Content-Based Recommender System using Word Embeddings for Pedagogical Resources

Mediani Chahrazed, Saad Harous, Mahieddine Djoudi

Content-Based Recommender System using Word Embeddings for Pedagogical Resources

Chahrazed Mediani
LRSD Laboratory, Computer Science department
Ferhat Abbas Sétif -1- University
Sétif 19000, Algeria
chahrazed.mediani@univ-setif.dz

Saad Harous
Computer Science Department
College of Computing and Informatics, University of Sharjah
Sharjah 27272, UAE
harous@sharjah.ac.ae

Mahieddine Djoudi
TECHNE, Université de Poitiers
1 rue Raymond Cantel, Bâtiment A3 TSA 11102
86073 Poitiers, France
mahieddine.djoudi@univ-poitiers.fr

***Abstract*—Recommender Systems are important systems operating within a system to ensure how certain types of data are managed on the internet. These systems help users with overwhelming data and provide a better user navigation experience. This paper presents a content-based recommender system for online resources using deep learning. We have included some deep learning techniques to allow a good semantic understanding of educational resources. However, we have used a pre-trained word2vec model owned by Google for the following three reasons: (1) Google is reliable; (2) the content of the Google news dataset is close to the content of the shared articles dataset; and (3) training a word2vec model is time-consuming and a domain-independent itself. We have also used techniques for the dimensionality reduction like t-Distributed Stochastic Neighbor Embedding and Principal Component Analysis to reduce the dimensions of users and items vectors. Our approach aims to ameliorate the recommendations accuracy and better satisfy the requirements of users. The results obtained when we tested our system are encouraging.**

***Index Terms*—Content-based, deep learning, pedagogical resource, recommender system, word embeddings, word2vec**

I. Introduction

Recommender systems (RS) are a recent research topic compared to the classical ones, such as information filtering and search engines [1]. They are considered an extension of user-customized systems and an intelligent class of information filtering that can propose a relevant items list for users [2]. The offered list is based mainly on user-collected preferences. For example, the user's preferred items might represent the articles he/she wants to learn [2]. Recommender systems aim to filter the overwhelming data and provide users with relevant information. Recommendation techniques are differentiated according to the various source of knowledge used as inputs to the recommender system, e.g., the data is coming from the items' content, the users' profile content, or the collaboration of other users' profiles in terms of items' interactions. Therefore based on those differences, recommender systems are classified into four main categories: content-based approaches, collaborative filtering, popularity-based, and hybrid-based, which represents the hybridization of the previous three categories.

Popularity-based approaches are a basic type of collaborative filtering, where items with the most common users' collaborations are considered popular [3]. popularity-based recommender systems prefer the most popular items due to the level of interactions reported by many users about these items. However, those systems provide non-personal recommendations. They recommend the majority of users' preferences [4].

Content-based filtering suggests items to users regarding their past behaviors and an ensemble of analyzed features about the item, such as the plain text content, an image, or tags [5]. Content-based approaches calculate the similarity between not experienced items and those that got some behavioral actions by the user in the past. They generally use items extracted information (e.g., items' likes and dislikes). Thus, unlike other techniques, content-based are domain-independent [2]. i.e., they do not need to match the users' interests. Every user is treated independently from other users. Moreover, items are recommended regardless of other users' tastes [6]. They focus on the content of the item itself.

Collaborative filtering techniques are a form of automated matchmaking of users' tastes. Unlike content-based approaches, a collaborative approach focuses more on the users than the item's content [1]. Each user is represented by a multidimensional vector of items, and each item has a state (e.g., a product rating score). Once the system gets the needed information about particular user preferences, it compares it to other users with common tastes and recommends similar items [7]. It uses heuristic calculation methods such as the cosine method to calculate user similarity or generate a user-based model [8]. Collaborative filtering RS are classified into two principle categories: memory-based and model-based [8].

