A NLP-LEARNING POWERED CUSTOMIZABLE APPROACH TOWARDS AUTO-BLOCKING DISTRACTING WEBSITES

Yulin Zhang[1], Yu Sun[2]

[1]University High School, 4771 Campus Drive. Irvine, CA 92612
[2]California State Polytechnic University, Pomona, CA, 91768,
Irvine, CA 92620

ABSTRACT

Over the past few decades, the problem of distraction and its accompanying side effects has taken its root deeply in all parts of our daily life and extended its ever-increasing influences among young generations [2]. In addition to its alarming prevalence, another characteristic of distraction that raises most concerns is how easily we can get distracted from our tasks at hand while using the electronic devices as a means of solving problems [3]. This paper attempts to address this society-wide problem thoroughly and universally through a technical approach of detecting, analyzing, and blocking the websites intelligently. Our design highlights the applications of machine learning and natural language processing, and is implemented purely in Python, Javascript, and several other web development languages. After retrieving the web content from the target websites through the web scraping process, summarizing the data to a number of short paragraphs via the use of NLP, we were able to perform data analysis on the result and finally block the websites accordingly [4]. With the help of this extension, students and those who wish to improve their concentration in work will be able to put more focus on the tasks at hand and thus boost their work efficiency under any working conditions.

KEYWORDS

NLP-learning, distraction, Auto-block

1. INTRODUCTION

Nowadays, the problem of distraction has become a society-wide problem that appears to be especially prevalent among teenage internet users. For decades, it hindered people's productivity at work and encouraged inattention and daydreaming, rather than focusing and solving the task at hand. In addition, the result of a research project conducted in UCI shows that being able to return to the concentrating state after distraction takes, on average, 23 minutes and 15 seconds, which demonstrates the influential effects distraction had on one's work mode [1]. On the other hand, being able to focus on the task will help students and employees to work more efficiently and systemically. Furthermore, according to an experiment focusing on how different levels of concentration impact the likelihood of being distracted, the data suggests that highly concentrated individuals are, in general, less likely to be distracted by irrelevant affairs in comparison to less concentrated groups.

Several approaches had been invented and widely used to combat this problem, including the practice of mediation, temporarily turning off the device, and using reward-and-punishment mechanism to motivate oneself; However, these techniques often failed to fully address the

David C. Wyld et al. (Eds): MLSC, ITCSS, ACSTY, SOFE, NATP, BDAB - 2023
pp. 117-125, 2023. CS & IT - CSCP 2023 DOI: 10.5121/csit.2023.130209

problem due to various reasons, some of them being the many forms the distraction may take on and the constant danger of not able to Concentration it poses. Such obstacles made distraction a seemingly unsolvable issue, and any successful attempts to address it to be rather rewarding and valuable.

Although there are already a number of existing anti-distraction techniques available, most of them failed to take into account users' potential actions after blocking the websites.

Some of the existing techniques and systems that have been proposed to address the problem of distraction can be loosely grouped into either manual approaches as discussed above, or "focus apps" that aim to help users to better concentrate on the tasks depending on their needs. However, most of these Apps/extensions either assume that users would never regret their choice of blocking a certain website and come back to unblock it during working hours, or provide little functionalities to prevent similar situations from happening. For example, one of the distraction-free extensions I tried allowed me to enter the websites I blocked at any time after an hour. In practice, chances are this is often the case among groups of users. Their implementations are also not intelligent enough to detect and block websites of certain categories that are specific to each user; Instead, they block a wide range of websites disregarding the information they contain, which may cause the inclusion of unnecessary websites that can be of value to users. A second practical problem is that some services block all but a number of tabs the user is currently working on, and many users find it hard to make a choice between having to deal with the constant distractions from their favorite websites and losing the access to any other websites aside from their relevance.

The goal of our extension is to significantly improve individuals' concentration skills and help them to cultivate a better study/working habit that will benefit them for a lifetime [5]. Our method is inspired by a number of existing programs that aim to address similar issues related to distraction; however, while most of these programs do not taking into account users' specific needs and how they may adjust the program to fit these needs, our extension allows users to update the preferences anytime and anywhere to blocking new categories of websites. In addition, different from those paid services, our extension is always free and openly accessible to users of all ages and backgrounds. Although we have made a number of important improvements from the existing methods, our extension still shares some major similarities with these methods, one of them being the same process of scraping down the web content, making relevant analysis with it, and blocking the website based on the result [6]. By using our extension, students who often find themselves struggling with the desire of looking at websites they like during the study sessions will be able to focus on the actual work and solve them more efficiently.

