A course hybrid recommender system for limited user information scenarios

Juan Camilo Sanguino
Universidad de los Andes
Systems and Computing Engineering
jc.sanguino10@uniandes.edu.co

Olga Mariño
Universidad de los Andes
Systems and Computing Engineering
olmarino@uniandes.edu.co

Nicolás Cardozo
Universidad de los Andes
Systems and Computing Engineering
n.cardozo@uniandes.edu.co

Rubén Manrique
Universidad de los Andes
Systems and Computing Engineering
rf.manrique@uniandes.edu.co

Mario Linares-Vásquez
Universidad de los Andes
Systems and Computing Engineering
m.linaresv@uniandes.edu.co

Recommender systems in educational contexts have proven to be effective in identifying learning resources that fit the interests and needs of learners. Their usage has been of special interest in online self-learning scenarios to increase student retention and improve the learning experience. In this article, we present the design of a hybrid course recommendation system for an online learning platform. The proposed hybrid system articulates the recommendation carried out by collaborative and content-based filter strategies. For the collaborative filtering recommender, we address the challenge of recommending meaningful content with limited information from users by using rating estimation strategies from a log system (Google Analytics). Our approach posits strategies to mine logs and generates effective ratings through the counting and temporal analysis of sessions. We evaluate different rating penalty strategies and compare the use of per-user metrics for rating estimation. For the content-based recommender, we compare different text embeddings that range from well-known topic models (LSA and LDA) to more recent multilingual contextual embeddings pre-trained on large-scale unlabelled corpora. The results show that the best model in terms of $P@5$ was the Collaborative filtering recommendation model with a value of 0.4, *i.e.,* two out of five courses recommended could be of the user's interest. This result is satisfactory considering that our models were trained from ratings inferred from implicit user data. The content-based strategies did not yield significant results, however, these strategies help to mitigate the cold start problem and validate the use of a combined hybrid strategy.

Keywords: recommender systems, collaborative filtering, content-based recommendations, hybrid recommendations, logs mining, contextual embeddings

1. INTRODUCTION

The number of users enrolled in learning web platforms (*i.e.,* MOOCs) (Kang, 2021) has constantly grown in the last 8 years (Sari et al., 2020). The recent COVID-19 pandemic has reinforced this phenomenon, and web virtual learning platforms are taking on importance not previously seen. The pandemic pushed learners, course designers, and instructors to migrate to virtual environments where the learning content is mainly delivered in digital formats with limited or no face-to-face interaction. In addition, existing work has shown the importance of simple but effective virtual learning environments, which allow a progressive adaptation according to learners' needs and preferences, in order to keep them motivated and engaged (Zakaria et al., 2016; Sousa and Rocha, 2020).

To adapt the learning environment and give meaningful content recommendations to learners, learning platforms should be able to profile the learner in terms of a set of features such as preferences, behavior, user navigation patterns, or learning needs. A question that arises in this context is, how to obtain relevant learners' information? While more traditional approaches ask the learner directly through surveys and registration forms, modern approaches complement such explicit information with implicit/latent data extracted from interactions with the platform (*i.e.,* logs) (Kew and Tasir, 2022; Hassan et al., 2021). The amount and quality of information that can be extracted depend largely on privacy regulations, the administrator guidelines, and the platform's technological capabilities. Therefore, it is not unusual to find free virtual learning scenarios where the available and collected information about learners is limited. This is the case for the GCFGlobal learning platform.[1]

The objective of the GCFGlobal Learning Program is to teach different basic skills necessary for the 21st century, online, in an open modality, and without cost. According to GCFGlobal 's Web page, `gcfglobal.org`, GCFGlobal offers training in more than 40 topics, ranging from Microsoft Office and email usage to reading, math, and more. The content there is organized in courses, and a GCFGlobal course contains several lessons. GCFGlobal offers more than 360 courses, counting more than 6,400 lessons, more than 2,500 videos, and more than 50 interactive activities and games. In 2021 the number of users for the English, Spanish, and Portuguese sites, visiting at least one course was around 41 million. GCFGlobal establishes as a regulatory principle of its operation that access to their learning content has to be open to anyone in the world, and the only requirement is internet access. The courses are online, self-paced, and self-directed (*i.e.,* without a tutor). Registration and authentication on the platform are not mandatory and it is estimated that only 2% of users are registered, thus, there is no explicit information about learners (*i.e.,* identity data). The information that is available comes mainly from logs via the Google Analytics tool. Google Analytics is a standard and popular tool used in e-commerce that has been positioned as a useful tool in learning platforms (Brandon, 2019; Gaur et al., 2016). Google Analytics' popularity is due to the ease of implementation and its capabilities to filter and analyze large volumes of logs. Although log analysis has limitations when it comes to identifying a user, it is the most likely type of data that can be found in any existing web-based platform.

This paper is an extension of our previous work (Sanguino et al., 2022). Our goal with this paper is to present a hybrid recommendation system that is able to work in limited information scenarios, *i.e.,* the proposed recommendation system uses as input information available

[1]GCFGlobal learning platform:`https://bit.ly/3tBpGa5`

in Google Analytics logs from GCFGlobal. In addition, we present an empirical study for rating estimation from Google Analytics logs, for the collaborative filtering (CF) approach. These ratings express the preferences of the learners and allow us to construct the collaborative filter model. Our hybrid recommender system also contains a content-based (CB) strategy to deal mainly with the cold start problem. Since the lessons are texts, the crucial step is the construction of an appropriate vector representation that allows correct similarity to be computed. We present a comparative study of different vectorization strategies that range from well-known topic models (LSA and LDA) to more recent multilingual contextual embeddings, pre-trained on large-scale unlabelled corpora.

Finally, to combine the CF and CB strategies, we use a weighted hybrid strategy (Yin et al., 2020; Do et al., 2017). Even though there are more outstanding strategies such as the analysis of the dispersion of the rating matrix (Xiao et al., 2018), these cannot be applied in our case as most of the users are "new users" and our matrix would be constantly sparse.

We use the term "limited information scenario" to refer to the scenario where there is no explicit information about the users and it is necessary to use alternative sources of implicit information, as Google Analytics logs. Because logs provide information about web pages visited by users, we process them to extract the courses and lessons that the users accessed. However, to work with these logs and build a recommendation system, we have to assume the following statements:

- A user visits the platform using a unique device. This is required as the user identifier in Google Analytics is the device identifier. In the way the logs are implemented, it is not possible to correlate the same user accessing on different devices.
- Most of the users are "new users". Approximately 70% of the visits correspond to devices not previously seen. This percentage is stable in the analyzed observation period (one year), and poses a huge challenge related to the well-known cold start problem.
- Users completely study the lessons they access. Although within our experiments, we explore the possibility of filtering access by session duration as an additional check, there is no guarantee that a user will cover all learning content of the lesson.

Our assumptions suppose a challenge in the construction of recommendation systems. Therefore, the way how we approached these challenges with our hybrid recommendation system constitutes the main contribution of this article. That contribution is supported on the following aspects described in the following sections:

(1) Dedicated strategies for counting lessons seen by a user, and processing the timelines from logs.

(2) A strategy to generate ratings that contemplate per-user metrics based on lesson counting for the CF model.

(3) A comparison of previous strategies in the course recommendation task for the CF model.

(4) A comparison of topic models (LSA, and LDA) and contextual embeddings (MPNet, RoBERTa, and MiniLM) as text vectorization strategies for the CB model.

(5) A comparison of non-multilingual and multilingual contextual embeddings as text vectorization strategies for the CB model.

(6) An analysis of the impact of the length of the text used to represent the course content, in the recommendation precision for the CB model.

The paper is organized as follows: Section 2 describes related work on recommendation systems in e-learning. Section 3 describes our proposed architecture for the recommendation system and the pipeline to build the model divided into 4 parts: First we explain the data processing steps. The following parts present the strategies and considerations to build the CF, CB, and hybrid models. Section 4 presents the experimental results of each model. Finally, Section 5 presents the discussion and future research directions, and Section 6 concludes with a summary of the main ideas and findings of the paper.

2. Related Work

2.1. Courses recommendation systems in e-learning

A recommender system can be characterized by answering the following four questions: What does the system recommend? On what sources of information? With which recommendation technique? And based on what technological paradigm?

Regarding the first question, in the context of learning, recommender systems can provide recommendations on different dimensions (Uddin et al., 2021): they can recommend educational resources at different levels of granularity: complete courses, particular lessons, and even specific content such as texts, videos, tutorials, etc. They can also recommend elements related to the learning process such as pre-requisites, learning objectives, or learning paths (Manrique Piramanrique, 2019) as well as supporting student peers.

To build the recommendations, e-learning recommender systems exploit three sources of information: the educational resources and paths, the users' (students') information, and the interaction between them. Information on these sources may be implicit or explicit. Implicit information is mainly obtained by automatically analyzing the interactions with the learning platform and the content of the educational resources, a source often exploited in MOOCs through learning Analytics (Uddin et al., 2021). Explicit information on users such as interests, goals, and background might be directly asked, for instance when enrolling in a MOOC course. Also, open educational resources (OER) are often enriched with educational metadata like the Learning Object Metadata (LOM) and IEEE Sharable Content Object Reference Model (SCORM) standards to facilitate a pedagogically oriented search in learning object repositories (LOR) like Merlot (Ghebghoub et al., 2008; Shmueli, 2017).

