Journal of Korea Multimedia Society Vol. 22, No. 3, March 2019(pp. 374-385)
https://doi.org/10.9717/kmms.2019.22.3.374

Design and Implementation of Web Crawler utilizing Unstructured data

Ahmed Md. Tanvir[†], Mokdong Chung[††]

ABSTRACT

A Web Crawler is a program, which is commonly used by search engines to find the new brainchild on the internet. The use of crawlers has made the web easier for users. In this paper, we have used unstructured data by structuralization to collect data from the web pages. Our system is able to choose the word near our keyword in more than one document using unstructured way. Neighbor data were collected on the keyword through word2vec. The system goal is filtered at the data acquisition level and for a large taxonomy. The main problem in text taxonomy is how to improve the classification accuracy. In order to improve the accuracy, we propose a new weighting method of TF-IDF. In this paper, we modified TF-algorithm to calculate the accuracy of unstructured data. Finally, our system proposes a competent web pages search crawling algorithm, which is derived from TF-IDF and RL Web search algorithm to enhance the searching efficiency of the relevant information. In this paper, an attempt has been made to research and examine the work nature of crawlers and crawling algorithms in search engines for efficient information retrieval.

Key words: Web Crawling, TF-IDF, Unstructured data, Hyperlink.

1. INTRODUCTION

With the flourishing availability of text documents on the World Wide Web (WWW), it has become very difficult for an average reader to gather information on individual issues, events, or topics by reading each and every document with an aim to find out the useful information. Therefore, the smart crawling system able to play a major role to overcome this problem. Because the crawler is a program that retrieves and stores pages from the web, commonly for a web search engine [2]. In this research, we have discussed how web crawling is highlighted in an efficient way. We preferred unstructured data in this article and the data was collected from the web page using a specific keyword. The unstructured data is a data or information that do not have any predefined data model or not organized pre-defined model. Earlier, in the research on unstructured data [1], [8] in web pages similarity but was very condensed. For this reason, their accuracy rate is not up to the satisfaction levels. Therefore, to solve this phenomenon, on this paper effective method of collecting unstructured data is demonstrated. First of all, we have collected automatically keywords related urls. Then created the data tree using the Breadth First Search (BFS) method from the all collected urls. Unstructured data was accounted and collected from the web page. We have collected multitudes unstructured

※ Corresponding Author : Mokdong Chung, Address: (48513) Yongso-ro 45, Nam-gu, Busan, Korea, TEL : +82-10-2858-6883, FAX : +82-51-629-6264, E-mail : mdchung@pknu.ac.kr
Receipt date : Nov. 12, 2018, Revision date : Jan. 10, 2019
Approval date : Jan. 21, 2019
[†] Dept. of Computer Engineering, Pukyong National University, Busan, South Korea (E-mail : tva.csai@gmail.com)
[††] Dept. of Computer Engineering, Pukyong National University, Busan, South Korea
※ This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF2017R1D1A1B03030033).

data. Therefore, we used word2vec to identify keyword related data from the multiple data pool. In this paper, filtration of the data acquisition level was performed for a large taxonomy.

The web crawler algorithm is used to traverse a web pages by extracting inter links and out links. The extracted links were seeded in URL list for future use. The web document are linked by multiple hypertext links with connecting diverse resources. The web crawler implements various web search algorithm for retrieving and locating web documents, such as BFS, Term Frequency-Inverse Document Frequency (TF-TDF) and Reinforcement Learning (RL). The implementation of different web search crawling algorithms provides search efficiencies. By analyzing various web search algorithms, this paper proposes a new method to use the TF-IDF algorithm by improving the equation. The rest of the paper has been constructed in this manner. The next section presents the related work, where we will describe our algorithms. In section 3 the design of web crawler. In section 4 out implementation and evaluation of the system. Finally, section 5 discusses the conclusion and future work.

2. RELATED WORK

2.1 Unstructured data

Unstructured data, in contrast, refers to data that does not fit neatly into the traditional row and column structure of relational databases. In web pages, it often includes text and multimedia content. Examples of text files like data processor, spreadsheets, presentations, webpages and many other kinds of business documents. While these sorts of files may have an internal structure, they are still considered unstructured because the data they contain does not fit neatly in a relational database. The learning resources and social networking sites are examples of unstructured data [3]. The video and audio streaming of classroom is also unstructured data. Our paper focuses on learning resources and web pages multimedia data for unstructured data [12]. The education system generates, maintains and analyzes large amount of data through various sources. This data is related to academic, non-academic, learning, examination, admiration, training and placement. The nature of such data as shown in figure 1, is varied in nature as given.

