Automatic Ranking of Swear Words using Word Embeddings and Pseudo-Relevance Feedback

Luis Fernando D'Haro, Rafael E. Banchs
Human Language Technologies, A*STAR, Singapore
E-mail: {luisdhe, rembanchs}@i2r.a-star.edu.sg, Tel: +65-6408 2000

***Abstract*— This paper describes a method for automatically ranking a dictionary of swear words based on their level of rudeness. The final ranking is generated by combining two baseline rankings: 1) using the normalized accumulated cosine similarity between the word embeddings of the swear word and the n-best list of closest neighborhoods, and 2) using a pseudo-relevance feedback and bootstrapping algorithm. The proposed methods are trained using dialogues extracted from movies scripts and evaluated against a list of swear words ranked manually in 5 categories by four different annotators. The Spearman correlation coefficient between the rankings generated by the proposed system and a consolidated gold standard reaches a similar value to the ones obtained among the different human annotators, proving that the proposed method is a good alternative to the manual process.**

I. INTRODUCTION

In recent years, the exponential grow in popularity of social media and networking sites like Facebook, Twitter, Youtube, Google+, etc. and, especially its popularity among child and youth people, has increased the effort of website administrators for detecting offensive language in order to block unpleasant, discriminatory or sexual comments that could hurt other users' sensitiveness. A similar problem, but in a different application, is found in the context of chatbot and Q&A engines that are trained on human generated contents. In this scenario, it could happen that some of the dialogues or answers the system uses to learn could contain offensive language that can lately be displayed to the users as part of the machine generated answers. Therefore, it is important to develop automatic methods that could help to detect those contents in training data in order to allow system developers to remove or modify those sentences accordingly.

One of the first things we need to define is the meaning of *swear*. In this paper, we have followed the adapted definition from [1] and [2], i.e. "it is any word or phrase that is likely to cause offense when used in middle class polite conversation". As we can see, this definition can cover a lot of referents, for instance: religion, sexuality, ethnic groups or nationalities, political affiliations, denigration or oppression of groups of people, diseases or undesirable behaviors [2].

Unfortunately, accurately detecting offensive language is a difficult task since even the most basic approach (e.g. based on keyword matching techniques), requires of a list of swear words that could be difficult to create and update. Although there is not a standard available dictionary, sites like Wikipedia[1], noswearing[2], or youswear[3] provide extensive list of words that can be used as starting point. However, the dictionary created from these kind of resources will have two main problems: (1) the words will not be classified according to their level or strength of rudeness, and (2) some of the included words can be considered as non-swear words depending on the context in which they are used due to factors like polysemy. In addition, we can mention other challenges to the classification of offensive language such as the following. (1) The lack of a large annotated corpus to train statistical systems. (2) Although swear words could have a recognized official spelling (e.g. given by the Oxford or Merriam-Webster English Dictionaries), there could be national variations in spelling and pronunciation of similar words. Moreover, written swear words can be bleeped (i.e. replacing some letters by * or - symbols), or by introducing numbers, punctuations, or special symbols that could resemble known letters (e.g. @ instead of the letter a, $ instead of the letter s, 1 for letter l or i, etc.). Additionally, there could be accidental or deliberated misspellings that significantly deteriorate the performance of techniques based on keyword or rule matching even when using well known offensive patterns. (3) The level of offensiveness of a sentence or swear word is very subjective and highly depends on several factors, such as the user's age, gender, background, vocabulary usage, the structure of the sentence, and the user's intention, among others [3]. Finally, (4) the continuous evolution of the language and its usage make it difficult to keep the dictionary updated. For instance, a word that can be considered offensive at some point in time, after sometime can be considered as mild (in [4], it is shown that this is true except for some particular words that haven't change along the time).

[1] https://en.wiktionary.org/wiki/Category:English_swear_words
[2] http://www.noswearing.com/dictionary
[3] http://www.youswear.com/index.asp?language=English

Despite of these problems, several attempts for addressing the problem of offensive language classification have been attempted in the past.

For instance, one of the first commercial products was *Smokey*, which is described in [5]. This system included a parser for syntactic analysis and then a series of semantic-rules applied before a C4.5 decision tree classifier used to categorize texts as offensive or not. In this system, the profanity, insults, polite and praise rules were based on the use of customized dictionaries of swear words.

