DOI: 10.1515/ausi-2015-0018

Sapiness–sentiment analyser

Katalin Tünde JÁNOSI-RANCZ
Sapientia Hungarian University of Transylvania
Dept. Mathematics and Informatics, Târgu-Mureş
email: tsuto@ms.sapientia.ro

Zoltán KÁTAI
Sapientia Hungarian University of Transylvania
Dept. Mathematics and Informatics, Târgu-Mureş
email: katai_zoltan@ms.sapientia.ro

Roland BOGOSI
Sapientia Hungarian University of Transylvania
Dept. Mathematics and Informatics, Târgu-Mureş
email: root@rolisoft.net

Abstract.
In our ever-evolving world, the importance of social networks is bigger now than ever. The purpose of this paper is to develop a sentiment analyzer for the Hungarian language, which we can then use to analyze any text and conduct further experiments. One such experiment is an application which can interface with social networks, and run sentiment analysis on the logged-in users friends' posts and comments, while the other experiment is the use of sentiment analysis in order to visualize the evolution of relationships between characters in a text.

1 Introduction

Sentiment analysis [10] is a technique used to determine the amount of positive and negative sentiment in a piece of text. Accurate opinion mining software

Computing Classification System 1998: G.2.2
Mathematics Subject Classification 2010: 68R15
Key words and phrases: Social networks, sentiment analysis, semantic web, prediction, classification

is very desirable due to the many uses it can serve, however this is one of the topics which does not easily carry over, as new parsers, dictionaries and neural networks have to be developed, compiled and trained, all while accounting for the features of the language.

Our sentiment analyzer targets the Hungarian language. Many challenges have presented themselves during the development, one of the main challenges was the handling of negation within a sentence. This is particularly hard, as without understanding the meaning of the words, a language parser will have a hard time deciding where to start and when to end a negation.

Suppose we are examining the following sentence:
"A kijelző minősége nem a legjobb és a kamerája is hagy kívánnivalót maga után, de ár/érték arányban verhetetlen a piacon!" ("The quality of the display is not the best and its camera also leaves something to be desired, but in value for the money it is unbeatable on the market!")

When analyzed by a human, this sentence is mostly neutral, as the device referred to in the sentence has both its ups and downs. However, an application will have trouble deciding how does the negation apply in the sentence above. Different implementation methods will yield different results. As described in section 3.2, if we try to invert the polarity of a fixed number of subsequent words, we will end up with a sentence which has a negative half, followed by a positive half, yielding a neutral sentence overall. However, if we decide to invert the polarity until the end of the sentence, we will get a negative sentence as the overall score. Unfortunately this can go both ways, as there are examples where the other implementation deems to be more accurate.

The Hungarian language (orthography) uses diacritics, and there are words which can have different meanings when written with and without diacritics. Throughout one of our experiments outlined in section 4.1, we have observed that the majority of posts and comments do not use diacritics. In an even worse situation, a sentence may contain mixed use of diacriticized and non-diacriticized words that should have been written with diacritics. In a sentence where diacritics are not used, and the word can have multiple meanings without the possibility of us distinguishing between them, we have to average the polarity of both meanings. However, in a mixed sentence, no averaging will be done, and the essentially misspelled word's meaning will be used, which may erroneously impact the overall score of the sentence.

Ambiguous words end up in the same boat as words with diacritics that were not diacriticized, however in this case it is not the user's fault, it is the fault of the language parser. Without knowing the proper meaning which the author intended to use in case of ambiguous words, we will have to average the polarity of the meanings, and use that. Unfortunately this is not an accurate solution, but a better solution is not possible with our implementation at this time.

Slang usage is another challenge, however this one can be solved by mapping the slang words to their non-slang counterpart. While this solves the issue, this also means an up-to-date database has to be kept with all the slangs, otherwise meanings may be missed when assessing sentences from the Internet.

Different domains may use words as different technical terms, and may even end up redefining the connotation of said words as a result. In order to correctly assess the connotation of a sentence in a specific domain, the application needs to be re-trained with regards to the connotation of the technical terms that it may encounter.

Sentences containing sarcasm and irony cannot be accurately assessed with simple language parsers. This results in a rather significant issue, as the use of sarcasm generally means the connotation of the sentence is completely inaccurate, as the meaning should have been negated, which is another challenge in and of itself. The sentiment analyzer presented within this paper does not address the use of sarcasm and irony.