2.2 An overview of TF-IDF

We will first introduce the mathematical background of the algorithm and also describe the TF-IDF algorithm working process.

2.2.1. Mathematical Framework

Basically, TF-IDF works by determining the relative frequency of words in a specific document compared to the inverse ratio of that word over the entire document corpus. Intuitively, this calculation determines how relevant a given word is in a special document. Words that are common in a single

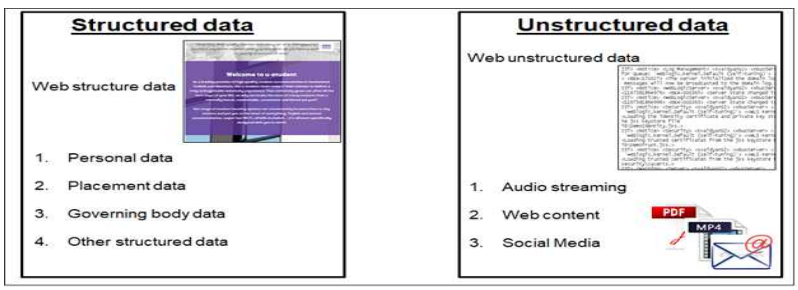

Fig. 1. Type of data and its application area.

Table 1. TF-IDF algorithm tools

Methods	Denoted by
Term Frequency	TF
Web document	x
Key words	K
Web page	P
Number of keyword document	K_i
Number of webpage document	P_i
Their interrelation coefficients	T
Total number of web page	$\|P\|$
Total number of Keyword	$\|K\|$
1st equation	$TF(P,K) = fre(P,K) / \sum_{K_i \in K} fre(P,K_i)$
2nd equation	$IDF(P,K) = \|P\| / \sum_{P_i \in P} dfre(P_i,K)$
3rd equation	T(P,K) = TF(P,K) × IDF(P,K)

or a small group of documents tend to have higher TF-IDF numbers than common words such as articles and prepositions. The formal procedure for implementing TF-IDF has some minor differences over all its applications, but the overall approach works as follows in table 1. Their paper first defines a topic keyword dictionary, at the same time the web document is represented by x heft words, and then use TF-IDF method to calculate keywords (K), webpage, (P) and Term Frequency (TF) as shown in equation (1).

$$TF(P,K) = fre(P,K) / \sum_{K_i \in K} fre(P,K_i) \quad (1)$$

Here, P, K indicate the number of things of the keyword K in the web document P. As shown in equation (2) Inverse Document Frequency (IDF).

$$IDF(P,K) = |P| / \sum_{P_i \in P} dfre(P_i,K) \quad (2)$$

For the keyword, K with the webpages, P. Their interrelation coefficients are calculated as shown in equation (3).

$$T(P, K) = TF(P, K) \times IDF(P, K) \quad (3)$$

TF-IDF is the multiple of the value of TF and IDF for a particular word. The value of TF-IDF increases with the number of occurrences within a document and with rarity of the term across the corps. Finally T has been used to establish a connection between TF and IDF [6].

2.3 Reinforcement Learning Framework

In this section, we propose a formal framework for the deep web crawling based on RL and formalize the crawling problem under the framework. First of all we give an overview of the RL framework. The relation between a crawler and a deep web database is illustrated in figure 2.

From the figure, one can conclude that at any given step, an agent (crawler) perceives its state and selects an action (query). The environment

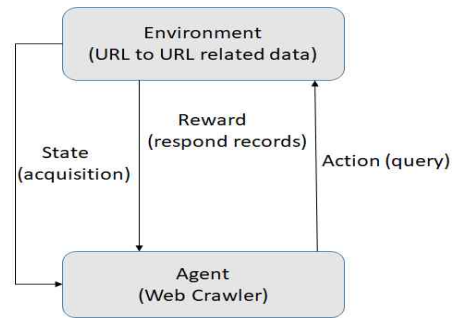

Fig. 2. Overview of the reinforcement learning frame work.

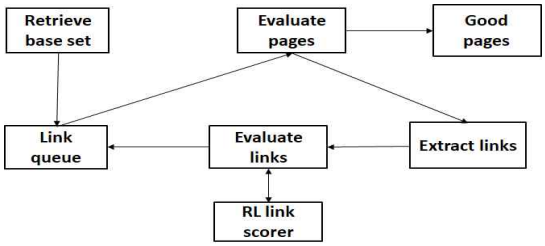

Fig 3 . Reinforcement Learning working Overview.

responds by giving the agent some (possibly zero) reward (new records) and changing the agent into the successor state [11]. In the total procedure of RL-algorithm in the proposed method is described by the figure 3.

