http://dx.doi.org/10.14236/ewic/HCI2017.100

Computational methods for text mining user posts on a popular gaming forum for identifying user experience issues

Ken McGarry
University of Sunderland
Sunderland, UK
ken.mcgarry@sunderland.ac.uk

Sharon McDonald
University of Sunderland
Sunderland, UK
sharon.mcdonald@sunderland.ac.uk

The advent of the social web such as twitter, facebook and the numerous social forums have provided a rich source of data representing human beliefs, social interactions and opinions that can be analysed. In this paper we show how extracting user sentiment by text mining posts from popular gaming forums can be used to identify user experience problems and issues that can adversely effect the enjoyment and gaming experience for the customers. The users posts are downloaded, preprocessed and parsed, we label the posts as negative, positive or neutral in terms of sentiment. We then identify key areas for game play improvement based on the frequency counts of keywords and key phrases used by the fora members. Furthermore, computational models based on complex network theory can rank the issues and provide knowledge about the relationships between them.

Text mining. Usability. Games industry. Graph theory.

1. INTRODUCTION

The computer gaming industry is a highly profitable business and in fact sales of computer games exceeds the revenues of the movie making industry, one estimate placed the gaming industry at $86 billion with Hollywood at $36 billion (UKI, 2017). Many popular games have user forums where players can post messages to each other and to the designers of their games. The majority of posts are requests to fellow players for help in solving difficult puzzles at various levels of gameplay or requests to the software developers for particular features they desire or features they find irksome.

Over the past 10-15 years text mining has seen massive expansion both in practical applications and research theory (Hearst, 1999). Several, quite diverse areas such as mining student feedback in educational domains (Romero and Ventura, 2010); Kumar and Jai, 2015), automatically creating ontologies from text (Missikoff et al. 2003), mining student requests for help on programming forums, mining customer emails/feedback for satisfaction or pinpointing problems with products have all benefited from this automated approach. There are many reasons for this explosive growth but the main factor is that the majority of human knowledge and experience is in the form of the written word and not structured databases (Bose, 2017). This presents some problems as the information contained in natural language statements is difficult to map to the rectangular/tidy data expected by machine learning and statistical algorithms (Wickham, 2011).

In recent years, usability and the delivery of an appropriate user experience has become a key determinant of success for digital products and services; particularly within the computer games industry. Typically, usability and the user experience are evaluated through two broad approaches to evaluation: analytical methods and empirical methods. Analytical approaches to evaluation, do not involve users and include popular techniques such as heuristic evaluation (Nielsen, 1993).

These techniques require that experts use their knowledge of usability principles to inspect the product in order to identify likely usability problems. However, while these methods are fast and relatively inexpensive to run, they have been widely criticised because of their lack of predictive power: many issues identified by experts never reveal themselves in actual use. Empirical methods involve the collection of data from real users, either in laboratory based usability tests where users are asked to complete representative tasks and problems in user are observed and field studies where researchers observe interactions with technologies in their context of use. These methods are considered to be more robust, however they are

© McGarry et al. Published by BCS Learning and Development Ltd.