2 Related work

In the field of Computer Science, Natural Language Processing is a wide subject, which has been broadly discussed. Most of the research done focuses on the English language, however due to the difference between the languages, the solutions proposed and implemented in those papers may not be easily applied to languages they did not focus on, as such this subject is highly language-dependent. The ascent of social media has attracted significant interest in sentiment analysis [10] techniques such as opinion mining.

Experiments with sentiment analysis which also use SentiWordNet[1] as a lexical resource, but focus on a language other than English have been conducted. A paper which discusses opinion mining user-generated movie and hotel reviews in the Portuguese language is [7], while a similar one exists for evaluating French movie reviews in [6]. A paper comparing various methods for sentiment analysis for the German language is in [12].

However, since opinion mining mainly targets user-generated content on social media, the use of humor, sarcasm and irony is rampant[13]. A paper targeting sentiment and irony analysis for the Italian language in [4] observes how users of social media generally use humor and irony, and how this affects methods used in sentiment analysis. In [14] a corpus of tweets is presented, where every tweet is annotated with an associated emotion, and can be used for further testing in this regard.

A publicly available corpus for the Hungarian language exists under the name of OpinHuBank[11], however we saw it unfit for our purposes as we are not doing entity-oriented analysis. The values assigned to sentences within OpinHuBank are -1, 0 and 1, which does not fit into our use case, as we would need to know the extent of positive and negative connotation. We ended up building our own corpus, as described in section 3.2.

A number of papers have been published previously which discuss sentiment analysis with a focus on the Hungarian language[16, 8]. In [3] the interaction of users is being analyzed and used to enhance traditional language processing techniques.

3 Methodology

3.1 Lexical approach

For the first version of the application, hereinafter referred to as *v1.0*, we took a lexical approach, with a database where every word is associated with a value representing its positive, negative and neutral connotations.

Since we were unable to find a fitting database openly available for the Hungarian language, but found enough for the English language, we decided to adapt an existing English database to Hungarian as a start. SentiWordNet[1] was ultimately found to be a fitting database whose structure suits our purposes. Their database contains 117,659 words in English with thesaurus attached and the words annotated with their polarity.

As we only had to do dictionary look-ups, we turned to Google Translate's API for translating the batch of words. Even though the translation was successful, and in theory it should have worked fine, upon inspecting the end result, we noticed multiple issues. The machine translation did not account for the correct meaning of words with multiple meanings, and as such most synonyms have been translated to the first Hungarian meaning, regardless what the actual meaning would have been. In an extreme edge-case, 30 different English words were translated to the same Hungarian word. To solve this, we tried to find synonyms for these Hungarian words. The translator would also ignore the part of speech of the translated word, for example the verb "*(to)* duck" would be translated to the noun "kacsa". In other cases, the assigned polarity would get invalidated, as the translated word does not share the same semantic orientation as the original, such as the relatively negative attribute "cheesy" would get translated to the neutral word "sajtos" in Hungarian.

After the failed machine translation attempt, we decided to manually translate the whole English database, carefully accounting for different meanings, part of speech, synonyms, and so on. This was a long, tedious and time-consuming operation, as even though we filtered the database to exclude words without significant polarity attached, we still had to manually comb through 50,973 words. Second step in the process was the processing of the input text we receive from the user, which we will query against the database. As we needed a Hungarian word stemmer, we used a library called magyarlánc[18], which was developed at the University of Szeged as a complete toolkit for linguistic processing of Hungarian texts. The database of the library was not complete at the time, and as such we had to extend it in order to support many other words. One example would be the word "román" (adj. "Romanian") which was stemmed to "rom" (n. "ruin") even though that is not the correct stem.

The web application, whose backend was developed in PHP, would process the input from the user, look up the polarities of the participating words, and compute the final arithmetic mean from their sum, in order to determine whether the specified text carried a predominantly positive, negative or neutral connotation.

This version is rather primitive as far as what is possible in the field of natural language processing. We compared it to several commercial products in subsection 4.2.

3.2 Neural networks

For the second version of our application, henceforth referred to as *v2.0*, we started from scratch as far as the algorithm is concerned, and ventured into the field of neural networks.