1. Link queue: Current set of links that have to be visited. Fetch link with highest score on queue.
2. Evaluate page this link points to: Based on set of text/content attributes. If relevant, store on Good Pages.
3. Get links from page
4. Evaluate links, add to link queue.
5. In the RL link scorer evaluate the links.

2.4 Word2vec

Word2vec is an algorithm for learning embedding's using a neural language model. It also an open source tool released by Google in around 2013 [12]. It is based on deep learning and can learn the vector representations of words. Our first challenge in the proposed method is to collect data from the web. For this reason, we have used specific keywords. We have selected specific four keywords (library, degree, student, university) and automatically collected hyperlink to the web according to the keywords described in figure 4. We have preferred web pages unstructured data. We have collected web hyperlinks in accordance with BFS method and selected unstructured related data. Therefore, we have selected keyword related web pages data using word2vec. We used

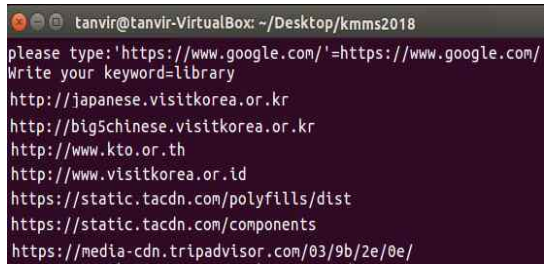

Fig. 4. Data gathering method according to the keyword.

word2vec to divide the closer neighboring words in our used keywords. The dense vector representations of words learned by word2vec can capture the semantic meanings, the logical relationships between words, where words that we know to be synonyms are more likely to have very closed vectors and antonyms tend to have dissimilar vectors. Because these vectors of words adhere surprisingly well to our friend word intuition and obey the laws of analogy.

3. DESIGN AND IMPLEMENTATION OF WEB CRAWLER

3.1 Web crawler architecture

Web data acquisition is the foundation of web data mining. The web crawler is an important tool for web data acquisition. A web crawler is a program that is an essential web search engine to find the URL to the web page then save and download it. Web crawlers are an important component of web search engine, where they are used to collect the corpus of web pages index by the search engine as shown in figure 5 [4].

The web crawler's work methods are described. The web page has different types of data. It is not possible to separate, when we are collecting this data, such as structured data or unstructured data in figure 1. Therefore, we created the architecture to separate these data. According to this architecture, we collect data from the web page and it is divided into structured and unstructured ways. Actually structure data is a data which is stored

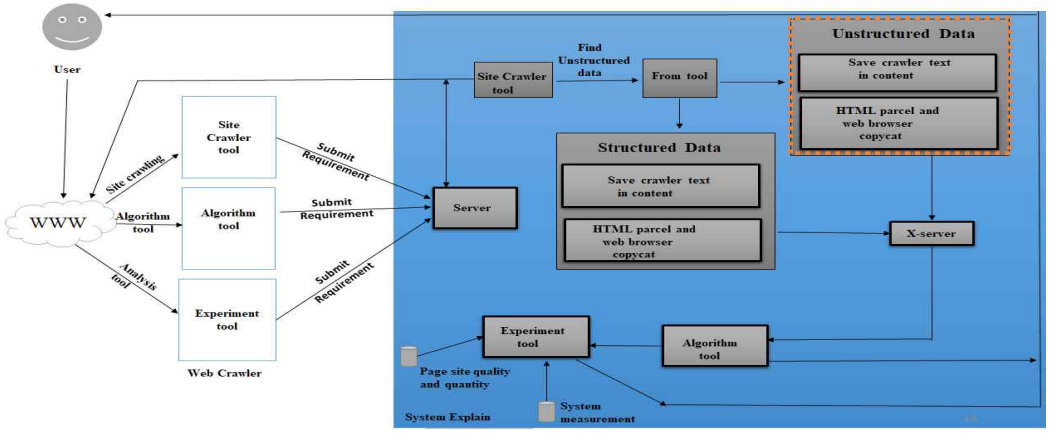

Fig. 5. Web crawler architecture.