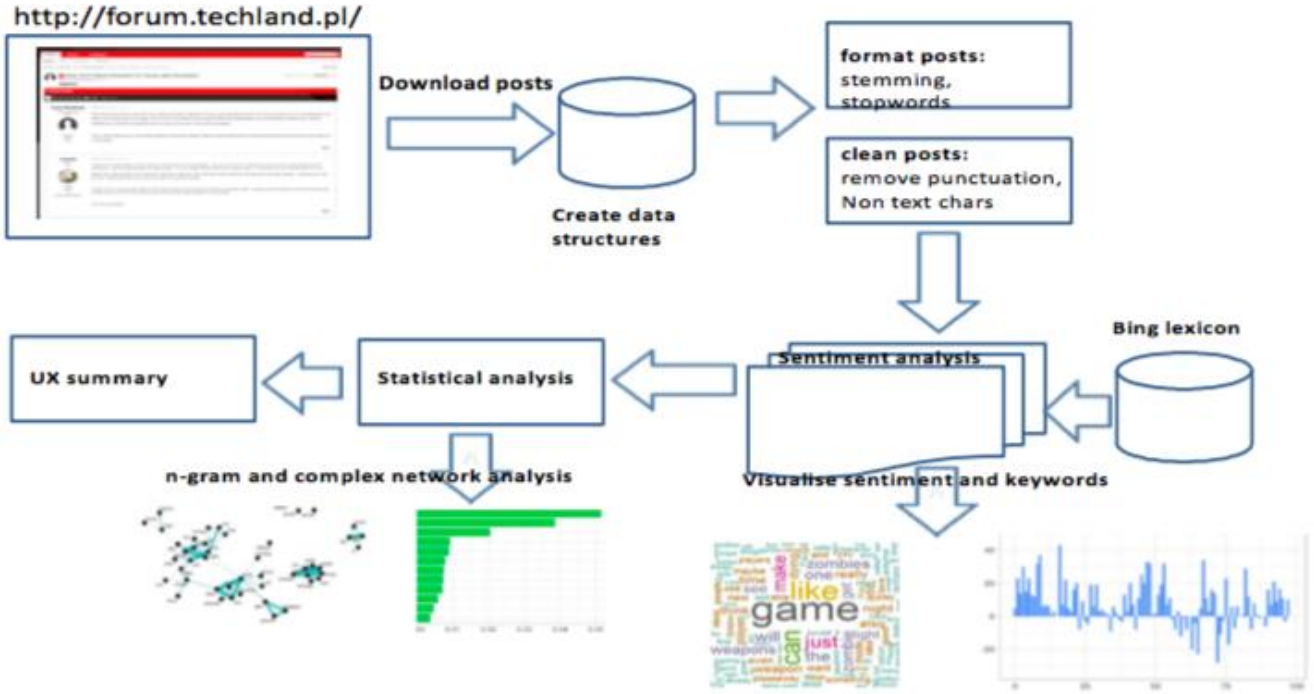

***Figure 1**: System overview: data download, preprocessing and model building*

more expensive and take considerably longer than inspection approaches.

Our overall system operation is highlighted by figure 1. The process is initiated by downloading the users posts which is the basic unit of data. This is a user-submitted text message enclosed into a block containing the user's details and the date and time it was submitted. Posts are usually short but in fact can vary in size as users communicate to previous posters, often providing detailed information to assist other members. The posts can usually edited or deleted my members. The posts have a certain structure called threads where the original poster (OP) creates a topic title. This first post creates the thread and subsequent replies to this post follow in logical order. The usual forum etiquette requires that subsequent posts should be on-topic and not to deviate to other subjects or issues, however this is often disregarded. Other useful information include the total count of each user's posts count (Nahm and Mooney, 2002).

The remainder of this paper is structured as follows; section two describes our methods, indicating the types of data used and how we download and preprocessed it, along with the computational statistics techniques used to model this data, section three presents the results, section four provides the discussion and finally section five summarizes the conclusions and future work.

2. METHODS

We implemented the system using the R language with the RStudio programming environment, on an Intel Xenon 64-bit CPU, using dual processors (3.2GHz) and 128 GB of RAM. R is primarily a statistical data analysis package but is gaining popularity for various scientific programming applications and is very extendable, using packages written by other researchers (R Core Team, 2015). It is freely available from CRAN and is supported by a large community of researchers. Since it is an interpreted language, R can be quite slow compared with a compiled language such as C++ etc, however it is possible to speed up R by recoding mission critical functions in C++, the application described in this paper did not require any speedups. The R code and the datasets are freely available on GitHub: https://github.com/kenmcgarry/TextMiner

Referring to the system diagram presented in figure 1, we have used the following R packages, the TM package by Feinerer which contains a comprehensive set of functions for creating a corpus (Feinerer et al, 2008). The RVEST package enables web page scraping of HTML documents creating data structures suitable for parsing (https://github.com/hadley/rvest). The posts are downloaded using special HTML functions from the RVEST package that remove the embedded structural information. The main URL with the OP topic is cut and pasted from a browser into our R code, but subsequent pages (each containing 25 posts) are automatically downloaded.

In order to successfully extract the users posts we need to know where in the HTML code the names for each CSS (cascading style sheet) node in the webpage. This unfortunately, has to be a manual process and we used http://selectorgadget.com/ to identify the post main body from the myriad of nodes

in the web page. The names are created by the web designers and obviously will be different from website to website. Many other nodes identify useful information such as the user names, the dates, the titles, the number of posts each user has submitted. Other nodes provide the usual HTML formating for positioning of text and graphics objects and are not useful to us. Once identified from the rest of the nodes, the text nodes of the posts are preprocessed by removing whitespace, newlines, tabs, punctuation, and any special embedded characters.

