RESEARCH ARTICLE OPEN ACCESS

Web Log File Analysis to Predict User Behavior Using the Multi-Context Recurrent Neural Network

Hadeel Ahmed [1], Mohand Kechadi [2]
[1] Computer Science, Sudan University of Science and Technology - Sudan
[2] Computer Science, University of College Dublin, Ireland

ABSTRACT
The web log file contains information about the user's interaction with the application. Understanding users' online behaviour has become necessity for the success of websites and other online businesses and applications. This will help in configuring applications based on user behaviour. Also, a better understanding of user behaviour and extracting meaningful information from the log file to develop the user personalization page. This paper describes a multi-context recurrent neural network. Some issues with the Elman network's accuracy and learning speed need to be resolved. To solve these limitations, they proposed a Multi–Context Recurrent Neural Network with three different training algorithms, Back Propagation (BP), back propagation through time (BPTT), and Estimation-Maximization (EM) algorithm. The Estimation-Maximization (EM) algorithm and the BPTT algorithm work together to solve the problem of BPTT's huge memory requirements that the context layers are unable to meet. As a result, the MCRN network's convergence is speeding up. MCRN is modified based on SRN. We study the performance of this network using various evaluation models such as the mean absolute percentage error (MAPE), the maximum difference (MAX), and accuracy. Experimental results show that MCRN is highly efficient in predicting user behaviour and access patterns when the EM and BPTT algorithms work together.

Keywords: - web log file, Simple recurrent neural network (SRN), Multi–context–recurrent neural networks (MCRN), Access patterns.

I. INTRODUCTION

In the last two decades, modern information systems and digital technologies have produced huge repositories of terabytes each. When we need to analyze large datasets, new challenges are created for efficient implementation and optimum system performance. When we need to analyze this big data, traditional methods are time-consuming and increase the cost. Data mining (DM) is the process of removing relevant information or patterns from large, non-trivial, implicit, previously unknown data sets, as well as finding patterns and correlations within large data sets to predict outcomes [1]. Data mining can help you determine the most suitable products for many customers by showing you what kinds of customers buy what things (clustering or classification). Use prediction to determine which factors will attract new customers [2]. After detailed analysis and interpretation, the Knowledge Discovery Process, or KDD (See Figure 1) is used to extract useful information from unstructured data [3].

A file that contains information about a user's access to the contents of a website is called a "web log file" so for each user request, there is a specific entry in the log file that it recognizes [4]. Websites, businesses, and other online applications need to be able to understand online user behavior in order to be successful. This will make it easier to configure applications based on user behavior. The dynamic nature of the Web and the complex techniques of user behavior have been studied.

Web mining is considered one of the most critical big data analytic applications. Web mining can be divided into three different types that are Web usage mining, Web content mining, and Web structure mining [4]. The web Usage Mining process consists of three steps: Data Pre-processing, Pattern Discovery, and Pattern Analysis [5]. The purpose of using data pre-processing is to convert the data into useful information from the weblog and then keep only the useful data for analysis [6]. The preprocessing of data includes data cleaning, user identification, and session identification [7].

This work aims to better understand users' behavior while using the Web in order to provide better support during use and to extract useful data from log files in order to provide a user-customizing page. This study presents a framework that uses the recurrent neural network (RNN) model.

The next section presents related works. A Multi-Context Recurrent Neural Network is described in Section 3.Section 4 explains the research methodology and Section 5 explains the results and discussion, finally, we conclude in section 6.

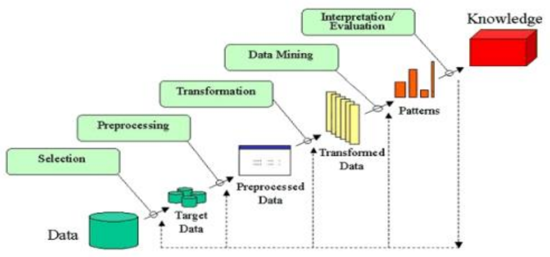


Fig. 1 the KDD process

II. RELATED WORKS

There are several existing research works done on web log file analytic. Reference [8] applied a clustering algorithm to find groups of websites with similar content. As a result of this paper, different segments of the content of Web log data are mined and evaluated by clustering evaluation measures. Reference [9] present a study of web usage mining based on PCA and KNN algorithms to analyze users' navigational patterns. For discovering the hidden pattern, visual clustering was suggested. Reference [10] presents two different clustering techniques, Fuzzy C-Means Clustering algorithms and the Markov model to predict the webpage. The FCM clustering algorithm provides better results than the K-means algorithm in this paper.

