ESP-JETA
ESP Journal of Engineering & Technology Advancements
ISSN: 2583-2646 / Volume 4 Issue 1 March 2024 / Page No: 133-140
Paper Id: JETA-V4I1P119/ Doi: 10.56472/25832646/JETA-V4I1P119

Original Article

AI-Enhanced Natural Language Processing for Improving Web Page Classification Accuracy

Dhruv Patel
Independent Researcher

Received Date: 19 January 2024 Revised Date: 21 February 2024 Accepted Date: 20 March 2024

Abstract: *Since the World Wide Web transformed the way of life, the number of web sites has grown rapidly. Techniques for classifying web pages must be created in order to correctly classify pages according to user queries. When it comes to classifying web pages, the text content is particularly important. They must be properly categorized in order to be used effectively, as the quantity of online sites is growing daily. This paper suggests a new Long Short-Term Memory (LSTM) network model that can better classify web text by capturing its sequential and contextual relationships. The WebKB dataset, consisting of web pages from several academic institutions, is used to evaluate the proposed approach. Features are extracted from structural elements such as titles, headings, body content, and URLs. Data experiments demonstrate the proposed LSTM model delivers an 88.73% overall accuracy, and precision rates reach 89.49%, as well as recall levels hit 88%, and F1-score performance marks 88.84%. The proposed LSTM-based approach delivers performance superior to Decision Trees and CNNs and BERT models and establishes itself as an effective solution for web page classification problems. Text sequences in the model structure enabled effective dependencies identification which led to performance enhancement in classification tasks. Web structure components added to the model made it more accurate and resilient in its performance.*

Keywords: *Web Page Classification, Text Mining, Information Retrieval, Web Mining, Machine Learning, Natural Language Processing, Webkb Dataset.*

I. INTRODUCTION

The digital age has produced explosive website growth which results in massive content quantities spread throughout different knowledge areas. Web page classification needs to progress efficiently as the Internet expands rapidly to improve user experience in search engines and content recommendation systems and targeted advertising systems. The classification of web pages based on content helps users find information efficiently because it enhances their navigation experience simultaneously [1]. Content classification stands as a crucial element for optimizing online advertising because it links appropriate ads to web pages based on their themes to achieve better user engagement and revenue [2].

Techniques for classifying websites can be broken down into two groups: those that use text data and those that use URL data [3]. Web addresses do not provide sufficient contextual understanding for accurate classification according to URL-based methods [4]. Web page content-based analysis techniques capture semantic page meanings directly from page texts, which results in better classification precision [5]. Web content contains valuable data that needs to be utilized for creating dependable classification systems.

Recent Artificial Intelligence advancements made with ML [6] and DL models, which allow to extraction of linguistic patterns from massive datasets and overcome previous technique restrictions, gave a breakthrough to the field of NLP. It has also been documented [7][8] that NLP achieves special importance in such information systems. NLP technology is necessary for it since machines can translate human language into meaningful interpretable meaning [9]. Once, old NLP systems categorized the rules on existing data taken as an input, therefore it did not have the attitude to identify as many languages patterns. By doing this, the web content analysis can be enhanced better and in a more specific way through the use of AI-enhanced NLP with content-based classification techniques. They learn better classification outcomes by the models themselves identifying important textual components.

A. Motivation and Contribution of Paper

With the speedy growth of web content, it has become an essential task for information retrieval, content organization and automatic web management systems to efficiently and accurately classify web pages. Web page content is a first order and highly sequential aspect that is relevant for making sense of parts of the web, but which is difficult for traditional ML models to capture. Together with this, web documents are complex in their structure and available in various formats that further complicate classification systems. However, these limitations motivate the research presented here to investigate the possibility of DL techniques, especially LSTM networks, to improve classification accuracy and reliability of web pages with respect to long-range dependency in textual data. This paper makes the following key contributions:

- The study utilizes the WebKB dataset, which is made up of web pages from different university computer science departments, to classify web pages.
- Implements a robust NLP preprocessing pipeline using NLTK, including stop word removal and stemming, to prepare raw web page content for modeling.
- Extracts features from multiple structural elements of web pages, including titles, headings, body text, and URLs, ensuring a rich representation of content for classification.
- Porter's stemming algorithm is used to break words down to their root form, which makes the feature space more semantically consistent.
- Developed a DL-based web page arrangement model using LSTM.
- Check the model's F1-score, accuracy, precision, memory, and confusion matrix for incorrect information, and use them to your advantage.