In three application scenarios, we demonstrate the consistency and accuracy of the extension's capability at blocking distracting websites based on the user input [7]. Each of these scenarios involves experimentally input a keyword into the extension and access 3 websites that are relevant to the keyword and should be blocked, as well as 2 websites that are irrelevant to the keyword and should not be blocked. The experiment was conducted at 5 different points in time, and we recorded whether Concentrate correctly blocked the page in 3 tables that correspond to 3 keywords. We then analyze the results and discover that the extension achieves on average an accuracy score of 98% in blocking relevant websites and 100% in letting irrelevant websites pass across 3 tables, indicating that the extension could effectively identify the similarity between the website content and keywords and block the website accordingly.

The rest of the paper is organized as follows: Section 2 gives the details on the challenges that we met during the experiment and designing the sample; Section 3 focuses on the details of our solutions corresponding to the challenges that we mentioned in Section 2; Section 4 presents the

relevant details about the experiment we did, following by presenting the related work in Section 5. Finally, Section 6 gives the conclusion remarks, as well as pointing out the future work of this project.

2. CHALLENGES

In order to build the project, a few challenges have been identified as follows.

2.1. Organizing not only Text Summarization but Subject Summarization

Being able to perform both text and subject summarization is a difficult problem to handle as it is hard to parse over unstructured web content, find an appropriate NLP package, extract only relevant information from large chunks of text and compare it against the possible categories the website may fall into [8]. Completing all these requirements requires a way to efficiently scrape over the website, summarize its content, and make data transmissions between front and back end. Through the use of several well-known libraries, we were able to scrape over the content with the requests module, summarize the web content under the help of the NLTK library, and finally use that page contents to block the website according to the result we received.

2.2. Managing Events through Content Scripts and Background Scripts in the Chrome API is Often Difficult

It is rather challenging to organize and maintain the bidirectional communication and data transmission between the background and content script due to the various restrictions chrome API applies to protect users' privacy. In order for the summarization and blocking process to function well, the content script has to undergo a multi-steps process, including collecting the web content, passing the data to the background script for analysis and summarization, then waiting patiently for the background script to pass back the decision of blocking or allowing the website [9]. During the process, each script may get triggered at unexpected times due to the shift of chrome API's permission, which requires careful examination of the issue and a once-and-for-all solution to allow for successful communications between two scripts. To address the issue, we added listeners to the background script part in order to regulate the transmission and keep relevant events under monitoring.

2.3. Designing a Clean User Interface to Allow the user to Make Changes to their Account info with Little Work

Given the small and confined area of the extension's pop-up, it's extremely difficult to design a clean and simple user interface that also includes comprehensive functionalities. Generally, when facing similar situations, some extension developers would choose to include an additional website with detailed descriptions on how to use the extension, or pay for professional designers to plan out and organize the interface for them; However, since neither of these solutions is applicable to our case due to the lack of resources and funding, we had to take a different approach. To exploit the given room to its fullest potential, we included a friendly welcome message, implemented the core function of taking inputs from users with the minimum amount of space required, and left the rest in either complete white or light green to give a clean visualizing effect.

3. SOLUTION

The implementation of our extension involves the acceptance of user inputs, finding the appropriate websites based on the user's need, maintaining data transmission between frontend and backend, summarizing and analyzing the web content gathered via web scraping, and finally using the result to decide whether or not to block the web pages [10]. To use the extension, the user first needs to go to the website that has the potential of posing a threat to their work efficiency, then enable the extension to allow the content script scraping down the website content. After receiving the content, the content script would pass the data to the background script for further processing. Inside the background script, various NLP and related machine learning techniques were applied to make a precise summarization of the textual data. With the result of summarization, the script would fetch the user's preference from the Firebase database system, compare it against the result, and make the final decision between blocking the website or allowing it to continue to exist, then hand the decision to the content script where it would come into practice. The extension also provides users with the right to adjust and update their preferences of category-based blocking at any moment: In order to make the adjustment, users need to enter a message that indicates their new preference in the extensions pop-up, which will be transferred to the Firebase system to replace the old preference. After the update, the background script will fetch the new preference from the database each time it's needed to make a new comparison.

