

SentiLangN: A Language-Neutral Graph-Based Approach for Sentiment Analysis in Microblogging Data

Muhammad Abulaish

Department of Computer Science
South Asian University, New Delhi, India
abulaish@sau.ac.in

Habeebullah Ebrahemi

Department of Computer Science
South Asian University, New Delhi, India
habeebullahf@gmail.com

Mohammad Baqir Rahimi

Department of Computer Science
South Asian University, New Delhi, India
baqerrahimi50@gmail.com

Amit Kumar Sah

Department of Computer Science
South Asian University, New Delhi, India
amitcsrs@students.sau.ac.in

ABSTRACT

In this paper, we present a language-neutral graph-based sentiment analysis approach, SentiLangN, which uses character n -gram graph for modelling textual data to handle language-neutral unstructured expressions and noisy data. Since ordering and positioning of characters and words in a document plays a vital role in content analysis, the SentiLangN employs the longest common subsequence and degree similarity to capture inherent semantics of the textual data.

SentiLangN introduces averaged character n -gram graph model and an application of *long-short-term memory* (LSTM) approach for sentiment analysis. The performance of SentiLangN is evaluated over real Twitter dataset, and it performs better than the individual n -gram graph models and traditional machine learning algorithms like C4.5. It is also compared with one of the state-of-the-art methods and performs significantly better.

CCS CONCEPTS

• **Information systems** → **Data analytics**; *Sentiment analysis*; • **Human-centered computing** → Social network analysis.

KEYWORDS

Text mining, Sentiment analysis, n -gram graphs, Machine learning, LSTM

ACM Reference Format:

Muhammad Abulaish, Mohammad Baqir Rahimi, Habeebullah Ebrahemi, and Amit Kumar Sah. 2019. SentiLangN: A Language-Neutral Graph-Based Approach for Sentiment Analysis in Microblogging Data. In *IEEE/WIC/ACM International Conference on Web Intelligence (WI '19)*, October 14–17, 2019, Thessaloniki, Greece. ACM, New York, NY, USA, 5 pages. <https://doi.org/10.1145/3350546.3352568>

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

WI '19, October 14–17, 2019, Thessaloniki, Greece

© 2019 Association for Computing Machinery.

ACM ISBN 978-1-4503-6934-3/19/10...\$15.00

<https://doi.org/10.1145/3350546.3352568>

1 INTRODUCTION

Microblogging sites like Twitter have been turned into a well-spring of varied information because they facilitate a large user-base to exchange views on different issues around the world. The dynamicity and variety of such systems resulted in the generation of informal tweets containing slangs, abbreviations, emojis, slams, misspelt and multilingual words, and unstructured expressions. Therefore, developing text information processing systems, including sentiment analysis, from such data is a challenging task. Sentiment analysis is a process of distinguishing and classifying the inclination or tendency which is expressed in a bit of content to demonstrate whether user's frame of mind towards a particular topic, product, public figure, or an event is positive, negative or neutral. Researchers have been working in this field for long, and they have come up with various techniques to analyse whether a user is happy or unhappy with the services availed. It helps equally other users, as well as the service providers to understand the requirements of the users.

Nowadays, Twitter is generally considered as a data source to test text information processing systems due to its extensive outreach and inherent complexities. As indicated by most recent investigations, Twitter is the second most prominent social networking site, with roughly 310 million users, posting about 500 million tweets every day¹. Inherent complexities in Twitter data is due to the character limit imposed on a post (*aka* tweet), which was recently upgraded from 140 to 280 characters. It leads to various data analysis issues such as sparsity, use of slang words, noise due to misspellings, and multilingualism.

In this paper, we propose a language-neutral graph-based sentiment analysis approach, SentiLangN, which uses character n -gram graph for modelling textual data to handle language-neutral unstructured expressions and noisy data. We utilise character n -grams instead of generating a bag of n -grams because it captures more information and makes no presumptions on the language of the text documents. Since ordering and positioning of characters and words in a document plays a vital role in content analysis, the SentiLangN employs the longest common subsequence and degree similarity to capture inherent semantics of the textual data. It also uses PageRank algorithm[10] to assign weight to the edges of the character n -gram graph, based on the importance of the connecting nodes. We have

¹<https://buffer.com/library/social-media-sites>

also used long-short-term memory (LSTM) [5], which is a variant of the recurrent neural network (RNN). It is mainly designed to handle long-range dependencies more accurately than conventional RNN, which suffers from vanishing gradient problem.

