electronics

Article

Deep Learning Recommendations of E-Education Based on Clustering and Sequence

Furkat Safarov [1], Alpamis Kutlimuratov [2], Akmalbek Bobomirzaevich Abdusalomov [1], Rashid Nasimov [3] and Young-Im Cho [1,*]

[1] Department of Computer Engineering, Gachon University, Seongnam-si 13120, Republic of Korea
[2] Department of AI Software, Gachon University, Seongnam-si 13120, Republic of Korea
[3] Department of Artificial Intelligence, Tashkent State University of Economics, Tashkent 100066, Uzbekistan
* Correspondence: yicho@gachon.ac.kr

Abstract: Commercial e-learning platforms have to overcome the challenge of resource overload and find the most suitable material for educators using a recommendation system (RS) when an exponential increase occurs in the amount of available online educational resources. Therefore, we propose a novel DNN method that combines synchronous sequences and heterogeneous features to more accurately generate candidates in e-learning platforms that face an exponential increase in the number of available online educational courses and learners. Mitigating the learners' cold-start problem was also taken into consideration during the modeling. Grouping learners in the first phase, and combining sequence and heterogeneous data as embeddings into recommendations using deep neural networks, are the main concepts of the proposed approach. Empirical results confirmed the proposed solution's potential. In particular, the precision rates were equal to 0.626 and 0.492 in the cases of Top-1 and Top-5 courses, respectively. Learners' cold-start errors were 0.618 and 0.697 for 25 and 50 new learners.

Keywords: recommendation system; modeling; sequence-aware; deep learning; embedding

check for updates

Citation: Safarov, F.; Kutlimuratov, A.; Abdusalomov, A.B.; Nasimov, R.; Cho, Y.-I. Deep Learning Recommendations of E-Education Based on Clustering and Sequence. *Electronics* **2023**, *12*, 809. https://doi.org/10.3390/electronics12040809

Academic Editor: Amir Mosavi

Received: 25 December 2022
Revised: 17 January 2023
Accepted: 28 January 2023
Published: 6 February 2023

1. Introduction

In the last decade, effective recommendation systems (RSs) have become an essential tool in modern online education applications, such as Coursera and Udemy. The RS suggests popular and most-enrolled courses to new learners of these applications, and also helps distance learning websites to suggest e-learning resources, such as books, lectures, and educational links, along with new and leading courses to existing learners. However, with the exponential increase in the amount of available online educational resources, the RSs have to overcome the challenge of resource overload and find the most suitable material for educators. Commercial e-learning platforms have been using RSs that assess and manage learner–item interactions to boost learner satisfaction, customize suggestions, and raise revenue. The e-learning platforms have established a few models for building RSs based on explicit and implicit features of learners and educational materials. For instance, the initial RS models were based on similarity [1]. More precise and quicker latent-factor methods were introduced immediately after the similarity-based paradigm; these use ratings to analyze customer input, and then categorize customers and products based on their features. Latent-factor methods are typically realized via matrix factorization [2] using a vector of variables based on a rating matrix to identify product and customer features. Due to the nature of the collected data, the aforementioned methods for designing recommendation models are categorized as content-based filtering (CBF) and collaborative filtering (CF) techniques. The accessibility to learner–item interaction input is evaluated using CBF, which involves the management of many explicit features [3,4]. In contrast, when generating suggestions, a CF-based RS [5] employs a customer's previous rating scores to anticipate unrated products and subsequently gathers customer requirements from a specified cohort to process predictions. In recent times, deep learning (DL) has radically changed recommendation algorithms and improved their effectiveness. In addition to enabling the modeling of increasingly complicated conceptions as data representations in the top layers, DL successfully records non-linear and complex customer-product associations [6]. Various RSs using DL algorithms [7–9] have recently attracted significant attention due to their unique potential. Neurons in neural models can be differentiated from one end to another and offer appropriate logical biases tailored to the data format. When a model with a specific form can be applied to a given data format, it may also be applied to other data that contain related structures. Furthermore, a differentiable network comprising many deep neural networks (DNNs) can be trained from start to completion. Consequently, hybrid RSs are easier to handle. However, a DL recommender mechanism may be unsuitable if both text metadata (tweets, tags, reviews, etc.) and graphic data (predicted photos) are used together, for instance. A combined (end-to-end) representation training is not possible in such cases, and hence conventional RS algorithms frequently fail. As an example, processing comments requires expensive preprocessing (such as extractive summarization), while content data (text) may be immediately processed by DL-based algorithms [10]. DNN characteristics are extensively and widely applied to assist RSs that use heterogeneous data [11,12]. However, DL and matrix factorization approaches [13,14] exhibit a continuous increase in prediction time when a system contains an enormous number of customers and products. To solve this issue, algorithms that generate top-K candidates have been suggested [15,16]. Unfortunately, these algorithms are often unable to simultaneously use both consecutive and heterogeneous inputs (i.e., semantic features). Therefore, we propose a novel DNN method that combines synchronous sequences and heterogeneous features to more accurately generate candidates in e-learning platforms that face an exponential increase in the number of available online educational courses and learners. Mitigating the learners' cold-start problem was also taken into consideration during the modeling. To the best of our knowledge, both features have not been previously used together, despite the substantial amount of research conducted to independently apply sequences and homogenous features in building RSs. Grouping learners in the first phase, and combining sequences and heterogeneous data as embeddings into recommendations using deep neural networks, are the main concepts of the proposed approach. The primary highlights of our study are as follows:

- ➢ Initialize learners as a homologous category via clustering;
- ➢ Combine synchronous sequences and heterogeneous data;
- ➢ Make time-aware course recommendations based on the sequence of learners' history;
- ➢ Overcome the learners' cold-start problem by concatenating additional features;
- ➢ Improve overall candidate generation performance when utilizing a large dataset.

