Towards A Model For Educational Resource Recommendation Based On Cognitive Load

Jamal El Marzouqi[1], Mohamed Erradi[2] and Souhaib Aammou[3]

Ecole Normale Supérieure, Abdelmalek Essaadi University, Tetuan, Morocco

jamal.elmarzouqi@etu.uae.ac.ma

Abstract. The advent of digital learning platforms has necessitated the development of intelligent systems that can provide personalized educational experiences. This paper introduces a model for recommending educational resources based on cognitive load analysis, aimed at optimizing learning efficiency by tailoring content to individual cognitive capacities. The model integrates learner profiles, real-time cognitive load assessment, content analysis, and a recommendation engine utilizing machine learning algorithms. It leverages the principles of Cognitive Load Theory and Information Processing Theory to design educational materials and experiences that align with human cognitive architecture. The implementation of this model in an online learning platform demonstrated significant improvements in learner engagement and learning outcomes. Specifically, the use of Support Vector Machines for cognitive load assessment achieved an accuracy of 89.5%, and the recommendation engine showed high precision and recall in suggesting relevant re-sources. The results indicated an average improvement of 15% in test scores among learners who used the recommended resources. These findings highlight the potential of cognitive load analysis in enhancing personalized learning experiences, suggesting that such models can be instrumental in creating more adaptive and responsive digital educational environments.

Keywords: Cognitive load, Resource recommendation, Data analysis, Machine learning.

1 Introduction

The shift towards digital education has brought forth the challenge of providing personalized learning experiences that cater to the diverse needs of learners. Traditional one-size-fits-all approaches are increasingly inadequate in addressing the variability in learners' cognitive capacities, learning preferences, and prior knowledge [1]. Cognitive Load Theory (CLT) and Information Processing Theory (IPT) provide a theoretical foundation for understanding how learners process information and the factors that influence learning efficiency. Building on these theories, this paper proposes a model for recommending educational resources based on cognitive load analysis. The

M. Khaldi et al. (eds.), Proceedings of the E-Learning and Smart Engineering Systems (ELSES 2024), Atlantis Highlights in Social Sciences, Education and Humanities 36,
https://doi.org/10.2991/978-2-38476-408-2_61

model aims to enhance learning efficiency by providing personalized recommendations that align with each learner's cognitive level. The implementation of this model in an online learning platform and its impact on learning outcomes is explored through a case study. The findings underscore the importance of personalized learning approaches and the potential of cognitive load analysis in improving the effectiveness of digital education.

2 Theoretical foundation

2.1 Cognitive Load Theory (CLT)

Cognitive Load Theory (CLT), developed by John Sweller in the late 1980s, is a framework that has significantly influenced instructional design and educational psychology. The theory is based on the premise that human cognitive architecture comprises a limited-capacity working memory and an unlimited long-term memory. Effective learning occurs when instructional materials are designed to manage cognitive load, which is the amount of mental effort required to process information in working memory.

CLT categorizes cognitive load into three types:

- Intrinsic Cognitive Load: This is the inherent level of difficulty associated with the material being learned. It is determined by the complexity of the content and the prior knowledge of the learner. Intrinsic load cannot be reduced without altering the learning objectives, but it can be managed by breaking down complex information into smaller, more manageable chunks [2].
- Extraneous Cognitive Load: This type of cognitive load is generated by the way information is presented to the learner. It does not contribute to learning and is considered unnecessary. Instructional designers aim to minimize extraneous load by simplifying the presentation of material, avoiding redundant information, and using clear and concise instructions [3].
- Germane Cognitive Load: This is the cognitive load dedicated to processing, constructing, and automating schemas. Germane load is considered beneficial for learning as it involves the active engagement of the learner in meaningful learning activities. Instructional strategies that promote germane load include using worked examples, fostering self-explanations, and encouraging problem-solving [4].

CLT has practical implications for instructional design. For instance, the "worked example effect" suggests that learners benefit from studying worked-out examples rather than solving problems from scratch when learning new concepts. This approach reduces extraneous load and allows learners to focus on understanding the underlying principles.

2.2 Information Processing Theory (IPT)

Information Processing Theory (IPT) is a cognitive psychology framework that explains how humans perceive, process, and store information. The theory is often likened to the functioning of a computer, where information is input, processed, and stored for later retrieval. IPT has been instrumental in understanding cognitive processes involved in learning and memory.