Based on this information, two common techniques used to generate a recommendation are: collaborative filtering and content-based recommendation. (Do et al., 2017; Shanshan et al., 2021). Collaborative filtering is based on the premise that if two users are "similar", they should be interested in the same OERs (Xiao et al., 2018; Shanshan et al., 2021). The content-based technique is based on the assumption that if a user has explored an OER, it is possible that they will be interested in another "similar" OER (Xiao et al., 2018; Do et al., 2017).

Finally, there are two major technological paradigms used in recommendation systems for e-learning. First, symbolic or knowledge-based approaches (Tarus et al., 2018). The recommendation system is based on an explicit representation of the knowledge and the content of both

users and OERs, and performs similarity analyses based on a comparison of said representations. Recent work uses ontologies and Semantic Web-based knowledge graphs for knowledge representation (Manrique et al., 2018), the related competencies (Paquette et al., 2021) and use graph similarity algorithms (Manrique et al., 2018), or knowledge inferences (Paquette et al., 2015; Marino and Paquette, 2010) to generate the recommendations. The knowledge-based approach relies on the explicit knowledge representation, either registered manually or extracted automatically from texts (Manrique et al., 2018), or from user interaction.

The second technological paradigm, which is the most widely used today, uses a connectionist approach that establishes similarity at the level, not of knowledge, but of data, using machine learning algorithms (Xiao et al., 2018; Shanshan et al., 2021). While the connectionist approach has some well-documented problems such as the data sparsity and the cold start, it does not need an explicit representation of the knowledge and content of users and resources.

It is worth noticing that the knowledge-based and the machine-learning-based paradigms can be combined, for instance by working with the explicit knowledge from the learning resources but keeping the user´s information at the level of the logs of the interactions with the platform. And they can both apply any of the recommendation techniques. The following section explains further the different recommendation techniques.

2.2. Recommendation techniques for courses recommendation systems in e-learning

In e-learning, one of the most popular recommendation techniques used is Collaborative Filtering (Shanshan et al., 2021; Mawane et al., 2020). Collaborative filtering (CF) analyzes users' behavior and course ratings to find a group of similar users. Then, the recommendation is made based on the courses viewed by the group. The rating could be explicit (*e.g.,* by a score that the learner gives to the course), or implicit (*e.g.,* deducted from the learner's interactions with the course content). These ratings are used to calculate the similarity between users (Tarus et al., 2018; Uddin et al., 2021). CF-based systems assume that if learner X has the same/similar rating as learner Y on a course, X is more likely to have the same/similar rating as Y on a different course (Aggarwal, 2016a, Chapter 3).

In the last years, different CF recommendation systems have been proposed. Although using CF has shown success, it also has limitations when the data has a reduced number of user ratings. For example, a CF-based system cannot cluster users with similar interests when the amount of ratings is very low. As a result, the precision of the recommendations is low. This problem is called cold start (Shanshan et al., 2021; Ma et al., 2021).

To address the cold start problem, recommendation systems implement auxiliary content-based (CB) systems that use additional information, in our context information about the lessons and courses. CB systems try to relate courses according to similar/common characteristics; thus, a course would be recommended with a higher probability if it has similar characteristics with other courses viewed by the user. With this concept, the system does not need a big amount of user data but it requires a normalized representation of the course content to find similar courses and recommend the ones similar to those seen/visited by the user. As consequence, the system recommends courses on the same area of the courses that the user sees, reducing the serendipity. The course content is usually text so the CB system requires a special representation of the text, such that it allows for similarity computation. The most common technique in the literature to represent the course content is by using topics models such as Latent Dirichlet Allocation

(LDA) (Yin et al., 2020; Campos et al., 2020) and Latent Semantic Analysis (LSA) (Cao et al., 2020), however, more recent approaches exist based on deep neural networks, such as auto-encoders and contextual embeddings (Wang et al., 2020).

Xiao et al. (2018) address the cold-start problem via model parallelization. In this approach, the selection of which model to use depends on the sparse matrix grade, *i.e.,* there is a threshold defining when the recommendation system uses one model (*e.g.,* CF) or the other (*e.g.,* CB). In the end, both systems work independently (this could be very useful to replace or modify a recommender system according to a new type of data users), and a ruler of switching decides which system to use. Shanshan et al. (2021) use an approach based on representing content via ontologies. They represent the relations between users and courses in an ontology, and use it as input for the CF model. Recommendations are refined from the CF by using association rules. Do et al. (2017) creates a factorization matrix and a knowledge model based on rules to classify users into different groups, generate different recommendations, and merge them with a weighted hybrid method.

Madani et al. (2020) exploit social networks to automatically create profiles and generate recommendations based on the most similar users. In this case, for the cold-start problem, the authors propose the use of a reinforcement learning model that receives as input the profile and generates the recommendations as output. Mawane et al. (2020) propose a CF system based on deep learning. They group the users using personal data, the volume of interactions with the site, and the score in the evaluations to cluster users, then they use two neuronal networks. The first one extracts the most relevant items in the cluster, and the second one predicts the score in the common courses of the cluster. In the end, they merge the results according to custom rules.

The aforementioned works operate in rich information scenarios. In these scenarios, demographic information, evaluation results, perception surveys, or social networks profiles are available to facilitate the construction and evaluation of the recommender system. We also found in other surveys the absence of works that address scenarios of low information about the user (Uddin et al., 2021). The recommender system we present in this paper combines the CF and CB strategies. However, we do not rely upon explicit user information. Our CF works only with ratings estimated from logs' events extracted from Google Analytics. Thus, we present dedicated strategies for counting lessons seen by a user, processing the timelines, and estimating ratings from this kind of logs. Another difference is related to the expected number of new users. Around 70% of the total traffic of GCFGlobal is from new users. This is a common behavior in the period that was analyzed (one year) and poses a challenge in relation to the cold start problem. Based on the above challenge, we propose a hybrid recommendation system that combines the CF with a CB system via a weighted strategy. Our goal is to combine the best strategy according to the literature review (CF) with a strategy that allows us to deal with the cold start problem (CB). Additionally, since we want to operate in multiple languages (Spanish, English, Portuguese), for our CB we propose the use of multilanguage contextual embeddings. According to Uddin et al. (2021) the research in languages like Spanish and Portuguese is quite limited, so this paper also constitutes a breakthrough in strategies to break language barriers in recommender systems.

3. Proposed recommendation system

Figure 1 shows the components of our recommendation system. The objective is to recommend to a user a course that meets their needs and interests. As noted above, the literature suggests that

for educational platforms, a CF strategy is the best recommendation model option (Shanshan et al., 2021). However, the web traffic of the GCFGlobal learning platform comes from new users (*i.e.,* they are using a device not previously seen in the logs), therefore a recommendation system based only on a CF model could be strongly impacted by the cold start problem. For this reason, we propose a hybrid model that combines a CF and a CB model which enables us to give suggestions to returning and new users alike.

We want our recommendation systems to be improved based on the new information available. Therefore, we propose an architecture based on interconnected independent components that allow future updates or changes without generating a big impact. Both content-based recommendation and collaborative filter-based recommendation can operate independently, and they are fed with different information sources.

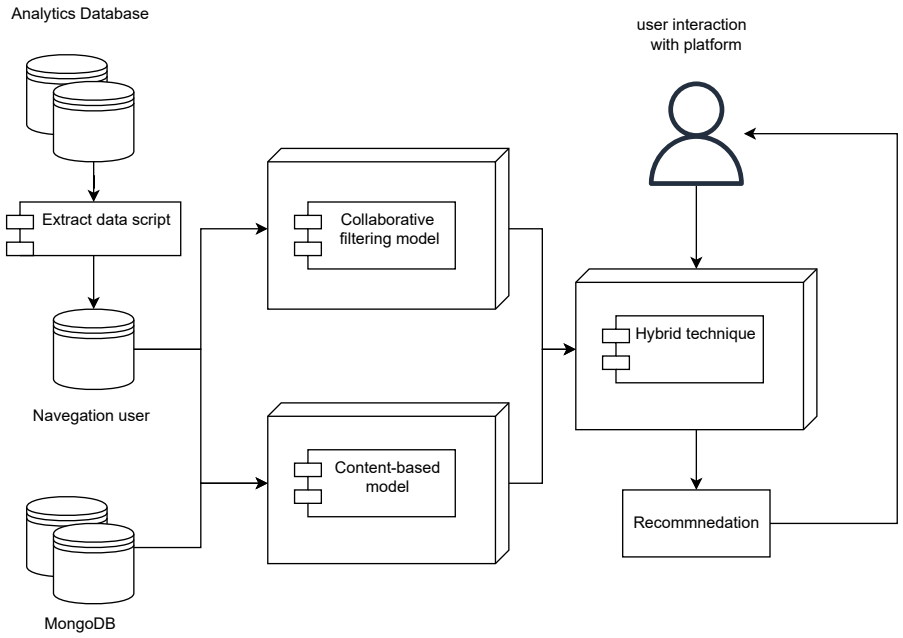


Figure 1: Recommendation system components.

The GCFGlobal data sources used by the recommendation systems are, on one hand, Google Analytics, which stores the user interactions with the courses and provides the information used to build the CF model. On the other hand, a Mongo database containing the contents of the lessons and courses of GCFGlobal . The Mongo database is the source of information for the CB recommendation model. The steps of data enlistment and data processing for each source are presented in Section 3.1. The construction of the recommendation models and the hybrid strategy is presented in Sections 3.3, 3.2, and 3.4 respectively.

3.1. Data processing

In the GCFGlobal platform a course is made up of lessons. Each lesson is self-contained at the content level and is stored as HTML in the MongoDB database. At the navigation level, each lesson is basically an independent web page within the platform. User interactions with the lessons are registered as logs on the Google Analytics platform. As there is no mandatory user registration on the GCFGlobal platform, a unique identification of the user is a challenge in

the logs. The following paragraphs present the processing performed on both the data extracted from Google Analytics and the data extracted from MongoDB.