In the literature, there are many algorithms used to analyze datasets such as the Artificial Neural Network (ANN) algorithm designed to recognize patterns in data sequences. Reference [11] introduced Recurrent Neural Networks (RNNs), which are a type of neural network and are mainly applied to detect patterns in data. Reference [12] proposed the Elman Recurrent Neural Network (ERNN), also known as Simple RNN. They reviewed five different architectures, ERNN, LSTM, GRU, NARX, and ESN. ERNN is competitive in most tasks and the implementation is simple. Reference [13] proposed a simple recurrent network based on Elman's network. They used three learning algorithms to improve the performance of the Elman network, resulting in a better training network. In back propagation (BP), the error function of the network is minimized by gradient descent. Reference [12] applied Back-propagation through time (BPTT) to train SRN. Back propagation through time (BPTT) builds on BP. The simplest way to deal with a recurrent network is to consider a limited number of iterations used for a finite period [10]. Reference [14] found that BPTT requires a lot of memory, which context layers cannot provide. In order to resolve this problem and speed up the network's convergence during the training phase, the BPTT algorithm and Estimation-Maximization (EM) algorithm were combined [15].The simple recurrent network (SRN) or the Elman network has some limitations in terms of the learning speed of the network and accuracy. However, the SRN architecture has only one small memory context layer (See Figure 2). Assigning hidden layer neurons to output layer neurons, and the cost computation due to the need for more hidden neurons.

A Multi-Context Recurrent Neural Network was developed using several learning algorithms to overcome the above limitations. The newly designed network includes the Multi Context layer (MCL) (See Figure 3) and feed-forward connections from the newly created network are added to the output layer. This will improve network learning [7].

iii. The MULTI-CONTEXT RECURRENT NEURAL NETWORK

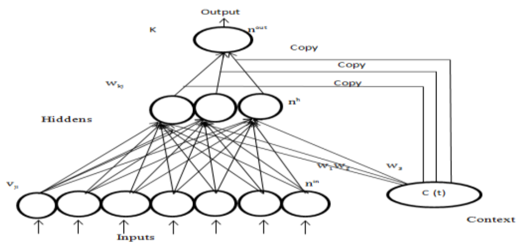


FIG. 2 the Elman network

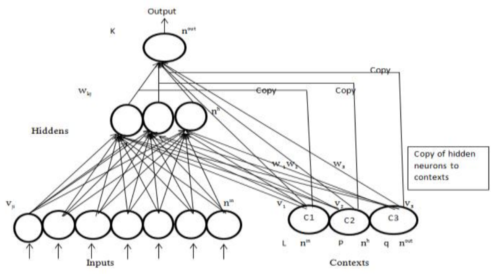


FIG. 3 The multi-context recurrent neural network

A. Basic Notion

The architecture of MCRNN is described using simple terms and symbols.

- *Layers and Neurons*

The indices of input, hidden and output neurons are I, J, and k respectively. The number of input, hidden and output layers aren^{in},n^h andn^{out} respectively. Context layer indices are L, p, and q.

- *Net Inputs and Output*

The neuron's output from the input, hidden, and output layers are I (t), H (t), and O (t). The neurons in the input, hidden, and output layers are copies of the previous time steps in C(1), C(2), and C(3), respectively.

- *Connection the Weights*

v_{ji} The weight connection from the input layer to the hidden layers v_1,v_2 and v_3 are the weight connection from the context layers to the hidden layer. w_1,w_2 and w_3 are the weight connections from the context layers to the output layer, w_{kj} is the weight connection from the hidden layer to the output layer.

B. Network Dynamics

According to how the MCRN works, at time zero, the input layer accepts input pattern values ranging from 0 to 1. The input and context layers activate the neurons in the hidden layer first, and then the hidden and context layers activate the neurons in the output layer. By removing the old context layer output and adding the output of the hidden layer to the first context layer, the context layer is updated. The target is compared to the output layer, and any errors are sent back through the network layers to change the connection weights. The subsequent time step includes the same procedure repeated.
Assuming that B is the overall amount of context layers and b is the number of active context layers (a layer responsible for holding data from the previous time step), then b can be written in Equation 1.

$$b = \begin{cases} t & \text{when } t < b \\ B & \text{when } t >= b \end{cases}$$

Where,
B: The overall number of context layers.
b: The number of active context layers.
First, we define the output of the hidden and context layers before calculating the network output. This is because the information in the MCRN network is transferred from the input to the output layers. The formula is as follows Equation

- *Output of the Hidden Layer*

$$h_j = \sum_{i=1}^{n} I_i(t)\, vji(t) + \sum_{p=1}^{n} \sum_{j=1}^{n} C_l(t - p)\, w_j^n(t)$$

(2)
Where,
h_j: The Hidden Layer's output

I_i: The input layer

$vji(t)$: The weight connection from the input layer to the hidden layers

C_l: The context layer

$w_j^n(t)$: The weight connection from the context layers To the output layer
p : The active context layer.