In [3], a multi-level classifier, boosted by a dictionary of insults and abuse words, is described. Here, the authors mention a dictionary with around 2700 words, phrases and expressions with different grades of rudeness labeled by hand in a 1-to-5 scale based on the potential impact of each entry on the classification of sentences. Unfortunately, this dictionary is not available online, but the amount of entries reflects an exhaustive amount of manual work that ideally should be made automatic.

In [6], an automatic detector of cursing-related offensive content in Twitter is proposed. The algorithm uses an initial seed list of offensive words that is increased automatically using bootstrapping and topic distributions (LDA). The results show that this approach is suitable for detecting new offensive words, even foreign and misspelled swear words. As in our current work, the reason for attempting this kind of approaches is that vulgarity is a type of linguistic style, as pointed by [6], meaning that it is expressed within a sentence with a certain rhythm; therefore, it will be frequent to find more than one swear words together in a sentence (see [7]). This allows for the use of distributional semantic techniques for detecting additional words which carry similar levels of rudeness in similar sentences or contexts.

Finally, in [8] an interesting classifier based on Lexical Syntactic Features (LSF) is proposed. This classifier is able to detect not only offensiveness at the sentence level, but also at the user level by using user profiles. The extracted features consider, among others: style, sentence structure, content, use of intensifiers, as well as the use of dictionaries with different levels of rudeness.

From the point of view of the scope of this paper, we are not pursuing the creation of a classifier, as it was done in the above systems, but proposing an automatic solution to the problem of ranking the entries in a dictionary of swear words according to their degree of offensiveness and rudeness. Our motivation is to use this resource to help cleaning up training corpora in the context of example-based chat engines [9]. The rest of the paper is organized as follows: in section II we describe the corpus of movie scripts used to extract, rank and classify the offensive words, as well as the seed dictionary of swear words we collected from different websites; then in section III, we describe the two proposed ranking approaches used to estimate the level of rudeness, which are based on (1) word embeddings and (2) Rocchio's pseudo-relevance feedback algorithm. Finally, in section IV, we present our main conclusions and some future work.

II. Database Description

For this work, we have used a refined version of the Movie-DiC corpus described in [10]. This corpus was extracted from movie scripts freely available from The Internet Movie Script Data Collection[4]. The creation process started by crawling this website and then identifying and extracting the relevant segments from the scripts. Three basic types of information elements were extracted from the scripts: speakers, utterances and context. The utterance and speaker information elements contain what is said at each dialogue turn and the corresponding movie character who says it, respectively. Context information elements, on the other hand, contain all additional information/texts appearing in the scripts, which are typically of narrative nature and explain what is happening in the scene. For detecting dialogue boundaries, some heuristics were implemented by taking into account the size and number of context elements between speaker turns.

In order to prepare the data for our experiments on offensive language categorization, we applied additional post-processing steps following recommendations from [6], specifically those mentioned in steps 7 and 8 (simplification of intentional repetitions of letters), 9 (stop-words removal) and 10 (removal of words containing mixtures of letters with numbers and/or symbols). We also performed several normalizations such as contraction expansions (e.g. *I'll* for *I will*, *don't* for *do not*, *ya'* for *you all*), omission of letters (e.g. *tryin'* for *trying*), resolving misspellings (by using the Wikipedia dictionary of common misspellings[5]), as well as several regular expressions rules for normalizing punctuation. Finally, we used NLTK TreeBank-WordTokenizer to obtain all words in a sentence [11].

Statistics	*Value*
Total number of movies	615
Total number of dialogues	65,215
Total number of speaker turns	512,582
Average amount of dialogues per movie	106.1
Average amount of turns per movie	824.8
Average amount of turns per dialogue	7.86
Average length of words in a turn	13.7
Total number of words	7,019,963
Vocabulary size	79,525
Number of different swear words found	1723
Average number of swear words found together in the same sentence	1.44 ± 0.92
Average number of swear words per sentence	0.19
Maximum number of swear words found in a sentence	27

TABLE 1. MAIN STATISTICS OF THE COLLECTED MOVIE DIALOGUE DATASET

[4] http://www.imsdb.com/

[5] https://en.wikipedia.org/wiki/Wikipedia:Lists_of_common_misspellings

Another important resource required for our proposed methods is a seed dictionary of swear words. For this, we collected all the words in the sites mentioned in the introduction (i.e. Wikipedia, noswearing and youswear), plus some few other sites, obtaining a total of 5845 words. In the generated dictionary, most of the entries are single words (i.e. 4587) while the rest were phrases. On the other hand, we found that the total number of swear words in this dictionary that also occur in the movie script data collection was 1723. From this list of words, we extracted the 75 highest frequent swear words with different levels of rudeness. The selection included 52 of the words mentioned in Table 6 from [2], and some others that were more frequent in our considered dataset. Then, we asked 4 researchers from our lab to classify those words in the same 5 different categories used in [2], i.e: very strong, strong, moderate, mild, and very mild.