3.1.1. Google Analitycs logs processing

The events generated by the actions of a user on the platform are consolidated in a Google Analytics session. The end of a session is an automatic event after 30 minutes of inactivity[2]. Inside the sessions, we are interested in the "PageView" events that allow us to know the URLs that the user visited. The URLs in the GCFGlobal platform are organized in such a way that it is possible to easily know the course and the lesson that the user has visited. URLs are specified as: "language/course/lesson/additional-params".

To group the sessions of the same user over time, we assume that a user accesses the platform through a single unique device. This statement is true in most cases, but we cannot guarantee it. User registration is not mandatory in GCFGlobal , and the number of users that are registered is low (2%). Therefore, there is no additional information that can be used to improve user session identification; under those conditions, the most suitable approach is to take advantage of the Google Analytics device identifier.

We use two filters to select users and lessons from Google Analytics. The first filter removes sporadic users, keeping users with at least 30 sessions during the year (2021-01-31 to 2022-01-31). The second filter ensures that the sessions belong to at least three different courses. This filter allows us to guarantee the future construction of the dataset and ground truth to evaluate the recommendation system. We use at least two courses to build the ratings for the collaborative filter and the third one for its evaluation. It is important to mention that the temporality of the sessions is taken into account in the construction of the dataset. The first two courses in temporal order are used in the construction of the collaborative filter and the last one visited is used to evaluate the recommendation. We also consider the session time factor for lesson counting, as explained in the following paragraph.

After building a base of users and sessions, we proceed to perform the pre-processing steps described in Figure 2. In step 1, the base of users and sessions extracted from Google Analytics is loaded into a local database in order to be able to perform transformations on the data in an easy and efficient way. In step 2, the timestamp column is transformed and standardized in such a way that the records can be ordered from the most recent session to the most distant. As mentioned previously, this allows us to identify the chronological route that the user made through the courses and their lessons. In step 3, we identify the time spent by the user in each lesson ("lesson duration"). Step 3 is not trivial as the different sessions in which the same lesson is visited must be reconciled. This reconciliation includes an intra-session analysis followed by an inter-session consolidation.

In the intra-session analysis, the time that a user spent in lessons of the same session is obtained. The time is calculated as the difference between the timestamp of the "PageView" event of the lesson URL and the immediately following event timestamp if it exists. In the case that the session ends with a visit to a lesson and there is no subsequent event for the calculation, we use imputation using the average lesson time over all users. For lessons with multiple visits in the same session, the times obtained were added. Table 1 shows an example of the calculation of the time spent in the lessons within the same session. Being the last event of the lesson, the time of lesson “D” is imputed using the average duration of the lesson over all users. Then

[2]Google Analytics forum: https://bit.ly/3IRlAAY

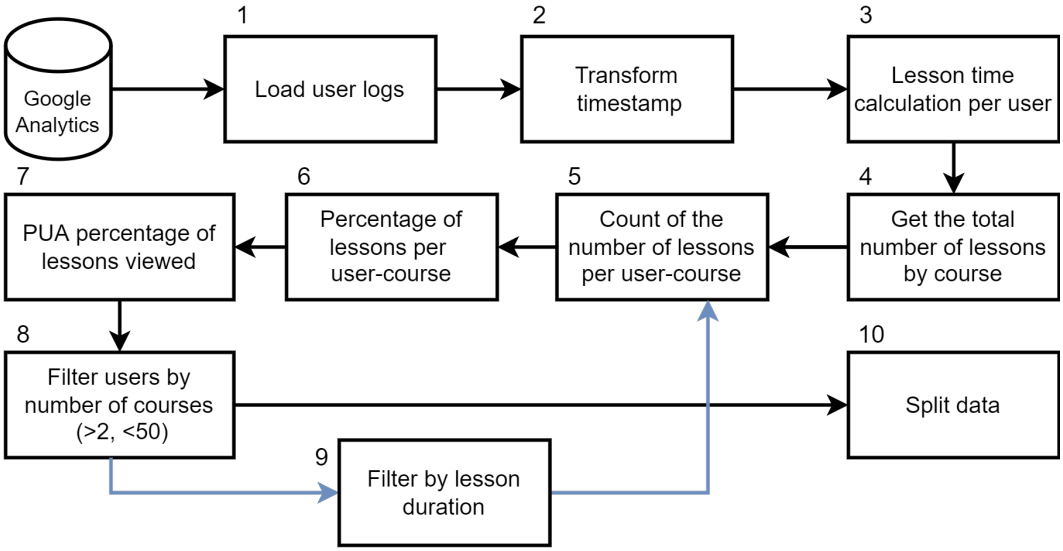


Figure 2: Data processing steps.

an inter-session level analysis is carried out where the times of the lessons that were visited in different sessions are added to consolidate a total "lesson duration" per user.

Table 1: Intra-session analysis example. Four "PageView" events of lessons A, B, C, and D. The time spent in lesson D is calculated as the average of all users.

Lesson	Timestamp	Time Spent (min)
A	2021-01-29 08:49:01	20
B	2021-01-29 09:09:01	18
C	2021-01-29 09:27:01	21
D	2021-01-29 09:48:01	17

In steps 4 and 5, the number of lessons that make up each course is consolidated. Then, for each user-course combination, the number of lessons viewed is counted. With these results, in step 6, the percentage of lessons viewed is calculated per user-course. In step 7, we estimate the average percentage of lessons accessed per course for each user. This personalized metric indicates how much of a course a particular user usually viewed. We named this metric “PUA (per-user-average)”.

In step 8, users who have seen lessons from more than 50 courses are removed. These are considered outliers because they are more than three standard deviations from the mean (Ilyas and Chu, 2019, p.19). From 7956 users considered, four were removed by this rule, one of them with a total of 131 courses viewed. Considering the minimum three courses filter explained at the beginning of this section, our dataset only contains users who have seen courses in the range [3, 50).

In step 9, we add a filter to the count made in step 5 according to the user’s lesson duration. Lessons below a stipulated time are not considered. The exclusion of lessons by duration is controlled by a parameter, and the effect of this parameter is evaluated in the experimentation. Our hypothesis is that the inclusion of this filter will allow discarding lessons that a user addressed

lightly and possibly incompletely, thus improving the results of the recommendation.

Finally, in the last step (step 10), we split the data into a subset for training (*i.e.,* construction of the collaborative filter) and another for testing. The split maintains the temporal order, therefore, the test set always has courses viewed after those in the training dataset per user. We use the first 70% of the courses for building the collaborative filter and the last 30% for evaluation. The resulting dataset constitutes our ground truth. It is important to mention that in the worst case, when a user has only seen three courses, two will be used for training and one for evaluation.

3.1.2. Course content processing

Different information was identified for the construction of the CB recommendation inside the MongoDB: (i) the description of the course (cou-desc) which is usually a short text of an average of 20 words, (ii) the descriptions of the lessons (le-desc) that compose the course, which are also short texts with an average of 127 words, and (iii) the lessons HTML content (le-content) which is the complete learning content of a course that is shown to the user. In total, three corpora were built, one for each source of information. For "le-desc" and "le-content", we concatenated all the descriptions and HTML content of the lessons that compose a course. Then, we removed all the HTML tags and other special tags used by the GCFGlobal staff. In Table 2 we present a summary of the characteristics of the corpus.

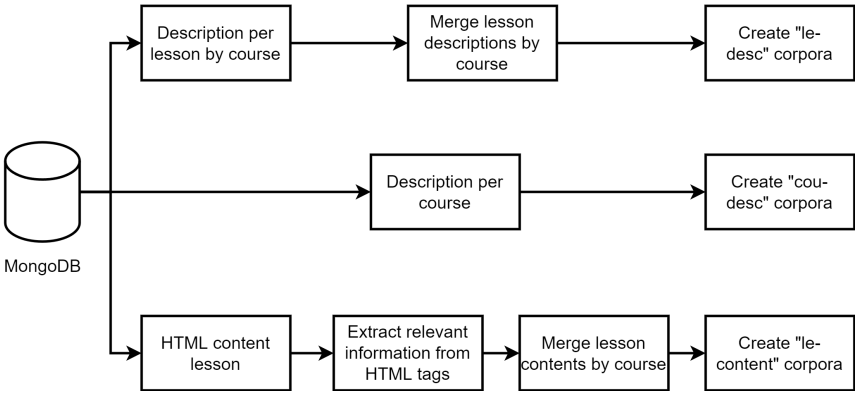


Figure 3: Corpora creating processing.

The content-based recommendation seeks to find courses similar to those that the user has already seen. It works under the assumption that the courses that the user has already seen are courses that they liked and therefore other courses similar to those already seen may be of interest to them. The key in the CB recommendation is the estimation of the similarity, which in our context means similarity between textual elements. In order to calculate this similarity, we propose the transformation of text to a vector representation using, on the one hand, topic modeling strategies (LDA and LSA) and, on the other hand, contextual embeddings (MPNET, RoBERTa, etc) that constitute the state of the art in many textual similarity tasks (Reimers and Gurevych, 2019). While for contextual embeddings no additional text processing is required, for topic-based models a typical workflow from the natural language processing area should be applied. In the next paragraph, we explain this pipeline.