2 RELATED WORKS

Sentiment analysis is one of the well-established text information processing problems, and initially, it was targeted to map reviews or surveys to bipolar classes like positive or negative [3, 6]. However, nowadays, many researchers have considered it as a ternary or multi-class problem. In [12], an unsupervised lexicon-based approach is proposed to develop a *semantic orientation calculator* which incorporates intensification and negation to the semantic polarity of the opinion words. In [13], authors proposed a graph-based text similarity measure, which incorporates named entity information and n -gram graph model to represent documents that are further clustered using k -means algorithm. In [9], the authors proposed an approach to represent documents as a collection of important sentences based on the presence of important words identified through PageRank algorithm. After that, they used k -means algorithm for cluster analysis. In [8], authors introduced a word's dynamic sentiment polarity based on its context and proposed a sentiment analysis technique over Chinese documents. In [1], authors proposed character n -gram graph model for text information representation and considered some graph-based similarity measures as input to traditional classification algorithms for sentiment analysis.

In [14], authors proposed an attention-based LSTM network for cross-lingual sentiment classification. In [11], authors used deep learning system for sentiment analysis of tweets by feeding convolutional neural networks (CNN) with high-quality embeddings to get good parameter initialisation. In [7], authors performed a series of experiments with CNN using pre-trained word vectors for sentence-level classification tasks. In [2], authors used a hybrid deep learning architecture for coarse-grained (i.e., sentence-level) as well as fine-grained (i.e., aspect level) sentiment analysis. It can be observed from the above discussion that a good number of works have been done for sentiment analysis over Twitter and other data sources, but very few of them are language-neutral. SentiLangN maintains language neutrality, and it is also noise-tolerant with significant accuracy.

3 PROPOSED APPROACH

This section presents the functional details of the proposed language-neutral graph-based sentiment analysis approach, SentiLangN. Figure 1 visualizes the work-flow of the SentiLangN, and a detailed description of the individual processes is presented in the following sub-sections.

3.1 Data Pre-processing

We have used Twitter dataset to evaluate SentiLangN. Tweets differ a lot from the correct grammatical and syntactical structure of a language, and they usually contain punctuations, numbers, special characters, URLs, @mention, retweets, hashtags, and ampersands, which does not contribute to the classification process. Rather, their presence leads to the production of a tremendous amount of vertices

in n -gram graphs, which eventually increases the computational complexity of the process. Therefore, we have filtered out all special characters mentioned above from the dataset.

3.2 Character n -Gram Graph Construction

In this work, we have used character n -gram graph model instead of a word n -gram graph model. Character n -grams are useful in analysing multilingual data taken from social media platforms as they consider characters instead of the entire word for identifying patterns. This property maintains language neutrality and also shows high tolerance to noise and spelling mistakes. Character n -gram graph model has been proposed for text summarization task in [4].

An n -gram graph G can be defined as a directed graph $G = (V^G, E^G, \omega^G)$, where V^G is the set of vertices labelled with n -grams, E^G is the set of edges, and $\omega^G \subseteq \mathbb{R}$ is the set of weights assigned to each edge of E^G . An n -gram graph is characterised by three parameters – (i) L_{min} : the minimum n -gram length, (ii) L_{max} : the maximum n -gram length, and (iii) δ_{win} : the window size (neighbourhood distance). In this work, we have taken $L_{min} = L_{max} = \delta_{win}$, which is empirically verified to give execution close to the optimal one, as discussed in [4].

The n -gram graph can model a single tweet or a collection of tweets uniformly through a single graph with the help of an update function. Given n^{th} tweet $t_n \in T$ of a tweet dataset T , it initially builds an empty graph G^x . Thereafter, tweet t_n is transformed into an n -gram graph G^y with co-occurrence frequency of the vertex-pairs as edge weights. Finally, G^y is merged with G^x to form a new updated graph $G^z = (V^{G^z}, E^{G^z}, \omega^{G^z})$, where $V^{G^z} = \{V^{G^x} \cup V^{G^y}\}$, $E^{G^z} = \{E^{G^x} \cup E^{G^y}\}$, and weight of an edge $e \in E^{G^z}$ is given by equation 1 as discussed in [1]. The division by n in the computation of the edge weight in this equation ensures that the weight converges to the mean value of the individual edge weights.

$$\omega^{G^z}(e) = \omega^{G^x}(e) + \frac{\omega^{G^y}(e) - \omega^{G^x}(e)}{n} \quad (1)$$

Once the n -gram graph is constructed, SentiLangN re-calculates weights of each edges using PageRank algorithm. To this end, PageRank algorithm is run on the n -gram graph which determines importance score of all vertices in the graph, and final weights of the edges are determined as the average of importance scores of the respective vertex-pairs.