III. PROPOSED METHODOLOGIES

In this section, we explain the two methodologies used to rank the list of the 75 selected swear words. The first methodology is based on the use of Rocchio's pseudo-relevance feedback [12] and bootstrapping [13]. The second method is based on the use of word embeddings trained using deep neural networks as proposed in [14].

A. Creation of the ranking using pseudo-relevance feedback and bootstrapping

One of the motivations for using bootstrapping in this method was to follow a similar approach to [6] but by replacing the LDA topic models with Rocchio's Pseudo-Relevance Feedback algorithm. Up to the best of our knowledge, this is the first time these two algorithms are combined for this task. In addition, we also implemented a weighting mechanism that improved the results. The procedure for ranking in the list of words into the five categories mentioned in section II is as follow:

1. We generated a binary word-document matrix[6] for all terms appearing more than twice in all turns in the dataset. The resulting matrix was of size ~ 40K x 500K.
2. Then, we ranked the columns of the matrix, which corresponds to the whole set of turns (~500K), based on the number of occurrences of swear words that appear in the seed dictionary of swear words we described in section II.
3. Next, we created two vectors. The first one corresponds to the weighted sum of the top 10K binary vectors in the ranked list of turns. For this, we applied a decreasing continuous weight (w_m in equation 1). The second vector corresponds to the weighted sum of the last 300K vectors in the ranking. In this case, we applied an increasing continuous weight (w'_m in equation 1). In Equation 1, M stands for the total number turns in the movie scripts.

$$w_m = \frac{M-m}{M}, \quad w'_m = \frac{m}{M} \tag{1}$$

4. Finally, we subtracted the two previous vectors and retained the N words with the highest weights. These N words were then used to update the original seed dictionary of swear words after each iteration. Eq. (2) shows the formula used for getting the final vector containing ranking of words; here, V stands for the total number of words in the vocabulary, K the total number of turns from the top of the turn ranking, and L the total number of turns from the bottom; w_k and w'_l are the corresponding weighting factors, and h is the original binary term-document matrix and h' is the same matrix with the order of columns inverted.[7]

$$H_v^{final} = \sum_{k=0}^{K}\sum_{v=0}^{V} w_k * h_{kv} - \sum_{l=0}^{L}\sum_{v=0}^{V} w'_l * h'_{lv} \tag{2}$$

The four-step process described above is repeated several times, and after each iteration the new updated list of swear words is used.

B. Creation of the ranking using word embeddings

In [2] it is mentioned that the strength of swearing varies depending on the percentage of people that would take offense at a particular usage. Measuring directly such a percentage will require a huge corpus of annotated sentences that is not publicly available. Therefore, in order to alleviate this problem, we propose an automatic procedure based on the use of word embeddings, which are created by using neural networks and trained from texts extracted from different domains. In [14] it is shown that these word embeddings are able to capture lexical-semantic information thanks to the incorporation of contextual information during the training stage. These embeddings are typically of lower dimensions than the original data space, so well known metrics such as the cosine or Euclidean distance can be used to measure the level of relationship between terms. Clearly, these embeddings present interesting properties that have been exploited in different tasks like machine translation [15], Q&A [16], information retrieval [17], or selection of distractor candidates for automatic evaluation of students [18]. On the other hand, it is important to mention that it has been found in many studies that the quality of these embeddings highly depends on (a) the amount of training data, (b) the tuning of several parameters (e.g. size of the embedding, window size, number of negative examples, minimum number of occurrences, number of classes, etc.), and (c) the number of times and contexts in which each word appears.

Taking into account the advantages of the embeddings, and that through them it would be possible to detect similar swear words occurring in similar contexts, we hypothesize

[6] In our experiments we also evaluated a TF-IDF matrix, but the results were slightly worst. Therefore we finally used the binary matrix.