- *Output of Context Layers*

$$c_j\ (t – P) = c_j\ (t – p + 1) \quad (3)$$

$$c_j\ (t – 1) = H_j\ (t) \quad (4)$$

Where,
c_j (t – P): The oldest data of context layer

c_j (t – p + 1): The new data of context layer

c_j (t – 1): The hidden layer output is copied to the first context layer

- *Network Output*

$$O_k(t) \sum_{j=1}^{n} Hj(t)\, wk\, j(t) + \sum_{p=1}^{n} \sum_{j=1}^{n} C_l(t - p)\, w_k^n(t) \quad (5)$$

(1)

Where,
$O_k(t)$: The output of the output layer

Hj: The hidden layer

wkj(t): The weight connection from the hidden layer to the output layers

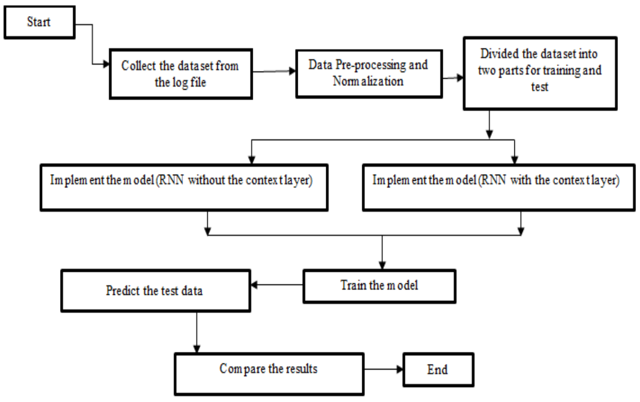
$w_k^n(t)$: The weight connection from the context layers

To the output layers

IV. METHODOLOGY

First, a review of the proposed work related to the development of a user behavior prediction model based on web server log files. Various data mining techniques should be reviewed to identify gaps in the current prediction model for this application.

The structure of the research methodology is shown in Figure 4. The following steps are taken in the processing of the proposed model: data collection; preprocessing and normalization; selection of the training and testing set; selection of the type of network architecture and network parameters; selection of an appropriate learning algorithm; and finally, the implementation model.



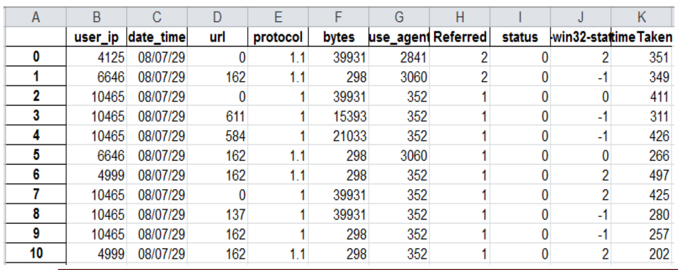
Start
Collect the dataset from the log file
Data Pre-processing and Normalization
Divided the dataset into two parts for training and test
Implement the model (RNN without the context layer)
Implement the model (RNN with the context layer)
Train the model
Predict the test data
Compare the results
End

Fig.4 Research Methodology structure

A. *Dataset Description*

The data was extracted from the Gezira University Web server. The data consists of server logs that record users' online activity. The data source covered the time frame of 7/11/2008 to 10/12/2009. The dataset (logs) for this time period was 66 MB in size and contained 293,236 cases and 15 features. Figure 5 shows the features that were used in the dataset.