In this work, we construct two different character n -gram graphs namely (i) *tweet representative graph*, which models each tweet as an n -gram graph, and (ii) *class representative graph*, which is constructed from the tweets of a particular polarity class. As discussed earlier, starting from an empty graph, *tweet representative graph* G_i is constructed for each tweet $t_i \in T$. In order to construct *class representative graph* G^c corresponding to each class, starting with an empty graph, we update character n -gram graph in a recursive manner for each tweets of the particular class.

3.3 Graph Similarity-Based Feature Generation

This section presents various graph-based similarity measures and feature extraction process to represent each tweet as a numeric feature vector.

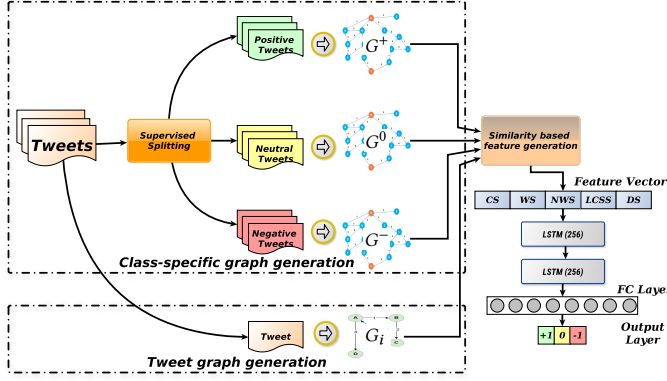


Figure 1: Work-flow of the proposed approach

3.3.1 *Similarity Measures.* A tweet $t_i \in T$ is converted into a feature vector \vec{t}_i containing five different similarity values between a tweet representative graph G_i and class representative graph $G^c \in \{G^+, G^0, G^-\}$, where G^+ , G^0 and G^- denote the *positive*, *neutral*, and *negative* class representative graphs, respectively. The similarity measures used in this study are briefly discussed in the following paragraphs.

- (1) *Correlative Similarity (CS):* It represents the proportion of the edges of G_i that are present in G^c , and formally defined in equation 2, where the value of $\mu(e, E^G) = 1$, if $e \in E^G$, otherwise 0.

$$CS(G_i, G^c) = \frac{\sum_{e \in E^{G_i}} \mu(e, E^{G^c})}{\min\{|E^{G_i}|, |E^{G^c}|\}} \quad (2)$$

- (2) *Weighted Similarity (WS):* It takes into account how many edges of G_i are contained in graph G^c along with the weight of the matching edges, and formally defined using equation 3.

$$WS(G_i, G^c) = \frac{\sum_{e \in E^{G_i}} \frac{\min\{\omega^{G_i}(e), \omega^{G^c}(e)\}}{\max\{\omega^{G_i}(e), \omega^{G^c}(e)\}}}{\max\{|E^{G_i}|, |E^{G^c}|\}} \quad (3)$$

- (3) *Normalised Weighted Similarity (NWS):* The NWS separates the weighted similarity from the dominance of the giant graph's size. It is a significant, inferred measure which is computed between using equation 4, where SS refers to size similarity given by equation 5.

$$NWS(G_i, G^c) = \frac{WS(G_i, G^c)}{SS(G_i, G^c)} \quad (4)$$

$$SS(G_i, G^c) = \frac{\min\{|E^{G_i}|, |E^{G^c}|\}}{\max\{|E^{G_i}|, |E^{G^c}|\}} \quad (5)$$

- (4) *Longest Common Substring Similarity (LCSS):* The LCSS demonstrates the longest sequence of strings that are common in G_i and G^c . Here, by string we refer to character n -grams. LCSS is calculated using equation 6, where \mathcal{E} refers to longest common substring, which is the longest subsequence of character n -grams connected in the same sequence in both G_i and G^c .

$$LCSS(G_i, G^c) = \frac{\sum_{e \in \mathcal{E}} \omega^{G_i}(e)}{\sum_{e \in E^{G_i}} \omega^{G_i}(e)} \quad (6)$$