B. Justification and Novelty of Paper

Web material is not organized and text depends on its context, making it hard to classify web pages. Often, these subtleties are missed by traditional ML models. If you want to model long-term dependencies in textual data, this paper argues in favor of using an LSTM-based deep learning method. The innovation of the work lies in its integration of structural web features (titles, headings, body, and URLs) with advanced NLP preprocessing and sequence-based modeling, resulting in significantly improved classification performance. The proposed model achieves higher accuracy compared to both classical and other DL methods, demonstrating its robustness and practical relevance.

C. Structure of Paper

The following describes how the document is structured overall. Additionally, Section II gives an outline of Enhanced Web Page Classification. Section III goes into great detail about the method. Section IV compares and contrasts the results, analysis, and conversation. Section V talks about the study's results and suggestions for more research.

II. LITERATURE REVIEW

Some important research on Improved Web Page Classification has been sorted through and studied in order to do this job.

Altaee (2023) used several different methods to find the best way to group Web pages into categories. In the first method, a group of machine-learning methods are used. The job was to sort things, and it checks how well they did with it. Ensemble stacking is the second method, which is used to better group websites into groups. After looking at how well each method performed, they conclude that the second option, Ensemble stacking, was superior at classifying websites. This is evident by its F1-score of 0.95 as well as its accuracy, precision, and memory ratings of 0.95. An F1 score of 0.93, precision of 0.94, recall of 0.93, and accuracy of 0.93 were all achieved by the first method, which made use of ML [10].

Odeh (2023) analyses the way the MNB algorithm is used to find and group computer languages in source code files. There is a method called MNB that is thought to be simple and quick for classifying text. There are a total of 396,090 lines of code in the dataset, which is made up of 12 programming languages and 12,003 lines of code used as examples. To train MNB and test its ability to classify computer language source code, a diverse dataset is used. According to the study, it can correctly identify and group computer languages with a 95.09% success rate [11].

Perdices (2021) the order of domains is like papers, and the words that make them up are like words. The best methods from NLP and NN can be used to turn this problem into a text classification problem. The websites that were seen can then be found with these. With peaks close to 100%, it can tell with over 90% accuracy which websites were visited on purpose after using TF–IDF, various contexts and datasets using Word2vec, Doc2vec, and custom neural networks. On its own, no one is facilitating this [12].

Mulahuwaish (2020) getting specific information based on clear features was seen as a way to deal with the issues mentioned above. There is a good way for the web to organize news stories into four groups, which are business, technology and science, health, and fun, shown in this piece. SVM, kNN, DT, and LSTM are the four ML models they look at and compare these days. Then, accuracy and receiver operating characteristic plots are used to compare them. Each of these classifiers is built on its own. The model kNN was the least accurate (88.72%) and SVM was the most accurate (95.04%) [13].

Zhao, Yang and Hua (2019) introduced in the short feature extraction network to improve the ability of short text information extraction. For longer text, the long feature extraction network takes information from the attention process of the word and the segment. In the last step of classification, the correction mechanism is used to combine models, which makes the classification even more accurate. The test results show that the suggested way is better at classifying things. All of the classification factors for the first-level label were at least 0.94, and all of them for the secondary label were at least 0.90. The combined feature extraction network created in this study is better at reducing noise and classifying things, and it can also get better results in terms of classification accuracy [14].

Markkandeyan and Devi (2015) combined feature selection method can automatically sort Web pages and help users use the online search engine to locate appropriate results. IMFS stands for "intermediate feature set." Improving these traits even more makes the data more accurate. The ML meta-classifier's attribute-selected classifier was used on the IMFS after the FFS was made. When compared to the IFS using the WebKb (Faculty and Course) and ODP (Sports) comparison datasets, the accuracy went up to 97% and the time it took to run all models decreased. When compared to the current versions, the suggested method sorts Web documents more accurately, takes up less room, and speeds up searches [15].