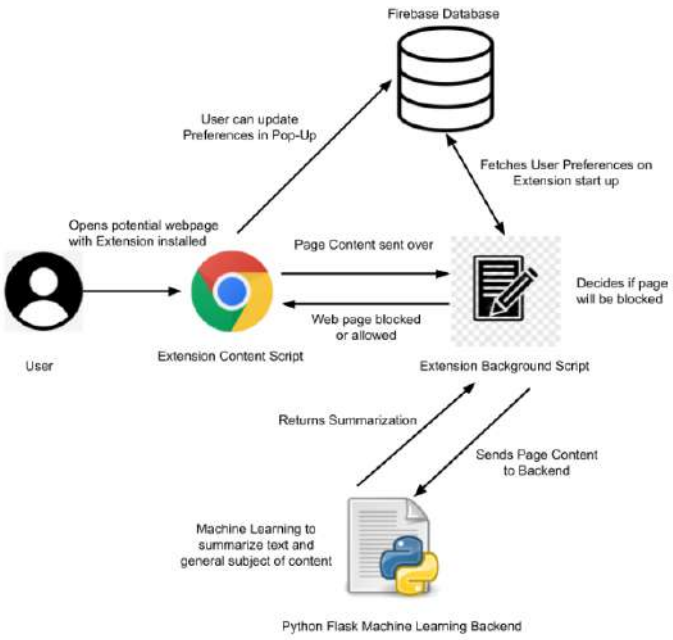

Figure 1. Overview of the solution

```python
@app.route("/getSummary/<data>")
def api_call(data):
  site_content = []
  page = data
  soup = BeautifulSoup(page, 'html.parser')
  articles = soup.find_all("p")

  for data in articles:
    if data is not None:
      data = data.text
      while "<" in data and ">" in data:
        index1 = data.find("<")
        index2 = data.find(">")
        tag = data[index1:index2+1]
        data = data.replace(tag, "")
      while "[" in data and "]" in data:
        index1 = data.find("[")
        index2 = data.find("]")
        tag = data[index1:index2+1]
        data = data.replace(tag, "")
      site_content.append(data)
  site_content = " ".join(site_content)
  site_content2 = site_content[(len(site_content)//2):]
  site_content = site_content[:(len(site_content)//2)]

  generate_summary(site_content, 2)
  return generate_summary(site_content2, 2)
```

Figure 2. Route code

```python
def generate_summary(data, top_n = 5):
  print("Generating the summary")
  stop_words = stopwords.words('english')
  summarize_text = []


  sentences = read_sentences(data)
  sentence_similarity_matrix = build_similarity_matrix(sentences, stop_words)
  sentence_similarity_graph = nx.from_numpy_array(sentence_similarity_matrix)
  scores = nx.pagerank(sentence_similarity_graph, max_iter=1000)
  ranked_sentence = sorted(((scores[i], s) for i, s in enumerate(sentences)), reverse = True)

  for i in range(top_n):
    summarize_text.append(" ".join(ranked_sentence[i][1]))

  print("Summarized Text: \n", ". ".join(summarize_text) + ".")
  return "Summarized Text: \n", ". ".join(summarize_text) + "."
```

Figure 3. Summary code

After users input a keyword to the extension popup, the content script will get executed and trigger the main body of the extension. First, the extension will scrape down the content on the current webpage and pass it to the backend part for summarization using natural language processing. Inside the back end, various NLP and ML techniques will be applied to the data in order to make a precise summarization. First, the script will read the data into separate sentences and filter out special characters. Then the sentences will be passed into another function to make a similarity matrix, which is based on the textual similarity and sentence ranking. Because of the function's ability to find out the similar words and put them into the matrix, the result will directly reflect the main topics discussed in the webpage. The matrix will then be used to rank each sentence and return the top selected ones as the summarization of the entire chunk of data. After having the summary, the extension will compare it against the keyword to see if it contains the word. If so, the current webpage will be categorized as a potentially distracting website and get blocked at the user end. However, if the result is negative, users will still have access to the website content. During this multi-step process, a number of third-party libraries are being used to assist with the precision of the summarization, including NLTK, Numpy, NetworkX, etc. The backbone of the extension lies on the web server created using the Flask library. It is responsible for not only setting up multiple routes for the users to navigate and explore, but also connecting the functionalities of the aforementioned NLP techniques. In addition, the server also takes the role of interacting with the database of the extension. Each time the users update a field in their