A conventional natural language processing (NLP) workflow is used, where all words are lowercased, punctuations and stop words are removed, stemming is applied to reduce words to

Table 2: Course average number of sentences and words per corpus.

	cou-desc	le-desc	le-content
Average number of sentences	1.17	11.21	216.68
Average number of words	20.30	127.47	3704.27

their root form, and a token is generated for each unique word in the corpus. Although the test corpus is in English, the system in the production environment must support other languages such as Portuguese and Spanish. For this reason, we opted to carry out a stemming process and not a lemmatization process since, although it is not perfect, there are stemming libraries for most languages. Finally, the tokens that have a frequency of appearance greater than 95% (calculated on the total number of courses) are removed. This operation is performed in order to remove words that, being common among all the courses, do not provide relevant information to the model. Figure 4 presents the processing performed in the construction of the corpus used.

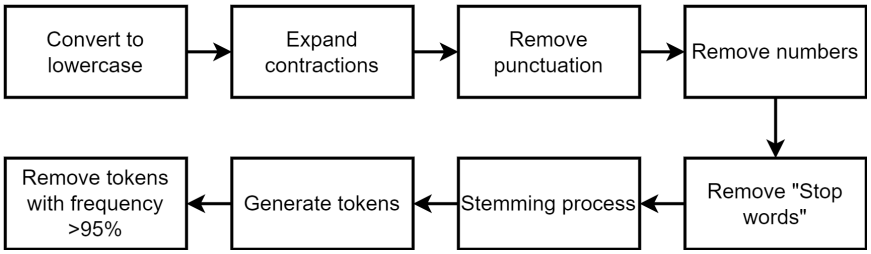


Figure 4: NLP processing for LDA and LSA topics models.

3.2. Collaborative filtering recommendation

CF requires a rating matrix to find similar learners/courses. It is necessary to infer these ratings from Google Analytics logs because the GCFGlobal platform does not ask for them explicitly. A course is composed of a set of lessons, and the number of lessons seen by the learner is used as an estimate of the rating. Being open courses where the courses and lessons are approached by the will of the learner, it seems valid to assume that the perception about a course is reflected in the number of lessons taken by the learner.

To estimate ratings we use a threshold between 0 and 5, common in recommender systems (Aggarwal, 2016a, Ch.2). Courses with a percentage of lessons viewed above a threshold will obtain the maximum grade (5), and those that are below will be penalized according to a penalty function. We use as threshold metric the average percentage of lessons viewed per course by a user, that is, the PUA metric explained in Section 3.1.1. The use of PUA is inspired by the different ways of calculating similarities (Aggarwal, 2016a, Ch.2), where knowing if a rating is above or below the average is considered more valuable than the rating itself. It is known that user ratings in different domains tend to be very close to the average. While a rating in the middle does not say much, a rating far from it is a clear indication of like or dislike.

For the rating penalty, five different functions were evaluated: logarithmic, square root, quadratic, linear, and step (Equations 1 to 5).

$$f_1(x) = \begin{cases} 5, & x \geq \text{threshold} \\ 5 * \sqrt{\frac{x}{\text{threshold}}}, & x < \text{threshold} \end{cases} \quad (1)$$

$$f_2(x) = \begin{cases} 5, & x \geq \text{threshold} \\ 5 * log_2((\frac{x}{\text{threshold}}) + 1), & x < \text{threshold} \end{cases} \quad (2)$$

$$f_3(x) = \begin{cases} 5, & x \geq \text{threshold} \\ 5 * \frac{x}{\text{threshold}}, & x < \text{threshold} \end{cases} \quad (3)$$

$$f_4(x) = \begin{cases} 5, & x \geq \text{threshold} \\ 5 * (\frac{x}{\text{threshold}})^2, & x < \text{threshold} \end{cases} \quad (4)$$

$$f_5(x) = \begin{cases} 5, & x \geq \text{threshold} \\ 0, & x < \text{threshold} \end{cases} \quad (5)$$

Figure 5 shows the penalty functions for a particular threshold from which it is possible to identify that the degree penalty order of the functions correspond to: F1 (less penalization)-F2-F3-F1-F5 (more penalization).

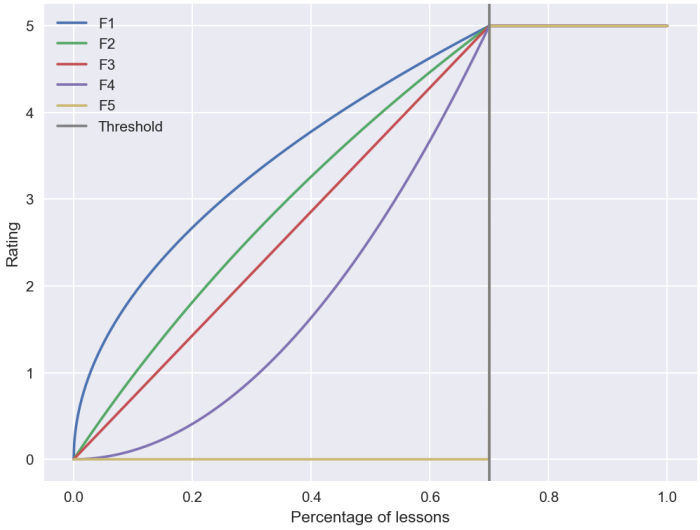


Figure 5: Behavior of penalty functions with a threshold of 0.7 (70%) for a given course. The estimated rating is 5 if the percentage of lessons accessed is greater than 70%. For lower values, the rating is given by the penalty functions F1-F5.

After estimating the rating matrix according to PUA and the penalty functions, we proceed to generate the recommendation. We follow a classic collaborative filter item-based approach. Our collaborative filter uses the `turicreate` and cosine similarity to identify similar courses based on ratings (Equation (6)).

$$CS(i,j) = \frac{\sum_{u \in U_{ij}} r_{ui} * r_{uj}}{\sqrt{\sum_{u \in U_i} (r_{ui})^2} * \sqrt{\sum_{u \in U_j} (r_{uj})^2}} \quad (6)$$

where U_i is the set of users who rated item i, U_{ij} is the set of users who rated both items i and j, r_{uj} is the rating for the item j by the user u, and r_{ui} is the rating for the item i by the user u.[3]

The library has a default threshold of 0.001 where users with a lower similarity coefficient are excluded when making the rating prediction. Given the similarity scores between items based on ratings, we generate a score for a particular user-course combination using a weighted average of the user's previous ratings. This score is used to generate the top-N recommendations.

3.3. CONTENT BASED RECOMMENDATION

As mentioned, the content-based recommendation system seeks to recommend courses that are thematically similar to those that the user has already seen. In general, the content-based recommendation process involves four steps:

1. Transform the text of the courses in a vectorized form that allows the calculation of the cosine similarity.

2. Build a simple user model composed of the vector representations of the courses that the user has seen.

3. For each course of the user's model, calculate the similarity with the rest of the courses not seen by the user. This generates a vector of similarities for each course in the user's model.

4. Average for each user model the similarity vectors obtained in step 3. Extract from this average vector the top-N similar courses.

For step 1 we used two topic-based representations widely used in recommender systems: LDA and LSA (Wang et al., 2020; Yin et al., 2020; Ma et al., 2021). In topic-based representations, each course is represented as a mixture of topics, and each topic consists of a collection of words. The resulting representation is, therefore, a vector of length "num-topics" representing the weighted presence of each topic in the doc. "num-topics" is an input parameter, that has to be estimated experimentally, so we try different values and select the one that maximizes the recommendation precision (Section 4.2).

Additionally, we evaluated recent representations based on contextual embeddings. In contextual embeddings, each word is associated with a representation that is a function of the entire input sequence (i.e. the entire sentence). Recent research has obtained state-of-the-art results using contextual embeddings pre-trained on large-scale unlabelled corpora in tasks such as text similarity, semantic search, question answering, and automatic summarization, among others (Liu et al., 2020). Of the different contextual embeddings, we are interested in the so-called "sentence transformers". The term "transformer" refers to a family of neural network architectures that compute dense, context-sensitive representations. A detailed description of this architecture can be found at (Reimers and Gurevych, 2019). The term "sentence", on the other hand, refers to the fact that a representation is produced for text composed of multiple tokens or words. Table 3 lists the selected sentence transformers.

The selected sentence transformers were the best reported by the "Sentence Transformers" library benchmark according to the metric of "performance sentence embeddings" (Reimers,

[3]Turicreate documentation: https://bit.ly/3y0vJsx

Table 3: Selected Sentence Transformers.

Sentence Transformer	Huggingface Model	URL	Embedding size
MPNet Song et al. (2020)	all-mpnet-base-v2	https://bit.ly/3Nlk2Bb	768
	paraphrase-multiligual-mpnet-base-v2	https://bit.ly/39y8cVO	768
RoBERTa Liu et al. (2019)	all-roberta-large-v1	https://bit.ly/3wmJK2v	1024
MiniLM Wang et al. (2020)	all-MiniLM-L12-v2	https://bit.ly/3lCIsdL	384
	paraphrase-multilingual-MiniLM-L12-v2	https://bit.ly/3lhUxF1	384

2022). The selected transformers differ from each other in their architecture or the way they are trained. MiniLM, for example, applies a compression strategy (termed deep self-attention distillation) to obtain a reduced version of the model. MPNet changed the optimization problem from an MLM (Masked Language Modeling) to a PLM (Permutative Language Modeling), and RoBERTa increased the training dataset by 10x. These differences are reflected in the resulting vector that represents the text, so it is expected a different recommendation performance. In addition to the English version of the selected transformers, its multilanguage versions (if they exist) were also selected. The multilanguage versions can operate in 50+ languages, including Spanish and Portuguese, languages in which the GCFGlobal platform also operates. The purpose of including them is to discover whether or not there is a degradation in the recommendation due to their use. This can give us clues about the most appropriate sentence transformer to use when including courses in various languages in the recommendation system. Adding the "topic-based" models and the "sentence transformers", we have a total of 7 different representations of the text to evaluate and compare. The representation size (*i.e.,* vector dimension) depends on the sentence transformer used.