The remainder of this paper is organized as follows. Section 2 provides an overview of the latest DL recommendation methods that use sequences and heterogeneous features to generate the top-N candidates, solve learner cold-start problems, and increase overall recommendation efficiency. Sections 3 and 4 explain the proposed method in detail and demonstrate its accuracy by comparing it with alternative approaches through experiments and evaluations. Section 5 summarizes the findings and scope of the study. In general, most of the relevant sources included are comparatively newer publications.

2. Literature Review

2.1. Sequence-Oriented Recommendation Methods

To tackle the ongoing challenges around course recommendations and course cold starts, several recent investigations [17–22] have used heterogeneous and sequence-oriented metadata as auxiliary features. Specifically, Zhao et al. [23] established a book RS based on customer history that considers the length of time between consecutive purchases. To mitigate the item cold-start problem, a novel methodology was created by Riedl and Yu [21], who built customized tales utilizing the greatest series of customer-observed events. In addition, a CF technique to compose dual probabilistic viewpoint approaches using pure customer data was designed by Lam et al. [18]. Likewise, extensive research [4,24,25] has concurrently incorporated multiple types of implicit and explicit data features in the prediction phase. Most candidate generation techniques and top-N RSs are relatively identical in terms of objectives. Therefore, similar to candidate generation, top-N recommendation algorithms exploit the prediction strength to select perspective products over large product spaces. For example, product-based carts [15] were suggested to consumers based on an analysis of product commonality. The basic autoencoder-based AutoRec [26] mechanism expressed by Formula (1) is another innovative top-N recommendation model:

$$h(r;\theta) = f(W \cdot g(Vr + \mu) + \beta) \tag{1}$$

where β and μ are the biases of each layer; g and f represent the model's activation functions; and r is the input's reconstruction. In this model, all customer ratings are processed as a training dataset by the input layer, which further reduces them in the encoder section. The decoder component of the model was based on the bottleneck at the midsection, where the layers of the model were widened. The original dimension in the input layer was reinstated in the output layer.

Moreover, the overfitting problem was tackled using L2 regularization to train the model, while also reducing the sum of the MSE between the input and output ratings. The parameters W and V in Formula (2) represent the first- and second-layer weights, respectively:

$$\min_{\theta} \sum_{i=1}^{n} \|r^i - h(r^i;\theta)\|^2 + \frac{\lambda}{2}(\|W\|_F^2 + \|V\|_F^2) \tag{2}$$

where λ is the regularization parameter, and r represents each learner's rating history.

Despite their flexible nature and high prediction accuracy, issues are encountered by methods using autoencoders when processing learner content to generate recommendation data because of the learner's history. To prevent this, a ground-breaking top-N RS was proposed in [27], which is based on a neural probabilistic language model. The model reduces the sparse-valued input into a fixed authentic-valued embedding vector given by Formula (3):

$$V^i = \left[v_1^i, \ldots, v_j^i \ldots, v_p^i\right] \in \mathrm{W}^{n \times p} \tag{3}$$

where p is the embedding vector size, i is the customer id, and W is the embedding matrix. The embedding vector is used to train the probabilistic component of the sentence sequence and a distributed description of every single word.

A further illustration of candidate selection is the deep recommendation algorithm of YouTube [16]. Candidate selection and product ratings are the two key components. By eliminating irrelevant films, the candidate selection phase reduces thousands of available films to a few hundred. The initial layer of the candidate-selection neural network mimics the factorization technique, using no deep layers and relying solely on customer sequence history; hence, the candidate-selection phase can be considered a nonlinear variation of the factorization technique. The approach proposed in this study closely resembles the neural network model of YouTube. Nonetheless, most recommendation engines do not usually consider the product order. In particular, calculating the mean of the embedded customer history causes the YouTube neural network algorithm to suffer from sequence features. In addition, the customer's past experience is transformed into a customized Bayesian rating [28], so that the ratings never store the customer's past order data.

2.2. Clustering-Based DL Recommendations

The accessibility to vast online educational content has made it more complex for learners to quickly choose the most pertinent content. To overcome this obstacle, a considerable number of DL recommendation techniques [29–31] have been developed using a combination of clustering methods as an initial step to guide learners overloaded with data.

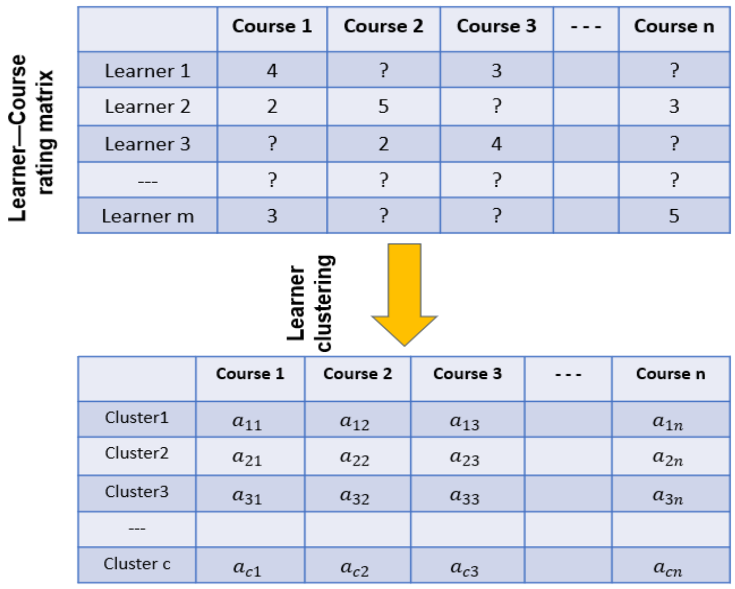
For example, a context-aware DL-based RS [32] was first proposed to derive contextual features utilizing a word-embedding technique. Then, a clustering method was applied to create positive, neutral, and negative sentiment clusters via the derived features. Finally, a deep recurrent neural network (RNN) was used to determine the most desirable customer from each cluster with the help of information entropy and biclustering [33], which offers a composite sorting approach to handle data diversity and sparseness. Biclustering can precisely identify the density components of a rating matrix, and an information entropy measure determines the similarity of a potential client to the density components. Moreover, to facilitate customer CF, Binbusayyis [34] developed a deep-clustering method that leverages the strengths of both fuzzy clustering and DNNs. The developed method first uses a deep autoencoder to extract the customers' underlying feature representation from the initial data matrix, and then performs fuzzy clustering operations on that information to create customer groups. The authors of [31] suggested using improved hierarchical reinforcement learning via a clustering approach that combines a hierarchical reinforcement neural network and a pretrained network to lessen the sparsity of book-renting information. Prior to recommending courses, clustering can be employed as a preparatory step to boost recommendation accuracy, as indicated earlier. This promotes the customization of suggestions by enabling computers to produce additional content representations from the learners' input. In summary, the proposed methodology first leverages clustering to initialize learners as a homologous category and decrease data sparsity, and then applies synchronous sequencing and heterogeneous features to generate top-N prospective courses and overcome the course cold-start problem, thus enhancing the overall candidate generation performance on big datasets. The possibility that the learners' tastes change over time sets the envisioned paradigm, with the exception of previous efforts. To the best of our knowledge, there is currently no other methodology that applies clustering to create homologous groups, and simultaneously integrates heterogeneous and sequence features within those groups.