Table 1 shows the background study of Advanced Natural Language Processing for Improving Web Page Classification Accuracy, including its dataset, models, performance, and input.

Table 1: Overview of Literature Studies on Improving WEB Page Classification Using Machine Learning

Author	Proposed Work	Dataset	Key Findings	Challenges/Gaps
Odeh (2023)	The Multinomial Naïve Bayes (MNB) method sorts computer languages into groups.	Dataset of 12 programming languages (12,003 samples, 396,090 lines of code).	Achieved 95.09% accuracy in finding and classifying programming languages.	May struggle with code similarities between languages and lack adaptability to new languages.
Perdices (2021)	Website identification using text classification techniques (TF-IDF, Word2Vec, Doc2Vec, neural networks).	Various datasets with different representation methods.	Accuracy over 90%, peaking close to 100%, with fully automated processing.	Performance may vary across different website structures and real-world scenarios.
Mulahuwaish (2020)	ML-based web news classification using SVM, kNN, DT, and LSTM.	News dataset categorized into Business, Tech & Science, Health, and Entertainment.	SVM performed best (95.04%), while kNN had the lowest accuracy (88.72%).	Need for real-time adaptability to new topics and news trends.
Zhao, Yang and Hua (2019)	Feature extraction network using attention mechanism and model fusion for text classification.	Various datasets for short and long text classification.	Classification accuracy ≥94% (first-level) and ≥90% (second-level), with strong anti-noise ability.	Performance may depend on dataset characteristics and parameter tuning.
Markkandeyan and Devi (2015)	A hybrid feature selection method (IMFS to FFS) for automatically categorising web pages.	WebKB (Faculty & Course) and ODP (Sports) datasets.	Accuracy improved up to 97%, reducing computational time and space requirements.	Effectiveness on large-scale datasets and dynamic web environments remains unexplored.
Odeh (2023)	employing the Multinomial Naïve Bayes (MNB) method to arrange computer language sets.	Dataset of 12 programming languages (12,003 samples, 396,090 lines of code).	Achieved 95.09% accuracy in finding and classifying programming languages.	May struggle with code similarities between languages and lack adaptability to new languages.

II. RESEARCH METHODOLOGY

The methodology for improving web page classification comprises various steps and phases that are shown in Figure 1. First, data collection is conducted by selecting web pages categorized into "Faculty," "Course," and "Student." After that, the NLTK toolkit is used for data preparation. Stop word removal and breaking are used here to make the text clean and correct. Later, phrases are taken from key HTML elements such as the web page's title, headings, body text, and URLs as part of feature extraction. The collected terms are then broken down into their root forms using Porter's stemming method. Following the cleaning, the dataset is split in two: 70% is used for training and 30% is used for testing. After the text data has been split up, an LSTM model is used to find the long-term relationships in it. The input, forget, and output gates in the LSTM design control the flow of data and make sure that important data is kept track of over time. Next, the model's memory, F1-score, accuracy, and ROC curve analysis are checked one last time.

Each step of a proposed flowchart for Improving Accuracy of Web Page Classification is provided below:

A. Data Collection

There are 8,282 web pages in the WebKB dataset. They are grouped into seven groups: others, curriculum, course, project, student, and teacher. It came from IT units at several universities in 1997. Also included are pages from Cornell, Texas, Washington, Wisconsin, and other schools. It is important to note that the 3,764 "other" pages are not main and do not belong to any of the next six groups. Web page classification jobs use the dataset. The data distributes in different classes that illustrated in Figure 2