in row and column format using traditional database management system. It follows all relational database management system properties like ACID, Normalization and etc. examples are Oracle, MySQL, ms-access, and so on. On the other hand, unstructured data is a data which is not stored in a table format. Which has no structure. It does not have a specific pattern like column or row. It includes audio, video, pdf file etc. Therefore, we preferred web pages unstructured data. Firstly, we have chosen a web page, after which we have separated all the unstructured data associated with that web page, which we saved on our server. The biggest advantage of this architecture is that the collected data will not be lost, all data can be stored separately on the server. In this step of the algorithm tool, we will design how to use the algorithm. Finally, the data reaches the experimental tool. The experimental tool uses the metrics computation tool's output to show how a given design differ from highly rated designs with a similar purpose. It uses several statistical models which derived from an analysis, that were rated according to their quality and usability and finally, it sends the result to the user.

3.2 Proposed Approach TF-Algorithm

The flow of measurement of the TF-algorithm of the syndrome is shown by the pseudocode as follows.

1. **START**
2. P = 0; x = 0
3. **While** (True) **do**
4. **While**(Keyword queue is not empty) **do**
5. Pop $|k|$ from the Keyword queue
6. **for**(k = 1..n) **do**
7. **if** (P [k] = 0 and $|p|$[k] < 0) or ($|p|$ [k] < 0 and t < x[k] + x) **then**
8. **if** (p[k_i] > 0 and t > x[k_i]+x) **then**
9. p [k_i] = $p_i[k_i]-1$
10. x [k_i] = t
11. **if** (k == n) **then**
12. Reject task $|k|$
13. **else**
14. **if** ($|p|$[$|k|$] < T and $|p|$ [$|k|$] > 0 and t > x [$|k|$] + x) **then**
15. $|p|$ [$|k|$] = $|p|$ [$|k|$] -1
16. x $|k|$ = t
17. $|k|$ is a controller by P
18. **else**
19. **if** ($|p|$[$|k|$] > + T and p [$|k|$] < fre p and k / (get_total (p and k_i)) **then**
20. p $|k|$ = p$|k|$ +1
21. x[k_i] =t
22. push $|k|$ in the internal keyword queue value

23. **break**
24. **while** (True) **do**
25. **if** (Keyword queue is empty for initial next state s' and a') **then**
26. **for** (k=1..n) **do**
27. **if**(p[k_i] < fre.P and k_i) **then**
28. p[k_i]=p[k_i]+1
29. x[k] = t
30. **END**

Step-1: Invocation and initialization (line 2 to 23). The block functionality is executed in an infinite loop (line 3) activated every time interval (line 23). The current P state is set to the lowest value, p_i and the vector of time of the previous p state change, x, is set to 0 (line 2).

Step-2: Task fetching (line 4 to 5), the tasks external queue is checked if empty end, if not a task total keyword value $|k|$ is fetched (line 5).

Step-3: The conditional rejection (line 7 to 13), The code related to this step is executed inside a loop whose iterator k ranges from 1 to n, which is the number of cores in the processor (line 6). If the output value of the k_i is controller ($|p|[k]$) is negative and the corresponding k_i core operates with highest performance and the interval between the previous p state switching time and current time (t) is long enough this value is assumed to have no capacity to execute task ki. Consequently, if all the cores in the analyzed processor have no capacity (line 11), task $|k|$ is rejected (line 13). Moreover, if P state of the k_i core is different from p and its p_i state has not been switched for at least time web document x (line 8), p state in the k_i core is decreased (line 9) and the k_i element of vector x, storing the previous time of the p_i state alteration of the value k_i, is updated (line 10). As the previous errors in the k controller have been observed in a different P state (line 11). Hence, these obsolete values do not influence future admittance decisions.

Step-4: Task conditional admittance (lines 14 to 23), If the output value $|k|$ controller is lower then threshold +T, $|p|$ state in the $|k|$ core is long enough (line 14), the $|p|$ state is lower (line 15), and x is updated accordingly (line 16). In case the $|k|$ dependent controller output value is above threshold +T, P state of the k_i core is different from the highest p state available in the processor (frequency p) and the previous switching of p state in the $|k|$ core is increased (line 19), $p|k|$ and x is updated (line 20 to 21). Finally $|k|$ is sent to the every state queue (line 22).

The second part of the algorithm is located between lines 24-30 and contains two steps only, as described below.

Step-1: Invocation (line 24 to 30), the block functionality is executed in an infinite loop (line 24), activated every time interval (line 29).