We greatly improved the Hungarian translation of our database in the meantime between the two versions, and then we went on to experiment with training a supervised learning linear classifier. After a few trial and error runs, we found the suitable configuration of the neural network to be consisting of 8 input values, 2 hidden layers and 1 output. The 8 input values are the positive and negative sentimental values, separately, of the verbs, nouns, adjectives and adverbs from the database. When given a sentence, we compute the sum of the sentimental values of verbs, nouns, adjectives and adverbs in the sentence, which is then fed to the input. The output of the neural network is a value between $[-1, 1]$ indicating the polarity of the specified sentence. For training, we used an annotated dataset consisting of 500 sentences from the Sentiment Treebank [15] which contains movie reviews.

While the initial results of the new application were promising, we set on to fine-tune the neural network by feeding it further data. However, in order to do this, first we had to compile another suitable training set.

In order to create a corpus in Hungarian for use as a training set for our sentiment analyzer application, we created a user-friendly web application whose purpose was to let prospective visitors give their feedback with regards to what level of positive, negative or neutral connotation they think each displayed sentence had. The sentences to be annotated consisted of user reviews fetched from various web sites, including car and hotel reviews, as well as the opinion of famous people about various topics. The sentences were stripped of any names, including brands and personal names. The annotation is a 2-way polarity with a numeric score for positive and negative sentimental value each. A total of 70 students used the application. The list of sentences to be annotated was generated in such a way, so as to always list the sentences with the least amount of annotations first, out of the 500 total, in order to ensure no sentences would be left without annotations. Students each annotated on average 60 to 120 sentences, resulting in each sentence being annotated at least 10 to 13 times.

The existing neural network was further trained with the 500 freshly annotated sentences. The application featuring the improved version of the neural network will be referred to as *v2.1* in later benchmarks, so as to easily distinguish the progress and improvements between the two networks.

We also implemented negation in this version. While there is no agreed-upon standard way to handle negation in a sentence, as described in papers [5, 9] the typical way to handle it is by inverting the polarity of the words surrounding the negation. As for the number of words to invert, aforementioned papers list the following possibilities to consider:

- Negate the sentence until the end.
- Negate a fixed number of subsequent words.
- Invert the polarity of the first sentiment-carrying word.
- Invert the polarity of the next non-adverb word.

We have determined the best approach is to invert the polarity of a fixed window of 3 words immediately following the negation, with the exclusion of stop words, and thus this is the solution we have ended up implementing in the application.

After much work on improving the efficiency of the sentiment analyzer, we have heavily benchmarked the results, which are available in subsection 4.2.

4 Experiments

In order to use the sentiment analyzer, we have come up with a few experiments which rely on real-world data and vary greatly in order to test all the edge-cases and implementation difficulties outlined in section 1.

4.1 Social network analysis

The idea of this experiment is to provide a web interface, on which the users can log in, and query a phrase they are interested in, after which the application will use data from the user's connected social networks, and determine the sentiment of the user's friends regarding the given query.

For the social network component, we have chosen to implement Facebook, as after the user has authorized us to do so, we had the ability to fetch the public posts and comments of the user's friends. As a result, a user may log in to our web interface, and they may search for a phrase, such as a movie or brand name, and the application will determine whether the friends of the user are mostly positive or negative with regards to it, if there are enough posts available regarding the queried phrase.

A few of the issues we have faced included the language variety, as not all the posts and comments of the examined users were in Hungarian. Secondly, the posts which were in fact in Hungarian, varied greatly on the use of diacritics: some were fully diacriticized, some were not at all, and some used diacritics interchangeably. If a text did not contain any diacritics, it is assumed to have been written without diacritics in the first place, and we implemented a special handling for those. We did not try to restore diacritics into the words, instead we generated a list from the database of the words which have the same letters when written without diacritics, and took the average of their polarity. In situations where diacritics were used interchangeably, the analyzer determined the sentence to have been written originally with diacritics due to the occurrence of at least one diacritic. In such cases, words which had meanings in both diacriticized and non-diacriticized forms were not properly accounted for. Last but not least, a significant amount of posts contained misspellings or unrecognized slang, which we had to try and solve by updating our synonyms database. We ended up manually checking 17,759 words and assigning them synonyms which were not yet in the database at that point.

We also implemented a complementary web browser extension for Google Chrome, which allowed users to select any phrase from any web page and perform sentiment analysis on it with our application, while also getting results on the sentiment of the user's friends regarding the queried phrase.