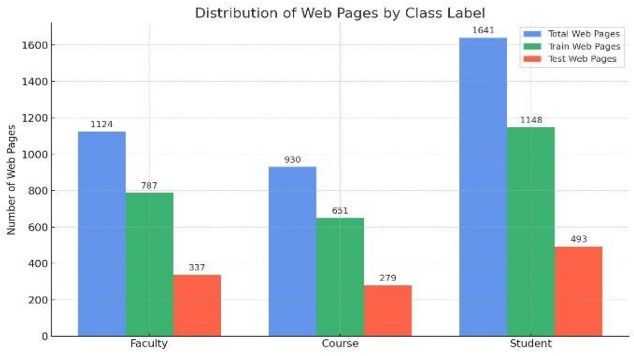

Figure 2 : Plot for Class Distribution

Figure 2 illustrates the number of web pages categorized under three different classes: Faculty, Course, and Student. For each class, the total number of web pages is subdivided into training and testing datasets. The student category contains the highest number of pages (1641 total), followed by Faculty (1124) and Course (930). To prepare a dataset, this distribution is very important for checking the class-wise balance, which has a direct effect on how well and fairly ML models learnt from this data.

B. Data Preprocessing

The first step in NLP is to get the data ready. This must also be done before ML and deep learning can be used. It used the Natural Language Toolkit (NLTK) to do a standard preprocessing method that can be used for many NLP jobs in this step. The steps of pre-processing are listed below:

- Stop Word Removal: Stop word removal in data preprocessing involves identifying and removing common, often meaningless words (like "the," "a," "is") to improve the proficiency and accuracy NLP tasks. This process helps you focus on the important words and get rid of the rest of the data's noise.
- Stemming: Stemming is an NLP technique for preprocessing text that strips words down to their base or root form (stem) by getting rid of their affixes. This makes text easier to use for tasks like finding information and analyzing it.

C. Feature Extraction

In order to extract features from a website, there are several tags employed. Title, header, content, text ease, and URL are all part of it. Not only were the words from these tags stolen, but so were the URLs of the appropriate training set webpages. After the terms are removed, Porter's dividing approach is used. Additionally, the corresponding tag or URL pair for every split sentence. Various pieces of information may be retrieved using the "tagged words" approach whenever the word "program" appears in the title, the "li" tag, or the URL.

D. Data Splitting

Training and testing use 70% and 30% of the information, respectively.

E. LSTM Model

LSTM is great for sorting text because it can recognize how words change over time and how they relate to each other. This classifier is an RNN, a kind of stacking network that feeds the output of one layer into the output of the next. Thanks to its feedback linkages, LSTM is able to process chains of data as well as individual data points. Every long short-term memory (LSTM) node has four gates: an input gate, an output gate, a forget gate, and a cell. Data flows through the cell according to the three gates, and the cell stores values that have been stored throughout time. Each of the LSTM layers is made up of three multiplicative gates and a memory block that is linked to other memory blocks over and over again [16]. Gates make sure that temporary data is used for a certain amount of time by continuously writing, reading, and resetting. As shown in Equations (1) through (6), the unit's inputs $\mathrm{x_t}, \mathrm{h_{t-1}}, \mathrm{c_{t-1}}$ and outputs $\mathrm{h_t}$, $\mathrm{c_t}$ are changed.

$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \quad (1)$$

$$f_t = \sigma\left(W_f x_t + U_i h_{t-1} + b_f\right) (2)$$

$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) (3)$$

$$g_t = tanh\left(W_g x_t + U_g h_{t-1} + b_g\right) \quad (4)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g_t \quad (5)$$

$$h_t = o_t \odot \tanh(c_t) \quad (6)$$

The logistic sigmoid function is represented by σ and element-wise multiplication is shown by $\odot$ in this equation. The LSTM unit's input gate is present at time step t.$\mathrm{i_t}$, a forget gate $\mathrm{f_t}$, an output gate $\mathrm{o_t}$, a hidden unit htt, and a memory cell $\mathrm{c_t}$. To illustrate, W and U represent the components that have been taught, while b denotes the additional bias. Each unit's update rate is modulated by three gates: the forget gate, which deletes data from the memory cell, the output gate, which displays the internal memory state, and the input gate, which updates data.