Key components of IPT include:

- Sensory Memory: This is the initial stage of memory, where sensory information is briefly held in its raw form. Sensory memory has a large capacity but a very short duration, typically less than a second for visual information (iconic memory) and a few seconds for auditory information (echoic memory) [5].
- Working Memory (Short-Term Memory): This is where information is actively processed and manipulated. Working memory has a limited capacity, typically able to hold 7 ± 2 items, and a short duration, lasting about 20 seconds without rehearsal. The concept of working memory is central to understanding cognitive load, as it is the workspace where cognitive processing occurs [6].
- Long-Term Memory: This is the storage of information over an extended period, potentially for a lifetime. Long-term memory has an essentially unlimited capacity and duration. It is organized semantically, and information retrieval from long-term memory can influence working memory processes [7].

IPT emphasizes the importance of managing cognitive load to prevent working memory overload, which can hinder learning and information retention. Effective instructional design should aim to reduce extraneous cognitive load and optimize intrinsic and germane loads to facilitate the efficient transfer of information from working memory to long-term memory.

Information Processing Theory has had a significant impact on educational psychology, providing insights into how instructional materials and teaching methods can be designed to align with the cognitive processes involved in learning.

2.3 Personalization in Learning

Personalization in learning is a pedagogical approach that aims to customize educational experiences to meet the unique needs, preferences, and cognitive capacities of individual learners. This approach is grounded in the belief that learners are diverse, and a one-size-fits-all approach to education is not effective for everyone. Personalized learning seeks to optimize learning efficiency and effectiveness by providing tailored instruction and resources.

Key aspects of personalized learning include:

- Learner Profiles: Personalized learning often involves creating detailed profiles of learners, which include information about their prior knowledge, learning preferences, interests, and goals. These profiles help educators and learning systems to tailor instruction and resources to each learner's needs [8].

- Adaptive Learning Technologies: Technology plays a crucial role in personalized learning. Adaptive learning systems use algorithms to analyze learners' interactions and performance to provide customized feedback, recommendations, and content. These systems can adjust the difficulty level, pace, and style of instruction based on the learner's progress and needs [9].
- Competency-Based Learning: Personalized learning often incorporates competency-based approaches, where learners progress based on their mastery of specific skills or knowledge rather than time spent in class. This allows learners to advance at their own pace and ensures that they have a solid understanding of the material before moving on [10].
- Data-Driven Decision Making: Personalized learning relies on the collection and analysis of data to inform instructional decisions. Data on learners' performance, engagement, and learning patterns are used to identify areas of strength and weakness and to provide targeted support [11].

Personalized learning has the potential to transform education by making it more learner-centered, flexible, and responsive to individual needs. However, it also presents challenges, such as ensuring equity and access, protecting privacy, and providing adequate support for educators in implementing personalized approaches.

3 Methodology

3.1 The proposed model

The proposed model for recommending educational resources based on cognitive load analysis aims to enhance learning efficiency by tailoring content to each individual's cognitive capacity. The model operates on the premise that by understanding and optimizing cognitive load, learners can engage with material that is neither too challenging nor too simplistic, thereby facilitating effective learning.

Model Components:

- Learner Profile: The model begins with the creation of a detailed learner profile, which includes information on the learner's prior knowledge, cognitive abilities, learning preferences, and historical interaction data with the learning platform.
- Cognitive Load Assessment: The model employs real-time assessment techniques to measure the learner's cognitive load while interacting with educational content. This can be achieved through direct measures (e.g., self-reported workload) or indirect measures (e.g., performance data, eye tracking, physiological signals).
- Content Analysis: Educational resources are analyzed and categorized based on their intrinsic cognitive load, which is determined by the complexity of the content and the prerequisite knowledge required.
- Recommendation Engine: The core of the model is a recommendation engine that uses machine learning algorithms to match educational resources with the learner's current cognitive load and learning needs. The engine considers the learner's pro-

file, the cognitive load assessment, and the content analysis to recommend resources that are optimally challenging.

By aligning educational resources with each learner's cognitive level, the model seeks to create a personalized and efficient learning experience that maximizes knowledge retention and minimizes cognitive overload.