- (5) *Degree Similarity (DS):* DS is the summation of ratio of degree of matching vertices between G_i and G^c . For any vertex $v \in V^G$, $deg(v) = indegree(v) + outdegree(v)$. Equation 7 formally defines DS, where $l \in \{V^{G_i} \cap V^{G^c}\}$ and $deg(l^G)$ represents the degree of vertex $l \in V^G$.

$$DS(G_i, G^c) = \sum_{l \in \{V^{G_i} \cap V^{G^c}\}} \frac{deg(l^{G_i})}{deg(l^{G^c})} \quad (7)$$

3.3.2 *Feature Vector Generation.* We model each tweet $t_i \in T$ as a tweet representative graph G_i and calculate its similarity with each of the class representative graphs $G^c \in \{G^+, G^0, G^-\}$ using the similarity measures discussed in the previous section. Finally, each tweet t_i is converted into a 15-dimensional numeric feature vector \vec{t}_i containing five similarities values of t_i with each class representative graphs.

Discretisation-based feature generation: Since the similarity values of a tweet representative graph with different class representative graphs may differ just by a small value; it becomes difficult for any classifier to learn discriminating patterns. To counter this problem, the authors in [1] adopted a mechanical strategy to discretise similarity values for improving classification accuracy. For a given tweet t_i , all its similarity values with different class representative graphs are considered, and the highest value is replaced with 1 and the rest of the values with 0. This is done for each similarity measure category.

Averaging-based feature generation: In this case, we vary the value of n to generate different character n -gram graphs, and similarity values obtained through each graph (bi-gram, tri-gram, and quad-gram graphs) are averaged for feature vector generation. Since each size of n -gram graph captures different contextual contents, the averaged feature vector is expected to perform better in the classification task.

4 EXPERIMENTAL SETUP AND RESULTS

SentiLangN is implemented in Keras, a high-level neural network API written in Python, and all experiments are performed on a PC with Intel Xeon E5 – 2650 16–Core 2GHz with 64GB RAM. The open-source library JInsect12 is used to implement the character n -gram graphs. We have used two-layer stacked LSTM with 256 memory cells at each layer. The LSTM layer is followed by a fully-connected (FC) layer with 512 neurons, which is further connected to a softmax (output) layer. The number of neurons at the softmax layer depends upon the number of classes, and it gives a class-wise probability of an instance to belong to a certain class.

Xavier Glorot initialiser is used to assign initial weights, and adam is used as an optimiser. To reduce the overfitting effect, our model has used dropout and L2 regularizer. Finally, C4.5 is implemented using Weka.

Table 1: Binary polarity classification (continuous)

Approach	Accuracy		Precision		Recall		F-score	
	C4.5	LSTM	C4.5	LSTM	C4.5	LSTM	C4.5	LSTM
SentiLangN (n=2)	77.70	80.00	78.00	80.00	79.00	80.00	78.40	80.00
SentiLangN (n=3)	79.50	85.60	81.20	83.40	81.50	83.00	81.35	83.20
SentiLangN (n=4)	81.50	87.20	82.00	86.00	82.00	86.00	81.53	86.00
SentiLangN (Avg)	81.80	87.80	81.30	87.00	81.50	87.80	81.40	87.40
SANGG [1] (n=2)	78.30	82.30	78.30	82.00	78.40	83.00	78.35	82.50
SANGG [1] (n=3)	80.60	85.40	79.00	85.00	80.19	83.00	79.59	83.99
SANGG [1] (n=4)	80.03	85.70	80.04	84.10	80.00	85.01	80.02	84.55
SANGG [1] (Avg)	82.14	86.45	81.30	86.00	81.45	86.12	81.37	86.06

Table 2: Binary polarity classification (discretised)

Approach	Accuracy		Precision		Recall		F-score	
	C4.5	LSTM	C4.5	LSTM	C4.5	LSTM	C4.5	LSTM
SentiLangN (n=2)	78.20	79.00	76.10	77.10	76.00	77.00	76.00	77.00
SentiLangN (n=3)	82.10	84.50	82.20	82.50	82.10	83.10	82.15	82.80
SentiLangN (n=4)	82.50	87.30	82.00	87.00	82.00	87.00	82.00	87.00
SANGG [1] (n=2)	80.20	82.19	80.00	82.00	80.01	82.00	80.00	82.00
SANGG [1] (n=3)	82.70	85.60	82.20	85.00	81.60	84.50	81.90	84.75
SANGG [1] (n=4)	82.60	86.30	82.00	85.90	82.60	86.00	82.30	85.95

Table 3: Ternary polarity classification (continuous)