Step-2: p state conditional increase (line 25 to 29), the functionality of this next state step is activated a' and s' (line 25). Under such condition, p state of all n cores are analyzed in a loop (line 26). If the core performance is not the lowest possible then the core's p state is increased (line 28), the web document value continues updated (line 29).

3.2.1 Proposed Approach RL-Algorithm

Reinforcement Learning or Q-Learning is learning method, which renews each synapse weights according to output signals. The learning rule can be applied to feed forward networks and recurrent networks. The flow of measurement of the renewal of the syndrome is shown by the pseudocode as follows.

1. Initialize state (s_n, a_n)
2. calculate the reward of action an
3. **for** each document $d_i \in r(s_{n-1}, a_n)$
4. **for** each keyword k in d_i do
5. **if** action a(k) $\neq$A then $A = A \bigcup a(k)$
6. **else** then restart Reinforcement-Learning (RL) of action a(k)

7. **end for**
8. **end for**
9. change the current state to s_{n+1}
10. $P_r = P_r \bigcup [a_n]$ restart candidate set S; S = S / $[a_n]$
11. **for** each $a_i \in S$ update its reward r_t
12. **for** each $a_i \in S$ update its each value γ
13. **for** each $a_i \in S$ calculated its RL-value
14. **return** $\arg.\max_a Q^*(s', a)$
15. **End**

Specifically, the surfacing algorithm first calculates the reward of the last executed action and then updates the action set through step 3 to Step 8, which motive the agent to transit from its state 's_n' to the successor state 's_{n+1}'. Then the training and candidate set are restart in accord with the new action set in step 10. After that, the algorithm catalog the reward and RL-value for each action in candidate set in step 11, step 12 and step 13 respectively. The action that maximizes RL-value will be returned as the next to be executed action.

3.3 Experimental Overview

In a good segmentation, sentences among different segments should belong to different subtopics. Hence, the goal is to find the segmentation with the maximum external (two consecutive segments) dissimilarity. There are two types of segmentation in the field of the web such as structured and unstructured data. Web researchers usually use these two types of data to complete their web research. In this paper, we have preferred only web pages unstructured data. In the proposed method, we have used unstructured data in the form of structuralization. In the existing method, they have preferred both types of data. As a result, they are not able to find their keyword data equally. The equation (1) used in the existing method, in the primary stage if the Web page (P), Keywords (K) and Number of keyword document (K_i) any value are "0" so, the result of the total TF value is zero, it means that the keyword is unable to choose a similar or closer word from the same document, which is not a good result. In this reason, the accuracy rate is not up to the satisfaction levels.

Therefore, to solve this problem we have used the total number of keyword document "**|K|**" in equation (4). In previous research, the total number of words has not been calculated. In this paper, we have calculated the total word value. This means that in the same document, the keyword can be correctly selected as a similar word. Therefore, if the more documents a term appears, the less important that term will be, and the weighting will be less. So the user can find their information easily and within in a short time.

$$TF(P, K) = fre(P, K) / \sum_{K_i \in K} fre(P, K_i) + |K| \qquad (4)$$

We can see the difference between the existing and proposed method using equation (1) and (4) and in figure 6 and 7.

4. IMPLEMENTATION AND EVALUATION

4.1 Implementation

At the beginning of implementation, we will describe our data collect process from the web page. By using the specific keywords, we collected all hyperlinks which belong to in that web page. In this paper, firstly we have presented the working design of our crawling system by an architecture. Here the process of execution of RL and TF-IDF

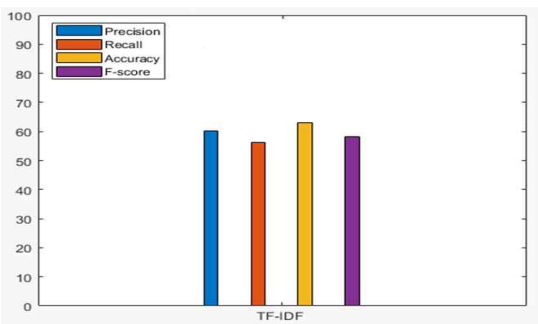

Fig. 6. Existing system TF algorithm result.

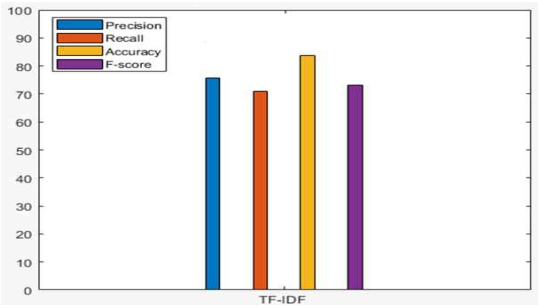

Fig. 7. Proposed system TF algorithm result.