3. Proposed Approach

This section explains the proposed approach, which combines sequences and heterogeneous features to more accurately generate candidates in e-learning platforms that face an exponential increase in the number of available online educational courses and learners. The following subsections describe the details of the recommendation model developed in this study.

3.1. Initializing Learners via K-Means

Because the numbers of available learners and courses on e-learning platforms have significantly grown, conventional RSs are unable to fulfil the online demand due to the time-consuming requests of potential learners in the entire e-learning platform. Furthermore, the efficiency in generating potential courses decreases as the number of cells augment. The initial sparseness of the input is the primary cause of poor efficiency. To mitigate the sparseness of the initial input while developing the candidate generation system, our study proposes a personalized course recommendation strategy that first applies a clustering technique based on the k-means algorithm to initialize learners as a homologous category, and then aggregates DNNs in each group of learners, utilizing synchronous sequences and heterogeneous features to generate the top-N prospective courses. Learners are categorized as homologous groups based on the input matrix. Depending on the resemblances between the potential learner and group center, the peers near the potential learner can be found and utilized to improve course generation. After the homologous groups have been established, course generations for a potential learner are produced by summing the responses from the remaining learners within the same homologous group. As illustrated in Figure 1, the idea is to use a learner clustering technique to divide the learners of a current e-learning platform into similar categories. The clustering process might result in a varying number of divisions of a particular size, or a defined set of divisions of varying sizes according to their similarity.



	Course 1	Course 2	Course 3	- - -	Course n
Learner 1	4	?	3		?
Learner 2	2	5	?		3
Learner 3	?	2	4		?
---	?	?	?		?
Learner m	3	?	?		5

	Course 1	Course 2	Course 3	- - -	Course n
Cluster1	a_{11}	a_{12}	a_{13}		a_{1n}
Cluster2	a_{21}	a_{22}	a_{23}		a_{2n}
Cluster3	a_{31}	a_{32}	a_{33}		a_{3n}

Cluster c	a_{c1}	a_{c2}	a_{c3}		a_{cn}

Figure 1. Learners' clusters.

Algorithm 1 is a widely investigated procedure for initializing learners using the k-means algorithm:

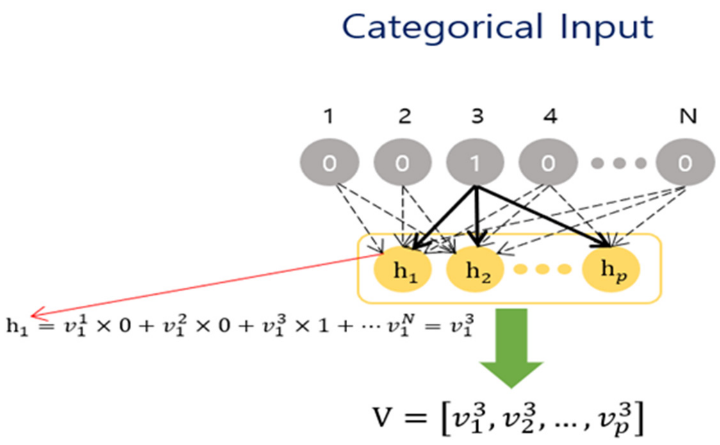
Algorithm 1: Learners' clustering algorithm

```
Input: learner–course interactions data, k-number of clusters
Output: dense learner clusters
Begin:
  Define learner set l= $\{l_1, l_2, l_3 \ldots, l_m\}$;
  Define course set co $= \{co, co_2, co_3 \ldots, co_n\}$;
  Choose primary r rating learners acting as the clustering cl $= \{cl_1, cl, cl_3 \ldots, cl_m\}$;
  The clustering kernel is null as c $= \{c_1, c_2, c_3 \ldots, c_k\}$;
  do
      for each learner $l_i \in l$
        for each cluster kernel $cl_i \in cl$
            calculate similarity sim($l_i$, $cl_i$);
        end for
        sim($l_i$, cl) = max{ sim($l_i$, $cl_1$), sim($l_i$, $cl_2$) ..., sim($l_i$, $cl_k$)};
        $cl_m = cl_m \cup l_i$
      end for
      for each cluster $cl_i \in cl$
        for each learner $l_j \in l$
            $cl_i = average\left(cl_i, l_j\right)$;
        end for
      end for
  while (c is not change)
Finish
```

The sparseness of the input data is probably the most crucial challenge in building RSs. We specifically initialize learners as a homologous group from the input data, and then apply the sequence-oriented course generation approach within each group separately. We determine dense learner interactions with certain courses by initializing the learner's homologous group. Eventually, the initial input data transforms into a dense learner–course data matrix for each homologous group.

3.2. Sequence-Oriented Embedding

A discrete categorical variable is mapped to a vector of continuous integers during embedding. Embeddings are low-dimensional, continuously learned vector representations of discrete variables used in neural networks. Neural network embeddings are helpful because they can reduce the dimension of categorical variables and accurately reflect categories in the converted space. Neural networks' weights that receive one-hot encoding of categorical data as their input, as shown in Figure 2, are called embeddings.