It should be noted that shortly after conducting the experiment, the API version used by our application (v1.0) was deprecated by Facebook on April 30th of 2015, thus we would be forced to switch if we were to continue the experiment. Newer versions of the Facebook API do not support the permissions we require in order to fetch posts and comments.

4.2 Comparison results

In order to determine the precision of our sentiment analyzer and get a progression indicator of our improvements, we continually tested our application against a few widely known existing implementations:

- *Alchemy API*[17] is a commercial product developed by IBM specializing in providing natural language processing services to developers via an API. The list of exposed services includes sentiment analysis.
- *Sentiment Treebank*[15] is a sentiment analysis application developed by the Stanford University. Their approach is to train a recursive neural network using a training set generated by users assigning a polarity to movie reviews.

For testing, we used $n = 100$ sentences containing movie reviews from imdb.com. After translating them to Hungarian we had two parallel sentence-sequences to compare the three applications. We denote by a_i, tb_i, s_i, $i = \overline{1, n}$ (**A**lchemy, **S**entiment **T**reebank, **S**apiness) the resulted sentiment polarity sequences. The performance of our in-house application against the aforementioned services is outlined in table 1.

The mean of deviation sequence between the two tested applications ($|a_i - tb_i|, i = \overline{1, n}$) is 32.29%, while the means of deviation sequences between our application and tested ones ($|a_i - s_i|, |tb_i - s_i|, i = \overline{1, n}$) range from 27.31% to 33.66%, well within the range of comparable results.

We conducted an experiment in which we take a 100 random sentences and their annotations from OpinHuBank's[11] database, and consider them to be the golden standard when testing against Sapiness. Out of the 100 sentences,

Comparison	**The means of deviation**
Alchemy – Treebank	32.29
Alchemy – Sapiness v1.0	52.16
Alchemy – Sapiness v2.0	36.54
Alchemy – Sapiness v2.1	33.66
Treebank – Sapiness v1.0	40.51
Treebank – Sapiness v2.0	30.34
Treebank – Sapiness v2.1	27.31

Table 1: Comparison of Sentiment Analyzers

Sapiness computed a correct sentimental value for 72 sentences, resulting in 72% correctness.

4.3 Plot evolution analysis

The main goal of our project is to detect and examine the informal e-communication network of the employees of Sapientia University (Faculty of Technical and Human Sciences), and to make organizational management suggestions based on this analysis (including the sentiment analysis of the members' e-messages). Since the employees of our faculty are not significantly present on social networks and we have not yet received permission to their email communication contents, we decided to test Sapiness by examining the relationship between characters from the bible.

In this experiment we tried to aim for a more advanced use of the analyzer, and made it our goal to analyze a character's evolution within a story as the plot progresses. We ran the analysis on a modern translation of the bible, examining the relationship between the characters, their interactions and relative emotional trajectory. For the purposes of this paper, we are going to present our findings when analyzing the relationship between David and God.

When preparing the text for processing, we removed all sentences which were irrelevant. We define "irrelevant" in this case as a sentence which does not have both the words "David" and "God" occurring in it. In such cases, we cannot infer a sentimental connotation onto these characters. The selected verses (containing the names of God and David and reflecting their relationship) were arranged in chronological order to characterize the relationship of the two characters over the time (horizontal axes). We assumed that the sen-