The next process is to remove stopwords, and to stem the document. This involves removing non informative words such as "if" , "and", "then" etc. Stemming simply replaces similar words with their common root e.g. "walked", "walking" and "walker" become "walk". This simplifies the number of tokens required for text mining without losing any meaning.

Sentiment analysis is conducted by the sentimentR package, we group the posts according to their time stamp and if there are any trends or patterns that occur over time these should be identified. It is highly likely that when a game is first introduced it may have either bugs or features that the users are unfamiliar with or have difficulties with and thus posts may have an overall negative sentiment. This process is based on single words without any regard for context or negation i.e. "I am not happy" would be identified as positive statement rather than negative. We use the Bing lexicon for a list of positive and negative sentiment words.

Algorithm 1 User keyword search/identification algorithm

```
1:  procedure SEARCHKEYWORDS(keywordlist, Corpus)   . keywords from wordcloud
2:      do initialize
3:      knum ← get number of keywords in list
4:      i = 1
5:      i_w = 0                                     . Words co-occurring with keyword i
6:      CUTOFF = 50                                 . Value determined heuristically
7:      end initialize
8:
9:      for i ≤ knum do
10:         i_w ← Corpus                            . all co-occurring words for i_w in Corpus
11:         wcount[i] ← get numberof words
12:         if wcount[i] > CUTOFF then
13:             Cw[i] ← i_w                         . keep these words in new structure
14:         end if
15:         for j ≤ knum do
16:             LCs[j] ← CalcNetworkStatistics(Cw[j])   . calc statistics of C w
17:         end for
18:     end for
19:     GCs ← CalcNetworkStatistics(Cw)             . call function for global statistics of C w
20:     return Cw, Cs, GCs                          . Return top keywords with local and global statistics
21: end procedure
```

We then take a more detailed approach whereby sentence level manipulation is implemented so negation and context may be better understood. The great challenge in text mining user posts from a gaming forum is that many of the words that would be normally associated as negative by the lexicon are in fact either neutral or positive because of the context of a violent shooting game.

We selected a set of keywords deemed important enough to uncover user issues and problems based on their occurrences highlighted by the wordcloud. Algorithm 1, operates by searching for our list of keywords for their occurrences in the corpus of posts. We chose a cutoff parameter based on heuristic experimentation and 50 was deemed to be a useful number. Complex network statistics were calculated for each keyword individually and then globally with all networks of keywords joined together.

3. RESULTS

The wordcloud presented in figure 2 highlights the relative frequency of the keywords, the larger the size of a word indicates it occurs quite often. This type of plot is useful for a quick scan of frequently occurring themes or issues. However, it is simply a "bag of words" method without any context of word order or relationships between them. It is an attractive and highly visual way of representing data but gives little in way of quantitative analysis.

***Figure 2**: Wordcloud for user posts to the Feature Requests topic.*

The only additional means is to organise words in terms of sentiment as per figure 3. Here we can see the positive and negative classifications as classified per the lexicon (Bing) we used.

It should be noted that several lexicons exist, each will provide slightly different classifications of words and different term weightings. We selected the Bing lexicon because of it wide applicability to text mining and did not experiment and compare it with the other lexicons.

An example of the raw text posts appears in table 1, the developer requests topic is the most useful source of information from a usability perspective. Interesting details can be gleaned from such as technology used e.g. PC, Apple or Xbox and issues with RAM memory and other system conflicts. Issues with bugs, glitches and other software behaviors are usually reported here.

***Figure 3**: Wordcloud for All topics organized by sentiment.*

***Table 1:** Example of posts from Developer requests topic*

text
I think that you should work on a house system so that you can just have your own ...
I would like too see new game achievements to have more things to do alone and...
To start, amazing job, I can't get enough of this game! Here are a few things I ...
In the begining of the story when Crane had got bitten he needs antizin to prevent...
Just take a normal mode.I use LG too, playing on PS4, with just normal, no Vivid...
I would like to see better vibrations on the xbox one controller during combat like...
Dear Dying Light developersI would like it if you would repair a small one Coop...
There should be 4 players on each team and each should have to race to an ultimate...
Some things that I think should be added. 1. I think they should add a way that any...
i think hands down this game is by far the best iv'e played with zombies! I think much...

The next stage is to conduct a sentiment analysis of all the downloaded posts, the posts are in the sequential order they appeared over time. Table 2 shows the first 10 posts in the FAQ topic, The index number uniquely identifies each post, with positive and negative counts, the net is simply the overall sentiment after subtracting the +ve from the -ne sentiments.