Categorical Input
1
2
3
4
N
0
0
1
0
0
h_1
h_2
h_p
$h_1 = v_1^1 \times 0 + v_1^2 \times 0 + v_1^3 \times 1 + \cdots v_1^N = v_1^3$
$V = [v_1^3, v_2^3, \ldots, v_p^3]$

Figure 2. Course-embedding vector of size "3".

Here, h is the hidden layer node, p is the hidden layer embedding vector size, V is the embedding vector for course three, v is the embedding vector weight, and N is the number of courses. These weights are stored as a matrix of variables to reduce the computational cost.

As part of the training phase, the network receives categorical values from the relevant row of the matrix, and the variables in this row and all other weights in the network are then trained. Pretrained or randomly initialized weights can be utilized as embedding vectors depending on the circumstances. Utilizing trained weights is advised when time is limited. Vectors obtained from techniques such as the Bayesian personalized ranking (BPR) [28] may be used to build a trained embedding. To validate the proposed method, we used the Ubob.com dataset (accessed on 24 December 2022), which contains categorical data in every field. Because every field has a different length, it is challenging to analyze the data using conventional techniques. In our approach, the number of courses unique to each learner constitutes the embedding layer. To address this issue, we used the length of the learner history sequence to generate a mask, which was then used in combination with the embedding vector to eliminate superfluous embedding. The mean vector of courses was initially determined through a conventional deep RS using Equation (4):

$$\frac{1}{n}\sum_{i=0}^{n} V_i \tag{4}$$

where n denotes the size of the learner history, and Vi is the embedding of the course that the learner has chosen.

The forecasts are identical across all combinations of member histories because this embedding strategy assumes that learner preferences are constant over time. This assumption may not be accurate in every circumstance. Initially, we create a learner–course connection matrix $R \in \mathbb{R}^{m \times n}$, where m and n represent the number of learners and courses, respectively, $r^{(u)} = (R_{u1}, \ldots R_{um})$ is the sequenced history of each learner, $u \in U = \{1, 2, \ldots m\}$, and $r^{(i)} = (R_{1i}, \ldots R_{mi})$ describes the sequenced history of every course $i \in I = \{1, 2, \ldots m\}$. Furthermore, assumptions are made as follows: if learner u_k studied "management" last year and is currently taking "economy", while learner u_l had already taken "economy" and is now learning "management", then:

- u_k: *{management, economy}*
- u_l: *{economy, management}*

Compared with u_l, who studied economy and management, in that order, u_k has a noticeably different preference. The first customer could have previously completed management, and eventually may change their mind and choose economy, whereas the second learner would want to enroll in the management course after completing economy. This demonstrates that the sequence should not be disregarded and suggests that proposing the same products may not render a reliable forecast. We provide a novel embedding vector that comprises sequence information to exploit the sequence data (learner history). The learner's category history should be input into the model without modifying its order (for example, (i_1, i_2, i_3 $_{\ldots}$ i_N)). Software was used to create the model. The history duration should be equalized by adding a certain constant number that will be subtracted in the following layer because N changes for various learners and neural networks can employ a constant matrix form. This integer number (the history of learners) is passed onto the subsequent embedding layer, which substitutes integers with the appropriate float dense vector. Large datasets typically use vectors with a length of 128; however, this may be altered. The importance of items that were consumed in the past diminishes over time, as previously indicated, and products that consumers have recently taken may be given a higher weight when forecasting the next items. To describe the order of the item in the learner's history, the values are multiplied by each embedding vector of the learner's history:

$$V' = \frac{V}{N - order + 1} \tag{5}$$

In this case, '*order*' is the number of courses in the learner's history, N is the number of courses, and V is the embedding vector of the current course. Equation (6) is used to obtain the average of these vectors following consecutive embedding of objects:

$$V_{sequential} = \frac{1}{N}\sum_{i=1}^{N} V'_i. \tag{6}$$

The input layer is then concatenated with V-sequential and actual embedding, as shown in Figure 3, to determine the layer about the learner's average and sequential preferences.

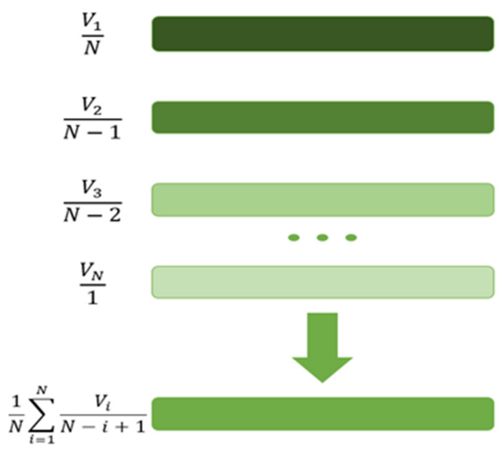


Figure 3. Sequence-oriented embedding.

3.3. Content Data

Generally, the customer content data comprise continuous and categorical (small and large) information. The continuous data are received by neural networks as inputs and large amounts of category data are used to function as embedding vectors. However, it is possible to enter categorical data into a network serving as a sparse vector (one-hot encoding) when the datasets are small. For instance, if the learner data include a field such as gender, it may include three categorical features: male, female, and missing information. In this situation, the data should be encoded using the one-hot method described below:

Female:	[1, 0, 1]
Male:	[0, 1, 0]
Missing data:	[0, 0, 1]

Our dataset comprised the learner's "jobcode" information, which utilized keywords to describe the learner's occupation. We calculate the mean vector and concatenate deep networks using these data as an embedding vector input because there are more than 100 job codes, and a single learner could have multiple job codes.