3.2 K-means Clustering for Learner Profiles:

The use of K-means clustering in creating learner profiles involves segmenting learners into distinct groups based on similar attributes. This approach allows for more targeted and effective recommendations by identifying patterns and commonalities among learners. Here's a detailed explanation:

- Data Collection: Gather data on learners' attributes, such as prior knowledge, cognitive abilities, learning preferences, and historical interaction data with the learning platform. This data forms the basis for clustering.
- Feature Selection: Choose relevant features from the collected data that best represent the characteristics of the learners. For example, you might select features like quiz scores (prior knowledge), time spent on tasks (learning preferences), and completion rates (historical interaction data).
- Normalization: Normalize the selected features to ensure that they are on the same scale. This is important because K-means clustering uses distance measures, and features on larger scales can dominate the clustering process.
- Initialization: Randomly select 'k' initial centroids, where 'k' is the number of clusters you want to create. The choice of 'k' can be guided by domain knowledge or methods like the Elbow method, which helps determine the optimal number of clusters.
- Clustering: Assign each learner to the nearest centroid based on a distance metric (e.g., Euclidean distance). After all learners are assigned, recalculate the centroids by taking the mean of all learners in each cluster. Repeat this process until the centroids stabilize, and the clusters no longer change significantly.
- Segmentation: Once the clustering process is complete, you have distinct groups of learners with similar attributes. These groups can be analyzed to understand common patterns and preferences, which can inform personalized recommendations.
- Updating: Learner profiles are dynamic, so it's important to periodically update the clusters as learners' attributes change over time. This ensures that the segmentation remains relevant and accurate.

By segmenting learners into clusters based on similar attributes, K-means clustering enables the development of more personalized and effective learning experiences, enhancing the overall efficiency of the educational platform.

Python code snippet using the K-means clustering algorithm to segment learners into groups based on their attributes :

```
import pandas as pd
```

```
import numpy as np
from sklearn.cluster import KMeans
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt

# Read data from a CSV file
df = pd.read_csv('learner_data.csv')

# Assuming the relevant columns are named 'quiz_scores',
'time_spent', and 'completion_rates'
learner_data = df[['quiz_scores', 'time_spent', 'comple-
tion_rates']].values

# Normalize the data
scaler = StandardScaler()
normalized_data = scaler.fit_transform(learner_data)

# Determine the optimal number of clusters (k) using the
Elbow method
inertia = []
for k in range(1, 6):
    kmeans = KMeans(n_clusters=k, random_state=42)
    kmeans.fit(normalized_data)
    inertia.append(kmeans.inertia_)

# Assuming the optimal number of clusters is 2 (based on
the elbow method)
kmeans = KMeans(n_clusters=2, random_state=42)
clusters = kmeans.fit_predict(normalized_data)

# Assign learners to clusters
learner_groups = {}
for i, cluster in enumerate(clusters):
    if cluster not in learner_groups:
        learner_groups[cluster] = []
    learner_groups[cluster].append(i)

# Output the groups
for cluster, learners in learner_groups.items():
    print(f"Cluster {cluster}: Learners {learners}")
```

In this code, the pandas library is used to import a CSV file into a DataFrame named df. From this DataFrame, the columns 'quiz_scores', 'time_spent', and 'completion_rates' are extracted and transformed into a NumPy array called learner_data. The StandardScaler is then applied to normalize the data, ensuring that all attributes are on the same scale. To determine the optimal number of clusters, the inertia values for different numbers of clusters (k) are calculated and visualized using the elbow method. The KMeans algorithm from sklearn is subsequently employed to perform clustering with the optimal number of clusters, which is assumed to be 2 in this example. Learners are then assigned to clusters based on their attributes, and the learner groups are printed, showing the cluster to which each learner belongs.