Fig. 5 Example of dataset



A	B	C	D	E	F	G	H	I	J	K
	user_ip	date_time	url	protocol	bytes	use_agent	Referred	status	-win32-stat	timeTaken
0	4125	08/07/29	0	1.1	39931	2841	2	0	2	351
1	6646	08/07/29	162	1.1	298	3060	2	0	-1	349
2	10465	08/07/29	0	1	39931	352	1	0	0	411
3	10465	08/07/29	611	1	15393	352	1	0	-1	311
4	10465	08/07/29	584	1	21033	352	1	0	-1	426
5	6646	08/07/29	162	1.1	298	3060	1	0	0	266
6	4999	08/07/29	162	1.1	298	352	1	0	2	497
7	10465	08/07/29	0	1	39931	352	1	0	2	425
8	10465	08/07/29	137	1	39931	352	1	0	-1	280
9	10465	08/07/29	162	1	298	352	1	0	-1	257
10	4999	08/07/29	162	1.1	298	352	1	0	2	202

B. *Data Pre-processing and Normalization*

The noise, missing data, and redundant data must be cleaned up through preprocessing steps. There are three steps in this stage: User identification and session data cleansing [16]. In order to make the Web log data suitable for pattern access and pattern analysis, metadata should be cleaned to remove irrelevant information from the original log file [17]. Our log file contained the following irrelevant entries: entries with a field for the status code, and Delete all records that don't contain the "GET" method. Remove the navigation session made by the robot, spider, and crawler, as well as any appended entries (.js, .css, .gif, .png, .jpg). Unique Visitor Identification is used for user identification (UVI). The IP address and user-agent fields can be used to identify that the unique user is each unique user on the website. Users who use the same IP address are considered the same person. Each different agent will point to a different user if the IP address is the same but the user agent is different.

C. *Data Normalization*

The input string is normalized to values within specific ranges as part of this process to convert the data into a format suitable for network inputs. Different normalization methods exist according to the values of the input data.

D. *Data Scaling*

In the same data sets, before the data can be entered into neural networks, it must be scaled and prepared. Usually, the input/output data is scaled between 0 and 1, or -1 and 1.

E. *Choice of Training and Testing Sets*

The training set is a subset of the trained model. The test set is a subset of the test model. The test set should be large enough to obtain statistically meaningful results. As our goal is to predict the next user's page and get a clear understanding of the user's online behavior we choose Test_ size = 0.15 and Train_ size = 0.85.

F. *Choose the Type of Network*

To solve these limitations in the Elman network, a new network (MCRN), which is updated depending on the

SRN, is introduced with different training algorithms. The network architecture (See Figure 3) adds two additional parts to Elman: A multi-context layer has been added [6]. This can speed up training sessions and maintain a more historical state. More states during training, faster learning. Network learning will also be improved by feed-forward connections from the multi-context layer to the output layer.

G. Learning Algorithms for MCRN

In this paper, we will describe some learning algorithms and how they work with the MCRN architecture. This will enable the updating of MCRN parameters, for example, learning rate, number of hidden layers, and number of context layers. Here we propose three learning algorithms to be used in training and optimizing network architectural parameters.

- *Back Bropagation*

The supervised learning back propagation algorithm (BP) is based on gradient descent error detection [18] .Here, implement the BP learning algorithm for the MCRN. BP includes the following steps: The input is then passed forward to the hidden layer until it reaches the output layer after it is entered into the network. "Forward propagation" is the name of this process. The target output is then compared to the output of the output layer. The error has been calculated. The required modification in the weight connection is then calculated using the difference between the output neuron and the target output. Backward propagation refers to the process by which errors are sent back in this way [19]. The mean square error is denoted as Equation 6.

$$E_s=1/2\sum_{k=1}^{nout}(d_{ks} - O_{ks})2=1/2\sum_{k=1}^{nout}(e_{ks})2 \quad (6)$$

Where,

E_s: The mean square error.

d_{ks} : The pattern goal.

O_{ks}: The actual outcome.

e_{ks}: The Difference between the pattern goal and its actual outcome

The primary goal is to reduce the total mean square error E_{total}, which can be obtained by Equation 7.

$$E_{total} = \sum_{s=1}^{S} E_s \quad (7)$$

Where,

E_{total} : The total mean square error.

- *Back Bropagation Through Time*

A "back propagation" technique called BPTT has been adapted for sequencing. The frequent application of the differentiation sequence principle is the core of back propagation. In theory, this allows the RNN to build a traditional feed forward neural network where we can apply back propagation. Therefore, we use the same notations for the RNN as suggested before [11].
Define the loss function L (O; Y) by passing our input X_T over the network we calculate the hidden state H_T and the output state O_T one step at a time. Equation 8 shows the difference between the output O_T and the target values Y_T.

$$L\,(O;\,Y) = \sum_{t=1}^{T} l(O_T, Y_T) \quad (8)$$

Where,

L (O; Y): The loss function.