In step 2, we take the representation of each course seen by each user to build a user model. It is important to clarify that we do not operate in any way the different course representations to obtain a single vector per user. Although it is common practice, in our production environment it is more appropriate to keep the different course representations as user models. This strategy allows us in steps 3 and 4 to generate recommendations working only on a pre-calculated similarity matrix (course-course) instead of having to calculate a new user vector and their similarities against all the courses.

At the implementation level, the Gensim library (Řehůřek and Sojka, 2010) was used for topic modeling, and the "Sentence Transformers" (Reimers and Gurevych, 2019, 2020) library with the pre-trained models of Table 3 for the contextual embeddings.

3.4. Hybrid approach recommendation

Based on the recommendations generated by CF and CB, a weighted strategy that combines them was proposed. This hybrid strategy uses as inputs per user two vectors:

- The score user vector of the CF model (V_{CF}), where each dimension is bounded in [0, 1].
- The similarity average user vector of the CB model (V_{CB}), where each dimension is bounded in [0, 1].

At the implementation level, we run into the challenge of reconciling the dimensions of both vectors. Due to the way the turicreate library implements CF, the resulting vector only has as dimensions the courses that have been seen by users and of which there is at least one rating. Of

the 213 courses, 23 do not appear in the dimensions of the CF vectors. The reconciliation process adds these 23 dimensions with zero values to the CF vector and ensures that each dimension in both vectors corresponds to the same course.

Using these vectors, a hybrid weighted score vector (V_{HY}) is created as in Equation (7).

$$V_{HY} = V_{CF}\alpha + V_{CB}(1 - \alpha) \tag{7}$$

where the α parameter controls the tradeoff between the contribution of CF and CB. α varies between 0 and 1; a value of 1 indicates a full contribution from the CF model, while a value of 0 indicates that only the CB model will be taken into account. In the experimentation phase (Section 4), we find the α value that leads to the best precision in the recommendation. To select the top-N courses to recommend, we calculate the V_{HY} for each user.

4. Experiments and Results

In this section, we present the experiments and results obtained by the CF, CB, and Hybrid strategies. Precision at k ($P@k$) was used as a metric to evaluate the recommendation performance (Pan et al., 2021; Yin et al., 2020; Liao et al., 2020; Manrique and Marino, 2018). The ground truth dataset of our evaluation is extracted from Google Analitycs logs, explained in Section 3.1.1. This dataset contains the courses viewed by 7071 users in temporal order. For each user, we use the first 70% of the courses viewed for building the recommendation system model (Sections 3.2, 3.3, and 3.4) and the last 30% of the courses for evaluation.

In the context of learning environments and learning content, it is common to provide top-N recommendations with $k = N \in [5, 10]$ (Pan et al., 2021; Yin et al., 2020). Delivering more than 10 recommendations (*i.e.,* courses) is not common because they can overwhelm the learner, particularly in unknown or partially known knowledge domains. On the other hand, values less than 5 can cause the set of results to be over-specialized in a single topic and not be diverse. In our case, the selection of a fixed value of k in the above range poses a challenge due to most of the users do not contain five relevant courses (*i.e.,* courses seen by the user) in our evaluation set. Therefore we selected a personalized value of $1 \leq k \leq 5$ for each user according to the number of courses in the test set. The results reported in all the experiments are the personalized $P@5$ average over all the 7071 users using the evaluation dataset.

We also run hypothesis tests to compare the different recommendation strategies or the influence of changes in their parameters or inputs. The tests were performed as follows:

- Generate 100 subsets of 3000 users from the full dataset.
- Calculate the global average $P@5$ of the recommendations generated by the two models to be compared over the 100 subsets.
- With the precision results, use a Wilcoxon rank-sum test with the Bonferroni correction to test statistical significance ($\rho < (0.05/\#\text{hypotheses})$).

4.1. Collaborative filter recommendation results

Table 4 presents the results obtained using the rating strategies explained in Section 3.3. According to the results on the dataset with the total number of users, F1 and F4 are the best penalty

Table 4: Precision results for the different penalty functions. In bold the best penalty functions.

Penalty Functions	Precision @ 5
F1 (Square root)	**0.3911**
F2 (Logarithmic)	0.3783
F3 (Lineal)	0.3759
F4 (Quadratic)	**0.3928**
F5 (Step)	0.1677

functions. To identify if there are significant differences in the precision obtained by different combinations of penalty we built the 100 subsets following the methodology explained above.

Table 5 presents the average precision obtained per penalty function over the 100 experiments. Regarding the penalty functions, an interesting behavior is evident: the lower the penalty, the better the recommendation. We tested if the results obtained via F1 were statistically significant in comparison with the other penalty functions. The test results are reported in Table 6 and show that results obtained via F1 are statistically significant compared to all other functions except for the quadratic function (F4).

Table 5: Average precision over 100 random datasets.

Penalty Functions	Avg. Prec. @ 5
F1 (Square root)	0.246
F2 (Logarithmic)	0.241
F3 (Lineal)	0.239
F4 (Quadratic)	0.225
F5 (Step)	0.09

Table 6: F1 vs other penalty functions via Wilcoxon test.

Penalty Functions	ρ-value ($\rho < (0.05/4)$)
F1 (Square root) vs F2 (Logarithmic)	8.400e-05
F1 (Square root) vs F3 (Lineal)	7.837e-09
F1 (Square root) vs F4 (Quadratic)	0.033
F1 (Square root) vs F5 (Step)	1.261e-34

The results suggest that less penalty leads to better recommendations. This indicates that a low number of lessons viewed per course cannot be directly related to low preference. Being open learning resources where the learning process is self-directed, there are many other variables that can affect the non-continuity in the learning process that are impossible to identify in our context (*i.e.,* learning priorities, time, health). It should also be taken into account that since it is an implicitly extracted rating, it is to be expected that it does not accurately reflect the user's preferences. The results also suggest that the F1 (square root) penalty function leads to better recommendation results, and the difference with F2, F3, and F5 is statistically significant.

Our final experimentation is related to the lesson duration (*i.e.,* Step 9 in Figure 2). Figure 6 shows the distribution of the lesson duration in minutes. Lessons lasting more than 35 minutes are considered outliers and thus deleted (a total of 4711 records were deleted from 289789). We wanted to see the effect on the recommendation precision of adding a lesson duration filter prior to counting and calculating the percentage of lessons viewed. The filter removes lessons with a duration less than x value. We used PUA as a threshold metric and F1 as a penalization strategy. Figure 7a shows the results obtained as the filter becomes more restrictive. One of the consequences is that as the minimum lesson duration time is restricted, the size of the dataset is reduced. This means that, after applying the filter, there were users with fewer than three courses and they were, thus, removed from the analysis. Figure 7b shows how the size of the number of

users decreases with the increase of the lesson duration filter.

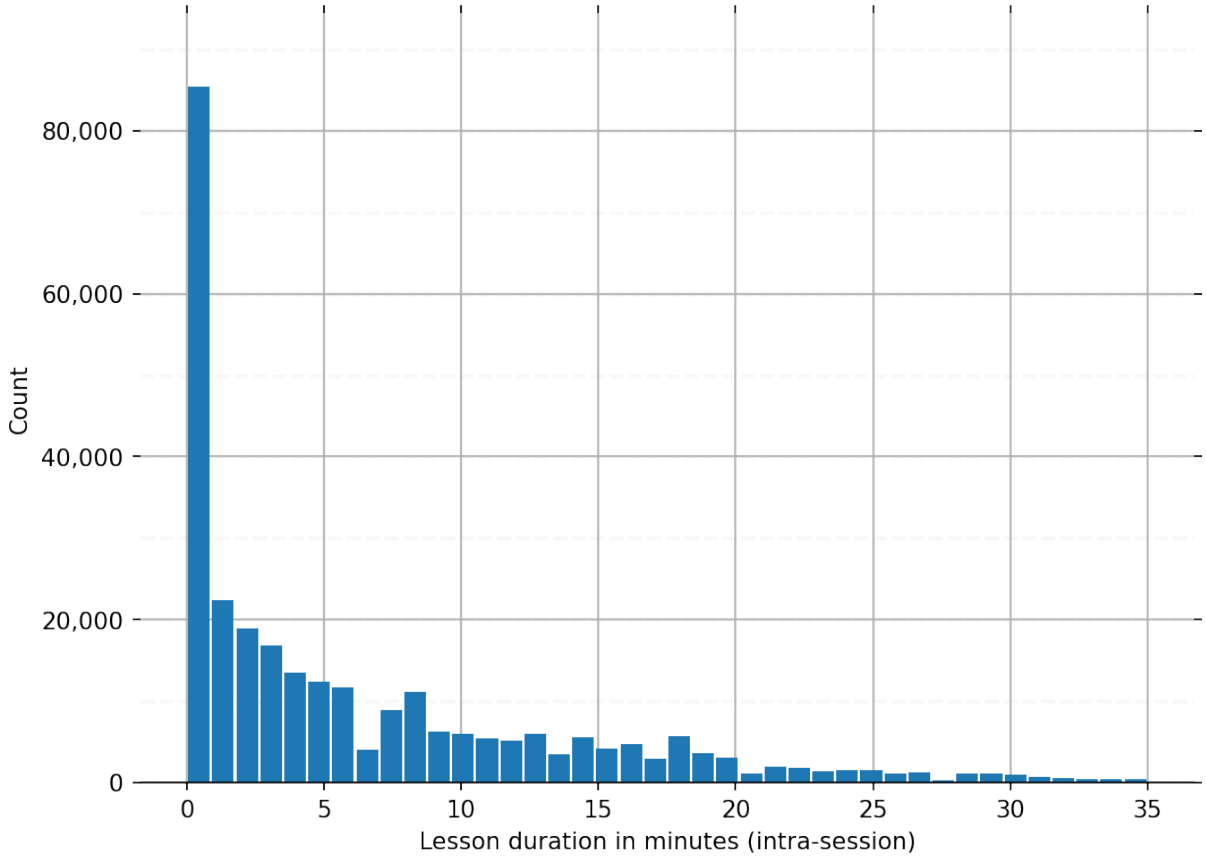


Figure 6: Distribution of the lesson duration after the intra session analysis.