The preceding recommender system approaches have

proven to be very strong and reliable. However, they face some vulnerabilities and weaknesses. Therefore, hybrid systems are introduced to improve recommendation systems. A hybrid system incorporates at least two principal techniques [9], in which the main idea of a hybrid system is to improve the weakness of one approach with the strengths of the others [2], [10]. The implementation of a hybrid recommender system demands precising a hybridization strategy. [11] proposed a taxonomy classification for hybrid recommender systems in which, they are devided to seven classes.

A. Problem statement

Recommender systems have become integral to many industries, especially e-commerce, entertainment, and social media [7]. The principle role of a recommender system is to give personalized recommendations to users based on their past behavior, preferences, and interests. For example, in e-commerce, recommender systems can suggest products a customer is more likely to buy based on their past purchase history, search history, and product ratings. This can increase customer satisfaction, drive sales, and increase customer loyalty [7]. In the entertainment industry, recommender systems can be used to suggest movies, TV shows, and music to users based on their viewing and listening history, ratings, and social media activity. This can help users discover new content they might enjoy and increase engagement with the platform. In social media, recommender systems can suggest friends, groups, and pages to users based on their interests and social network. This can help users connect with like-minded people and increase engagement with the platform. However, Recommender systems can also be used in other industries like healthcare, education, and finance. They can significantly impact e-learning by providing personalized recommendations to learners. They can help learners personalize their learning by recommending learning resources, courses, and activities tailored to their interests, learning styles, and knowledge levels [12]. Additionally to the impact of recommender systems concerning user navigation experience and data overwhelming, they still face some challenges, such as the increased volume, heterogeneity, and the fact that they are not sufficiently adapted to the users' needs. Therefore, the advancement of machine learning leads to the introduction of deep learning (DL) technologies. In our research, we compared different content-based recommender systems to find the deep learning model that performed the best overall. Finding a deep learning model that could be used in a recommender system was the research's main goal.

B. Approach and results

In recent years, recommender systems have started including the efficiency of deep learning techniques to understand users' behaviour and increase recommendations accuracy [13]. Recently, E-learning platforms try to benefit from recommender systems to facilitate and improve learning. They furnish users (teachers and students) with the appropriate pedagogical tools to help them build a better educational environment that includes collaboration and resource sharing. Our work proposes a deep learning-based recommendations system of pedagogical resources. This deep learning system implements a word embedding technique in a content-based approach to tackle pedagogical resources' semantics.
Our contributions are summarized as follows:

- We proposed a content-based recommender system for educational resources (WordToVec_CB Model). The WordToVec_CB converts the higher-order feature interactions from the original data. This architecture seeks to decrease user-item interactions and increase content-based performance.
- With the help of WordToVec technology, WordToVec_CB enabled the content-based method to better comprehend the semantic importance of words in the instructional resources. It generates word embeddings that catch the semantic significance of words based on their context in a text. This enables content-based recommender systems to capture the semantic meaning of items like articles, movies, or music and recommend similar preferred items to users.
- Word2Vec_CB generates low-dimensional word embeddings that can represent the content of items compactly and efficiently. This can reduce the feature space dimensionality and make the recommender system more efficient and scalable.
- The WordToVec_CB benefit from embedding representations' advantages for content modeling. This recommender extracts useful attributes from the items, which produce more precise predictions.
- To evaluate WordToVec_CB, two other instances are used to compare with their results: The popularity and CB-TFIDF developed in [14].
- The WordToVec_CB framework can be used to provide an efficient and effective way to represent the semantic meaning of items and make personalized recommendations.
- The WordToVec_CB model surpasses state-of-the-art content-based algorithms on a variety of real-world datasets.

In the following, we discuss related work to content-based recommender systems. Section 3 presents a set of concepts related to this work. Section 4 presentes in detail our WordToVec_CB framework. Section 5 provides the dataset and discusses the results of our experimentations on WordToVec_CB. Finally, we summarize the conclusions.