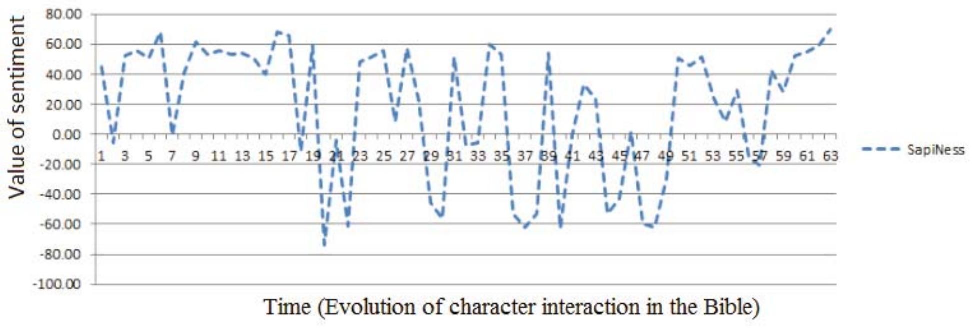

Figure 1: Relationship between David and God as determined by Sapiness

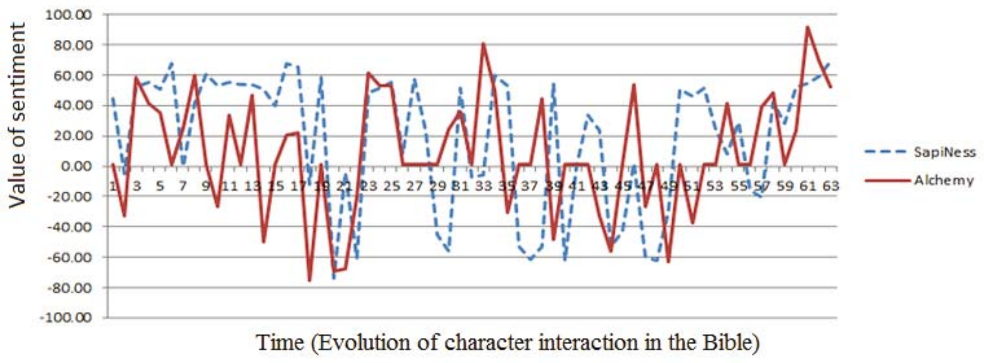

Figure 2: Comparison between Sapiness and Alchemy API

timent value sequence of these verses will model the analyzed relationship appropriately. As we expected, negative sentiment value sub-sequences reflect negative periods in the relationship between God and David. The analysis was done on a verse-by-verse basis, meaning that we assigned a sentimental value to each verse separately. If the verse contained multiple sentences, we calculated the sentimental value of each sentence, and then used the arithmetic mean of these values as the final sentimental value of the verse.

After running the analysis, we plotted the initial results in figure 1.

In order to benchmark the precision of our sentiment analyzer, we tried to re-run the analysis using a commercial application intended for this purpose, namely Alchemy[17]. As the chosen web service did not support the Hungarian language, we had to search for an English translation, which also used a modern language.