O_T: The output.

Y_T: The target values

- *Estimation-Maximization (EM) Algorithm*

The Expectation Maximization (EM) algorithm was introduced to maximize likelihood functions and is defined as the combination of different unsupervised machine learning algorithms [20].
The EM algorithm is an iterative method for processing cycles between two modes. The estimation step, often known as the E step, is the first mode of the attempt to estimate the latent or missing variables. The second mode, also known as the maximization step or M step, attempts to optimize the model's parameters in order to best explain the data.
As an EM algorithm with alternate maximization steps, or as a coordinate descent algorithm See Equation 9.

$$F(q) = E_q[\log l(x;z)] + H_{(q)} \quad (9)$$

Where,

X: The Observed variables

Z: The latent (unobserved) variables

$H_{(q)}$: The distribution's entropy

q: The any arbitrary probability distribution over the unobserved data z

The EM algorithm's steps may be viewed as:

The Expectation step: To maximize F, choose q

$$q^{(t)} = \arg\max_q F(q, \theta^{(t)}) \quad (10)$$

Where,

$\theta^{(t)}$: The estimate of the parameters at iteration t.

The Maximization step: To maximize F, choose θ

$$\theta^{(t+1)} = \arg\max_\theta F(q^{(t)}, \theta) \quad (11)$$

Where,

$\theta^{(t+1)}$: The estimate of the parameters at iteration t + 1.

H. *MCRN Parameters*

There are some very critical parameters when using MCRN, such as the choice of activation function, momentum, and computation complexity.

- *Activation Function*

Neural networks support a wide range of activation functions. Here we use the logistic sigmoid function.

- *Momentum Technique*

As a result, the network's performance will be enhanced because it will be able to disregard small features in the cost function's error surface. The training phase is accelerated using the momentum technique and adaptive learning rate.

- *Complexity Computations*

In a recurrent network, information about previous inputs is stored in the context layer. Network accuracy is usually increased by using more than one context layer, but this also makes the network more complex due to the more weighted connections that need to be calculated and updated.

I. *Selection of MCRNN Structure and Learning Parameters*

Figure 3 explains the multi-context recurrent neural network structure. Here we use different network structures, implemented with different parameters (learning rate, momentum, and the number of training cycles).The first structure consists of 7–3–1–1 (7 input neurons, 3 hidden, 1 context,(10) and one output neuron). The second network structure consists of 7–3–2–1 (7 input neurons, 3 hidden, 2 contexts, and one output neuron), and the third network structure consists of 7–3–3–1 (7 input neurons, 3 hidden, 3 contexts, and one output neuron). All networks were trained for 2000 cycles. The learning rate and momentum of the trained networks were 0.1 and 0.005, respectively.

In this study uses the server side to collect data. There are different types of user behaviors that online businesses and applications should track on a website. We have used the user experience here to make effective decisions regarding visitor retention, personalizing content, and improving the navigation of our website. Data collection is followed by data preprocessing based on the algorithm. Once the preprocessing is complete, the next stage of our algorithm is the building of the model.

To build the model, First, we generated an excel file containing 123064 cases and 8 features (user_ip, use_agent, protocol, bytes, URL, referred, sc-win32-status, and time taken). Second, we analyzed one of the critical features of the web log file (URL felid) in order to predict the following user's page, as well as this felid entry to the model as output and the other felids (user_ip, use_agent, protocol, bytes, referred, sc-win32-status, and time taken) entries to the model as input. Then we have a structure consisting of 7–3–1–1 (7 input neurons, 3 hidden, 1 context, and one output neuron), and also Different network structures are implemented with different parameters. Third, the model was trained with a small amount of data. This training is done in 89999 cases. After training the model by back propagation through time (BPTT), and Estimation-Maximization (EM) algorithm, it can be tested. Then, to predict the next user's page, the input data is entered into the network and passed forward to the hidden layer until it reaches the output layer. Then the target output is compared to the output of the output layer, and an error has been calculated, which requires adjusting the weight connection using the logistic sigmoid function.

The final step was to record the results of the test model with different network structures (a change in the number of context layers). As a result of these tests, we were able to determine the maximum difference (MAX), the mean absolute percentage error (MAPE), and the accuracy of the system in terms of performance. Using the results also helped us to explain how to improve the system and make decisions.

