



Optimising Student Internet Navigation: A Comparative Analysis of Machine Learning Algorithms for Action Prediction

Omar Zammit, Serengul Smith, Clifford De Raffaele
Middlesex University Faculty of Computer Science,
London

Abstract

Web-based learning has been promoted in education and students are required to retrieve online information to complete their assignments and study for exams. Research shows that challenges exist during information retrieval, especially with novice students. In this research, we aim to lessen these challenges by introducing a collaborative framework that gathers students' searched keyphrases and analyses trends to predict the most effective subsequent keyphrase to search. The proposed solution encourages students to contribute by sharing their information retrieval trends while collectively benefiting from each other's searching strategies. In addition, novice students will enrich their domain knowledge since the prediction results contain keyphrases searched by students from previous cohorts. Next-word prediction is a well-known area of NLP that is used to forecast the next word given a sentence or predict trends based on time-series data. Word suggestions are popular in mobile devices and studies show that users rely on them while they are typing. The methodology involves the implementation of a framework designed to collect online browsing activity. Undergraduate students studying a BSc in Computer Science were engaged to participate in an experiment wherein they installed a Google Chrome extension capable of collecting data and predicting suitable content related to the researched domain. The collected data consisted of URLs containing keyphrases that students searched during their studies. A feature engineering process was performed to analyse and transform the data into a time-series sequence of actions and to ensure that it is fit for the intended purpose. A grid-search method was employed on various machine learning models to identify the most effective hyper-parameters that can predict the next best keyphrase. The results obtained during an in-class test show that students relying on the predictions generated by the machine learning models outperformed those who depended solely on the Internet.

Keywords: Next best action prediction, Internet activity monitoring, Hyper-parameters tuning and Enhancing learning experience.

1. Introduction

Web-based learning is increasingly gaining popularity in education. Students now have access to a wealth of online information and educational resources, which assist them in completing assignments and preparing for assessments [1]. However, searching for online information requires an effective strategy and some argue that such strategies present challenges, particularly for novice students who lack the domain knowledge required to build effective search queries [2]. Keyphrases searched by students in queries across cohorts are not retained and are forgotten over time. Our research suggests that collecting such data and transforming it into a time-series format, alongside a next-word prediction model, can assist students who are novices to a domain to be exposed to new keyphrases and also enrich the teaching experience for educators.

The methodology adopted in this research included the implementation of a data collection framework that is capable of collecting Internet activities and browsing data [3]. The framework was distributed among students studying a BSc in Computer Science to generate a dataset and evaluate the effectiveness of the proposed approach. The dataset was transformed and organised into a sequence of actions and used in an experiment that aims to identify the best machine learning algorithms and hyper-parameters to use to predict the next best action. The best models were implemented within the data collection framework and used during an in-class test by students to measure their effectiveness. The results show that students relying on the proposed Google Chrome Extension and the predicted results performed better considering the short time that was allowed. In addition, the data collected during the session can be used by lecturers to perform data analytics on the keyphrases searched by their students. Following a brief review in Section 2 on related work and existing technologies, the paper presents in detail the data processing and transformation methodology used in Section 3.



Section 4 explains the model selection process while Section 5 discusses the evaluation results obtained during an in-class test. Lastly, a conclusion is drawn in Section 6.

2. Background

Next word prediction is an application of machine learning within Natural Language Processing (NLP) that is capable of predicting the next most suitable word in a sentence based on the context and preceding words [4]. The more words one has to train the machine learning model the more accurate the predictions will be [4]. Such algorithms are very commonly used while typing, especially on mobile devices. In contrast, time-series predictions are defined as applications that rely on historical data to build models capable of forecasting future values and trends [5]. Such data can be collected from any device that generates data and can be used in various domains, from financial market prediction to weather predictions [6]. Studies show that when users rely on next-word predictions, they perceive that they require less effort to type thus improving their typing experience [7]. In their study, Lehmann et al. [7] outlined that even incorrect predictions might be useful it was observed that some users select incorrect predictions and then manually correct them to match their intended keyphrase. In addition, the authors outlined that there are various selection strategies that users adopt while using systems that predict text. Users can be divided into two main categories. *Non-Suggestion Category* - users who never rely on the suggestion but use it as a source to create their keyphrase. In contrast, *Suggestion Category* - are users that rely on the suggestions more frequently. Walsh et al. [8] recommended a structure to report machine-learning-based analysis applied to biological studies. They divided their recommendations into four main topics; Data, optimization, model and evaluation. A similar structure was suggested by Rathee and Yede [4], the authors outlined that a data preparation process is composed of; gathering data and transforming it into sequences, model designing and training, predicting the next best word and measuring the accuracy of the model. The model parameters are fine-tuned and the process is repeated until the desired accuracy is obtained.