***Table 2:** Count of positive and negative words with overall net sentiment outcome for first 10 posts in FAQ topic*

positive	negative	net	index	topic
1.00	*1.00*	*0.00*	*1.00*	*FAQ*
9.00	*2.00*	*7.00*	*2.00*	*FAQ*
13.00	*14.00*	*-1.00*	*3.00*	*FAQ*
10.00	*3.00*	*7.00*	*4.00*	*FAQ*
8.00	*1.00*	*7.00*	*5.00*	*FAQ*
9.00	*10.00*	*-1.00*	*6.00*	*FAQ*
4.00	*3.00*	*1.00*	*7.00*	*FAQ*
6.00	*3.00*	*3.00*	*8.00*	*FAQ*
15.00	*10.00*	*5.00*	*9.00*	*FAQ*
19.00	*10.00*	*9.00*	*10.00*	*FAQ*

In table 3, we have displayed the top 15 words, their sentiment class and the number of times they appear in all posts. This is the data we use for the complex networks and statistical calculations when creating word pairs and word linkages.

***Table 3**: Count of positive and negative words with individual*

	word	sentiment	n
1	love	positive	495
2	safe	positive	411
3	dead	negative	363
4	kill	negative	355
5	hard	negative	312
6	fun	positive	289
7	cool	positive	281
8	damage	negative	274
9	infected	negative	269
10	awesome	positive	241
11	skill	positive	213
12	nice	positive	209
13	easy	positive	206
14	survival	positive	193
15	survivor	positive	172

In order to assess the likelihood of the sentiment analysis misinterpreting words because of negation we ran an analysis searching for the number occurrences of "not" and listing the words it precedes. In figure 4 we can see that like and good have the highest scores at 42 and 18 respectively. Taking the not into account will make our sentiment more negative and should really mean not like and not good.

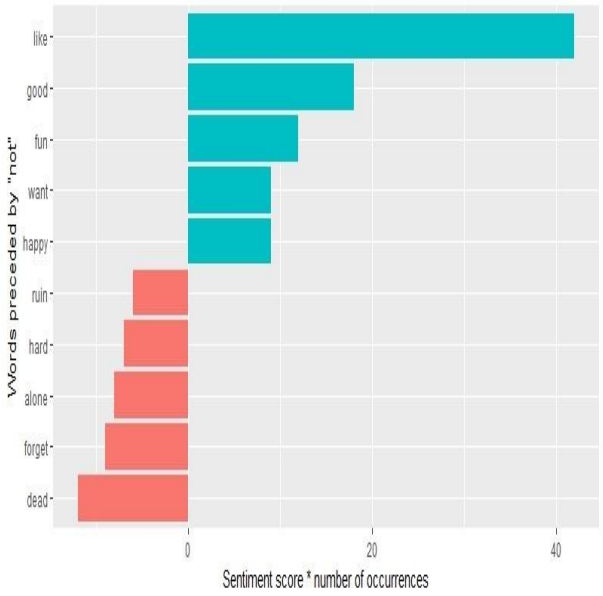

Figure 4: Word sentiment misclassification based on negation by not

Keeping track of the potential for bias through negation, we performed the sentiment analysis as shown in figure 5 for the four main topics. Each column in figure 5 represents approximately 10 posts taken in consecutive order as they were posted by the forum members. It will be noted that Feature requests has accumulated far more posts than the other three topics. This is to be expected, as any issues are reported here.

We find that the Feature requests topic is consistently negative in terms of its sentiment. The Developer Tools topic is generally negative as this contains posts from those trying to modify the game based on their own programming skills user the

software development kit. This is a complex and generally frustrating endeavour, and from reading