F. Evaluation Metrics

To make predictions, this is the last step. Use different evaluation tools, like the confusion matrix, F1-score, and classification accuracy, to look at the forecast results now. The statistical numbers that each measure is based on are shown in a confusion matrix. There are times when you can use a confusion grid:

- TP - True Positive: Refers to the positive tuples that the classifier correctly named.
- FP - False Positive: This term refers to the good tuples that the algorithm put in the wrong group.
- FN - False Negative: These are the negative tuples that the classifier put in the wrong group.
- TN - True Negative: Describes the negative tuples that the classifier correctly named.

Accuracy: A, which stands for "accuracy," is the number of right predictions compared to all the predictions in the testing dataset. The following Equation (7) is used to figure out the accuracy:

$$Accuracy = \frac{\mathrm{TP+TN}}{\mathrm{TP+Fp+TN+FN}} \quad (7)$$

Precision: Precision, on the other hand, is the number of right predictions for a certain class of activities compared to the total number of predictions for that class in the testing dataset. This Equation (8) is used to figure out precision:

$$\text{Precision (P)} = \frac{\mathrm{TP}}{\mathrm{TP+FP}} \quad (8)$$

Recall: an indicator of how many correctly identified positives of a certain class there are compared to the total number of true class events in the test dataset. Recall can be found using Equation (9):

$$\text{Recall (R)} = \frac{\mathrm{TP}}{\mathrm{TP+FN}} \quad (9)$$

F1 score: The balanced F1-measure is another name for the F1-score. It thinks about both false negatives (FN) and false positives (FP). In this case, the F1-score is a better way to evaluate something than the accuracy measure. It comes out as Equation (10):

$$\mathrm{F1-score} = 2 \times \frac{\mathrm{Precision} \times \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}} \quad (10)$$

ROC Curve: The TPR times the FPR is what it's measured as. It puts the FPR and the TPR, which is also known as memory or sensitivity, next to each other at different starting points:

All of these metrics together show how well and how accurately the model can predict the goal variable.

II. RESULTS AND DISCUSSION

This section displays the test results for the proposed Enhanced Web Page Classification. Research requires a system with 16 GB of RAM and an Intel Xenon E5-2643 CPU running at 3.30 GHz. The four measurement metrics—accuracy, precision, recall, and F-measure—can be used for experimental analysis. Table II displays the results of the proposed LSTM model. With a lot of correct guesses, the model was 88.73% accurate. Its accuracy of 89.49% shows that most of the positive classifications were correct, and its recall of 88% shows how well it could find relevant web pages. The model's general reliability and robustness in classifying web pages is shown by its F1-score of 88.84%, which is a good balance between accuracy and recall.

Table 1: Experiment Results of Proposed Models for Improving WEB Page Classification

Performance Matrix	LSTM
Accuracy	88.73
Precision	89.49
Recall	88
F1-score	88.84

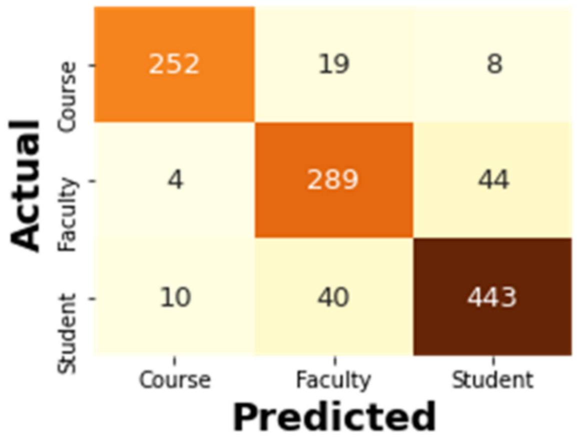

Figure 3 : Confusion Matrix for LSTM

Figure 3, there is a confusion matrix that measures how well a classification model can guess whether a piece of text is in the "Course," "Faculty," or "Student" group. The categories that are actually there are shown in the rows, and the expected categories are shown in the columns. The numbers on the diagonals show how many cases were correctly put into each category: It correctly identified 252 "Course" texts as "Course," 289 "Faculty" texts as "Faculty," and 443 "Student" texts as "Student." Out of the cross.