As suggested by Zammit et al. [9], data collection can be accomplished through a Google Chrome Extension utilizing JavaScript event listeners to gather browsing data, which is then sent to a remote server for additional processing over Hypertext Transfer Protocol (HTTP). Such data can then be processed and converted into a time-series format that is then used by machine learning algorithms to predict the next word. Various studies outlined different machine learning algorithms that can predict the next word based on time-series data. Rathee and Yede [4] worked with Long-Short Term Memory (LSTM) and Bidirectional LSTM (BiLSTM) models to predict the next word. Their research showed that the BiLSTM model was better than LSTM since it can capture past and future information. In fact BiLSTM scored an accuracy of 85% as opposed to 57% scored by LSTM. LSTM can model natural languages effectively since they can remember information over a sequence of tokens and produce the probability of the next word in a sequence. There is a study that uses LSTM indicating its potential to predict the next word (code syntax) for the Java programming language [10]. Random forests also reported promising results, such an algorithm is an ensemble model that consists of many decision trees. In Liu et al. [11] it was outlined that the approach taken by this machine learning algorithm avoids overfitting and improves stability. In addition it was also mentioned that researchers are using Random forests to solve regression problems. In contrast, Decision trees are designed to traverse from the root to its leaves, their main advantage is that they are simple to interpret and they can scale to solve bigger problems involving large datasets. Liu et al. [11] also outlined that K-Nearest Neighbor (k-NN) can be used since it is independent from pre-trained models and can address classification and regression problems. Support vector machines have also been used to predict datasets that have linguistic terms [6]. Such an algorithm separates data points by introducing hyperplanes and they exhibit good performance in solving classification problems.

3. Data Acquisition and Preprocessing

The proposed framework is composed of two main components; a Google Chrome Extension and a remote server. The extension is installed in Google Chrome Browser and is capable of collecting the URLs visited and sending them to the remote server. In contrast, the remote server can extract the keyphrases searched in search engines by the extension user from the received URL. Having a sequence of keyphrases, the server uses machine learning to predict the next best keyphrase to search based on historical searches. The predicted results are collected back and displayed in the extension. As shown in Figure 1, the collaborative framework can be used by both lecturers and students. Lecturers will contribute with keyphrases searched while preparing for the lecture or



structuring assignments. In contrast, students will receive the predicted results based on past data and contribute to future students.