3.4. Deep Neural Network

As shown in Figure 4, the network begins by concatenating each feature vector into a single layer. Two or three completely linked layers may be added, depending on the length of the training data (three throughout all of our tests). Each hidden layer included 64, 32, and 16 units. While the customers' "jobcode" and "company information" were encoded employing 16 units each, the courses' embedding dimension was encoded employing 32 units. Subsequently, ReLU, Leaky ReLU, and sigmoid activation functions were implemented. It was determined that a ReLU as given by Equation (7) was optimal for DNNs:

$$\text{ReLU} = \begin{cases} x: & if(x > 0) \\ o: & if\ (x < 0) \end{cases} \tag{7}$$

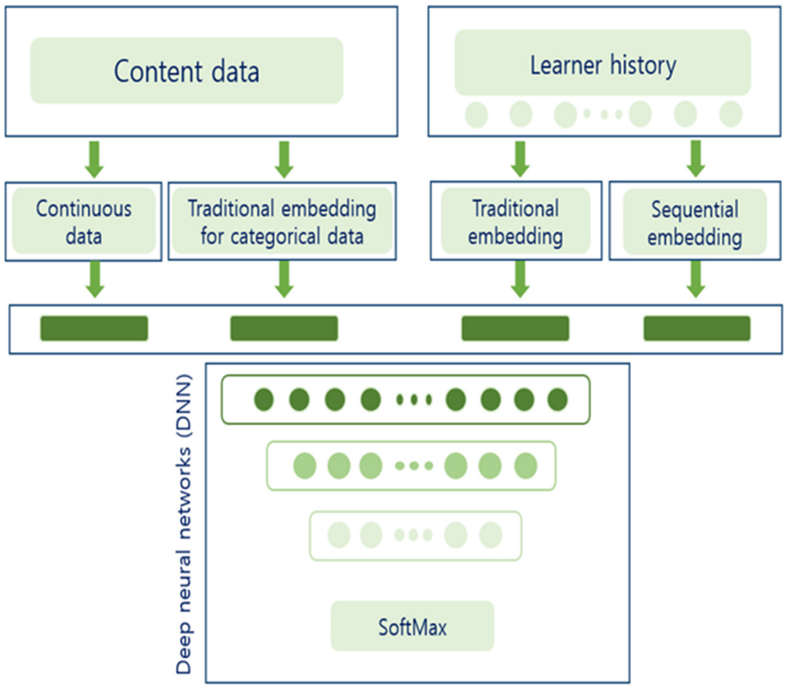


Figure 4. Sequence-oriented DNN (SODNN) recommendation.

Finally, with an identical number of active courses in the dataset, the SoftMax function expressed by Equation (8) was deployed in the output layer of the proposed DNN:

$$\text{Softmax } = \sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{8}$$

It should be noted that the output value corresponds to the number of courses in the dataset.

4. Experimental Results

4.1. Dataset

The Korean e-learning platform "www.ubob.com" (accessed on 24 December 2022) provided a dataset to design and implement the recommendation approach. The dataset was primarily used to assess and compare the recommendation approach with other related methods. The dataset structure comprises four tables that contain sequence and heterogeneous information, as illustrated in Figure 5.

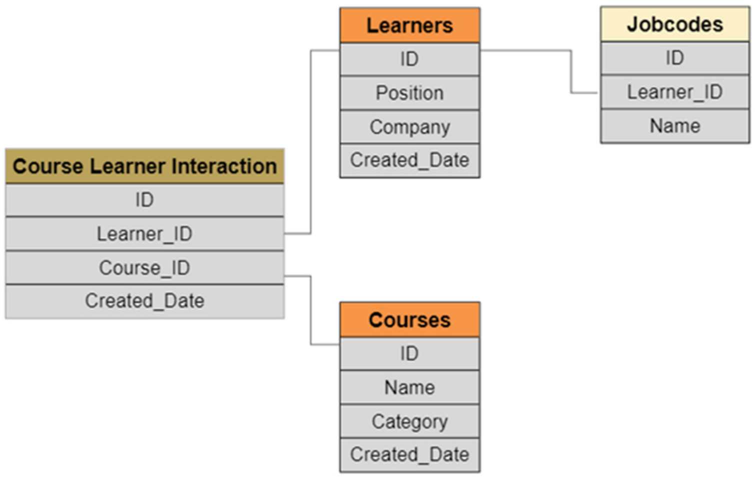


Figure 5. Components of the Ubob.com dataset (accessed on 24 December 2022).

The dataset comprises variables with varying sizes, each of which included categorical features. For example, the table "student" stores information about learners. Therefore, it is difficult to establish approaches to examine these features. Moreover, several features, including sex and age, lack sufficient data to be employed in the recommendation engines. Consequently, the table no longer contained these features. There were approximately 222,000 learners on www.ubob.com (accessed on 24 December 2022); however, only 82,000 learners were associated with the learner–course interactions that were implemented in our model. Additionally, learners provide "jobcode" data, which summarizes their occupation with a single term. Every learner's "jobcode" feature in the platform has a distinct length. There were 528 eligible "jobcode" features serving 82,000 learners. The network's final layer, known as "SoftMax", could not learn all the classifications when the dataset was trained only using the features that were already available. Therefore, to mitigate this problem, the dataset was expanded using a data-augmentation process based on "the sliding-window technique". The data augmentation process is presented in Table 1.

Table 1. Data augmentation process.

Learner History Sequence	Following Course
$L_1 = [C_1, C_2, C_3]$	C_4
$L_2 = [C_1, C_2]$	C_3
$L_3 = [C_1]$	C_2

For instance, one learner may have a C_1, C_2, C_3, and C_4 course history sequence, meaning that after taking courses C_1, C_2, and C_3, the learner has chosen course C_4. Moreover, if there is a C_1 and C_2 sequence history, the following course is C_3. Based on this strategy, data augmentation was implemented to increase the dataset size.

After implementing data augmentation, 320,000 examples were obtained, which impact the efficiency of the proposed network model. To evaluate the performance of the proposed model, 80% of the dataset was used for training and the remaining 20% was used for testing.

4.2. Implementation Settings

The proposed approach was implemented using software, hardware configurations, and model parameters, as illustrated in Table 2.

Table 2. Configurations.