[7] This formula is expressed in a general form so it can be used with a TF-IDF matrix or with our binary matrix. In addition K and L can be set to M to account for a continuous weighting scheme.

that it would be possible to measure the level of rudeness of a word based on the number and distances to similar words in the embeddings that are also swear words. Here, we propose to use a normalized accumulated distance ($\overline{w_r}$) between all words in the vocabulary and the list of swear words found among the K-Nearest Neighbors. For identifying the nearest neighbors we use the cosine distance between the vectors in the embedding as shown in equation 3. Based on this accumulated distance, we generate the final ranked list of words.

$$w_r = \sum_{k=0}^{K} cos(w, w_k), \qquad being\ \overline{w_r} = \frac{1}{max(w_r)} \qquad (3)$$

In this work, we used two different kinds of word embeddings. (1) A publicly available word embedding trained on part of Google News dataset (containing ~100 billion words) of text data extracted from news[8]. This model contains 300-dimensional vectors for around 3 million words and phrases. (2) Word embeddings trained using the full text of dialogue turns in the movie script dataset under consideration. Since the amount of data in the last case is significantly lower than in the first case, the dimension of these embeddings was set to 150 to avoid that some of the embedding dimensions could not be well trained due to data scarceness. Finally, in both cases the embedding were obtained using the skip-gram method. The motivation for using both embeddings is to account for domain independent and domain dependent information, which considers general and specific usage frequency of each word, minimizing some of the problems with the embeddings mentioned before. In this way, the general embedding from Google news can provide coverage, while the movie's embedding will provide domain specificity. Unfortunately, for our proposed task, it is not possible to avoid the problem that the Google's embedding was extracted from news, where many swear words (especially stronger ones) might not occur at all or only appear few times. Therefore, the resulting nearest neighbors will be unrelated or common words. On the other hand, our domain specific embeddings will face a data scarcity problem, since more training data is required in order to properly model swear word contexts (i.e. take into account that in our data only around 6.5% of the sentences were found to have at least one swear word).

The procedure to combine both embedding is as follow:

1. For each word in the vocabulary, we extracted the K-Nearest Neighbors and their cosine distance using first the embedding from Google and then from our database.
2. Then, for each word we calculated the cumulative cosine distance considering only the distance of those neighbors that were swear words according to our full dictionary. Since the combination of both kind of embedding is done at the cosine distances, the differences in the sizes of the embedding do not pose a problem.
3. Then, only the known swear words are extracted and a new rank is created based their normalized cosine distance.
4. Optionally, it is possible to extract new unknown swear words by considering as candidates words obtaining high cumulative values, or by considering words that appear frequently as neighbors of well known swear words.

[8] Available at https://code.google.com/p/word2vec/

IV. Experiments and Results

In order to test the quality of the ranks generated by both methods, we decided to compare them with a list of 75 swear words ranked in 5 levels of rudeness. We restricted the analysis to the 75 swear words that occurred at least more than 30 times in our movies' dataset. The generated list contained most of the terms appearing in the list provided in [2], with the exception of those terms not occurring in our dataset or occurring with a very low frequency. Then, we asked four research colleagues to rank them using a list of 5 categories, ranging from 1 (very mild) to 5 (very strong). We requested them to do the classification based on their opinion about how rude each word would be in case that word would appear in a conversation with the chat agent.

After this, we averaged their ranks to generate a gold standard reference. The reason for doing this was to minimize the effect of the different backgrounds of the annotators that played an important role in the process of ranking the words, something that became evident when checking the differences between their proposed ranks. In a future work, we plan to provide them each word in the context of a reference sentence that could be useful to establish context and reduce background effects.

In order to test the correspondences between the rankings generated by the annotators, we used the Spearman correlation coefficient. More specifically, we randomly generated 10 thousand 5-sample rankings by randomly selecting one word from each category in the lists of categorized words produced by a *reference* annotator. Then, we created similar 5-sample test rankings for each one of the others annotators, by using the same words selected from the *reference* annotator. Then, we calculated the Spearman correlation for each of the experiments generated and averaged all the results. Table 2 shows that the resulting average Spearman coefficient between annotators is 0.614. This value is considered as our oracle result for the proposed algorithms. On the other hand, we can also see that the average Spearman correlation between the annotators and the produced gold standard is 0.899, which shows that the generated reference provides a good compromise between the different rankings proposed by the annotators.

Then, we used the same procedure for estimating the Spearman correlations between the gold standard and the

rankings generated by the two proposed methods: pseudo-relevance algorithm with bootstrapping and the word embeddings. In this case, the position in the 5-sample rankings depended on the relative positions of the selected words in the generated ranks using any of the methods. In Table 2 we show a summary of the results obtained with the different techniques. Below we provide some insights about the obtained experiments.