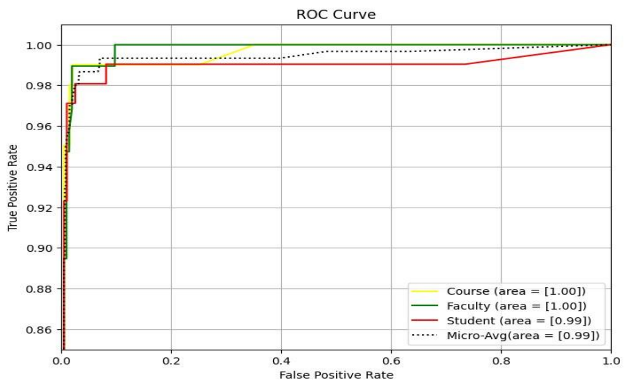

Figure 4 : ROC Curve for LSTM

The predictor did well in all three groups shown in Figure 4, with AUC values of 1.00 for Course, 1.00 for Faculty, and 0.99 for Student. The AUC for the micro-average ROC curve, which adds up the effects of all the groups, is also 0.99. The TPR is plotted against the FPR on the ROC curve. This shows how sensitivity and specificity are related. The high AUC numbers show that all classes were correctly classified.

G. Comparison With Discussion

This section shows a contrast between ML and DL models for correctly classifying web pages using the WebKB dataset shown in Table III. A slightly lower accuracy (79% for the BERT model based on deep learning) was achieved by the ML model using DT. While traditional models did better, deep learning models like CNN and LSTM did even better. CNN got the best score of 88.14%, while LSTM got the highest score of 88.73%. Finally, it demonstrates that deep learning, especially LSTM, is very good at handling complicated patterns and relationships for classifying web pages.

Table 2 : Comparison Between ML And DL Models For Improving WEB Page Classification On

Models	Accuracy
BERT [17]	79
DT [18]	83.19
CNN [19]	88.14
LSTM [20]	88.73

A traditional ML model that employs an LSTM for web page classification has several advantages over traditional methods for web page classification. The model successfully captures long-term dependencies in textual data in order to increase classification accuracy to 88.73% on the WebKB dataset. On the other hand, combining structural features from the web pages (titles, headings and URLs) makes the semantic understanding of the content better which helps to improve the classification performance. Two key features of the LSTM that help it more than look as simple web page structures are that it is capable of modeling sequential relationships in text, which more shallow feature extraction and bag of words models are not.

III. CONCLUSION AND FUTURE STUDY

Web page describing is a very important part of managing and finding information. It needs to pick the right features for Web Page classification so that it is done right. There are many features in web page and the more the number of features decreases the classification accuracy. The aim of this paper is to present an LSTM-based model for web page classification with an accuracy of 88.73% in the WebKB dataset. The suggested approach does a better job of capturing the sequential relationships in the text than traditional ML algorithms and other DL models like CNN and BERT. However, the model's performance is constrained by the quality and diversity of the dataset, and the preprocessing steps may limit its generalization to web pages with highly varied structures or noisy content. Future work will focus on integrating more robust data preprocessing techniques, leveraging pre-trained models such as BERT, and exploring the scalability of the model for real-time classification tasks in large-scale web environments.

IV. REFERENCES

[1] A. Razali, S. M. Daud, N. A. M. Zin, and F. Shahidi, “Stemming text-based web page classification using machine learning algorithms: A comparison,” Int. J. Adv. Comput. Sci. Appl., 2020, doi: 10.14569/ijacsa.2020.0110171.

[2] L. Deng, X. Du, and J. Shen, “Web Page Classification Based on Heterogeneous Features and a Combination of Multiple Classifiers,” Front. Inf. Technol. Electron. Eng., vol. 21, no. 7, pp. 995–1004, Jul. 2020, doi: 10.1631/FITEE.1900240.

[3] S. H. Apandi, J. Sallim, R. Mohamed, and N. Ahmad, “Automatic Topic-Based Web Page Classification Using Deep Learning,” Int. J. Informatics Vis., 2023, doi: 10.30630/joiv.7.3-2.1616.