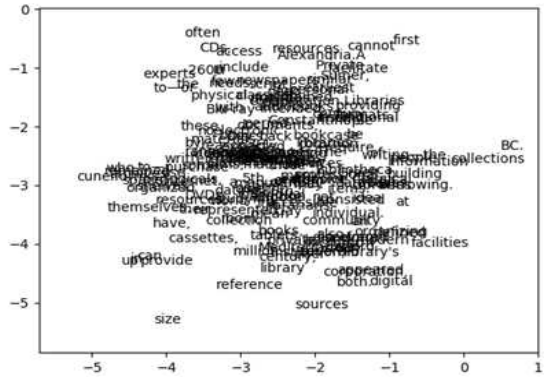

Fig. 8. Word2vec Overview.

algorithm work is displayed. When the crawler finishes downloading a page from web, the relevance of the page needs to be identified to improve the efficiency of the crawling process and save the storage space. For this reason we have used BFS and word2vec algorithm. At first we explain an interesting feature of the BFS algorithm [10]. The BFS algorithm traverses the graph by following its links. The distance of each crawled page from the root (seed) is always less than or equal to that of the uncrowded pages. The BFS ordering is not the best method for crawling but breadth-first has an interesting feature. It can discover pages with high page rank in the initial stages of the crawling process.

We collected unstructured data from web pages using the BFS algorithm. Then, we used word2vec to divide the closer neighboring words in our used keywords. This of one is an experiment to distinguish document features by extracting features of document semantics and another one is word embedding (word2vec), which is used to distinguish the meaning of words in a document. In figure 8 we described the method of using word2 vec. We collected four data related to the library, here the results of the total data were shown using word2vec. Which is very close to our specific keyword (library). In table 2, we have described the keyword related data after using word2vec. Here we have emphasized the above data on value 0.5. We have used the keyword related data using word2vec, now we will calculate Precision, Recall, Accuracy and F-score between these data using TF and RL algorithm. The method of using TF method is already described in our design point.

A task is defined by a set of state, $s_n \epsilon$ S, a set of actions, a ϵ A, a state action alteration function, T : S × A → S, and reward function, r : S × A → R. At each step, the RL equation selects an action and then as a result is given a reward and its new state. The goal of RL is to learn a policy, a mapping from states to actions, π : S → A, that maximize the sum of its reward over time. We use the unlimited-horizon discounted model where reward over time is a geometrically discounted sum in which the discount, $0 \leq \gamma \leq 1$, devalues rewards received in the future. Therefore, when following policy 'π', we can define the value of each

Table 2. Word2vec Performance Measurements

Input: Library		Input: Degree		Input: Student		Input: University	
sources	0.62163	assessment	0.51244	kindness	0.55665	graduation	0.59895
facilities	0.80525	assembled	0.53224	reader	0.5786	versify	0.66181
sanctum	0.86355	designation	0.57001	pupil	0.59032	education	0.74821

state to be

$$V^{\pi}(s) = \sum_{t=0}^{n} \gamma r_t \quad (5)$$

Where 'r_t' is the reward received 't' time steps after starting in state 's' and following policy 'π'. The optimal policy written 'π*', is the one that maximizes the value V^{π}(s), for all initial states 's' and next state 's_n'. In order to learn the optimal policy we learn its value function 'V^{*}' and its more specific correlate called 'Q'. Let Q*(s, a) be the value of selecting action 'a' from initial state 's' and next state s_n, and thereafter following the optimal policy [13]. This is expressed as

$$Q^{*}(s,a) = r(s,a) + \gamma V^{*}(T(s_n,a)) \quad (6)$$

We can now define the optimal policy in terms of Q by selecting from each the action with the highest expected future reward $\pi^{*}(s) = \arg.\max_a Q^{*}(s_n, a)$. In RL-Algorithm and we have used data from word2vec. Using the above mentioned equation (6), our results are shown in figure 9.