II. Related work

There have been many studies on content-based recommender systems that use word embeddings in recommendation. [15] proposed a novel approach to contextual recommender systems. They addressed the challenge of providing relevant recommendations to users in the context of a document, such as a news article or blog post. The new Convolutional Matrix Factorization (CMF) approach combined matrix factorization with convolutional neural networks (CNNs) to integrate the document context in the recommendation. The CMF model considers the user's historical preferences, the document's content, and the document's context, such as the category or topic of the article. The model learns to encode the document context and user preferences as latent vectors, which

are combined to generate recommendations. The authors have used two datasets to assess the efficiency of the CMF model and compare it to several baseline methods. As a result, the CMF model surpasses the baseline methods in terms of accuracy and coverage.

In [16], Cataldo Musto et al. proposed a preliminary examination into using Word Embedding techniques in the content-based recommendation. They compared the efficacy of three widely used methods in developing a vector space representation of recommended items and user profiles: Latent Semantic Indexing, Word2Vec, and Random Indexing. They used two datasets (Movielens and DBbook), and the results revealed valuable insights that paved the way for future initiatives. The result shows that only F1@5 is the best-performing configuration in DBbookW2V. On the other hand, on F1@10 and F1@15, the Latent Semantic Indexing did the lowest on MovieLens but it surpasses WordtoVec and Random Indexing.

[17] suggested a content-based recommendation algorithm built on CNN. It predicts the latent factors from the text information of the multimedia resources to solve the training input. The CNN used the language model, and for the output, it proposed the latent factor model. The Bregman iteration method is used to solve the model. They used a Book-Crossing dataset. The results showed that this recommendation algorithm could be used to recommend new learning resources. Furthermore, the Bregman iteration method has ameliorated the training efficiency.

[18] proposed a recommender system for user sessions that used Long Short-Term Memory (LSTM) networks. They have used the MovieLens dataset. The movie LSTM-based RS has been evaluated in different ways. The research compared the LSTM networks to Recurrent Neural Networks (RNN), a similar deep learning method, and collaborative filtering that used item-based nearest neighbors (item-KNN). They obtained that when the hyperparameters of the LSTM are optimized, the LSTM-based movie RS can obtain higher recommendation performance.

[19] proposed a collaborative filtering model that incorporated word embeddings to catch items content and make personalized recommendations. They proposed a new approach based on neural networks, which they argue can handle the challenges of collaborative filtering more effectively. The proposed model, NeuMF, combines two neural networks: a matrix factorization network and a multi-layer perceptron network (MLP). The matrix factorization network captures the latent correlation between users and items, while the MLP network learns the non-linear interactions between them. The authors combine these two networks to make more accurate user preference predictions. The authors assess NeuMF efficiency on two large-scale datasets and find that it surpasses existing state-of-the-art methods in terms of prediction accuracy and ranking quality.

[20] presented a movie recommendation system that uses the Word2vec algorithm to generate movie embeddings, which are used to recommend movies to users. The aim of ExMrec2vec is to produce an explainable recommendation system, where the recommendations are based on the semantic meaning of the movies, as captured by the Word2vec embeddings. The authors evaluated the ExMrec2vec model using two metrics: precision and recall. The authors found that ExMrec2vec outperformed several baseline models in terms of precision and recall, demonstrating the effectiveness of the Word2vec embeddings in capturing the semantic meaning of the movies.

These works demonstrate the effectiveness of using word embeddings in content-based recommender systems across different domains, such as movies, music, and articles. They highlight the potential of word embeddings in capturing the semantic meaning of text data and making personalized recommendations.

III. Background

We attempt to define and explain some concepts relevant to our proposed model, such as:

A. User-Item Utility Matrix

In recommender systems, the user-item utility matrix represents the user behavior. It contains the interactions recorded between users and items. The matrix's rows and columns each stand for a user and an object, respectively. The matrix cells represent a user's level of interaction or preference for a particular item. Different manners describe user-item interactions in the matrix: binary representation, explicit, and implicit feedback.

B. User Profile

The user profile is significant in recommendation systems. Its model represent the user's information. Most personalization systems require the creation of a user profile or model to determine the needs of individual users. It has different representations. In our content-based approach, every user and item is represented by their embedding because the item, which in our case is an article, is a sequence of semantically interconnected words. A user profile is also defined as a sequence of words from the texts he reads.