Software	Programming tools	Python, Pandas, Keras-TensorFlow,
	OS	Windows 10
Hardware	CPU	AMD Ryzen Threadripper 1900X 8-Core Processor 3.80 GHz
	GPU	Titan Xp 16 GB
	RAM	128 GB
Parameters	Epochs	20
	Learning rate	0.001
	Optimal Clusters	8

4.3. Model's Results

4.3.1. Top-N Predication Performance

First, the proposed sequence-oriented deep network approach initializes learners to make similar groups. Then, the whole candidate generation process synchronously integrates the learners' history sequence data and content data accomplished in each initialized group. In the implementation process, precision was utilized for the top-N performance results, and the mean absolute error was used to evaluate the cold-start case. In addition, to compare the performance of the proposed approach with its competitors, we contrasted it with the following benchmarks, and the prediction results are illustrated in Table 3.

1. AutoRec [26]: To deliver individualized suggestions, a method based on the autoencoder approach attempts to use customer preference data for various products.
2. YouTube model [16]: The authors utilized categorical features and continuous data to make recommendations, whereas the sequence of customer history was disregarded.
3. LightGCN [35]: This model learns customer and product embeddings by linearly distributing them on a customer-product interaction graph. The weighted sum of the embeddings was then utilized as the last embedding layer.
4. FISM [36]: The authors offer a technique that creates two low-latent factorized matrices by learning the product–product matrices to represent and retain the relationships between products.
5. E-LCRS [37]: This recommendation model was built based on the history and preferences using a collaborative filtering mechanism.
6. Cluster-IHRS [38]: The recommender evaluates and learns the styles and features of the learners automatically. Split and conquer strategy-based clustering is used to process the various learning styles. The algorithm then makes intelligent suggestions based on the ratings of frequently occurring sequences.

Table 3. Top-N performance.

Model	Precision@1	Precision@5	Precision@10
AutoRec [26]	0.309	0.213	0.175
YouTube model [16]	0.599	0.461	0.294
FISM [36]	0.553	0.3238	0.2358
LightGCN [35]	0.589	0.4702	0.3768
E-LCRS [37]	0.611	0.453	0.209
Cluster-IHRS	0.46	0.352	0.216
SODNN (proposed)	0.626	0.498	0.385

Furthermore, using a trained model, we attempted to illustrate the similarity in the predicted courses (Table 4) related to the course called "Real Estate Broker_Introduction 2017 (1)".

Table 4. Predicted course similarity.

Courses	Similarity
Real Estate Disclosure Act	0.801
Real Estate Finance	0.775
Real Estate Investing	0.698
Real Estate Course Introduction	0.623

Moreover, the network was trained without overfitting, as evidenced by the visual representation of the loss functions of the training and validation sets. Furthermore, we may conclude that the training model performed the best in the 25th epoch (Figure 6).

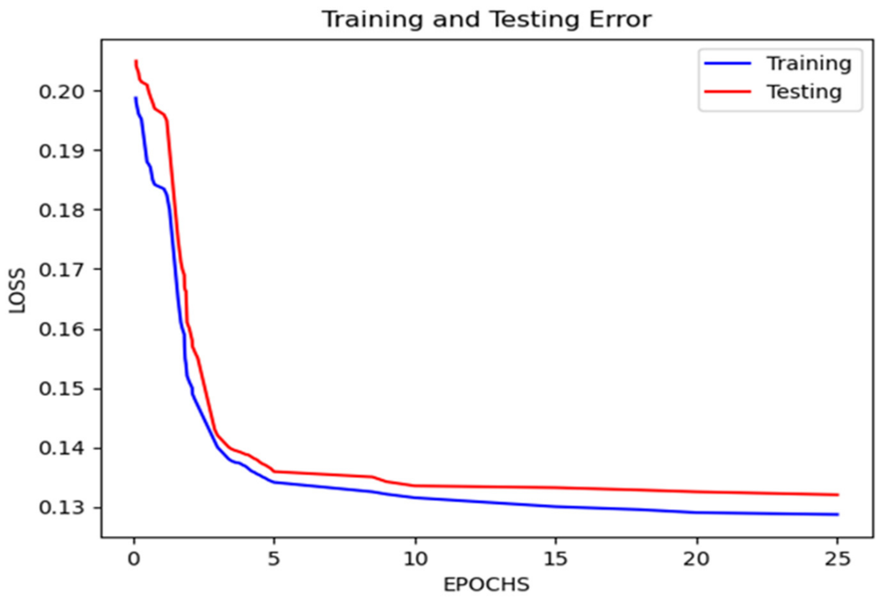


Figure 6. Training and testing error.

4.3.2. Cold-Start Case

The cold-start issue may be minimized because the Ubob.com (accessed on 24 December 2022) dataset will have sufficient interaction data on learners and courses after clustering. Additionally, because courses were permanent and never altered, adding new content is a challenge for this online educational platform. Moreover, generating recommendations only for new subscribers could be challenging. To address the issue of new subscribers, the combined default embedding vector, with the help of new learners' "company_id" and "jobcode" features, was utilized to predict courses where the system already had those features for the new learners. Table 5 demonstrates that the proposed approach performed better than the compared benchmarks. Thus, simultaneously utilizing the learners' sequence history and content data generated good recommendation results in a cold-start scenario.

Table 5. Learners' cold-start performance.

Cold-Start Cases		**Model**				
		AutoRec [26]	**YouTube Model [16]**	**FISM [36]**	**Light-GCN [35]**	**SODNN (Proposed)**
MAE	New 25 learners	0.771	0.664	0.792	0.784	**0.618**
	New 50 learners	0.835	0.753	0.843	0.840	**0.697**

5. Conclusions and Future Work

In this research, we attempted to build a novel deep neural network that combines synchronous sequences and heterogeneous features to help e-learning platforms generate candidates when there is an exponential increase in the number of learners and available online educational courses. Specifically, we first focused on applying a clustering technique based on the k-means algorithm to initialize learners as a homologous group. Then, based on aggregating deep neural networks to each group of learners, synchronous sequences and heterogeneous features are utilized to generate the top-N prospective courses. The goal of our research was to initialize learners as a homologous category by clustering and combining synchronous sequences and heterogeneous data. Moreover, the proposed methodology overcomes the learners' cold-start issue by concatenating additional features and enhancing the overall performance of candidate generation on big datasets. The empirical results show that the proposed approach outperforms baseline methods. However, some limitations need to be addressed to further its development:

- Combine the approach with other more advanced and accurate clustering techniques [39,40];
- Address the need to offer the simplest possible dynamic candidate generation;
- Address the "gray sheep" problem, in which a learner cannot be associated with any homologous cluster and the online platform is incapable of suggesting relevant courses.