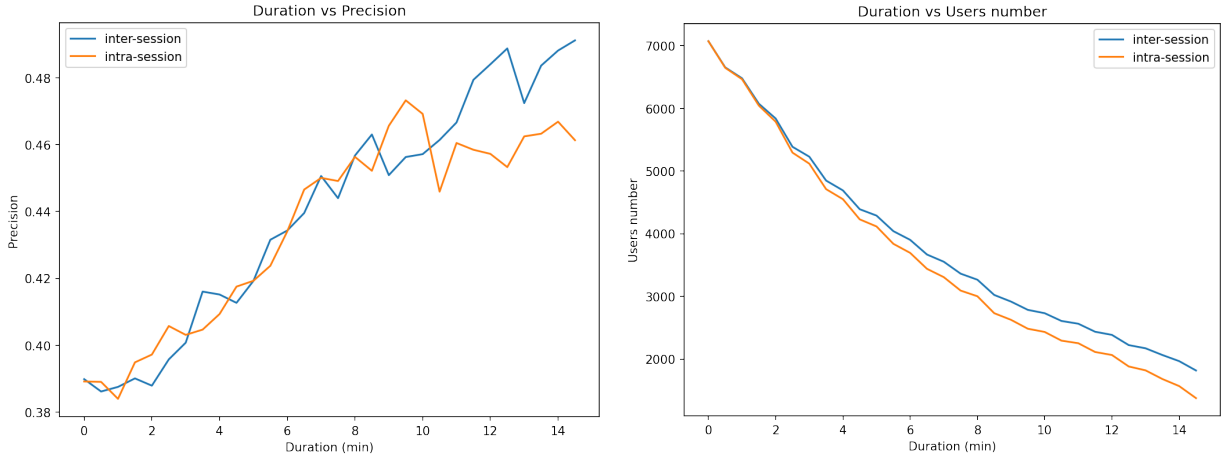
4.2. CONTENT BASED RECOMMENDATION RESULTS

We first present the results of selecting the "num-topics" parameter for topic-based models. This parameter was varied in the interval [5, 360] in increments of 5 in order to find the value that maximizes the accuracy of the recommendation. We used the corpus "le-content" in this experiment and a maximum number of 50 iterations through the corpus when inferring the topic distribution. Figure 8 presents the results obtained for LSA and LDA.

For the LDA model, different peaks are generated for different values of "num-topics". The highest value recorded is found when the number of topics is 225 (0.161). For the LSA model, after 160 topics the precision tends to stabilize around 0.167. Although the results of both models are very similar, it should be noted that the LSA model has a lower computational complexity and execution time. For the following experiments, the values of 225 and 160 were used as the number of topics for LDA and LSA respectively.

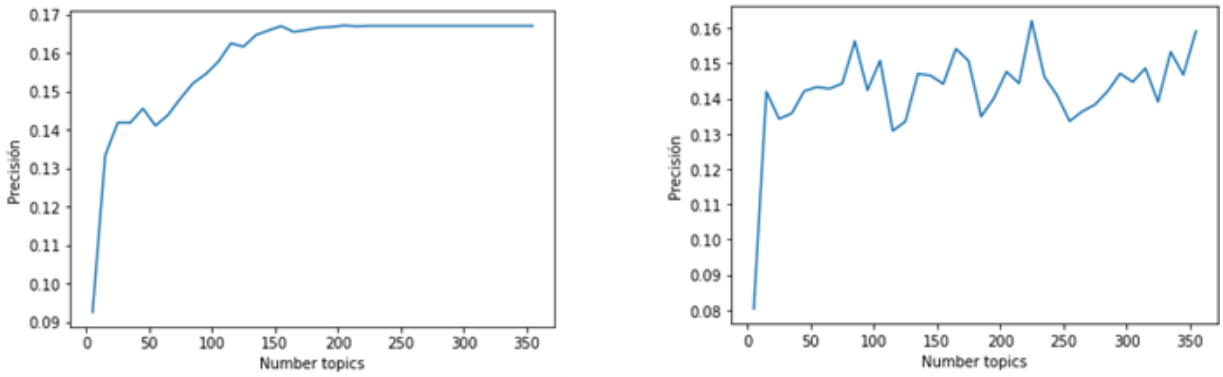
Next, we compare all representations (topic-modeling and sentence transforms) using the "le-content" corpus. The results obtained are presented in Table 7.

These results show that in terms of recommendation precision, MPNet is superior as a textual representation. We perform a statistical significance following the methodology explained above where the null hypothesis is that the $P@5$ obtained by MPNet has the same distribution that one



(a) Lesson duration filter vs recommendation precision. (b) Lesson duration filter vs number of users in dataset.

Figure 7: Lesson duration filter effect.



(a) LSA. (b) LDA.

Figure 8: LSA and LDA “num-topics” parameter influence.

Table 7: Recommendation results using different text representations. Topic modeling representations (LSA and LDA), and Sentence Transformers representations (MPNet, RoBERTa and MiniLM). The word "multilingual" describes those models that can operate in multiple languages, in particular English, Spanish, and Portuguese.

Text Representations	$P@5$
MPNet	0.1872
LSA	0.1655
LDA	0.1621
RoBERTa	0.1530
MPNet (multilingual)	0.1507
MiniLM	0.1493
MiniLM (multilingual)	0.1398

of the other text representations. The alternative hypothesis is that the distribution of $P@5$ obtained by MPNet is stochastically greater than the distribution of the other. We repeated this test for each text representation, therefore a total of six hypotheses were evaluated. The results of the tests are shown in Table 8.

Given that all the tests are below the acceptance threshold of the null hypothesis, we can affirm that the MPNet results are statistically significant and constitute our best textual representation strategy.

Table 8: MPNet vs other text representations via Wilcoxon test.

Text Representations	ρ**-value** ($\rho < (0.05/6)$)
MPNet vs LSA	1.34e-20
MPNet vs LDA	1.26e-18
MPNet vs RoBERTa	1.26e-18
MPNet vs MPNet (multilingual)	1.26e-18
MPNet vs MiniLM	1.26e-18
MPNet vs MiniLM (multilingual)	1.26e-18

In another experiment, we wanted to test the influence of the length of the text that represents the course content used for CB construction. In Table 2 three corpora were presented where the description of the course text varies. "le-content" has all the textual content of the lessons that make up a course and is by far the corpus with the longest descriptions. "le-desc" is a medium-length corpus containing lessons descriptions. These descriptions are made by the content designers and can be interpreted as short summaries of the content of each lesson. Finally, "cou-desc" is a corpus that contains a simple description of the course as a whole. The results of using MPNet on these three corpora are presented in Table 9.

The results show that the more information (*i.e.,* more text about the content of the course), the better the recommendation. However, it is worth noticing that the difference is only 0.012 with "le-desc" and 0.0327 with "cou-desc". As the amount of additional processing required for "le-content" is higher than for the other, a significant improvement was expected. The last column in Table 9 presents the results of the Wilcoxon test that validate the statistical significance of the results obtained with the corpus "le-content".

Several conclusions can be drawn from these results. First, the best textual representation was MPNet, which is a pre-trained language model. We did not perform a model refinement process (*i.e.,* fine-tuning); in other words, we used the version reported in the literature without any modification. This is an advantage compared to LSA/LDA models that require a training

Table 9: Recommendation results using MPNet over different corpora. The last column presents the Wilcoxon statistical significance test comparing le-content vs (le-desc, cou-desc).

Corpus	$P@5$	ρ**-value** ($\rho < (0.05/2)$)
le-content (large)	0.1872	-
le-desc (medium)	0.1754	4.26e-13
cou-desc (short)	0.1545	1.26e-18

process to generate the distribution of topics. Second, the results show that the accuracy of the recommendation is improved when the "le-content" corpus is used. This corpus contains the complete content that is shown to the user in HTML, and it is necessary to carry out an extensive cleaning process. The other corpora require a less rigorous process and they show a relatively low decrease in the recommendation precision: 1.18% in the case of "le-desc" and 3.27% in the case of "cou-desc". In a production environment with limited resources, the use of "le-desc" might be more suitable. Finally, LDA and LSA present similar results in all the experiments. However, the training time of LSA is much shorter compared to LDA. [4]

4.3. HYBRID APPROACH RECOMMENDATION EVALUATION

In this section, we evaluate the hybrid strategy. The best models obtained for both CB and CF were chosen. For CF we use the F1 penalty function and for CB we use the MPNet representation and the "le-content" corpus. The hybrid model is based on the combination of the outputs of these two models according to Equation (7). To choose the best alpha value we followed a methodology similar to the statistical significance test, 100 subsets of 3000 users each were generated, then the precision of the recommendation for different values of α in the 100 groups was evaluated and averaged. Table 10 shows the results.

