off by discussing the importance of sentiment analysis and community detection with reference to social media, we then went ahead to discuss several generally accepted definitions of the problems and then discussing algorithms which are generally used such as Naive Bayesian Classification, Support vector machine classification, deep learning based classification for sentiment analysis and louvain method, label propagation method for community detection. After that we ended by discussing two recent papers in which community detection and sentiment analysis have been used for purposes such as creating a recommendation system and for showing diversified news.

6. ACKNOWLEDGMENT

This research was carried out under the guidance of Mr. Suraj Patil, MPSTME.

7. ADDITIONAL DOCUMENT STYLE OPTIONS

The following additional style option is available with the ijcaArticle class file:

Please place any additional command definitions at the very start of the LaTeX file, before the `\begin{document}`. For example, userdefined `\def` and `\newcommand` commands that define macros for technical expressions should be placed here. Other author-defined macros should be kept to a mininum.

Commands that differ from the standard LaTeX interface, or that are provided in addition to the standard interface, are explained in this guide. This guide is not a substitute for the LaTeX manual itself. Authors planning to submit their papers in LaTeX are advised to use `\ijcaArticle.cls` as early as possible in the creation of their files.

8. REFERENCES

[1] Data never sleeps, (accessed March 20, 2020).

[2] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10):P10008, 2008.

[3] Prerna Chikersal, Soujanya Poria, and Erik Cambria. Sentu: sentiment analysis of tweets by combining a rule-based classifier with supervised learning. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 647–651, 2015.

[4] Georgios Drakos. Support vector machines vs logistic regression, (accessed March 20, 2020).

[5] Pablo Gamallo and Marcos Garcia. Citius: A naivebayes strategy for sentiment analysis on english tweets. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014*. Citeseer, 2014.

[6] Jonathan Gemmell, Andriy Shepitsen, Bamshad Mobasher, and Robin Burke. Personalizing navigation in folksonomies using hierarchical tag clustering. In *International Conference on Data Warehousing and Knowledge Discovery*, pages 196–205. Springer, 2008.

[7] Mohammad Rezwanul Huq, Ahmad Ali, and Anika Rahman. Sentiment analysis on twitter data using knn and svm. *IJACSA) International Journal of Advanced Computer Science and Applications*, 8(6):19–25, 2017.

[8] Zhao Jianqiang, Gui Xiaolin, and Zhang Xuejun. Deep convolution neural networks for twitter sentiment analysis. *IEEE Access*, 6:23253–23260, 2018.

[9] Arzum Karataş and Serap Şahin. Application areas of community detection: A review. In *2018 International Congress on Big Data, Deep Learning and Fighting Cyber Terrorism (IBIGDELFT)*, pages 65–70. IEEE, 2018.

[10] George Karypis. Evaluation of item-based top-n recommendation algorithms. In *Proceedings of the tenth international conference on Information and knowledge management*, pages 247–254, 2001.

[11] István A Kovács, Robin Palotai, Máté S Szalay, and Peter Csermely. Community landscapes: an integrative approach to determine overlapping network module hierarchy, identify key nodes and predict network dynamics. *PloS one*, 5(9), 2010.

[12] Xiaowei Li, Changchang Wu, Christopher Zach, Svetlana Lazebnik, and Jan-Michael Frahm. Modeling and recognition of landmark image collections using iconic scene graphs. In *European conference on computer vision*, pages 427–440. Springer, 2008.

[13] Mcstrother. Two layer ann, (accessed March 17, 2020.

[14] Vivek Narayanan, Ishan Arora, and Arjun Bhatia. Fast and accurate sentiment classification using an enhanced naive bayes model. In *International Conference on Intelligent Data Engineering and Automated Learning*, pages 194–201. Springer, 2013.

[15] Symeon Papadopoulos, Yiannis Kompatsiaris, and Athena Vakali. A graph-based clustering scheme for identifying related tags in folksonomies. In *International conference on data warehousing and knowledge discovery*, pages 65–76. Springer, 2010.

[16] Symeon Papadopoulos, Yiannis Kompatsiaris, Athena Vakali, and Ploutarchos Spyridonos. Community detection in social media. *Data Mining and Knowledge Discovery*, 24(3):515–554, 2012.

[17] Symeon Papadopoulos, Athena Vakali, and Yiannis Kompatsiaris. Community detection in collaborative tagging systems. In *Community-Built Databases*, pages 107–131. Springer, 2011.

[18] F Radicchi, C Castellano, F Cecconi, V Loreto, and D Parisi. Self-contained algorithms to detect communities in networks. *Proc. Natl. Acad. Sci. USA*, 101:2658, 2004.

[19] Usha Nandini Raghavan, Réka Albert, and Soundar Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical review E*, 76(3):036106, 2007.

[20] Hassan Sayyadi, Matthew Hurst, and Alexey Maykov. Event detection and tracking in social streams. In *Third International AAAI Conference on Weblogs and Social Media*, 2009.

[21] Jonathan Scott, David Millard, and Pauline Leonard. Identifying similar opinions in news comments using a community detection algorithm. In *International Conference on Social Informatics*, pages 98–111. Springer, 2015.

[22] Vishnu Sundaresan, Irving Hsu, and Daryl Chang. Subreddit recommendations within reddit communities, 2014.