V. RESULTS ANALYSIS

Our experiments were performed on python and Java Net Beans IDE 8.0.2. After the preprocessing step is applied, the data cleaning algorithm reduces the number of records. Table 1 shows the comparison between the number of records before and after data cleaning.

	The Number of records
Before cleaning	293236
After keeping only "GET"	292622
after removing the missing value rows	263341
after keeping only the status =200	223780
after removing irrelevant request	123064
Total No. of Unique Visitors	2710

Table 1 The Statistical report of data cleaning

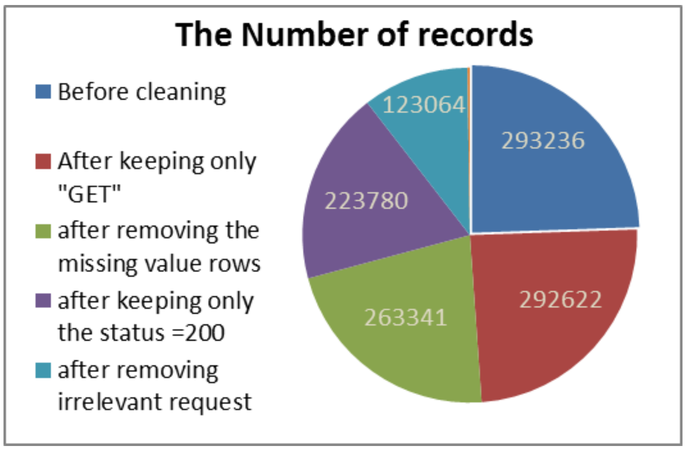


Fig. 6 The number of records before and after data cleaning

After training the model on the training data, we enter the testing data. The model predicts the output, and then we compare the outputs by calculating the errors. Various types of error measures are used to compare the models, such as the maximum difference (MAX), mean absolute percentage error (MAPE), and accuracy.

$$\text{MAP}=\frac{100}{n}\sum_{i=1}^{n}|L_{ri} - L_{Pi}/L_{Ri}| \quad (12)$$

$$\text{MAX}=\max(|L_{Ri} - L_{Pi}|) \quad (13)$$

$$\text{Accuracy} = (100\text{- MAPE})\ \% \quad (14)$$

Where,

N: The predicted number of outputs from the network.

L_{Ri}: The target value

L_{Pi}: The predicted value

Table 2 presents the MCRNN's performance based on different network structures and parameters. The SRN and MCRN structures each contain 7 input neurons, 3 hidden neurons, and 1 output neuron. The MCRN structure also has a different context layer. Learning rates and momentum vary from 0.009 to 0.3 and from 0.3 to 0.7, respectively. Also, the training cycles vary between 18000 and 3000 cycles.

In Table 3, we show the error values for the different network structures in terms of error performance, using mean absolute percentage error (MAPE), the maximum difference (MAX), and accuracy. The results obtained have proven that MCRN can be successfully applied to analyze log files; the accuracy of MCRN was 96.3%, while MAPE was better than the other models by 3.7%.

Table 4 shows the details of predicted values produced by the SRN, MCRN (FI), and MCRN (FII) models.

Table 2 the network structures

Network	SRN	MCRN(FI)	MCRN(FII)
Input layers	7	7	7
Hidden layers	3	3	3
Output layers	1	1	1
Context layer	-	2	3
Learning rates and momentum	0.1&0.005	0.1 & 0.005	0.1 & 0.005
Activation Function	Logistic	Logistic	Logistic
Epochs	2000	2000	2000

Table 3 Forecasting errors for the SRN, FI and FII

Performance	SRN	MCRN(FI)	MCRN(FII)
MAPE%	10.1	4.8	3.7
MAX	269.6	76.8	95.3
Accuracy %	89.9	95.2	96.3
weights	60	69	78
the training Error	4.49853229003E-4	0.0132982931042	0.0061226067176135
the Testing Error	2.46420507160473	2.2902795521E-7	1.0544416811396E-7

Table 4 predicted the values produced by models

SRN	MCRN(FI)	MCRN(FII)	Target
775	755	768	751
726	772	734	703
699	724	690	677
741	698	653	718
762	739	701	738
759	759	721	709
732	730	693	745
769	767	730	749
773	771	734	734
758	755	711	679
701	699	666	748
772	770	734	739
763	761	717	756
780	778	742	763
788	785	749	752
776	774	738	738
762	759	716	699
722	719	678	782
1037	805	768	782
858	805	768	792
869	815	778	801
879	824	786	781
857	804	767	731
802	752	718	708
777	729	695	798
866	812	775	791
876	821	783	776
868	814	777	792
624	799	778	763
818	785	749	743