Approach	Accuracy		Precision		Recall		F-score	
	C4.5	LSTM	C4.5	LSTM	C4.5	LSTM	C4.5	LSTM
SentiLangN (n=2)	69.72	70.00	70.70	71.00	70.00	71.00	70.00	71.00
SentiLangN (n=3)	71.60	75.01	71.60	78.00	72.70	72.00	72.15	74.88
SentiLangN (n=4)	71.80	76.40	75.00	77.00	72.00	75.00	73.00	76.00
SentiLangN (Avg)	72.40	76.81	72.00	76.00	71.40	75.40	71.70	75.70
SANGG [1] (n=2)	68.50	72.34	69.10	72.30	68.20	72.00	68.65	72.15
SANGG [1] (n=3)	69.90	74.60	70.00	74.00	70.01	74.01	70.00	74.00
SANGG [1] (n=4)	68.90	74.08	69.00	74.60	68.00	73.00	68.50	73.79
SANGG [1] (Avg)	72.40	75.00	72.15	74.80	71.40	74.00	71.77	74.39

Table 4: Ternary polarity classification (discretised)

Approach	Accuracy		Precision		Recall		F-score	
	C4.5	LSTM	C4.5	LSTM	C4.5	LSTM	C4.5	LSTM
SentiLangN (n=2)	67.20	71.00	67.40	68.00	67.00	68.00	67.00	68.00
SentiLangN (n=3)	71.70	74.80	71.80	74.00	71.70	73.00	71.75	73.50
SentiLangN (n=4)	73.50	76.02	74.00	75.00	74.00	74.00	74.00	74.50
SANGG [1] (n=2)	70.80	71.26	71.30	71.00	70.00	71.01	70.64	71.00
SANGG [1] (n=3)	73.12	73.98	73.10	73.40	73.10	73.60	73.10	73.50
SANGG [1] (n=4)	73.07	73.86	72.00	73.40	73.00	72.00	72.49	72.69

4.1 Dataset

We have used publicly available dataset² having 8922 real-world tweets. It contains 7605 tweets as training data and 1317 as test data. The dataset has 3 polarity classes, namely positive, negative, and neutral.

For binary polarity classification, we selected only positive and negative tweets from training and test datasets, whereas for ternary polarity classification, we have used the entire dataset.

²<http://www.kaggle.com/rabinandan/twitter-sentiment>

4.2 Evaluation Results and Comparative Analysis

The performance of SentiLangN is evaluated using LSTM and C4.5 in terms of *accuracy*, *precision*, *recall*, and *f-score* for both binary and ternary classification tasks. It is also compared with one of the state-of-the-art approaches “*sentiment analysis of social media content using n-gram graphs* (SANGG)” proposed in [1]. It may be noted that we have considered non-discretised values as continuous, and the averaging-based *n*-gram model is not applicable on discretised feature vectors.

4.2.1 Binary Polarity Classification. Table 1 shows the experimental results of binary polarity classification using continuous *n*-gram graph model. It can be observed that *accuracy* of SentiLangN using LSTM is better in all the cases except for *n* = 2. It can also be observed that performance of SentiLangN using LSTM in terms of *precision*, *recall*, and *f-score* is better in case of 4-gram and averaging-based *n*-gram graph model.

Table 2 presents the experimental results of binary polarity classification using discretised *n*-gram graph model. It can be observed that SANGG using LSTM performs better in terms of all the evaluation metrics for *n* = 2, 3, whereas SentiLangN using LSTM performs better for *n* = 4.

4.2.2 Ternary Polarity Classification. Table 3 presents the experimental results of ternary polarity classification using continuous *n*-gram graph model. It can be observed that *accuracy*, *precision*, and *f-score* of SentiLangN using LSTM is better in all the cases except for *n* = 2, whereas it performs better in terms of *recall* for *n* = 4 and averaging-based *n*-gram graph model.

Table 4 presents the experimental results of ternary polarity classification using discretised *n*-gram graph model. It can be observed that SentiLangN using LSTM performs better in terms of all the evaluation metrics for *n* = 3, 4, whereas SANGG performs better for *n* = 2 in terms of *accuracy* using C4.5, and in terms of *precision*, *recall*, and *f-score* using LSTM. In summary, SentiLangN outperforms SANGG in most of the cases. It is worth to mention that SentiLangN using LSTM performs extremely well for *n* = 4 in all the experiments which reflects its ability to handle long matching sequence of substrings.