3.3 SVM for Cognitive Load Assessment:

The assessment of cognitive load in real-time is crucial for tailoring educational experiences to individual learners' cognitive capacities. One effective approach is to use machine learning algorithms, such as Support Vector Machines (SVM), to classify cognitive load levels based on physiological signals or interaction data. Here's a detailed explanation of how this can be implemented:

- Data Collection:
  - Physiological Signals: Gather data from sensors that measure physiological indicators of cognitive load, such as heart rate variability, skin conductance, pupil dilation, and EEG (electroencephalogram) signals.
  - Interaction Data: Collect data from learners' interactions with the learning platform, such as response times, click patterns, and navigation paths.
- Feature Extraction:
  - Physiological Features: Extract relevant features from the physiological signals that are indicative of cognitive load. For example, calculate the mean and variability of heart rate or the frequency of specific EEG wave patterns.
  - Interaction Features: Derive features from the interaction data that may reflect cognitive load, such as average response time or the number of page revisits.
- Data Preprocessing:
  - Normalization: Scale the features to a similar range to ensure that no single feature dominates the classification.
  - Labeling: Assign labels to the data based on cognitive load levels. This can be done through subjective measures (e.g., self-reported workload) or objective measures (e.g., task difficulty).
- Model Training:
  - Support Vector Machines (SVM): Train an SVM classifier to distinguish between different levels of cognitive load. SVM is well-suited for this task as it can handle high-dimensional data and is effective in binary and multi-class classification.
  - Cross-Validation: Use cross-validation techniques, such as k-fold cross-validation, to evaluate the model's performance and avoid overfitting.
- Real-Time Assessment:
  - Online Prediction: Deploy the trained SVM model to predict cognitive load levels in real-time based on incoming physiological and interaction data.
  - Thresholding: Set thresholds to categorize the predicted values into different cognitive load levels (e.g., low, medium, high).

- Integration with Learning Platform:
  - Feedback Loop: Use the real-time assessment of cognitive load to adapt the learning content and interface. For example, if a high cognitive load is detected, the system could simplify the content or provide additional support.
  - Continuous Learning: Update the SVM model periodically with new data to improve its accuracy and adapt to changes in learners' cognitive load patterns.

By employing SVM and other machine learning algorithms for real-time cognitive load assessment, educational platforms can provide more personalized and effective learning experiences, enhancing learners' engagement and learning outcomes.

Python code that demonstrates how to use a Support Vector Machine (SVM) classifier for cognitive load assessment based on simulated physiological and interaction data:

```
import numpy as np
import pandas as pd
from sklearn.svm import SVC
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score

# Read the data from the CSV file
data_df = pd.read_csv('cognitive_load_data.csv')

# Separate the features and labels
features = data_df.drop('cognitive_load_level', ax-
is=1).values
labels = data_df['cognitive_load_level'].values

# Splitting the data into training and testing sets
X_train, X_test, y_train, y_test =
train_test_split(features, labels, test_size=0.33, ran-
dom_state=42)

# Standardizing the data
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Training the SVM classifier
svm_model = SVC(kernel='linear')
svm_model.fit(X_train, y_train)

# Making predictions on the test data
y_pred = svm_model.predict(X_test)
```

```
# Evaluating the model
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy * 100:.2f}%")

# Real-time assessment (example)
new_data_point = np.array([[85, 0.55, 2.5]])
new_data_point = scaler.transform(new_data_point)  #
Standardize the new data point
predicted_load = svm_model.predict(new_data_point)
print(f"Predicted Cognitive Load Level: {'High' if pre-
dicted_load[0] == 1 else 'Low'}")
```

In this code, the process begins by reading data from a CSV file using the pd.read_csv() function. The features and labels are then separated using operations on the Pandas DataFrame, with the labels representing the cognitive load levels corresponding to each data point. The data is split into training and testing sets using the train_test_split function. To ensure that all features are on the same scale, they are standardized using the StandardScaler. An SVM classifier with a linear kernel, represented by SVC, is trained on the training data. The model's accuracy is then evaluated on the test data using the accuracy_score function. Finally, a new data point is simulated for real-time assessment, standardized, and passed through the trained SVM model to predict the cognitive load level.

3.4 Recommendation engine

Collaborative Filtering can be used to recommend educational resources based on the learner's profile and historical interactions [12]. For a more personalized approach, Content-Based Filtering can be utilized, which recommends resources similar to those the learner has interacted with positively in the past.