Description	Value
Average Spearman correlation between annotators	0.614
Average Spearman correlation between the annotators and the gold standard	0.899
Spearman correlation between Rocchio algorithm (after 3 iterations) and the gold standard	0.268
Spearman correlation between Word Embeddings and the gold standard	0.281
Spearman correlation between the interpolated system (alpha = 0.5) and the gold standard	0.551

Table 2. Summary of correlation results obtained with the different techniques

Regarding the Rocchio algorithm, we found that 3 iterations of the bootstrapping algorithm provided the best result (i.e. 0.268), after the third iteration the value remained the same. This could mean that turn ranks do not change with the updated dictionary after the third iteration. One possible solution is that the number of words to weight from the top positions or the bottom positions needs to be updated as the algorithm iterates. At the same time, it could happen that our strategy of only adding words to the dictionary could require a different updating mechanism as, for instance, to allow for removing words from the dictionary too. On the other hand, we detected that our original proposal for ranking the turns based only on the number of swear words occurring in the sentence and normalizing this value by the number of words in the sentence, produced the undesired effect that most of the top ranked sentences were too short (i.e. containing only one or some few words). Therefore, we added a penalization term that favors longer phrases up to a certain limit to avoid selecting too long sentences. In our case, our best results were found using the penalization factor in equation 4. With this factor, we favored sentences containing between 10 and 30 words.

$$\min(1.3, \log_{10}(\text{sentence\_lenght})). \quad (4)$$

Finally, we decided to evaluate the linear interpolation between the rankings created by using both techniques. In this case, the combination was done by generating the 5-sample test rankings for each technique and combining them using a linear interpolation between the rank positions generated by each method. Then, based on the obtained interpolation values we generated a new 5 categories ranking with independence of the differences between the actual floating values generated by the interpolation. In our experiments, we found that the best interpolation factor was 0.5; however, we also found that a higher correlation could be obtained if we applied first a floor operation to the interpolated value before creating the ranking. The reason for this, although more experiments need to be done to confirm our guess, is that this procedure allows several words to share the same rank category minimizing the effects that small differences in the floating values produce completely different categories.. Besides, we also observed a similar situation during the calculation of the correlations between different human annotators; where it happened that the reference annotator considered that the five randomly selected words in the experiment belonged to different categories, but the test annotator put several of them in the same category. Therefore, by introducing this discretization mechanism we allow that closed words in the combined rank can remain closer also for the correlation calculation. As future work, we plan to consider additional combination formulas and procedures.

V. CONCLUSIONS

In this paper, we have described a methodology for automatically categorizing a dictionary of swear words in 5 discrete categories that measure the level of rudeness of swear words. The proposed methodology first ranks the words using two different approaches: pseudo-relevance feedback with bootstrapping and word embeddings. The first approach relies on the creation of a ranking of sentences or turns from which the algorithm extracts swear words by exploiting the difference of word frequency distributions between the top and the bottom ranked sentences. The second approach is based on word embeddings trained on different domain texts. The process in this case was to rank the swear words based on the cumulative cosine distance between the given swear word and other swear words found in a list of its K-Nearest Neighbors. The motivation for this approach is that stronger swear words will have more and closer swear words as neighbors.

To evaluate the rankings produced by the proposed techniques, we calculated Spearman correlation coefficients between the generated rankings and a gold standard reference on multiple subsets of selected swear words. The gold standard reference was created from averaging rankings generated by human annotators. These experiments proved that an interpolated system combining the output of both approaches provides a Spearman coefficient that is close to the one found between human annotators (0.551 vs 0.614). Therefore, the proposed combined system can be seen as a good alternative to the manual process of creating the ranking.

As future work, we propose to improve the rankings, in terms of its correlation with human annotations, by taking into account lexical-syntactic information as the ones proposed in [3], and combining them with emotional information (e.g. polarity and subjectivity as described in

[19], [20] and [21]). On the other hand, since our proposed system will be included as a pre-processing stage for cleaning chat engine training corpus, we will also work on a mechanism for creating replacement dictionaries where suitable milder synonyms will be proposed for more strong offensive words. In this way, rude words can be either replaced from the current training material or, at runtime, used to detect user's rudeness and then the chatbot will redirect the dialogue by using milder words.

ACKNOWLEDGEMENTS

We want to thank the 4 annotators for their contribution on annotating the list of words, and for their useful comments about how to improve the paper.

REFERENCES

[1] McEnery, Tony. “Swearing in English: Bad language, purity and power from 1586 to the present”. Routledge, 2004.

[2] Mike Thelwall. "Fk yea I swear: cursing and gender in MySpace." *Corpora* 3.1 (2008): 83-107.