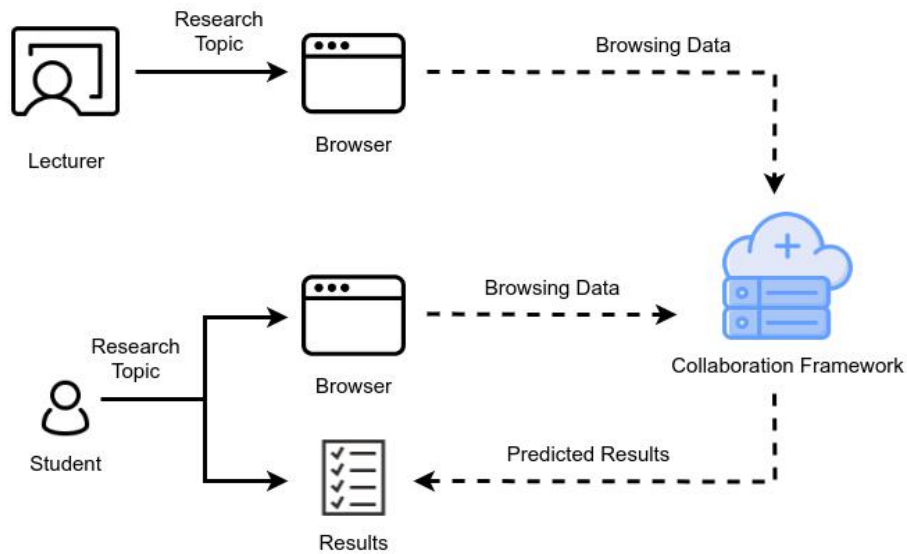



Figure 1 Collaborative Framework Overview

While browsing the Internet students are engaged in various web search sessions. A web search session starts when a student issues a query to find online information, the query is processed by a remote server and the result may surface on a website [12]. Student then will validate the result and visit the website respectively. Students are engaged with such sessions to find information online while studying, they start with a query and can continue formulating different queries until the required result is obtained [13]. All actions take the form of URLs and the Google Chrome extension that was implemented in this research had the capability of collecting such data and storing it on a remote server. Students who volunteered to participate in this research were encouraged to use the Google Chrome extension for a month during their studies and the data collected was arranged into sequences of actions. A sequence S is a collection of actions done by a student within an hour and is defined as $S = \{a_1, a_2, a_3, \dots, a_n\}$ and $a_n \in A_s \cup A_v$ where a_n can be either a keyphrase search A_s in a search enabled websites or a visit to a website A_v . The three main search-enabled websites A_s that were identified during the experiment are Google, Stackoverflow, and Wikipedia. Keyphrases searched by students are embedded within the URL or included in the URL query string as shown below:

 <https://www.google.com/search?q=machine+learning>

 <https://stackoverflow.com/search?q=machine+learning>

 https://en.wikipedia.org/wiki/Machine_learning

Table 1 shows an example of a sequence generated by a student in an hour. The *Sequence Key* is a string composed of the student's unique identifier and the timestamp when the action happened. While the *Action Data* contains the keyphrase searched or the URL visited by the student, determined in the *Action Type*.

Sequence Key	Action Data	ActionType
22fc4e4e_2023_8_24_18	machine learning	Keyphrase Searched
22fc4e4e_2023_8_24_18	artificial intelligence	Keyphrase Searched
22fc4e4e_2023_8_24_18	data science	Keyphrase Searched



22fc4e4e_2023_8_24_18	supervised learning	Keyphrase Searched
22fc4e4e_2023_8_24_18	www.techtarget.com	Visited Link
22fc4e4e_2023_8_24_18	data science	Keyphrase Searched

Table 1 Example of actions performed by a student

Using the Python programming language, a word embedding technique was used to represent each sequence as a vector and build a vector space. This approach enables machine learning algorithms to work with textual data in a way to preserve its context. Various studies show that word embeddings are extremely effective for text classification since they can detect recurring associations in texts [14]. For a supervised machine learning algorithm, the sequence should be divided into features X and labels y . This will allow the algorithm to train the model based on the input features given. In this research, the learning task is trying to predict a sequence similar to time-series forecasting given a context X to predict the action y . Since when dealing with languages, a small vocabulary will fail to represent enough words in the corpus [10], in this research data oversampling was done by duplicating the sequences and changing the starting point. For example, a sequence $S = \{a_1, a_2, a_3, a_4\}$ is divided into subsequences, and their respective features and labels are shown in Table 2. Additionally, since when training machine learning algorithms, the X values of each entry must be of the same size, each vector was left padded with a zero value to the longest vector.

Start	Sequence (s_i)	Input Features (X)	Label (y)
a_1	$\{a_1, a_2\}$	$\{a_1\}$	a_2
a_1	$\{a_1, a_2, a_3\}$	$\{a_1, a_2\}$	a_3
a_1	$\{a_1, a_2, a_3, a_4\}$	$\{a_1, a_2, a_3\}$	a_4
a_2	$\{a_2, a_3\}$	$\{a_2\}$	a_3
a_2	$\{a_2, a_3, a_4\}$	$\{a_2, a_3\}$	a_4
a_3	$\{a_3, a_4\}$	$\{a_3\}$	a_4

Table 2 Example of subsequences

Walsh et al. [8] explained that reporting statistics on the dataset distribution can help understand if there is a good representation of the domain being explored. Figure 2 shows the distribution of labels after transforming the dataset into sequences. Each label l_n on the x axis represents a distinct keyphrase while the y axis shows the total number of occurrences of an instance. Students' search behaviors depends on various factors such as diversity of interests or the popularity of certain topics, these factors can lead to an uneven distribution of the search frequencies among different keyphrases.