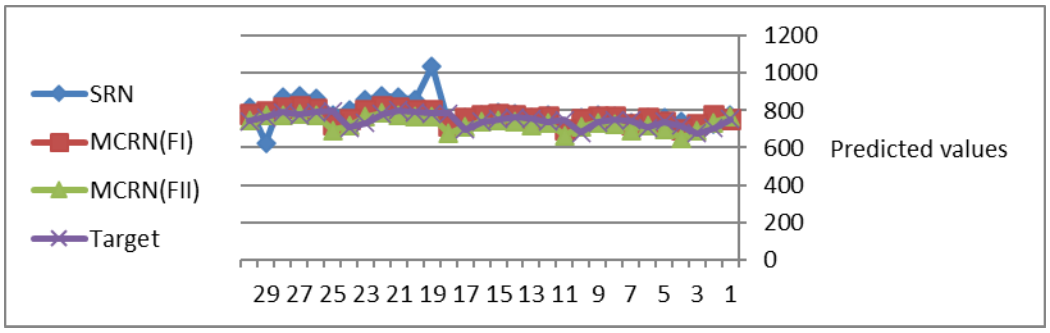


Fig.7 predicted the values produced by the different models.

Table 4 and Figure 7 show that the obtained results prove that MCRN can be successfully applied to analyse log files more efficiently than other models, discovering hidden patterns and using these patterns to develop the user's personalization pages and then predicting the user's next web page.

- *Complexity Computations*

Network accuracy is increased by more than one context layer because of the additional connections; the network will become more complex in terms of the number of connections weights that need to be calculated and updated. The number of hidden layers and the number of connection weights are summarized in Table 5.

Table 5 Explain the number of hidden layers and the number of connection weights.

Hidden(1)	Hidden(2)	weight
10	2	390
11	2	448
12	2	510
13	2	407
14	2	450
15	2	496
16	2	542
17	2	302
18	2	318
19	2	334
20	2	350
21	2	366
22	2	382
23	2	398
24	2	414
25	2	430
26	2	446
27	2	462
28	2	478
29	2	494
30	2	510
31	2	526
32	2	542
33	2	558
34	2	574
35	2	590
36	2	606
37	2	622
38	2	638
39	2	654
40	2	670
41	2	686
42	2	702
43	2	718
44	2	734

• DISCUSSION AND IMPLICATIONS

This work introduces the problem of the dynamic nature of the Web and the complex techniques of user behavior. An understanding of the dynamic nature of the Web and how to analyze the online behavior of users has become essential for the success of websites and other online businesses and applications. It will help to configure the applications based on user behavior. The selection of an appropriate method and technique for these applications is crucial.

In this study, we analyzed one of the critical features of the web log file (URL felid) in order to predict the following user's page and to gain an understanding of how they deal with the dataset's features, as well as this felid entry to the model as output and the other felids (user_ip, use_agent, protocol, bytes, referred, sc-win32-status, and time taken) entries to the model as input. A newly designed network includes a Multi Context layer (MCL) with three different training algorithms: back propagation (BP), back propagation through time (BPTT), and Estimation-Maximization (EM). Because BPTT requires a huge amount of memory that the context layers are not able to meet, the Estimation-Maximization (EM) algorithm is used along with the BPTT algorithm to solve this problem. As a result, the MCRN network's convergence is speeding up.

The results of the analysis showed that the data cleaning algorithm reduced the number of records (See Figure 6). The goal of data preprocessing is to transform web log data into useful information and then use that data for analysis. We have presented the MCRNN's performance based on different network structures and parameters (See Table 2) and explained the various error values for the different network structures, including the mean absolute percentage error (MAPE), the maximum difference (MAX), and accuracy (See Table 3). The results obtained have proven that MCRN can be successfully applied to analyze log files; the accuracy of MCRN was 96.3%, while MAPE was better than the other models by 3.7%.

According to Table 4 and Figure 7, the obtained results demonstrate that MCRN is more effective at analyzing log files than other models; it is more effective in revealing hidden patterns and then using these patterns to create a user's personalization page and then predict the next web page for the user.

In this work, we also discussed complexity computation. The network accuracy is increased by more than one context layer because of the additional connections; the network will become more complex in terms of the number of connections weights that need to be calculated and updated (See Table 5).

To summarize the above discussion, we recommend companies use data mining techniques to analyze other online applications such as handwriting recognition, speech recognition, and weather forecasting. They can use this information to plan for the future.