Table 10: Hybrid recommendation results for different α values. Best value in bold.

Alpha	**Average** $P@5$
1	0.400782
0.9	**0.402959**
0.8	0.400917
0.7	0.402134
0.6	0.398614
0.5	0.395000
0.4	0.384213
0.3	0.359337
0.2	0.321633
0.1	0.272146
0	0.197228

The best result of the hybrid recommendation is with a $\alpha = 0.9$, which indicates that the contribution of CF is much higher than that of CB. Finally, we performed the statistical significance test of the hybrid model with a $\alpha = 0.9$ vs. the hybrid models using another value of α. As shown in Table 11, according to the p-values obtained, three of the 10 comparisons accept the null hypothesis. This is because the changes in precision for values of $0.7 \leq \alpha \leq 1$ are very small and on the order of 0.21% - 0.08%.

[4]Ten training processes were carried out for each model, and the average time was used for comparison. For the LSA model, the average time to train the model was $3.24s \pm 9.09ms$, and for the LDA model was $1min36s \pm 299ms$. We use the Gensim library implementation of both models https://radimrehurek.com/gensim/, the same input corpus, and the following hyperparameters: 200 topics, and 50 iterations.

Table 11: Hybrid model with $\alpha = 0.9$ vs Hybrid model with $\alpha = (1, 0.8, 0.7, 0.6, 0.5, 0.4, 0.3, 0.2, 0.1, 0)$ values via Wilcoxon test. In bold the cases that accept the null hypothesis (i.e there is no difference with the hybrid model with $\alpha = 0.9$).

Alpha	**ρ-value ($\rho < (0.05/10)$)**
1	**0.0056221**
0.8	**0.0122907**
0.7	**0.18363191**
0.6	1.3573e-06
0.5	2.1734e-15
0.4	1.6632e-31
0.3	1.2620e-34
0.2	1.2620e-34
0.1	1.2620e-34
0	1.2620e-34

5. Discussion and Future Work

The results clearly show the superiority of CF over CB. In general, according to the literature, when explicit or implicit user information is known, CF outperforms CB (Shanshan et al., 2021). In the context of the hybrid model, the contribution of CB is also very low. The best hybrid models are those that consider the CF model to a greater extent, that is, those with a α between [0.7, 1]. Moreover, via the statistical significance test, we can conclude that there is no difference against a purely CF model ($\alpha = 1$).

Despite these experimental findings, there are reasons why we prefer the hybrid approach. As mentioned in the motivation of the hybrid model, much of the traffic to GCFGlobal comes from new users for whom there is no prior information (cold start problem). Additionally, CF has a limitation regarding the courses it can recommend. If there is a course that has not been visited by users, its probability of being recommended is zero. This phenomenon is related to the long tail effect in recommender systems (Aggarwal, 2016b). In the hybrid model, thanks to the contribution of the CB model, these non-previously seen courses can be recommended.

As an alternative to the weighted hybrid model and based on the low contribution of the CB model, a switch-type hybrid model could be considered. When the user is new or there are courses that have not been seen by any user, the CB recommendation is used, otherwise the recommendation comes from CF. This switch strategy has the additional advantage that it does not have to compute V_{HY}, thus, reducing the computation required to generate the recommendation.

It is difficult to directly compare our approach with others due to the difference in the information available about the learner. As established in Section 2, the reviewed recommendation systems make use of demographic information, evaluation results, surveys, or social networks to build a learner profile (Uddin et al., 2021). Since we don't have any information other than the event logs, it is not possible to directly compare our results with other course recommenders systems in the literature. Our approach operates in low users information scenarios insofar as we deduce the learner's preferences implicitly through the logs of interaction events with the platform. Although our approach has limitations as the unique identification of a learner, it is possible to generate a recommendation with good precision ($P@5 = 0.4$).

The strategies proposed in this paper can be applied to contexts other than course recommendations. The rating estimation presented in Section 3.3 can be used in any recommendation scenario where there is no explicit information about the user, but there is implicit information in the form of access event logs. At a minimum, said event log must contain a session identifier, the access event to a particular URL, and the respective timestamp. The use of contextual embeddings as textual representation could be also used in other contexts, particularly in those where the elements to be recommended are in multiple languages. Multilingual embeddings make it easy to manipulate content in different languages through a single representation. The above feature is ideal since it avoids the design of multiple content-based recommenders, one for each language in which you want to operate. We believe that this may be a promising idea for further research, in particular, to break language barriers in recommender systems (Uddin et al., 2021).

Finally, now that we have found an appropriate strategy to build the ratings from logs and text representations, we want to explore how to adapt recent deep learning approaches (Wu et al., 2022) as future work. We also want to explore content representations using semantic descriptions that can be enriched via knowledge graphs (Grévisse et al., 2018). Semantic enrichment strategies allow finding links between concepts that, although they are not explicit in the content to be recommended, are in the knowledge graph. By enriching the course representation with these new semantic links, we hope to improve the recommendation.

5.1. Limitations

A limitation of our approach is related to the assumptions on which the ratings were estimated. As a consequence of assuming that a user accesses the learning platform using a single device, the following scenarios could threaten the validity of our results. In the first scenario, a single user using two different devices could have been analyzed in our experimentation as two different users, or the interactions with one of the devices could have been discarded if they did not meet the minimum number of events needed in the logs. We can summarize this scenario as a "data fragmentation" scenario, and it has the consequence that a user sees different recommendations depending on the device used. A second scenario is related to the fact that multiple users access the learning platform using the same device. This scenario can occur, for example, in shared computer rooms in colleges or universities. Due to the impossibility of individualizing users' interactions, the recommendation operates for the collective, and as a consequence, the individual user experience could be harmed.

We hypothesize that the better the users and their interactions can be identified (by promoting or making user registration on the platform mandatory), the better our recommendation system will be. Although it seems obvious, we currently have no way to prove it. Our future work will be aimed at validating this hypothesis and evaluating strategies for the complete and unique identification of users. This undoubtedly requires profound changes in the platform and the enrichment of the logs with a broader set of events.

6. Conclusion

In this paper, we presented a course hybrid recommender system for limited information scenarios. An important challenge that we addressed is the estimation of user course ratings from implicit log information. Our estimation technique with penalization using events from Google

Analytics logs allows us to produce a CF model with a P@5 of 0.4. In other words, two of the five recommended courses are of interest to the user. We consider this result outstanding in the absence of explicit user information. These results also validate the efficiency of the CF system in the e-learning field as the literature review suggests.

We also reach important practical conclusions for rating estimation from event logs. First, the proposed "per-user-average" (PUA) threshold was appropriate for the recommendation task. A second practical conclusion is about the use of penalty functions: the lower the penalty, the better the quality of the recommendation. This indicates that a low number of lessons viewed per course cannot be directly related to low preference, so in limited information scenarios, the penalty has the negative effect of further reducing the knowledge about the user's tastes. We also validated that a time session filter prior to the rating estimation improves the final results of the recommendation, yet causes a drastic reduction in the data size. In a production environment, this filter is not appropriate, since it could reduce the number of users that could benefit from the CF system.

Regarding the CB system, the best vectorization strategy was the use of the MPNet transformer model. Although the difference with classical topic models like LDA is not high, the transformers have been used without any further training or fine-tuning process. Additionally, there are versions of these transformers that can operate in multilingual environments without a significant impact in terms of precision, as our results suggest. In comparison with the CF model, the CB has less precision in terms of the recommendation but it only needs the course content and the information of at least one course visited by a user to generate a recommendation. In the context of GCFGlobal, this factor is very important because the number of new users on the site is around 70% of the total traffic.

Finally, the hybrid system using a weighted strategy slightly improves the results of the CF system. The improvement obtained is not significant in comparison with CF alone, yet it is necessary to face the cold-start problem. Other hybridization strategies can be easily explored (e.g switching) because the current architecture allows for improving each system model independently.

7. ACKNOWLEDGMENTS

The authors would like to thank GCFGlobal Learning Program for funding this project, its continuous interest in improving its learning services, and its support in consolidating the dataset used in this paper.

REFERENCES

AGGARWAL, C. C. 2016a. *Ensemble-Based and Hybrid Recommender Systems*, 1st. ed. Springer International Publishing, Cham, Chapter 6, 199–224.

AGGARWAL, C. C. 2016b. *Evaluating Recommender Systems*, 1st. ed. Springer International Publishing, Cham, Chapter 7, 225–254.

BRANDON, D. 2019. Google analytics. *J. Comput. Sci. Coll. 34*, 81–82.

CAMPOS, R., DOS SANTOS, R. P., AND OLIVEIRA, J. 2020. A recommendation system based on knowledge gap identification in moocs ecosystems. In *XVI Brazilian Symposium on Infor-*

mation Systems, F. E. Horita, C. A. Kamienski, S. D. Ávila E Silva, A. M. Magdaleno, and D. Viana, Eds. SBSI'20. Association for Computing Machinery, New York, NY, USA, 1–8.

CAO, B., CHEN, J., LIU, J., AND WEN, Y. 2020. A topic attention mechanism and factorization machines based mobile application recommendation method. *Mobile Networks and Applications 25,* 4 (Aug), 1208–1219.

DO, P., NGUYEN, K., VU, T. N., DUNG, T. N., AND LE, T. D. 2017. Integrating knowledge-based reasoning algorithms and collaborative filtering into e-learning material recommendation system. In *Future Data and Security Engineering*, T. K. Dang, R. Wagner, J. Küng, N. Thoai, M. Takizawa, and E. J. Neuhold, Eds. FDSE '17, vol. 10646. Springer International Publishing, Cham, 419–432.