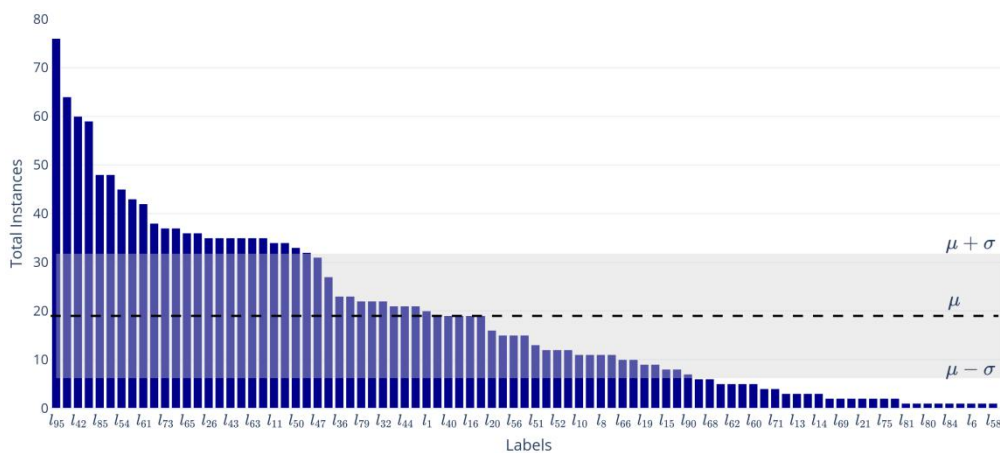


Figure 2 Label distribution showing the area between the upper and lower standard deviation



4. Model Selection

In this research, a variety of machine learning models were taken into consideration. Their implementation was sourced from popular libraries, including scikit-learn [15] for Logistic Regression, Naive Bayes, K-Nearest Neighbors, Support Vector Classifier, Random Forest, and Decision Tree. In addition, the PyRCN [16] implementation of Echo State Network and Extreme Learning Machine was used. TensorFlow [17] was used for the implementation of GRU, and LSTM. Since such libraries are widely used, they provide a standard and efficient implementation that can be easily used and evaluated. The documentation provided for each model was analysed to identify the hyper-parameters that can be fine-tuned relevant to each algorithm.

Liu et al.[11] used various machine learning algorithms such as SVM, Random Forests and k-NN to detect DDOS attacks on software-defined networks. Such machine learning algorithms were used to learn normal traffic patterns, detect anomalies and identify DDOS automatically. In addition, they outlined the importance of feature engineering and they suggest various evaluation matrices such as precision, recall and F1 scores, that can be used to evaluate the quality of machine learning algorithm predictions. In this research, precision (Equation 1) and recall (Equation 2) were used together with their F1 ratio. Where T_P and T_N refers to true positive and true negative results while F_P and F_N refers to false positives and false negatives respectively.

$$Precision = \frac{T_P}{T_P + F_P} \quad (1) \quad Recall = \frac{T_P}{T_P + F_N} \quad (2)$$

To avoid overfitting and underfitting a good choice of optimization strategy is important when selecting a machine learning algorithm [8]. In addition, various studies show that a proper evaluation can be achieved by dividing the dataset into training and testing batches [5], [6], [10]. In this research, the computed dataset was divided into 80% training and 20% testing and scikit-learn *GridSearchCV* was used to perform an exhaustive search over all the parameters values for each machine learning algorithm with a cross-validation of three (3). Table 4 shows the algorithms used and the best hyper-parameters that were identified during the GridSearchCV.