Moreover, future studies should investigate more advanced models to calculate the significance of the hidden user and object features and mitigate the above-mentioned problems. In addition, it should be possible to create an emotion-based candidate generation model that involves heterogeneous speech data [41] and learners' history sequence data, and to develop a recommendation model for visually impaired people [42] based on their characteristics.

Author Contributions: This manuscript was designed and written by F.S., A.K., A.B.A. and R.N., who analyzed and validated the proposed model. Y.-I.C. supervised the study and contributed to the analysis and discussion of the algorithm and experimental results. All authors have read and agreed to the published version of the manuscript.

Funding: This study was funded by Korea Agency for Technology and Standards in 2022, project numbers are K_G012002234001, K_G012002236201, by Korea Institute of Marine Science & Technology Promotion (KIMST) funded by the Ministry of Oceans and Fisheries, Korea (20220534) and by the Gachon University research fund of 2020 (GCU-202008460006).

Acknowledgments: F.S., A.K., A.A. and R.N. would like to express their sincere gratitude and appreciation to Young-Im Cho (Gachon University) for her support, comments, remarks, and engagement over the period in which this manuscript was written. Moreover, the authors would like to thank the editor and anonymous referees for the constructive comments in improving the contents and presentation of this paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ricci, F.; Rokach, L.; Shapira, B.; Kantor, P.B. *Recommender Systems Handbook*; Springer: Berlin/Heidelberg, Germany, 2011.
2. Koren, Y.; Bell, R.; Volinskiy, C. Matrix factorization techniques for recommender systems. *IEEE Comput.* **2009**, *42*, 30–37. [CrossRef]
3. Su, X.; Khoshgoftaar, T.M. A survey of collaborative filtering techniques. *Adv. Artif. Intell.* **2009**, *2009*, 421425. [CrossRef]
4. Kutlimuratov, A.; Abdusalomov, A.; Whangbo, T.K. Evolving Hierarchical and Tag Information via the Deeply Enhanced Weighted Non-Negative Matrix Factorization of Rating Predictions. *Symmetry* **2020**, *12*, 1930. [CrossRef]
5. Wang, J.; de Vries, A.P.; Reinders, M.J.T. Unifying Learner-based and Item-based Collaborative Filtering Approaches by Similarity Fusion. In Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Seattle, WA, USA, 6–11 August 2006; pp. 501–508.
6. Zhang, S.; Yao, L.; Sun, A.; Tay, Y. Deep Learning Based Recommender System: A Survey and New Perspectives. *ACM Comput. Surv.* **2020**, *52*, 5. [CrossRef]
7. Okura, S.; Tagami, Y.; Ono, S.; Tajima, A. Embedding-Based News Recommendation for Millions of Learners. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, 13–17 August 2017.
8. Ilyosov, A.; Kutlimuratov, A.; Whangbo, T.-K. Deep-Sequence–Aware Candidate Generation for e-Learning System. *Processes* **2021**, *9*, 1454. [CrossRef]
9. Chen, M.; Xu, Z.; Weinberger, K.; Sha, F. Marginalized denoising autoencoders for domain adaptation. *arXiv* **2012**, arXiv:1206.468. [CrossRef]
10. Zheng, L.; Lu, C.-T.; He, L.; Xie, S.; He, H.; Li, C.; Noroozi, V.; Dong, B.; Yu, P.S. MARS: Memory Attention-Aware Recommender System. In Proceedings of the 2019 IEEE International Conference on Data Science and Advanced Analytics (DSAA), Washington, DC, USA, 5–8 October 2019; pp. 11–20.
11. Zhang, F.; Yuan, N.J.; Lian, D.; Xie, X.; Ma, W.-Y. Collaborative knowledge base embedding for recommender systems. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016; pp. 353–362.
12. Lee, H.; Ahn, Y.; Lee, H.; Ha, S.; Lee, S.G. Quote Recommendation in Dialogue using Deep Neural Network. In Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval, Pisa, Italy, 17–21 July 2016; pp. 957–960.
13. Guo, H.; Tang, R.; Ye, Y.; Li, Z.; He, X. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. *arXiv* **2017**, arXiv:1703.04247. [CrossRef]
14. Ruining, H.; Julian, J. VBPR: Visual bayesian personalized ranking from implicit feedback. In Proceedings of the AAAI-16 Conference on Artificial Intelligence, Phoenix, AZ, USA, 12–17 February 2016; Volume 30.
15. Deshpande, M.; Karypis, G. Item-based top-N recommendation algorithms. *ACM Trans. Inf. Syst.* **2004**, *22*, 143–177. [CrossRef]
16. Covington, P.; Adams, J.; Sargin, E. Deep neural networks for Youtube recommendations. In Proceedings of the 10th ACM Conference on Recommender Systems, Boston, MA, USA, 15–19 September 2016; pp. 191–198.
17. Agrawal, R.; Srikant, R. Mining sequential patterns. In Proceedings of the 11th International Conference on Data Engineering (ICDE), Taipei, Taiwan, 6–10 March 1995; pp. 3–14.
18. Lam, X.N.; Vu, T.; Le, T.D.; Duong, A.D. Addressing cold-start problem in recommendation systems. In Proceedings of the 2nd International Conference on Ubiquitous Information Management and Communication, Suwon, Republic of Korea, 31 January–1 February 2008; ACM: New York, NY, USA, 2008; pp. 208–211.
19. Abdusalomov, A.; Baratov, N.; Kutlimuratov, A.; Whangbo, T.K. An Improvement of the Fire Detection and Classification Method Using YOLOv3 for Surveillance Systems. *Sensors* **2021**, *21*, 6519. [CrossRef]
20. Schein, A.I.; Popescul, A.; Ungar, L.H.; Pennock, D.M. Methods and metrics for cold-start recommendations. In Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, New York, NY, USA, 11–15 August 2002; pp. 253–260.
21. Yu, H.; Riedl, M.O. A sequential recommendation approach for interactive personalized story generation. In Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems, Valencia, Spain, 4–8 June 2012.
22. Mobasher, B.; Dai, H.; Luo, T.; Nakagawa, M. Using sequential and non-sequential patterns in predictive Web usage mining tasks. In Proceedings of the 2002 IEEE International Conference on Data Mining, Maebashi, Japan, 9–12 December 2002; pp. 669–672.