setting, the change will get passed back into the Firebase system through the API defined inside the web server [12]. On the user level, if they want to modify, add or remove the categories of websites they want the extension to block, they will be able to do so by clicking the extension popup and making their changes by interacting with it. In this way, the Flask server essentially connects the front end, back end, and the database system for user convenience. The parental control feature also allows parent users to set up a password right after the installation. After the setup of the password, each time the user attempts to modify the block list or update the password, they will be asked to re-enter the password to validate their identity as parents. Through this authentication process, if the user fails to prove their identity, their attempt to make changes will get rejected until the correct password is being entered.

```
IRR_SITES = [

    "The Useless Web." The Useless Web, n.d. https://theuselessweb.com/.

     "Yahoo | Mail, Weather, Search, Politics, News, Finance, Sports & Videos." Yahoo! Yahoo!, n.d.
https://www.yahoo.com/. "
]

GAMES_SITES = [

    "Games - Free Online Games at Addicting Games." Addicting Games, n.d. https://www.addictinggames.com/.

     Jonathan Bolding published 5 November 22, Kerry Brunskill published 5 November 22, Mollie Taylor published
5 November 22, Ted Litchfield published 4 November 22, Christopher Livingston published 4 November 22, Andy
Chalk published 4 November 22, Imogen Mellor published 4 November 22, et al. "PC Gamer." pcgamer, November 5,
2022. https://www.pcgamer.com/.


    "Your Favorites PC/Mac Games up to 70% off! Digital Games, Instant Delivery, 24/7!" Instant, n.d.
    https://www.instant-gaming.com/.

]
MOVIE_SITES = [

    "Ratings, Reviews, and Where to Watch the Best Movies & TV Shows." IMDb. IMDb.com, n.d.
    https://www.imdb.com/.


    "Unlimited Movies, TV Shows, and More." Netflix, n.d. https://www.netflix.com/.

    "Watch Free TV & Movies Online: Stream Full Length Videos." Tubi, n.d. https://tubitv.com/.

]

CAT_SITES = [

    Vicente, Rogério. "HTTP CATS." HTTP Status Cats API. Accessed November 5, 2022. https://http.cat/.


     TheCatSite, n.d. https://thecatsite.com/.


    "Great Artists' Mews." FatCatArt. Accessed November 5, 2022. https://fatcatart.com/.

]
```

Figure 4. Screenshot of code

4. EXPERIMENT

4.1. Experiment 1

To evaluate the efficiency of our approach of website blocking, we have performed 5 trials at different points of time. In each trial we enter a specific keyword into Concentration, then proceed to access 3 sets of websites (each set consists of 3 should-be-blocked websites that are relevant to the keyword and 2 should-not-be-blocked websites that are irrelevant to the keyword) and record the results (blocked/not blocked) in the tables below. Each table corresponds to a specific keyword as indicated by the table name.

Table 1. Testing websites with the keyword “gaming”

Testing Websites with the keyword "gaming"					
Timestamp	Irreverent Website 1 [2]	Irreverent Website 2 [3]	Relevant Website 1 [4]	Relevant Website 2 [5]	Relevant Website 3 [6]
7am, 10/17/22	TN; Not Blocked	TN; Not Blocked	TP; Blocked	TP; Blocked	TP; Blocked
1pm, 10/17/22	TN; Not Blocked	TN; Not Blocked	TP; Blocked	TP; Blocked	TP; Blocked
5pm, 10/17/22	TN; Not Blocked	TN; Not Blocked	TP; Blocked	TP; Blocked	TP; Blocked
7am, 10/18/22	TN; Not Blocked	TN; Not Blocked	TP; Blocked	TP; Blocked	TP; Blocked
9pm, 10/18/22	TN; Not Blocked	TN; Not Blocked	TP; Blocked	TP; Blocked	TP; Blocked

Table 2. Testing websites with the keyword “movie”

Testing Websites with the keyword "movie"					
Timestamp	Irreverent Website 1 [2]	Irreverent Website 2 [3]	Relevant Website 1 [7]	Relevant Website 2 [8]	Relevant Website 3 [9]
7am, 10/17/22	TN; Not Blocked	TN; Not Blocked	TP; Blocked	TP; Blocked	TP; Blocked
1pm, 10/17/22	TN; Not Blocked	TN; Not Blocked	TP; Blocked	TP; Blocked	TP; Blocked
5pm, 10/17/22	TN; Not Blocked	TN; Not Blocked	TP; Blocked	TP; Blocked	TP; Blocked
7am, 10/18/22	TN; Not Blocked	TN; Not Blocked	TP; Blocked	TP; Blocked	TP; Blocked
9pm, 10/18/22	TN; Not Blocked	TN; Not Blocked	TP; Blocked	TP; Blocked	TP; Blocked