5 CONCLUSION AND FUTURE WORK

In this paper, we have proposed a graph-based language-neutral sentiment analysis approach SentiLangN. The proposed approach models microblogging data using character *n*-gram graphs that are mainly used to convert each tweet into a numeric vector containing various graph similarity values. It uses PageRank algorithm to assign weights to the vertices and edges in the *n*-gram graph. Simple and intuitive similarity measures used in this study make the model more robust and language-neutral. Exploring more graph similarity measures and development of a hybrid model using contents and similarity values seems one of the promising areas of research.

REFERENCES

- [1] Fotis Aisopos, George Papadakis, and Theodora Varvarigou. Nov 30-11, 2011. Sentiment Analysis of Social Media Content Using N-Gram Graphs. In *Proceedings of the 3rd ACM SIGMM Int'l Workshops on Social Media, NY, USA*, Vol. 11. ACM, 9-14.

- [2] Md. Shad Akhtar, Ayush Kumar, Asif Ekbal, and Pushpak Bhattacharyya. December 11–16, 2016. A Hybrid Deep Learning Architecture for Sentiment Analysis. In *Proceedings of the 26th Int'l Conf. on Computational Linguistics (COLING), Osaka, Japan*. 482–493.
- [3] Khalid Al-Rowaily, Muhammad Abulaish, Nur Al-Hasan Haldar, and Majed Al-Rubaiana. 2015. BiSAL – A Bilingual Sentiment Analysis Lexicon to Analyze Dark Web Forums for Cyber Security. *Digital Investigation* 14 (2015), 53–62.
- [4] George Giannakopoulos, Vangelis Karkaletsis, George Vouros, and Panagiotis Stamatopoulos. 2008. Summarization System Evaluation Revisited: N-gram Graphs. *ACM Transactions on Speech and Language Processing* 5, 5 (2008), 1–36. <https://doi.org/10.1145/1410358.1410359>
- [5] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.* 9, 8 (Nov. 1997), 1735–1780. <https://doi.org/10.1162/neco.1997.9.8.1735>
- [6] Ahmad Kamal, Muhammad Abulaish, and Jahiruddin. 2016. OntoLSA – An Integrated Text Mining System for Ontology Learning and Sentiment Analysis. In *Sentiment Analysis and Ontology Engineering, Studies in Computational Intelligence 639*, Witold Pedrycz and Shyi-Ming Chen (Eds.). Springer, 399–423.
- [7] Yoon Kim. October 25–29, 2014. Convolutional Neural Networks for Sentence Classification. In *Proceedings of the Conf. on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar*.
- [8] R. Liu, R. Xiong, and L. Song. 2010. A Sentiment Classification Method for Chinese Document. In *Proceedings of the 5th Int'l Conference on Computer Science Education*. 918–922. <https://doi.org/10.1109/ICCSE.2010.5593462>
- [9] Yukari Ogura and Ichiro Kobayashi. August 4–9, 2013. Text Classification Based on the Latent Topics of Important Sentences Extracted by the PageRank Algorithm. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL), Sofia, Bulgaria*. 46–51.
- [10] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. 1999. *The PageRank Citation Ranking: Bringing Order to the Web*. Technical Report 1999-66. Stanford InfoLab. <http://ilpubs.stanford.edu:8090/422/> Previous number = SIDL-WP-1999-0120.
- [11] Aliaksei Severyn and Alessandro Moschitti. Aug 9–13, 2015. Twitter Sentiment Analysis with Deep Convolutional Neural Networks. In *Proceedings of the 38th Int'l ACM SIGIR Conf. on Research and Development in Information Retrieval, Santiago, Chile*. 959–962.
- [12] Maite Taboada, Julian Brooke, Milan Tofiloski, Kimberly Voll, and Manfred Stede. 2011. Lexicon-Based Methods for Sentiment Analysis and Computational linguistics. 37 (2011), 267 – 307.
- [13] Leonidas Tsekouras, Iraklis Varlamis, and George Giannakopoulos. Sep 04, 2017. Twitter A Graph-based Text Similarity Measure That Employs Named Entity Information. In *Proceedings of Recent Advances in Natural Language Processing, Varna, Bulgaria*. 765 – 771.
- [14] Xinjie Zhou, Xiaojun Wan, and Jianguo Xiao. 2016. Attention-Based LSTM Network for Cross-Lingual Sentiment Classification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP), Austin, Texas*. ACL, 247–256. <https://doi.org/10.18653/v1/D16-1024>