4.2 Evaluation

How to evaluate the web data in crawler system it is a difficult question. Generally, there have two kinds of evaluation methods: intrinsic evaluation and extrinsic evaluation. Intrinsic evaluation directly analyzes the data to judge the quality of themselves. Extrinsic evaluation judges the quality of data by its affection to some other task. In this paper, an intrinsic evaluation method is presented. In figure 10, we shows our overall performance. We have shown here the performance of Precision, Recall, F-score and Accuracy. When a user searches on his document, he will get results in a short time because our algorithm does not face any overlap and can show the correct results. In the evaluation, we will explain it more detail. First, we have selected four keywords (library, degree, research, and student) and we have collected data from the web page. We have provided data with 100 hyperlink related data per webpage according to each keyword. We collected 400-hyperlink information from each of words. We have created a data tree using the BFS method. By using word2vec we have collected the keyword related data. Therefore, we use the TF algorithm to calculate the weight value of each hyperlink data. The results of the TF and RL algorithm used in the existing and proposed systems are described in the table 3. Here we have described each keyword's total precision, recall accuracy and F-score value. In the figure 10 we can see that the result of Precision, Recall and F-score of RL algorithm is better than TF algorithm but the results of Accuracy are not good. Because the TF algorithm follows the word frequency, in this reason, the number of mutations of mutual vibration occurs in the word. Therefore, the accuracy rate is higher in the TF algorithm.

This section reports on our and existing system

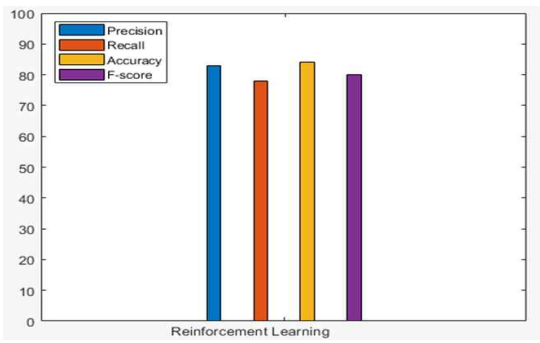

Fig. 9. Proposed system Reinforcement Learning result.

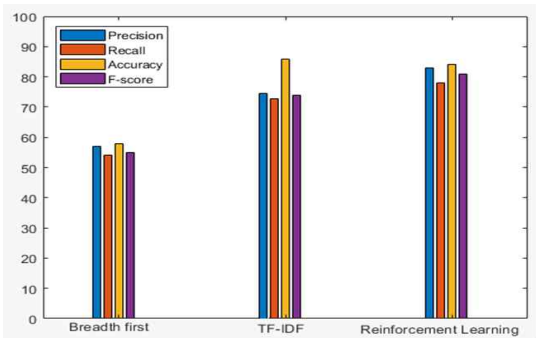

Fig. 10. Overall Performance.

Table 3. Performance Measures

Existing System				
Algorithm	Precision	Recall	Accuracy	F-score
Breadth First	48.8	40.4	43.1	44.2
TF-IDF	62.1	58.6	65.3	60.2
Proposed System				
Algorithm	Precision	Recall	Accuracy	F-score
Breadth First	57.3	53.6	58.4	55.3
TF-IDF	74.2	72.4	87.2	73.2
Reinforcement Learning (RL)	83.2	78.2	84.1	80.6

performance during the crawl. In our analysis of web crawler performance, we contrast it with the performance of the Google, Internet archive crawlers and existing system. We make no attempt to adjust for different hardware configurations since the papers describing the two other crawlers do not contain enough information to do so. Our main intention in presenting this performance data is to convey the relative speeds of the various crawlers. The overall performance between proposed and existing method are described in the table 4.

5. CONCLUSION

In this paper, we emphasized the use of web unstructured data, we have proved that we can use this unstructured data through structuralization. In structuralization of unstructured data, we emphasized the frequency of the word, so we have used the TF algorithm by improving the equation. The structured data typically gives us information about what happened, on the other hand, unstructured data provides us with information on why it happened, for example, web crawling. Our conceptual research on web crawling and improving its performance by the various crawling algorithms has previously explained and we have described a new method by using the word2vec. Compared to the baseline method only changing term frequency (TF), the obtained results demonstrated better efficiency and accuracy. Finally, we have used RL-algorithm to increase the performance of the crawling. These results demonstrated a strong evidence that reinforcement learning is an excellent framework to perform web spearing. Additionally, to investigate other value function criteria that relax some of current assumptions meet the merit of future study. In this paper, we have clearly described that the accuracy rate of our used algorithm is much better than other systems. In the fu-

Table 4. Performance area

Category		System Descriptions	
Index	Title	Existing System	Proposed System
1	Alpha Processor	1.70GHz	270.GHz
2	RAM	16GB	16GB
3	Local Disk	258GB	229GB
4	Programming Language	Java	Python and Matlab
5	HTTP request in 7 days	81.4 million	90.3 million
6	Download rate	120 document/sec 1682 KB/sec	120 document/sec and 1872KB/sec

ture, we will improve the accuracy and efficiency of short texts feature extension, expand the scale of the experiment, and apply the method to various kinds of unstructured data.