GAUR, L., SINGH, G., JEYTA, AND KUMAR, S. 2016. Google analytics: A tool to make websites more robust. In *Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies*. ICTCS '16. Association for Computing Machinery, New York, NY, USA, 1–7.

GHEBGHOUB, O., ABEL, M.-H., AND MOULIN, C. 2008. Learning object indexing tool based on a lom ontology. In *2008 Eighth IEEE International Conference on Advanced Learning Technologies*, P. Díaz, Kinshuk, I. Aedo, and E. Mora, Eds. IEEE, 576–578.

GRÉVISSE, C., MANRIQUE, R., MARIÑO, O., AND ROTHKUGEL, S. 2018. Knowledge graph-based teacher support for learning material authoring. In *Advances in Computing*, J. E. Serrano C. and J. C. Martínez-Santos, Eds. Springer International Publishing, Cham, 177–191.

HASSAN, J., LEONG, J., AND SCHNEIDER, B. 2021. Multimodal data collection made easy: The ez-mmla toolkit: A data collection website that provides educators and researchers with easy access to multimodal data streams. In *LAK21: 11th International Learning Analytics and Knowledge Conference*. LAK21. Association for Computing Machinery, New York, NY, USA, 579–585.

ILYAS, I. F. AND CHU, X. 2019. *Data Cleaning*. Association for Computing Machinery, New York, NY, USA.

KANG, B. 2021. How the covid-19 pandemic is reshaping the education service. In *The Future of Service Post-COVID-19 Pandemic*, J. Lee and S. H. Han, Eds. The ICT and Evolution of Work, vol. 1. Springer, Singapore, 15–36.

KEW, S. N. AND TASIR, Z. 2022. Learning analytics in online learning environment: A systematic review on the focuses and the types of student-related analytics data. *Technology, Knowledge and Learning 27,* 2 (Jun), 405–427.

LIAO, T., FENG, X., SUN, Y., WANG, H., LIAO, C., AND LI, Y. 2020. Online teaching platform based on big data recommendation system. In *Proceedings of the 5th International Conference on Information and Education Innovations*. ICIEI '20. Association for Computing Machinery, New York, NY, USA, 35–39.

LIU, Q., KUSNER, M. J., AND BLUNSOM, P. 2020. A survey on contextual embeddings. arXiv preprint arXiv:2003.07278.

LIU, Y., OTT, M., GOYAL, N., DU, J., JOSHI, M., CHEN, D., LEVY, O., LEWIS, M., ZETTLEMOYER, L., AND STOYANOV, V. 2019. Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692.

MA, B., LU, M., TANIGUCHI, Y., AND KONOMI, S. 2021. Courseq: the impact of visual and interactive course recommendation in university environments. *Research and Practice in Technology Enhanced Learning 16,* 1 (Jun), 18.

MADANI, Y., EZZIKOURI, H., ERRITALI, M., AND HSSINA, B. 2020. Finding optimal pedagogical content in an adaptive e-learning platform using a new recommendation approach and reinforcement learning. *Journal of Ambient Intelligence and Humanized Computing 11,* 10 (Oct), 3921–3936.

MANRIQUE, R., CUETO-RAMIREZ, F., AND MARIÑO, O. 2018. Comparing graph similarity measures for semantic representations of documents. In *Advances in Computing*, J. E. Serrano C. and J. C. Martínez-Santos, Eds. Springer International Publishing, Cham, 162–176.

MANRIQUE, R., GRÉVISSE, C., MARIÑO, O., AND ROTHKUGEL, S. 2018. Knowledge graph-based core concept identification in learning resources. In *Semantic Technology*, R. Ichise, F. Lecue, T. Kawamura, D. Zhao, S. Muggleton, and K. Kozaki, Eds. Springer International Publishing, Cham, 36–51.

MANRIQUE, R. AND MARINO, O. 2018. Knowledge graph-based weighting strategies for a scholarly paper recommendation scenario. In *Workshop of Knowledge-aware and Conversational Recommender Systems RecSys*, V. W. Anelli, T. D. Noia, P. Lops, C. Musto, M. Zanker, P. Basile, D. Bridge, and F. Narducci, Eds. KaRS'18. RecSys, Vancouver, Canada.

MANRIQUE PIRAMANRIQUE, R. F. 2019. Towards automatic learning resources organization via knowledge graphs. Ph.D. thesis, Universidad de los Andes.

MARINO, O. AND PAQUETTE, G. 2010. A competency—driven advisor system for multi-actor learning environments. *Procedia Computer Science 1,* 2, 2871–2876. Proceedings of the 1st Workshop on Recommender Systems for Technology Enhanced Learning (RecSysTEL 2010).

MAWANE, J., NAJI, A., AND RAMDANI, M. 2020. Unsupervised deep collaborative filtering recommender system for e-learning platforms. In Smart Applications and Data Analysis, M. Hamlich, L. Bellatreche, A. Mondal, and C. Ordonez, Eds. *Communications in Computer and Information Science 1207*, 146–161.

PAN, Z., ZHAO, L., ZHONG, X., AND XIA, Z. 2021. Application of collaborative filtering recommendation algorithm in internet online courses. In *Proceedings of the 6th International Conference on Big Data and Computing*. ICBDC '21. Association for Computing Machinery, New York, NY, USA, 142–147.

PAQUETTE, G., MARINO, O., AND BEJAOUI, R. 2021. A new competency ontology for learning environments personalization. *Smart Learning Environments 8,* 1 (Aug), 16.

PAQUETTE, G., MARIÑO, O., ROGOZAN, D., AND LÉONARD, M. 2015. Competency-based personalization for massive online learning. *Smart Learning Environments 2,* 1 (Feb), 4.

ŘEHŮŘEK, R. AND SOJKA, P. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. ELRA, Valletta, Malta, 45–50.

REIMERS, N. 2022. Pretrained models — sentence-transformers documentation. https://www.sbert.net/docs/pretrained_models.html. Accessed: 2022-06-02.

REIMERS, N. AND GUREVYCH, I. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Hong Kong, China, 3982–3992.

REIMERS, N. AND GUREVYCH, I. 2020. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Online, 4512–4525.

SANGUINO, J., MANRIQUE, R., MARIÑO, O., LINARES, M., AND CARDOZO, N. 2022. Log mining for course recommendation in limited information scenarios. In *Proceedings of the International Conference on Educational Data Mining*, A. Mitrovic and N. Bosch, Eds. EDM'22. International Educational Data Mining Society, 430–437.

SARI, A. R., BONK, C. J., AND ZHU, M. 2020. Mooc instructor designs and challenges: what can be learned from existing moocs in indonesia and malaysia? *Asia Pacific Education Review 21,* 1 (Mar), 143–166.

SHANSHAN, S., MINGJIN, G., AND LIJUAN, L. 2021. An improved hybrid ontology-based approach for online learning resource recommendations. *Educational Technology Research and Development 69,* 5 (Oct), 2637–2661.

SHMUELI, E. 2017. Merlot — a reliable framework for oer. In *2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)*. Vol. 2. IEEE, Turin, Italy, 697–699.

SONG, K., TAN, X., QIN, T., LU, J., AND LIU, T.-Y. 2020. Mpnet: Masked and permuted pre-training for language understanding. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds. Vol. 33. Curran Associates, Inc., 16857–16867.

SOUSA, M. J. AND ROCHA, Á. 2020. Learning analytics measuring impacts on organisational performance. *Journal of Grid Computing 18,* 3 (Sep), 563–571.

TARUS, J. K., NIU, Z., AND MUSTAFA, G. 2018. Knowledge-based recommendation: a review of ontology-based recommender systems for e-learning. *Artificial Intelligence Review 50,* 1 (Jun), 21–48.

UDDIN, I., IMRAN, A. S., MUHAMMAD, K., FAYYAZ, N., AND SAJJAD, M. 2021. A systematic mapping review on mooc recommender systems. *IEEE Access 9*, 118379–118405.

WANG, C., ZHU, H., ZHU, C., ZHANG, X., CHEN, E., AND XIONG, H. 2020. Personalized employee training course recommendation with career development awareness. In *Proceedings of The Web Conference 2020*. WWW '20. Association for Computing Machinery, New York, NY, USA, 1648–1659.

WANG, W., WEI, F., DONG, L., BAO, H., YANG, N., AND ZHOU, M. 2020. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds. Vol. 33. Curran Associates, Inc., 5776–5788.

WU, L., HE, X., WANG, X., ZHANG, K., AND WANG, M. 2022. A survey on accuracy-oriented neural recommendation: From collaborative filtering to information-rich recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 1–1.

XIAO, J., WANG, M., JIANG, B., AND LI, J. 2018. A personalized recommendation system with combinational algorithm for online learning. *Journal of Ambient Intelligence and Humanized Computing 9,* 3 (Jun), 667–677.

YIN, S., YANG, K., AND WANG, H. 2020. A mooc courses recommendation system based on learning behaviours. In *Proceedings of the ACM Turing Celebration Conference - China*. ACM TURC'20. Association for Computing Machinery, New York, NY, USA, 133–137.

ZAKARIA, I., JAMALUDIN, M., ISMAIL, W. S. A. W., AND ARIFIN, N. 2016. Measuring user's usage intentions on e-learning portal. In *Envisioning the Future of Online Learning*, J. E. Luaran, J. Sardi, A. Aziz, and N. A. Alias, Eds. Springer Singapore, Singapore, 347–357.