VI. CONCLUSION

A multi-context recurrent neural network (MCRN) is proposed in this paper to overcome the limitations of the Simple Recurrent Network (SRN), or Elman Network, in terms of network learning speed and accuracy. The MCRNN is trained on Back Propagation (BP), Back Propagation through Time (BPTT), and Estimation-Maximization (EM) techniques to predict the next user's page and obtain a clear understanding of users' online behavior. Understanding users' online behavior has become a necessity for websites. In the proposed model, the data will be collected from the University of Gezira Web server logs. The raw log files should be cleaned up because they contain entries for unrelated things like unsuccessful entries, image access, etc. Preprocessing is the first step used to prepare online usage data for analysis. It involves removing or modifying any unnecessary history or noisy data. The results obtained have proven that MCRN can be successfully applied to analyze log files and produce correct decisions. Convergence is faster when we use the Estimation-Maximization (EM) algorithm with the BPTT algorithm. In terms of predicting user behavior, the accuracy of MCRN was 96.3%, while MAPE was better than the other models by 3.7%. For future work, there are many features that can be applied for prediction that cannot be presented here and are not referred to here. Implement other forecasting methods, such as decision trees, Consider using other complex networks, such as wavelet networks.

REFERENCES

[1] J. F. Pinto da Costa and M. Cabral, "Statistical Methods with Applications in Data Mining: A Review of the Most Recent Works", (2022).

[2] A. S. tiwarkhede and V. Kakde, "A review paper on big data analytics". International Journal of Science and Research, 4(4), 845-848, (2015).

[3] D. Rajeshwari, "State of the art of big data analytics: A survey". International Journal of Computer Applications, (2015).

[4] A. Kumar, V. Ahirwar and R. K. Singh, "A study on prediction of user behavior based on web server log files in web usage mining". International Journal of Engineering and Computer Science, (2015).

[5] J. Umarani, G. Thangaraju and J. Anitha, "Prediction of user behavior in educational web sites by applying web mining". IJCSIT, (2017).

[6] J. Mehra and R. S. Thakur, "An effective method for web log preprocessing and page access frequency using web usage mining", (2018).

[7] K. D. Patel and S. M. Parikh, "Preprocessing on Web Server Log Data for Web Usage Pattern Discovery". International Journal of Computer Applications, (2017).

[8] T. A. Al-Asdi and A. J. Obaid, "An efficient web usage mining algorithm based on log file data", (2016).

[9]A. Vishwakarma and K. N. Singh, "A survey on web log mining pattern discovery". IJCSIT-International Journal of Computer Science and Information Technologies, (2014).

[10] V. R. Rathod and G. V. Patel, "Prediction of User Behavior using Web log in Web Usage Mining" .International Journal of Computer Applications, (2016).

[11] R. C. Staudemeyer and E. R. Morris, "Understanding LSTM--a tutorial into long short-term memory recurrent neural networks", (2019).

[12] F. M. Bianchi, E. Maiorino, M. C. Kampffmeyer, A. Rizzi and R. Jenssen, "An overview and comparative analysis of recurrent neural networks for short term load forecasting", (2017).

[13] T. A. Rashid and M. T. Kechadi, "Multi-Context-Recurrent Neural Network for Load Forecasting".In Artificial Intelligence and Applications, (2005).

[14] B. Q. Huang, T. Rashid and T. Kechadi,"A new modified network based on the elman network". In Proceedings of IASTED International Conference on Artificial Intelligence and Application, ed., MH Hamz, (2004).

[15] B. Q. Huang, T. Rashid and M. T. Kechadi,"Multi-context recurrent neural network for time series applications". International Journal of Computer and Information Engineering, 1(10), (2007).

[16] M. H. Elhebir and A. Abraham, "Discovering Web Server Logs Patterns Using Clustering and Association Rules Mining" .Journal of Network and Innovative Computing, (2015).

[17] A. Sherstinsky, "Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM)

network", (2020).
[18] A. Singh, S. Kushwaha, M. Alarfaj and M. Singh, "Comprehensive Overview of Backpropagation Algorithm for Digital Image Denoising", (2022).
[19] V. Ranganathan and S. Natarajan, "A new backpropagation algorithm without gradient descent", (2018).

[20] T. Lartigue, S. Durrleman and S. Allassonnière, "Deterministic approximate EM algorithm; Application to the Riemann approximation EM and the tempered EM", (2022).