Model Name	Parameters
Dummy Classifier <i>sklearn.dummy</i>	strategy: most_frequent
Logistic Regression <i>sklearn.linear_model._logistic</i>	C: 0.1 penalty: l1 solver: liblinear
Gaussian Naive Bayes <i>sklearn.naive_bayes</i>	priors: None
Extreme Learning Machine Classifier <i>pyrcn.extreme_learning_machine._elm</i>	activation: tanh activation_func: sigmoid alpha: 1.0 chunk_size: 10 n_hidden: 50 ridge_alpha: 0.1 solver: lsqr
K-Nearest Neighbors <i>sklearn.neighbors._classification</i>	n_neighbors: 3 weights: distance
Echo State Network Classifier <i>pyrcn.echo_state_network._esn</i>	alpha: 1.0 bias_scaling: 1.0 input_scaling: 1.5 leakage: 0.8 n_reservoir: 50 sparsity: 0.2 spectral_radius: 0.9
Multi-layer Perceptron <i>sklearn.neural_network._multilayer_perceptron</i>	activation: relu alpha: 0.01 batch_size: 5 hidden_layer_sizes: [100, 50] learning_rate: adaptive max_iter: 1000 solver: sgd
SVC <i>sklearn.svm._classes</i>	C: 1 gamma: 0.001 kernel: poly max_iter: 500
GRU <i>tensorflow.keras.layers</i>	epochs: 200 model__activation: softmax model__layer: GRU model__learning_rate: 0.01 model__loss: mean_squared_error model__recurrent_dropout: 0.25 model__units: 50
LSTM <i>tensorflow.keras.layers</i>	epochs: 200 model__activation: softmax model__layer: LSTM model__learning_rate: 0.01 model__loss: mean_squared_error model__recurrent_dropout: 0.75



	model__units: 50
Random Forest	max_depth: 20 min_samples_leaf: 1 min_samples_split: 2
<code>sklearn.ensemble._forest</code>	n_estimators: 200
Decision Tree	max_depth: 30 min_samples_leaf: 1 min_samples_split: 2
<code>sklearn.tree._classes</code>	

Table 3 Best Hyper Parameters Identified

Once the best hyper-parameters were identified, each model was used to predict the next-best word using the training and testing sequence data. Such a technique facilitates the detection of overfitting and evaluating the effectiveness of the machine learning model. As part of the evaluation process, scikit-learn *DummyClassifier* was used as a baseline classifier. Such a classifier makes predictions using simple techniques such as predicting the most frequent label or generating random predictions. Figure 3 shows the F1 score computed from the precision and recall.

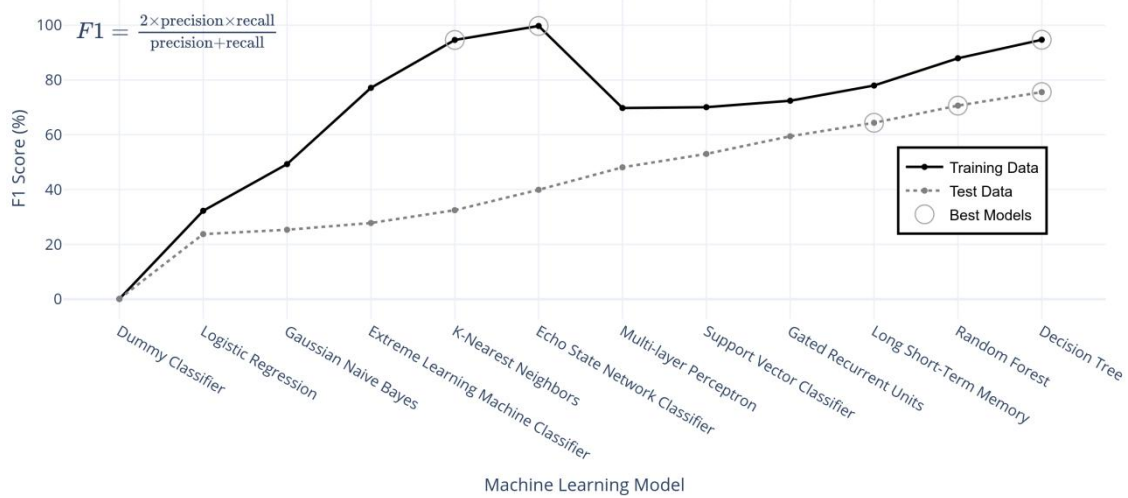


Figure 3 Algorithms evaluation results (F1 Scores)

All trained models scored better than the *DummyClassifier*, suggesting that each model has some element of learning. Scoring was higher when predicting training data since such data was used to train the internal logic of the models. Lower scores were expected when predicting testing data, given that some of the data was unseen during model training. Selection of the best models was based on accuracy and F1 scores on the training data, with GRU, LSTM, and Decision Trees models ranking highest in accuracy, and Random Forest, LSTM, and Decision Trees in F1 scores. The best models were trained and deployed on a remote server, with their prediction results accessible to students' via a Google Chrome Extension (refer to Figure 4). This extension collects browsing data from students' Chrome browsers and sends it as a sequence to the remote server. The server then uses scikit-learn *VotingClassifier* to predict results using all models, returning the most accurate prediction between the models.