C. Word Embedding

The word representation based on the principle: *"words in similar contexts have similar meanings"* is called Word embedding [21]. It means: words in the same context are semantically connected and share similar clusters [22]. Word embedding is used in language modeling like natural language processing (NLP), text classification, etc. The word is represented by a vector space or a dense vector that contains real values of the likelihood estimation (embedding) of the word and other contextual words in a sentence [23].

Word2vec, or word-to-vector was introduced by [24]. It is an unsupervised neural network architecture for learning word embedding, where it captures the related semantics and nearby words in a text [23]. Therefore, each word vector is trained to capture the co-occurrence (i.e., word meaning) by maximizing the log-conditional probability associated to the word given the

context word appearing in a window of fixed size. (e.g., in a five-word sentence, a window of three takes two context words next to the target word in the sentence) [23].

For example, given two sentences: "The **kid** said he would grow up to be spiderman" and "The **child** said he would grow up to be spiderman" Those two sentences contain two different words (kid and child), but they are in the same context. Thus, based on word2vec, *kid and child* have similar word embeddings.

Word2vec came with two main models: the first model is Continues Bag of Words (CBOW) and the second is skip-gram (SKG). In CBOW, the context is used to predict the targeted word, whereas skip-gram does the inverse because it predicts the context based on the word. (Figure 1) [25].

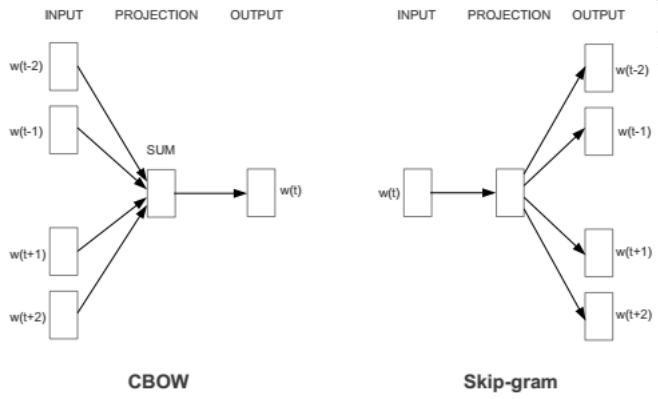

Fig. 1. The CBOW and Skip-gram Neural Architectures [25]

Word2vec architecture takes, as input, a text corpus, and as output, it generates word vectors. It starts building a vocabulary from the training of the text data. Then, It learns the words's vector representation. Many machine-learning and natural language processing (NLP) applications can use the resulting word vector file. However, since we are building a recommender system, not an NLP application, building a deep neural network of word2vec takes serious time and effort. Therefore, due to machine learning and deep learning technologies that allow transfer learning, we decide to use a pre-trained word2vec architecture [26].

We adapted the word embedding technique in our approach because it is a very efficient deep learning technique of computing words semantics through low-dimensional matrix operations, unlike others, such as the high dimensional one-hot encoding. Or TF-IDF that is limited only to the term frequency, not the semantic of words. A simple example that shows the efficiency of word embeddings, is that given three words (king, man and woman) that are represented as embedding vectors, let's say: $\vec{king} = \{0.36, 0.41, ... - 1.21\}, \vec{man} = \{0.34, -1.5, ... - 0.09\}$, $\vec{woman} = \{0.98, 1.1, ... - 0.12\}$. So, we can do some mathematical calculations like addition or multiplication to get a resulted vector that might predict the word *queen*.