23. Zhao, G.; Lee, M.L.; Hsu, W.; Chen, W. Increasing temporal diversity with purchase intervals. In Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval, Portland, OR, USA, 12–16 August 2012.
24. Bao, Y.; Fang, H.; Zhang, J. TopicMF: Simultaneously Exploiting Ratings and Reviews for Recommendation. In Proceedings of the AAAI Conference on Artificial Intelligence, Québec City, QC, Canada, 27–31 July 2014.
25. Qiao, Z.; Zhang, P.; Cao, Y.; Zhou, C.; Guo, L.; Fang, B. Combining Heterogenous Social and Geographical Information for Event Recommendation. In Proceedings of the AAAI Conference on Artificial Intelligence, Québec City, QC, Canada, 27–31 July 2014; pp. 165–174.
26. Sedhain, S.; Menon, A.K.; Sanner, S.; Xie, L. AutoRec: Autoencoders Meet Collaborative Filtering. In Proceedings of the 24th International Conference on World Wide Web, Florence, Italy, 18–22 May 2015; pp. 111–112.
27. Bengio, Y.; Ducharme, R.; Vincent, P.; Janvin, C. A Neural Probabilistic Language Model. *J. Mach. Learn. Res.* **2003**, *3*, 1137–1155.
28. Rendle, S.; Freudenthaler, C.; Gantner, Z.; Schmidt-Thieme, L. BPR: Bayesian Personalized Ranking from Implicit Feedback. *arXiv* **2012**, arXiv:1205.2618. [CrossRef]
29. Rostami, M.; Oussalah, M.; Farrahi, V. A Novel Time-Aware Food Recommender-System Based on Deep Learning and Graph Clustering. *IEEE Access* **2022**, *10*, 52508–52524. [CrossRef]
30. Kutlimuratov, A.; Abdusalomov, A.B.; Oteniyazov, R.; Mirzakhalilov, S.; Whangbo, T.K. Modeling and Applying Implicit Dormant Features for Recommendation via Clustering and Deep Factorization. *Sensors* **2022**, *22*, 8224. [CrossRef]
31. Wang, X.; Wang, Y.; Guo, L.; Xu, L.; Gao, B.; Liu, F.; Li, W. Exploring Clustering-Based Reinforcement Learning for Personalized Book Recommendation in Digital Library. *Information* **2021**, *12*, 198. [CrossRef]
32. Boppana, V.; Prasad, S. Web crawling based context aware recommender system using optimized deep recurrent neural network. *J. Big Data* **2021**, *8*, 144. [CrossRef]
33. Jiang, M.; Zhang, Z.; Jiang, J.; Wang, Q.; Pei, Z. A collaborative filtering recommendation algorithm based on information theory and bi-clustering. *Neural Comput. Appl.* **2019**, *31*, 8279–8287. [CrossRef]
34. Binbusayyis, A. Deep embedded fuzzy clustering model for collaborative filtering recommender system. *Intell. Autom. Soft Comput.* **2022**, *33*, 501–513. [CrossRef]
35. He, X.; Deng, K.; Wang, X.; Li, Y.; Zhang, Y.; Wang, M. LightGCN: Simplifying and Powering Graph Convolution Network for Recommendation. *arXiv* **2020**, arXiv:2002.02126.
36. Kabbur, S.; Ning, X.; Karypis, G. Fism: Factored item similarity models for top-n recommender systems. In Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Chicago, IL, USA, 11–14 August 2013; pp. 659–667.
37. Jena, K.K.; Bhoi, S.K.; Malik, T.K.; Sahoo, K.S.; Jhanjhi, N.Z.; Bhatia, S.; Amsaad, F. E-Learning Course Recommender System Using Collaborative Filtering Models. *Electronics* **2023**, *12*, 157. [CrossRef]
38. Bhaskaran, S.; Marappan, R.; Santhi, B. Design and Analysis of a Cluster-Based Intelligent Hybrid Recommendation System for E-Learning Applications. *Mathematics* **2021**, *9*, 197. [CrossRef]
39. Peng, X.; Li, Y.; Tsang, I.W.; Zhu, H.; Lv, J.; Zhou, J.T. XAI Beyond Classification: Interpretable Neural Clustering. *arXiv* **2018**, arXiv:1808.07292. [CrossRef]
40. Ji, P.; Zhang, T.; Li, H.; Salzmann, M.; Reid, I. Deep subspace clustering networks. In Proceedings of the 29th Advances in Neural Information Processing Systems, Montreal, QC, Canada, 4–9 December 2017.
41. Makhmudov, F.; Kutlimuratov, A.; Akhmedov, F.; Abdallah, M.S.; Cho, Y.-I. Modeling Speech Emotion Recognition via Attention-Oriented Parallel CNN Encoders. *Electronics* **2022**, *11*, 4047. [CrossRef]
42. Abdusalomov, A.B.; Mukhiddinov, M.; Kutlimuratov, A.; Whangbo, T.K. Improved Real-Time Fire Warning System Based on Advanced Technologies for Visually Impaired People. *Sensors* **2022**, *22*, 7305. [CrossRef] [PubMed]