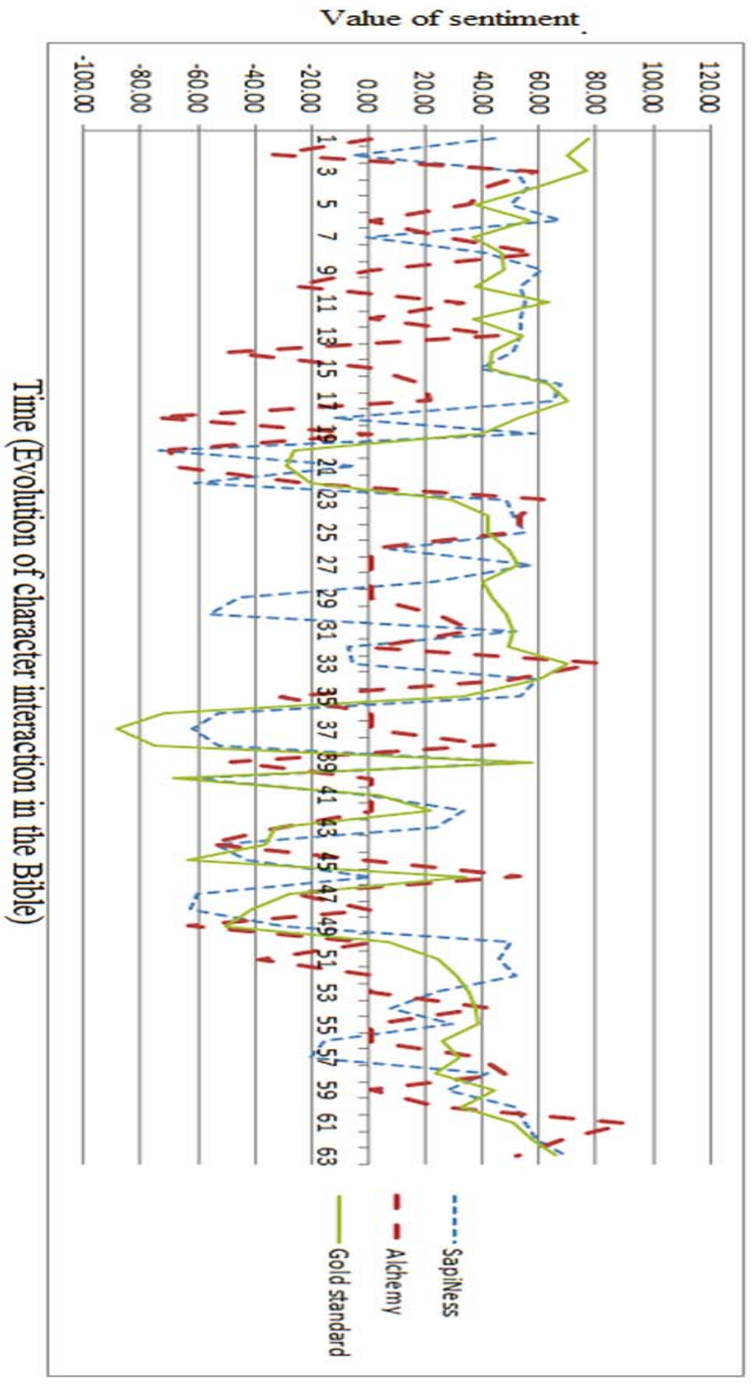

Figure 3: Comparison between AI and gold standard

The matches and discrepancies between our application and the commercial one is visualized in figure 2:

As the two results in figure 2 are the results of two automated non-human sentiment analyzers, we asked three independent person to provide their own opinions on what positive, negative and neutral connotation each analyzed sentence had.

The results of this manual analysis, overlapped with the previous two results, are visible in figure 3:

We chose as gold standard (GS: $g_i, i = 1, n$) the sentiment polarity sequence that was given by the three independent person ($g_i = (e1_i + e2_i + e3_i)/3, i = 1, n$). The means of deviation sequences "GS vs. Sapiness" ($|g_i - s_i|, i = 1, n$) and "GS vs. Alchemy" ($|g_i - a_i|, i = 1, n$) were 23.82 and 36.77, respectively. Applying a paired t-test we found that the Sapiness result was significantly better than the Alchemy's one ($p = 0.005 < 0.05$).

Analyzing the bible was our own idea, however after analysis we found that similar experiments have already been conducted, but no paper has been published on the subject.

5 Conclusions and future work

We took the idea of sentiment analysis, and implemented it using two different methods, while testing it against existing commercial products and research applications. We also conducted two experiments which mirrored practical applications and whose main building block was our sentiment analyzer. After examining the results of our benchmarks in subsection 4.2, the results of the experiment in subsection 4.3, we can conclude that we have significantly improved our sentiment analyzer application with every iteration, and we are currently at a point where our application produces results which are comparable to commercial applications and research products.

Future work on the project may include a variety of objectives, but the main goal is improving the accuracy of the analyzer. We may take multiple roads to approach this goal, such as trying to compile better training sets, try to tweak the configuration of the neural network, and even trying other types of neural networks which can be used for our purposes, such as Support Vector Machines and Naïve Bayes.

As for the experiment conducted in section 4.3, we may try to compile a training set from a literature piece of an author, then try to use the newly trained neural network on a different work of fiction from the same author, and test how much improvement, if any of significance, can be had by training the neural network to recognize connotations for a specific style of writing, instead of generalizing it.

Further experimentation can be conducted in order to compare the accuracy of different handling modes for negation, as listed in section 3.2.

We should also study the use of links between SentiWordNet[1] and wordnet synsets in other languages, as presented in paper [2], in order to automatically generate a corpus for any language, solving the issue presented in section 3.2 regarding words with multiple meanings.

We are planning to use the presented application (Sapiness – Sentiment Analyser) for detecting, modeling and characterizing the informal network of the Faculty of Technical and Human Sciences of Sapientia University (based on the day-to-day electronic interactions of its members.) After the informal social digraph has been established each arc will be characterized by the sentimental content of the corresponding messages. We hope that the formal management of our Faculty will take advantage of the expected results of this research.