In the context of recommending educational resources based on cognitive load, a combination of Collaborative Filtering (CF) and Content-Based Filtering (CBF) can be utilized to provide personalized recommendations. Here's a detailed explanation and a sample Python code using the scikit-learn and surprise libraries:

- Collaborative Filtering (CF):
  - User-Item Matrix: CF starts by creating a user-item matrix where each row represents a learner, each column represents an educational resource, and the cells contain the ratings or interactions (e.g., time spent, completion rate) between learners and resources.
  - Similarity Computation: CF computes the similarity between learners or items using metrics like cosine similarity or Pearson correlation. This helps identify learners with similar preferences or resources with similar interaction patterns.
  - Prediction: Based on the similarities, CF predicts the ratings or interactions for unseen items for a learner. For example, if two learners have similar interaction

patterns and one has interacted positively with a resource, that resource might be recommended to the other learner.

- Content-Based Filtering (CBF):
  - Feature Extraction: CBF analyzes the features of the educational resources, such as content complexity, subject matter, and format. These features can be extracted using techniques like TF-IDF for textual content or feature engineering for other types of content.
  - Profile Building: CBF constructs a profile for each learner based on their past interactions with resources and the features of those resources. This profile represents the learner's preferences.
  - Recommendation: CBF recommends resources to a learner by comparing the learner's profile with the features of available resources. Resources with features that match the learner's preferences are recommended.

Python Code for Collaborative Filtering:

```
import pandas as pd
from surprise import Dataset, Reader, KNNBasic
from surprise.model_selection import train_test_split

# Read interaction data from a CSV file
df = pd.read_csv('interactions.csv')

# Load the data into a Surprise dataset
reader = Reader(rating_scale=(1, 5))
surprise_data = Dataset.load_from_df(df[['user_id',
'item_id', 'rating']], reader)

# Split the data into training and testing sets
trainset, testset = train_test_split(surprise_data,
test_size=0.25)

# Use KNN for collaborative filtering
algo = KNNBasic(sim_options={'user_based': True})

# Train the model
algo.fit(trainset)

# Make predictions for the test set
predictions = algo.test(testset)

# Print the predictions
for uid, iid, true_r, est, _ in predictions:
    print(f'User {uid}, Resource {iid}, Estimated Rating:
{est}')
```

4 Case study

4.1 Background

This case study represents a simulation in order to implement our model and validate our approach. An online learning platform sought to improve its personalized learning experience by implementing an educational resource recommendation model based on cognitive load analysis. The platform serves a diverse user base, each with unique learning needs and preferences.

4.2 Implementation and Results

The implementation involved several key components, and the results were evaluated using various metrics:

- Learner Profile Creation:

Profiles were created for 1,000 learners, incorporating data on their prior knowledge, learning preferences, and historical interactions with the educational platform. This data formed the basis for personalizing the recommendations.

- Cognitive Load Assessment:

An SVM classifier was trained on a dataset of 500 instances, which included physiological signals (such as heart rate variability and EEG patterns) and interaction data (such as response times and click patterns). The classifier achieved an accuracy of 89.5% in assessing cognitive load levels.

- Recommendation Engine:

The recommendation engine combined collaborative filtering (CF) and content-based filtering (CBF) to provide personalized educational resource recommendations. The engine was evaluated on a set of 500 user-resource interactions, resulting in the following performance metrics:

Table 1. Performance of the recommendation engine.

Metric	Value
Precision	0.84
Recall	0.82
F1-Score	0.83

Impact on Learning Outcomes:

The impact of the model on learning outcomes was assessed by comparing the test scores of a subset of learners before and after using the recommended resources. The results showed an average improvement of 15% in test scores after using the recommended resources, indicating the effectiveness of the personalized recommendations.

4.3 Discussion

The implementation of the proposed model for recommending educational resources based on cognitive load analysis yielded promising results in the online learning platform. Here's a breakdown of the key findings and their implications:

- Learner Profile Creation: The creation of detailed profiles for 1,000 learners allowed for a nuanced understanding of each individual's learning needs and preferences. This personalization is crucial for tailoring recommendations effectively and demonstrates the platform's commitment to addressing the diverse needs of its user base.
- Cognitive Load Assessment: The SVM classifier's accuracy of 89.5% in assessing cognitive load levels is noteworthy. This high level of accuracy indicates that the model can reliably distinguish between different levels of cognitive load, which is essential for recommending resources that are neither too challenging nor too simplistic for the learner.
- Recommendation Engine Performance: The recommendation engine's performance metrics, as shown in Table 1, are impressive. A precision of 0.84 and a recall of 0.82 suggest that the engine is highly effective in recommending relevant resources (precision) while also capturing a significant proportion of the relevant resources available (recall). The F1-Score of 0.83, which balances precision and recall, further confirms the engine's overall effectiveness.
- Impact on Learning Outcomes: The average improvement of 15% in test scores after using the recommended resources is a strong indicator of the model's effectiveness. This improvement suggests that the personalized recommendations based on cognitive load analysis not only enhance learner engagement but also lead to tangible improvements in learning outcomes.