[3] Amir H. Razavi, Diana Inkpen, Sasha Uritsky, and Stan Matwin. 2010. “Offensive language detection using multi-level classification”. In *Proceedings of the 23rd Canadian conference on Advances in Artificial Intelligence* (AI'10), Atefeh Farzindar and Vlado Kešelj (Eds.). Springer-Verlag, Berlin, Heidelberg, 16-27. DOI=10.1007/978-3-642-13059-5_5.

[4] Jay, Timothy. "The utility and ubiquity of taboo words." *Perspectives on Psychological Science 4.2* (2009): 153-161.

[5] Spertus, Ellen. "Smokey: Automatic recognition of hostile messages." *AAAI/IAAI*. 1997.

[6] Guang Xiang, Bin Fan, Ling Wang, Jason Hong, and Carolyn Rose. 2012. “Detecting offensive tweets via topical feature discovery over a large scale twitter corpus”. In *Proceedings of the 21st ACM international conference on Information and knowledge management* (CIKM '12). ACM, New York, NY, USA, 1980-1984. DOI=10.1145/2396761.2398556.

[7] Martin, James R., and Peter R. White. “The language of evaluation: Appraisal in English”. Palgrave Macmillan, 2003.

[8] Ying Chen, Yilu Zhou, Sencun Zhu, and Heng Xu. 2012. “Detecting Offensive Language in Social Media to Protect Adolescent Online Safety”. In *Proceedings of the 2012 ASE/IEEE International Conference on Social Computing and 2012 ASE/IEEE International Conference on Privacy, Security, Risk and Trust* (SOCIALCOM-PASSAT '12). IEEE Computer Society, Washington, DC, USA, 71-80. DOI=10.1109/SocialCom-PASSAT.2012.55.

[9] Banchs, Rafael E., and Haizhou Li. "IRIS: a chat-oriented dialogue system based on the vector space model." *Proceedings of the ACL 2012 System Demonstrations*. Association for Computational Linguistics, 2012.

[10] Banchs, Rafael E. "Movie-DiC: a movie dialogue corpus for research and development." *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*. Association for Computational Linguistics, 2012.

[11] Bird, Steven. "NLTK: the natural language toolkit." *Proceedings of the COLING/ACL on Interactive presentation sessions*. Association for Computational Linguistics, 2006.

[12] Rocchio, Joseph John. "Relevance feedback in information retrieval." in The SMART Retrieval System - Experiments in Automatic Document Processing. Prentice Hall. Edited by Salton, Gerard, pps. 313-323. 1971.

[13] Efron, Bradley. "Bootstrap methods: another look at the jackknife." The annals of Statistics (1979): 1-26.

[14] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. “Efficient Estimation of Word Representations in Vector Space”. *In Proceedings of Workshop at ICLR, 2013.*

[15] Mikolov, Tomas, Quoc V. Le, and Ilya Sutskever. "Exploiting similarities among languages for machine translation." *arXiv preprint arXiv:1309.4168* (2013).

[16] Belinkov, Yonatan, et al. "VectorSLU: A Continuous Word Vector Approach to Answer Selection in Community Question Answering Systems." Proceedings of the 9th International Workshop on Semantic Evaluation, SemEval. Vol. 15. 2015.

[17] Liu, Xiaodong, et al. "Representation Learning Using Multi-Task Deep Neural Networks for Semantic Classification and Information Retrieval." Proc. NAACL, May 2015.

[18] Kumar, Girish, Rafael E. Banchs, and Luis F. D’Haro. "RevUP: Automatic Gap-Fill Question Generation from Educational Texts." The Tenth Workshop on Innovative Use of NLP for Building Educational Applications, NAACL HLT (2015): 154-161.

[19] Montoyo, Andrés, Patricio Martínez-Barco, and Alexandra Balahur. "Subjectivity and sentiment analysis: An overview of the current state of the area and envisaged developments." Decision Support Systems 53.4 (2012): 675-679.

[20] Kumar Ravi, Vadlamani Ravi, A survey on opinion mining and sentiment analysis: Tasks, approaches and applications, Knowledge-Based Systems, Available online 29 June 2015, ISSN 0950-7051, http://dx.doi.org/10.1016/j.knosys.2015.06.015.

[21] Shirani-Mehr, Houshmand. "Applications of Deep Learning to Sentiment Analysis of Movie Reviews." Available at http://cs224d.stanford.edu/reports.html [July 2015]