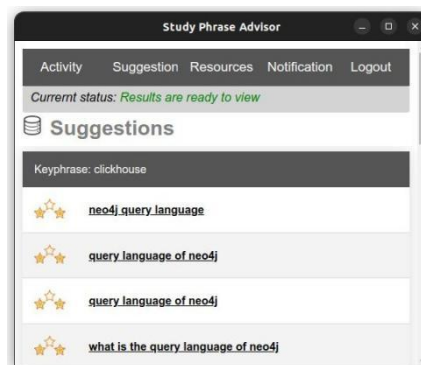


Figure 4 Google Chrome Extension showing next word prediction



5. Evaluation

The evaluation methodology involved eleven (11) undergraduate students studying for a BSc Degree in Computer Science at the University Of Wolverhampton who volunteered to participate in this research. Furthermore, a collaborative session with various lecturers was held to select a topic that students are not familiar with so that it will be used as an assessment during evaluation. ClickHouse <https://clickhouse.com/> emerged as the chosen topic, given its relevance to data science and software engineering projects. An introduction session was held to explain to the students the rationale behind the study and to provide a tutorial on how to install and use the Google Chrome Extension. All students were subjected to a multiple choice pre-test on ClickHouse to void a potential biasing factor and to provide a baseline on the subject knowledge as suggested by De Raffaele et al. [18]. Five (5) students were chosen randomly to serve as a Control group, while the remaining students were assigned to the Evaluation group and instructed to install the Google Chrome Extension. Both groups were instructed to do another multiple-choice test related to ClickHouse having questions structured differently from the pre-test to avoid possible influence. The control group was instructed to use the Internet during the test to find the correct answers while the evaluation group was instructed to use the Internet and the proposed Google Chrome extension suggestions.

$$\text{Mean difference} = \frac{1}{n} \sum_{i=1}^n (X_i - Y_i) \quad (3)$$

n is the number of paired tests done.

Y_i is the pre-test score for paired test i .

X_i is the final test score for paired test i .

The pre-test and the test scores obtained were aggregated and statistical analysis was performed as reported in Table 5. The low pre-test score (\bar{P}) shows that the biasing factor was very low and students had very low knowledge of the test topic (Clickhouse) before the evaluation process. To void the biasing factor, the pre-test score was deducted from the test score and the mean difference ($\bar{P} - \bar{T}$) was computed for all paired tests as outlined in Equation 3. The result shows that students within the evaluation group that relied on the proposed approach performed better (82%) than the control group (52%). Such difference indicates that the proposed approach provided educational advantages to the evaluation group. As reported in Table 5, the independent and the dependent t-statistic values when observing the scores obtained by both the evaluation and the control groups. A $p < 0.05$ denotes that the results obtained have a high statistical significance and are unlikely to have occurred by chance. Thus sustaining that the proposed approach impacted the data observed.

	Total Students	Pre-Test Average Score (\bar{P})	Test Average Score (\bar{T})	Difference ($\bar{P} - \bar{T}$)	Std Dev (σ)	T-statistic (t)
Evaluation	5	2.00%	84.00%	82.00%	19.24	9.53 $p < 0.05$
Control	6	6.67%	58.33%	51.66%	19.41	6.52 $p < 0.05$
Independent						2.59 $p < 0.05$

Table 5 Evaluation Results

6. Conclusion

This research aimed to address some of the challenges faced by students while retrieving online information for educational purposes. By introducing a collaborative framework that harnesses the power of machine learning to predict subsequent keyphrases based on historical search trends, and thus enriching the student's learning experience. The methodology adopted, included data collection, transformation, and model selection, where various machine learning algorithms were evaluated. The best algorithms that were identified included, GRU, LSTM, and Decision Trees based on accuracy and F1 scores. The framework evaluation showed that students relying on the Google Chrome Extension predictions exhibited improved scores when compared to those relying solely on internet searches. This highlights the potential of such an approach to assist students and improve their experience. Further research could explore the evaluation of the framework across diverse educational domains and student populations.