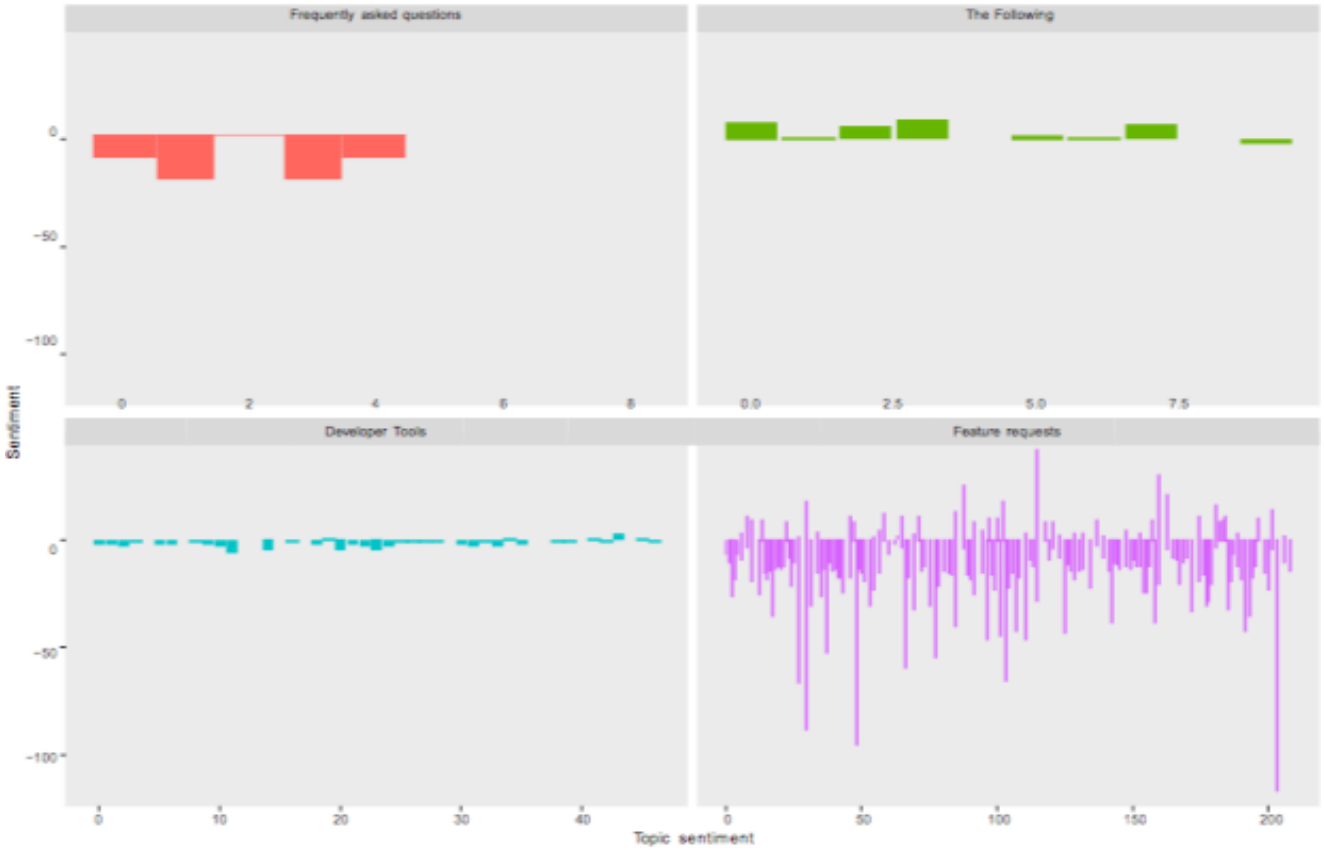

***Figure 5**: Sentiment analysis for four main topic*

the posts majority of gamers find the task difficult and their efforts do not succeed. The FAQ topics starts negative but builds up to a small overall positive sentiment. The Following topic is very well received as this was the second game in the series with the bugs, glitches and annoying (game spoiling) features were more or less solved.

Using Algorithm 1 we are able to assess the impact of the keywords selected as important for usability analysis based on hubness and centrality measures. The initial set of 18 keywords was used to create complex networks of co-occurring keywords, but only if the keyword appeared more than 25 times, else it would be discarded. This produced a list of 10 usable keywords and their co- words that would be investigated further. The keywords are shown in table 4 along with the number of co-words.

We then built a complex network and preformed the statistical analysis on its structure and connectivity patterns for the top 20 co-words as defined by hubness. However, the overall structure of the network consisted of 226 nodes (words) with 704 connections between them. The modularity was 0.79, this unit-less measurement exists between 0 and 1. Closer to unity suggests there is structure between the words rather than a random connection pattern. The avepath was 4.76 (average distance of the path between any two words). The closeness, betweenness, hubness and power will vary for each word depending on number of connections.