References

[1] S. Baccianella, A. Esuli, F. Sebastiani, SentiWordNet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining, *LREC 2010, Seventh International Conference on Language Resources and Evaluation*, Valetta, Malta, May 17–23, 2010, pp. 2200–2204. ⇒188, 189, 198

[2] V. Basile, M. Nissim, Sentiment analysis on Italian tweets, *Proc. 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, June 14, 2013, Atlanta, Georgia, USA, pp. 100–107. ⇒198

[3] G. Berend, R. Farkas, Opinion mining in Hungarian based on textual and graphical clues, *Proc. 8th WSEAS International Conference on Simulation, Modelling and Optimization*, September 23–25, 2008, Santander, Spain. ⇒189

[4] C. Bosco, V. Patti, A. Bolioli, Developing corpora for sentiment analysis: The case of Irony and SentiTUT, *Intelligent Systems, IEEE* **28,** 2 (2013) 55–63. ⇒188

[5] M. Dadvar, C. Hauff, F. de Jong, Scope of negation detection in sentiment analysis, *DIR 2011: the eleventh Dutch-Belgian information retrieval workshop*, 2011, pp. 16–19. ⇒191

[6] H. Ghorbel, D. Jacot, Sentiment analysis of french movie reviews, *Advances in Distributed Agent-Based Retrieval Tools*, Vol. 361, 2011, pp 97–108. ⇒188

[7] L. A. Freitas, R. Vieira, Ontology based feature level opinion mining for portuguese reviews, *Proc. WWW '13 Companion Proceedings of the 22nd International Conference on World Wide Web*, Rio de Janeiro, May 13–17, 2013, pp. 367–370. ⇒188

[8] V. Hangya, R. Farkas, G. Berend, Entitásorientált véleménydetekció webes híranyagokból, In: Tanács, A., Varga V., Vincze V. (eds.) *XI. Magyar Számítógépes Nyelvészeti Konferencia (MSZNY 2015)*, Szeged, Jan. 14–15, 2015. Univ. Szeged, pp. 227–234. ⇒189

[9] A. Hogenboom, P. van Iterson, B. Heerschop, F. Frasincar, U. Kaymak, Determining Nnegation scope and strength in sentiment analysis, *2011 IEEE International Conference on Systems, Man and Cybernetics – SMC*, October 9–12, 2011, Hilton Anchorage, Anchorage, AK, USA, pp. 2589–2594. ⇒191

[10] B. Liu, Sentiment analysis and opinion mining, *Synthesis Lectures on Human Language Technologies* **5**, 1 (2012) 1–167 ⇒186, 188

[11] M. Miháltz, OpinHuBank: szabadon hozzáférhető annotált korpusz magyar nyelvű véleményelemzéshez. *IX. Magyar Számítógépes Nyelvészeti Konferencia*, 2013, pp. 343–345. ⇒189, 193

[12] S. Momtazi, Fine-grained German sentiment analysis on social media, *Proc. LREC'12 Eighth International Conference on Language Resources and Evaluation*, Istanbul, Turkey, May 21–27, 2012, pp. 1215–1220. ⇒188

[13] A. Reyes, Paolo Rosso, D. Buscaldi, From humor recognition to irony detection: The figurative language of social media. *Data & Knowledge Eng.* **74** (2013) 1–12. ⇒188

[14] K. Roberts, M. A. Roach, J. Johnson, J. Guthrie, S. M. Harabagiu, Empatweet: Annotating and detecting emotions on Twitter. *Proc. LREC'12 Eighth International Conference on Language Resources and Evaluation*, Istanbul, Turkey, May 21–27, 2012,pp. 3806–3813. ⇒189

[15] R. Socher, A. Perelygin, J. Y. Wu, J. Chuang, Ch. D. Manning, A. Y. Ng; Ch. Potts, Recursive deep models for semantic compositionality over a sentiment treebank *Proc. EMNLP 2014: Conf. on Empirical Methods in Natural Language Processing*, October 25–29, 2014, Doha, Qatar, vol. 1631, p. 1642. ⇒191, 193

[16] M. K. Szabó, V. Vincze, Egy magyar nyelvű szentimentkorpusz l etrehozásának tapasztalatai, *XI. Magyar Számítógépes Nyelvészeti Konferencia (MSZNY 2015)*, Szeged, Jan. 14–15, 2015. Univ. Szeged, pp. 219–226 ⇒189

[17] J. Turian, *Using AlchemyAPI for Enterprise-Grade Text Analysis.* Technical Report, AlchemyAPI, August 2013. ⇒193, 195

[18] J. Zsibrita, V. Vincze, R. Farkas, Magyarlanc: A Toolkit for Morphological and Dependency Parsing of Hungarian, *Proc. RANLP 2013*, September 7–17, Hissar, Bulgaria, pp. 763–771. ⇒190

Received: November 5, 2015 • Revised: December 30, 2015