Table 3. Testing websites with the keyword “cats”

Testing Websites with the keyword "cats"					
Timestamp	Irreverent Website 1 [2]	Irreverent Website 2 [3]	Relevant Website 1 [10]	Relevant Website 2 [11]	Relevant Website 3 [12]
7am, 10/17/22	TN; Not Blocked	TN; Not Blocked	TP; Blocked	TP; Blocked	TP; Blocked
1pm, 10/17/22	TN; Not Blocked	TN; Not Blocked	TP; Blocked	FN; Not Blocked	TP; Blocked
5pm, 10/17/22	TN; Not Blocked	TN; Not Blocked	TP; Blocked	TP; Blocked	TP; Blocked
7am, 10/18/22	TN; Not Blocked	TN; Not Blocked	TP; Blocked	TP; Blocked	TP; Blocked
9pm, 10/18/22	TN; Not Blocked	TN; Not Blocked	TP; Blocked	TP; Blocked	TP; Blocked

The experiment shows highly consistent and accurate results among 3 website sets. The data from the 1st and 2nd table indicates that Concentration correctly blocked every relevant website and did not block every irrelevant website, showing 100% accuracy in website blocking when the keywords "gaming" and "movie" were entered. The 3rd table shows that Concentration failed to block 1 relevant website and inaccurately blocked 1 irrelevant website, leading to an 93.3% accuracy in correctly blocking relevant websites and 100% accuracy in not blocking irrelevant websites when the keyword "cats" was entered. Overall speaking, Concentration performed fairly well in determining whether to block the page as judged by the high accuracy it achieves in the experiment, which is in alignment with my previous expectation that Concentration will yield accurate and reliable results on most user inputs.

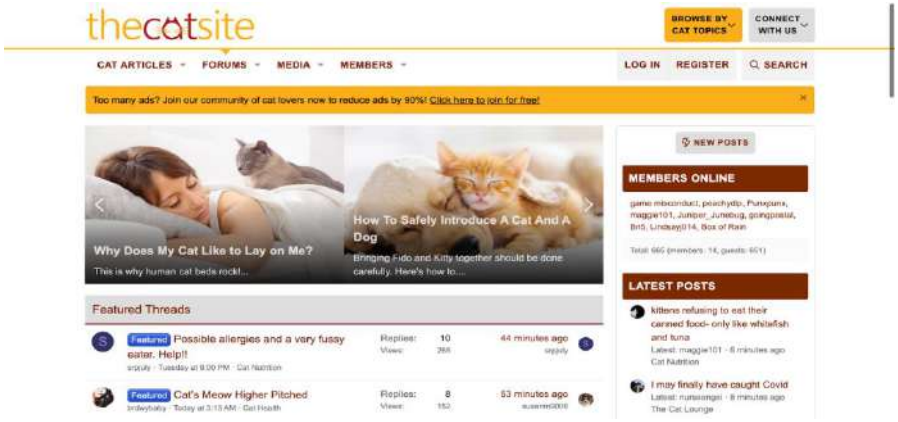

Figure 5. Screenshot of the website that Concentration fails to block when the keyword is "cats"

5. RELATED WORK

N K Nagwani [13] performed text summarization on large chunks of textual data through the use of MapReduce framework. By comparison, our work combines summarization and categorization of website contents in relatively smaller sizes. Our algorithm is capable of monitoring real-time websites status, taking note of the specific categories the websites fall into, and blocking the sites accordingly based on the result, while Nagwani's work focuses solely on implementation of efficient means of text summarization and relevant information extraction.

Igor Kotenko et al. [14] presented an automated approach to detect, evaluate, and block websites of inappropriate content intelligently. In their model, they used techniques such as text and html analysis, as well as methods of machine learning and data mining to construct a systematic algorithm of identifying and blocking various web pages and -sites. Our method shares several major similarities with this approach, including the content summarization, category evaluation, and denial of access on the user's end. As different from their model which implements the system using F-Secure platform, ours exploits natural language processing (NLP) technique and Flask server to compose our main functionalities in Python.