REFERENCES

- [1] M.-J. Tsai, "Online information searching strategy inventory (oissi): A quick version and a complete version," *Computers and Education*, vol. 53, no. 2, pp. 473–483, 2009, doi: <https://doi.org/10.1016/j.compedu.2009.03.006>.
- [2] S. Debowski, "Wrong way: Go back! An exploration of novice search behaviors while conducting an information search," *The Electronic Library*, vol. 19, pp. 371–382, Dec. 2001, doi: [10.1108/02640470110411991](https://doi.org/10.1108/02640470110411991).
- [3] O. Zammit, S. Smith, D. Windridge, and C. D. Raffaele, "Exposing students to new terminologies while collecting browsing search data (best technical paper)," in *International conference on innovative techniques and applications of artificial intelligence*, Springer, 2020, pp. 3–17. doi: [10.1007/978-3-030-63799-6_1](https://doi.org/10.1007/978-3-030-63799-6_1).
- [4] V. Rathee and S. Yede, "A machine learning approach to predict the next word in a statement," in *2023 4th international conference on electronics and sustainable communication systems (icesc)*, IEEE, 2023, pp. 1604–1607.
- [5] L. Shi et al., "An improved robust kernel adaptive filtering method for time series prediction," *IEEE Sensors Journal*, 2023.
- [6] O. C. Yolcu and U. Yolcu, "A novel intuitionistic fuzzy time series prediction model with cascaded structure for financial time series," *Expert Systems with Applications*, vol. 215, p. 119336, 2023.
- [7] F. Lehmann, I. Kornecki, D. Buschek, and A. M. Feit, "Typing behavior is about more than speed: Users' strategies for choosing word suggestions despite slower typing rates," *Proceedings of the ACM on Human-Computer Interaction*, vol. 7, no. MHCI, pp. 1–26, 2023.
- [8] I. Walsh et al., "DOME: Recommendations for supervised machine learning validation in biology," *Nature methods*, vol. 18, no. 10, pp. 1122–1127, 2021.
- [9] O. Zammit, S. Smith, D. Windridge, and C. De Raffaele, "Reducing the dependency of having prior domain knowledge for effective online information retrieval," *Expert Systems*, vol. 40, no. 4, p. e13014, 2023, doi: <https://doi.org/10.1111/exsy.13014>.
- [10] B. Boldt, "Using lstms to model the java programming language," in *Artificial neural networks and machine learning—icann 2017: 26th international conference on artificial neural networks, alghero, italy, september 11-14, 2017, proceedings, part ii 26*, Springer, 2017, pp. 268–275.
- [11] Z. Liu, Y. Wang, F. Feng, Y. Liu, Z. Li, and Y. Shan, "A ddos detection method based on feature engineering and machine learning in software-defined networks," *Sensors*, vol. 23, no. 13, p. 6176, 2023.
- [12] J. Y. Kim, K. Collins-Thompson, P. N. Bennett, and S. T. Dumais, "Characterizing Web content, user interests, and search behavior by reading level and topic," in *WSDM 2012 - proceedings of the 5th acm international conference on web search and data mining*, New York, NY, USA: ACM, 2012, pp. 213–222. doi: [10.1145/2124295.2124323](https://doi.org/10.1145/2124295.2124323).
- [13] A. Usta, I. S. Altıngövede, I. B. Vidinli, R. Özcan, and Ö. Ulusoy, "How K-12 students search for learning? Analysis of an educational search engine log," *SIGIR 2014 - Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 1151–1154, 2014, doi: [10.1145/2600428.2609532](https://doi.org/10.1145/2600428.2609532).
- [14] S. Daenekindt and J. Schaap, "Using word embedding models to capture changing media discourses: A study on the role of legitimacy, gender and genre in 24,000 music reviews, 1999–2021," *Journal of computational social science*, vol. 5, no. 2, pp. 1615–1636, 2022.
- [15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.
- [16] P. Steiner, A. Jalalvand, S. Stone, and P. Birkholz, "PyRCN: A toolbox for exploration and application of reservoir computing networks." 2021. Available: <http://arxiv.org/abs/2103.04807>
- [17] M. Abadi et al., "TensorFlow: Large-scale machine learning on heterogeneous systems." 2015. Available: <https://www.tensorflow.org/>
- [18] C. De Raffaele, S. Smith, and O. Gemikonakli, "The aptness of tangible user interfaces for explaining abstract computer network principles," in *2016 IEEE Frontiers in Education Conference (FIE)*, IEEE, 2016, pp. 1–8.