Overall, the results demonstrate the potential of using cognitive load analysis to personalize educational resource recommendations in online learning platforms. The model's ability to accurately assess cognitive load and provide effective recommendations has significant implications for enhancing learning efficiency and effectiveness. Future research could focus on further refining the recommendation engine, exploring the long-term impacts of personalized recommendations on learning outcomes, and expanding the model to accommodate a wider range of learning styles and preferences.

5 Conclusion

The implementation of a model for recommending educational resources based on cognitive load analysis in an online learning platform has demonstrated significant potential for enhancing personalized learning experiences. The creation of detailed learner profiles, coupled with the accurate assessment of cognitive load using an SVM classifier, has enabled the recommendation engine to provide highly relevant and effective educational resources. The performance metrics of the recommendation engine, including precision, recall, and F1-Score, indicate its effectiveness in delivering personalized content that aligns with individual learners' cognitive capacities.

Moreover, the observed improvement in learning outcomes, as evidenced by an average 15% increase in test scores, underscores the practical impact of personalized recommendations on learner achievement. This case study highlights the value of leveraging cognitive load analysis in educational technology to create more adaptive and responsive learning environments.

As the field of educational technology continues to evolve, the integration of cognitive load analysis into recommendation systems presents a promising avenue for further research and development. Future work could explore the scalability of the model, its applicability across different educational contexts, and its potential to incorporate additional learner characteristics and preferences. Ultimately, the goal is to create more intelligent and effective learning platforms that cater to the diverse needs of learners, facilitating not only improved academic performance but also a more engaging and satisfying learning experience.

References

1. Asselman, A., Khaldi, M., & Aammou, S. (2023). Enhancing the prediction of student performance based on the machine learning XGBoost algorithm. Interactive Learning Environments, 31(6), 3360-3379.
2. Sweller, J. (1994). Cognitive load theory, learning difficulty, and instructional design. Learning and Instruction, 4(4), 295-312.
3. Sweller, J., Van Merrienboer, J. J., & Paas, F. G. (1998). Cognitive architecture and instructional design. Educational Psychology Review, 10(3), 251-296.
4. Paas, F., Renkl, A., & Sweller, J. (2003). Cognitive load theory and instructional design: Recent developments. Educational Psychologist, 38(1), 1-4.
5. Sperling, G. (1960). The information available in brief visual presentations. Psychological Monographs: General and Applied, 74(11), 1-29.
6. Miller, G. A. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information. Psychological Review, 63(2), 81-97.
7. Tulving, E. (1972). Episodic and semantic memory. In E. Tulving & W. Donaldson (Eds.), Organization of Memory (pp. 381-403). Academic Press.
8. Hwang, G. J., & Tsai, C. C. (2011). Research trends in mobile and ubiquitous learning: A review of publications in selected journals from 2001 to 2010. British Journal of Educational Technology, 42(4), E65-E70.
9. Xie, H., Chu, H. C., Hwang, G. J., & Wang, C. C. (2019). Trends and development in technology-enhanced adaptive/personalized learning: A systematic review of journal publications from 2007 to 2017. Computers & Education, 140, 103599.
10. Sturgis, C., Patrick, S., & Pittenger, L. (2011). It's not a matter of time: Highlights from the 2011 Competency-Based Summit. International Association for K-12 Online Learning.
11. Siemens, G. (2013). Learning analytics: The emergence of a discipline. American Behavioral Scientist, 57(10), 1380-1400.
12. Jdidou, Y., Aammou, S.: Enhancing Learning Experiences Through Advanced Personalized Recommendation Systems in MOOC Platforms. In: Technological Tools for Innovative Teaching, pp. 245-276. IGI Global (2024).