Zhang et al. [15] conducted an experiment on the suitability of users' feedback for Web content summarization across several representative social services and concluded the superiority of bookmarking service over others. In addition, they further proved the experimental result by implementing the SSNote system and comparing its output with manual summaries. The paper put the focus on the evaluation of potential impacts of user's activities to the quality of auto-summarization, while ours takes only the website content into account and leaves alone with user's preferences.

6. CONCLUSIONS

In this project, we proposed an intelligent approach to auto-block distracting websites based on user inputs using web scraping and natural language processing. The application is implemented as a Chrome extension that scrapes down and summarizes the web content, obtains the keyword that the user previously inputted, compares the summary to the keyword and finally blocks the website if the summary is similar to the keyword. Experiment indicates that the extension performs well across most user inputs, achieving 100% accuracy for input keywords "gaming" and "movie" and ≥93% accuracy for input keyword "cats" and showing high consistency across all blocking results.

After successfully retrieving the web content from the user's browser, it is rather challenging to perform accurate text summarizations on the web content. As a result, the prediction of the exact category the website falls into is not always entirely accurate, especially when a website covers many topics. In addition, since we have to design our UI in limited spaces, it's also difficult to come up with a clean design that provides all necessary functionalities the users may need to use. The project's UI does not increase usability or practicability as much as it could. Finally, the extension takes longer-than-usual time on summarizing for websites that consist of a lot of textual content.

Given these limitations at hand, we can do more research on how we may use more performant NLP libraries and algorithms to achieve more accurate results of summarization in a more timely manner [11]. We can also use users' feedback and suggestions on how we may further improve our UI appearance as the guidelines of our future designs.

REFERENCES

[1] Sörqvist, Patrik, and John E Marsh. "How Concentration Shields Against Distraction." Current directions in psychological science vol. 24,4 (2015): 267-272. doi:10.1177/0963721415577356
[2] Connelly, S. Lisa, Lynn Hasher, and Rose T. Zacks. "Age and reading: the impact of distraction." Psychology and aging 6.4 (1991): 533.
[3] Dontre, Alexander J. "The influence of technology on academic distraction: A review." Human Behavior and Emerging Technologies 3.3 (2021): 379-390.
[4] Thomas, David Mathew, and Sandeep Mathur. "Data analysis by web scraping using python." 2019 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA). IEEE, 2019.
[5] Monsma, Eva, Melanie Perreault, and Robert Doan. "Focus! Keys to developing concentration skills in open-skill sports." Journal of Physical Education, Recreation & Dance 88.7 (2017): 51-55.
[6] Herring, Susan C. "Web content analysis: Expanding the paradigm." International handbook of Internet research. Springer, Dordrecht, 2009. 233-249.
[7] Livingston, Samuel A., and Charles Lewis. "Estimating the consistency and accuracy of classifications based on test scores." Journal of educational measurement 32.2 (1995): 179-197.
[8] Cambria, Erik, and Bebo White. "Jumping NLP curves: A review of natural language processing research." IEEE Computational intelligence magazine 9.2 (2014): 48-57.
[9] Kikuchi, Yukio. "Numerical simulation of the blocking process." Journal of the Meteorological Society of Japan. Ser. II 47.1 (1969): 29-54.
[10] Abdullah, Hanin M., and Ahmed M. Zeki. "Frontend and backend web technologies in social networking sites: Facebook as an example." 2014 3rd international conference on advanced computer science applications and technologies. IEEE, 2014.
[11] Cheng, Xinyun, et al. "A combined method for usage of NLP libraries towards analyzing software documents." International Conference on Advanced Information Systems Engineering. Springer, Cham, 2020.
[12] Gu, Xiaodong, et al. "Deep API learning." Proceedings of the 2016 24th ACM SIGSOFT international symposium on foundations of software engineering. 2016.
[13] Nagwani, Naresh Kumar. "Summarizing large text collection using topic modeling and clustering based on MapReduce framework." Journal of Big Data 2.1 (2015): 1-18.
[14] Kotenko, Igor, et al. "Analysis and evaluation of web pages classification techniques for inappropriate content blocking." Industrial Conference on Data Mining. Springer, Cham, 2014.
[15] Park, Jaehui, et al. "Web content summarization using social bookmarks: a new approach for social summarization." Proceedings of the 10th ACM workshop on Web information and data management. 2008.