***Table 4**: Keywords that appeared more than 25 times were retained, these produced between 5-269 co-words. A zero entry indicates the keyword was discarded*

Keyword	Number of co-words
inconvenience	*0*
problem	*0*
confusion	*0*
complicated	*0*
issue	*5*
obstacle	*0*
glitch	*0*
bug	*0*
annoying	*14*
stupid	*8*
unfair	*0*
difficult	*35*
hard	*269*
bad	*80*
issues	*8*
hate	*15*
wrong	*18*
cheat	*3*

The use of the power measure (Bonacich) attempts to define cliques of individuals that may cooperate as a group and is borrowed from social web mining and is probably more controversial in text mining. The value indicates the effect of one's neighbour's connections on ego's power. Where the attenuation factor is positive (between zero and one), being connected to neighbours with more connections

makes one powerful. On the other hand, if a node (word) has neighbours who do not have many connections to others, those neighbours are likely to be dependent on that node, making it more powerful. Negative values of the power factor (between 0 and -1) compute power based on this idea. Thus a node may not have many connections but may well have the 'right' connections to powerful nodes.

The main word that is central and occurs many times is 'hard' , the betweeness measure for this word is 18,442 well in excess of any other word. The word 'idea' has a value of 7,203 the rest of the words have tiny fractional values. Betweeness is based on the idea of shortest paths between nodes (words), and is a measure of how a given node stands 'between' the other nodes in a network - the higher the value then that node is very central in the network.

4. DISCUSSION

There are limitations to our study, we only used one games forum, our software would need to be more generic to tackle this. The initial activities in post downloading are manual and this would have to be repeated for other forums. It became clear that the normal lexicon based approach of assigning every word in English a score that is either negative or positive is inefficient for this particular application. Words such as "scary", "damage", "enemy" and "kill" are negative scores but are generally expressing satisfaction on the part of the gamers as they are describing what appeals to them in game play. The wordclouds were useful in providing keywords to augment the terms we had devised prior to running the analysis. Words included: Hard- refers to difficulty of last level, impossible for some players to complete the game. Video scenes - spoils pace of game. Time critical missions - complete the mission in 3-5 minutes. Guns - limited in variety. Cheating - in multiplayer mode, access to better weapons.

The overall response to the game (Dying Light) by the customers is very positive, bugs and issues having been sorted by the development team over a short period of time. Purchase and maintenance of the game is through the Internet, so downloads of fixes/patches are easy to obtain. These included: Fun - majority of players enjoy the game. Ideas - suggestions for various improvements. Sequel - keen to have updates on progress on Dying Light 2. However, many nodes or words such as the use of number '4' to represent text speak for 'for' and were uninformative for our purposes.

5. CONCLUSION

Overall, the system was able to detect trends in sentiment over time as the gaming product became more mature and bugs/issues were sorted out. However, the usual method of sentiment analysis does give a rather skewed picture of the usability issues. The Graph theoretic statistics provided a better understanding of the usability issues than mere frequency count of individual words. The bi-grams of co-occurring words can now be linked together for a deeper analysis of the issues. As far as we are aware, our approach is novel for detecting patterns or issues in game usability.

6. ACKNOWLEDGMENT

The authors would like to thank Julia Silge for providing her helpful information on tidytext.

7. REFERENCES

Bose, S. (2017). RSentiment: A Tool to Extract Meaningful Insights from Textual Reviews, pp. 259–268. Springer Singapore.

Feinerer, I., K. Hornik, and D. Meyer (2008). Text mining infrastructure in r. Journal of Statistical Software 25(1), 1–54.

Hearst, M. (1999). Untangling text data mining. In Proceedings of ACL '99: the 37th Annual Meeting of the Association for Computational Linguistics, pp. 126–136.

Kumar, A. and R. Jai (2015). Sentiment analysis and feedback evaluation. In in 2015 IEEE 3rd International Conference on MOOCs, Innovation and Technology in Education (MITE), pp. 433–436.

Missikoff, M., P. Velardi, and P. Fabriani (2003). Text mining techniques to automatically enrich a domain ontology. Applied Intelligence 18, 323–340.

Nahm, U. and R. Mooney (2002). Text mining with information extraction. In U. Nahm and R. Mooney. Text Mining with Information Extraction. In Proc. AAAI 2002 Spring Symposium on Mining Answers from Texts and Knowledge Bases.

Nielsen, J. (1993). Usability Engineering. Academic Press, Boston, USA.

R Core Team (2015). R: A Language and Environment for Statistical Computing. Vienna, Austria: R Foundation for Statistical Computing.

Romero, C. and S. Ventura (2010). Educational data mining: a review of the state of the art. IEEE Transactions on Systems, Man and Cybernetics. Part C Appl. Rev. 40(6), 601–618.

UKI. (2017). The games industry in numbers. Association for UK Interactive Entertainment

Wickham, H. (2011). The split-apply-combine strategy for data analysis. Journal of Statistical Software 40(1), 1–29.