REFERENCE

[1] S. Saranya, B.S.E. Zoraida, and P.V. Paul, "A Study on Competent Crawling Algorithm (CCA) for Web Search to Enhance Efficiency of Information Retrieval," *Proceeding of Artificial Intelligence and Evolutionary Algorithms in Engineering Systems, Springer, New Delhi*, pp. 9-16, 2015.

[2] K.S. Kim, K.Y. Kim, K.H. Lee, T.K. Kim, and W.S. Cho, "Design and implementation of web crawler based on dynamic web collection cycle," *Proceeding of The International Conference on Information Network, IEEE*, pp. 562-566, 2012.

[3] Y. Kim, H. Hong, and M. Chung, "Application of Cohesion Devices for Improvement of Distributional Representation," *Proceeding of The 14th International Conference on Multimedia Information Technology and Applications (MITA),* pp. 84-87, 2018.

[4] M.Y. Ivory and M.A. Hearst, "Improving web site design," *Proceeding of IEEE Internet Computing 2,* Vol. 6, No. 2, pp. 56-63, 2002.

[5] D. Debraj and P. Das, "Study of deep web and a new form based crawling technique," *International Journal of Computer Engineering and Technology (IJCET),* Vol. 7, No. 1, pp. 36-44, 2016.

[6] Z. Guojun, J. Wenchao, S. Jihui, S. Fan, Z. Hao, L. Jiang, et al., "Design and application of intelligent dynamic crawler for web data mining," *Proceeding of 2017 32nd Youth Academic Annual Conference of Chinese Association of Automation (YAC) IEEE,* pp. 1098-1105, 2017.

[7] K.A. Pakojwar, R.S. Mangrulkar, and V.G. Bhujade, "Web data extraction and alignment using tag and value similarity," *Proceeding of 2015 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS),* pp. 1-4, 2015.

[8] S. Kolhatkar, M.M. Pati, M.S. Kolhatkar, and M.S. Paranjape, "Emergence of Unstructured Data and Scope of Big Data in Indian Education," *International Journal of Advanced Computer Science and Applications (IJACSA)*, Vol. 8, No. 1, pp. 150-157, 2017.

[9] M. Afsharizadeh, H. Ebrahimpour-Komleh, and A. Bagheri, "Query-oriented text summarization using sentence extraction technique," *Proceeding of 4th International Conference on Web Research (ICWR)*, pp. 128-132, 2018.

[10] S. Ringe, N. Francis, and A.H.S.A. Palanawala, "Ontology Based Web Crawler," *International Journal of Computer Applications in Engineering Sciences,* Vol. 2, No. 3, pp. 194-197, 2012.

[11] L. Jiang, Z. Wu, Q. Feng, J. Liu, and Q. Zheng, "Efficient deep web crawling using reinforcement learning," *Proceeding of Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, Berlin, Heidelberg,* pp. 428-439, 2010.

[12] Y. Kim, B. Kim, and M. Chung, "Unstructured data analysis and multi-pattern storage technique for traffic information inference," *The Journal of Multimedia Information System,* Vol. 21, No. 2, pp. 211-223, 2018.

[13] R. Jason and A. McCallum, "Using reinforcement learning to spider the web efficiently," *Proceeding of International Conference on Machine Learning (ICML)*, Vol. 99, 1999.

Ahmed Md. Tanvir

2015: BS in Computer Science and Engineering, The Millennium University, Bangladesh
2017~Present: MS in Computer Engineering, Pukyong National University
Research Interests: Computer Security for Application, Ubiquitous Computing, Context Aware Computing, Neural Network.

Mokdong Chung

1981: BS in Computer Engineering, Kyungpook National University
1983: MS in Computer Engineering, Seoul National University
1990: Ph.D in Computer Engineering, Seoul National University
1984~1985: Researcher, Goldstar Semiconductor Co,
1985~1996: Professor, Department of Computer Engineering, Pusan University of Foreign Studies
1996~Present: Professor, Department of Computer Engineering, Pukyong National University
1999~2000: Visiting Professor, Iowa State University, USA
Research Interests: Computer Security for Application, Bigdata Analysis, Context Aware Computing, Intelligent Agent.