[4] E. Baykan, M. Henzinger, L. Marian, and I. Weber, “Purely URL-based Topic Classification Categories and Subject Descriptors,” Proc. 18th Int. World Wide Web Conf. (WWW 2009), 2009.

[5] E. Baykan, M. Henzinger, L. Marian, and I. Weber, “A comprehensive study of features and algorithms for URL-based topic classification,” ACM Trans. Web, 2011, doi: 10.1145/1993053.1993057.

[6] K. K. Nimavat and R. Kumar, “Updating Machine Learning Training Data Using Graphical Inputs,” 17178360, 2022

[7] S. Perazzoli, J. P. de Santana Neto, and M. J. M. B. de Menezes, “Systematic analysis of constellation-based techniques by using Natural Language Processing,” Technol. Forecast. Soc. Change, 2022, doi: 10.1016/j.techfore.2022.121674.

[8] W. Zhang et al., “Neuro-Inspired Language Models: Bridging the Gap between NLP and Cognitive Science,” 2023.

[9] T. B. Lalitha and P. S. Sreeja, “Potential Web Content Identification and Classification System using NLP and Machine Learning Techniques,” Int. J. Eng. Trends Technol., 2023, doi: 10.14445/22315381/IJETT-V71I4P235.

[10] H. Altaee, “Webpage Classification Using Ensemble Machine Learning,” Iraqi J. Intell. Comput. Informatics, 2023, doi: 10.52940/ijici.v2i1.27.

[11] A. H. Odeh, M. Odeh, H. Odeh, and N. Odeh, “Using Natural Language Processing for Programming Language Code Classification with Multinomial Naive Bayes,” Rev. d’Intelligence Artif., 2023, doi: 10.18280/ria.370515.

[12] D. Perdices, J. Ramos, J. L. García-Dorado, I. González, and J. E. López de Vergara, “Natural language processing for web browsing analytics: Challenges, lessons learned, and opportunities,” Comput. Networks, 2021, doi: 10.1016/j.comnet.2021.108357.

[13] A. Mulahuwaish, K. Gyorick, K. Z. Ghafoor, H. S. Maghdid, and D. B. Rawat, “Efficient classification model of web news documents using machine learning algorithms for accurate information,” Comput. Secur., 2020, doi: 10.1016/j.cose.2020.102006.

[14] Q. Zhao, W. Yang, and R. Hua, “Design and research of composite web page classification network based on deep learning,” in Proceedings - International Conference on Tools with Artificial Intelligence, ICTAI, 2019. doi: 10.1109/ICTAI.2019.00219.

[15] S. Markkandeyan and M. I. Devi, “Efficient Machine Learning Technique for Web Page Classification,” Arab. J. Sci. Eng., vol. 40, no. 12, pp. 3555–3566, Dec. 2015, doi: 10.1007/s13369-015-1844-1.

[16] R. Tarafdar and Y. Han, “Finding Majority for Integer Elements,” J. Comput. Sci. Coll., vol. 33, no. 5, pp. 187–191, 2018.

[17] A. Gupta and R. Bhatia, “Ensemble approach for web page classification,” Multimed. Tools Appl., 2021, doi: 10.1007/s11042-021-10891-3.

[18] S. Choudhury, T. Batra, and C. Hughes, “Content-based and link-based methods for categorical webpage classification,” pp. 1–9, 2018.

[19] V. Gokula Krishnan, J. Deepa, P. Venkateswara Rao, and V. Divya, “Web Page Classification Based on Novel Black Widow Meta-

Heuristic Optimization with Deep Learning Technique," in Lecture Notes on Data Engineering and Communications Technologies, 2022. doi: 10.1007/978-981-19-2347-0_15.

[20] A. K. Nandanwar and J. Choudhary, "Semantic features with contextual knowledge-based web page categorization using the glove model and stacked bilstm," Symmetry (Basel)., vol. 13, no. 10, 2021, doi: 10.3390/sym13101772.