IV. Proposed Approach

In this section, we developed a model that recommend resources to learners according to the their interests. We have used a content-based method. The proposed method uses a deep learning model to overcome the weaknesses of recommendation approaches. Our approach explores user profiles and generates recommendations according to their past behavior. Generally, plain texts (unstructured textual data) are the essential content in the educational resources context. Thus, we aim to use deep learning techniques to allow the system to better comprehend the semantic of words in plain text. So, we adopt the deep neural network architectures of word2vec to improve content-based recommendation accuracy. For this module, we used the following algorithm:

Algorithm : WordToVec_CR

Input: I: Items vector of size m
U: Users interactions vector of size n
Output: L: Recommendation list of size k

1) Convert each text to a sequence of tokens: unify the text's case (lower or upper case), remove special characters, punctuation, accent marks, and other diacritics.

$$tokens(i) = \{t1, t2, ..., tk\} \tag{1}$$

Where: t is the token, and k is the number of tokens in item i.

2) Construct the user profile $\vec{p}(u)$ as a sequence of tokens of his interacted items, which are his read articles in our case:

$$\vec{p}(u) = \{i_1, i_2, ..., i_k\} = \{tokens(i_1), tokens(i_2), ..., tokens(i_k)\} \tag{2}$$

With: $tokens(k)$ is the sequence of tokens of the k'th item interacted with the user u.

3) Construct word embedding by applying a pre-trained word2vec architecture. Where each word is converted to a real-value dense vector.

4) Calculate a centroid of vectors of words for both items and users vectors:

$$\vec{i_j} = \frac{1}{N} \sum_{j \in K=1}^{N} word\_vector_i \tag{3}$$

$$\vec{u} = \frac{1}{N} \sum_{j \in P=1}^{N} \vec{i}_j \tag{4}$$

5) Calculate the distance between each pair of user-items embedding vectors:

$$\vec{dist}(u, i) = |\vec{u} - \vec{i}| \tag{5}$$

6) Produce a candidate list L of items that are close to a user profile (i.e., the preference $pr(u, i)$ is high)

$$L = min(\vec{p}(u), \vec{dist}(u, i)) = max(pr(u, i)) \tag{6}$$

with $i \notin \vec{p}(u)$, (the user has not already interacted with the item).

7) For each user, sort the provided list in descending order, with the closest items at the top of the list.

Figure 2 covers all the explained steps of our content-based approach presented in WordToVec_CR algorithm.

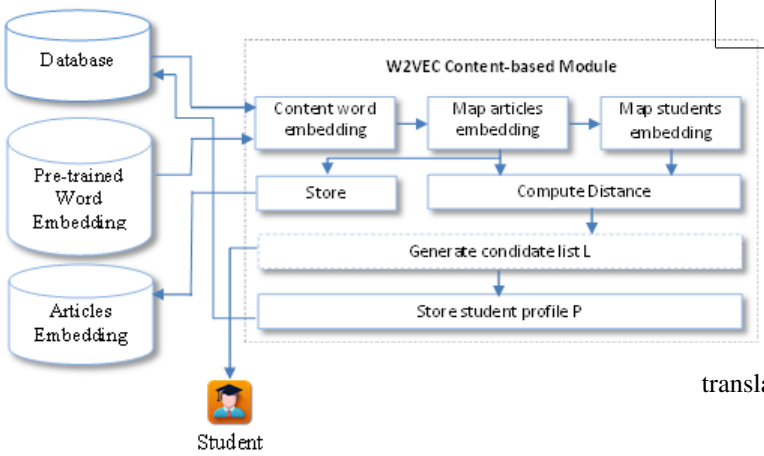

Fig. 2. Our system detailed architecture

V. Experimental Results

We discuss the results of the experiments that assess the proposed approach in this section. The used dataset and the evaluation metrics will also be explained.

A. Dataset Description and preprocessing

Our content-based recommender system is used to recommend educational resources to users. We have chosen articles to represent these resources. Therefore, to train and test our recommender system, we conducted experiments on the articles dataset. The dataset description and preparation process are explained as follows:

1) Definition: We used a real dataset, namely the Articles dataset, to validate the proposed approach. It consists of two files : Articles sharing and users interactions. The dataset contains more 3k shared papers, and more than 72k logged users interactions. It is composed of the features presented in (table I).

2) Dataset pre-processing :

The users interactions are implicitly represented in the dataset like views and comments. Associated weights are given to each type of interaction based on interaction level (Table: II) [27]. The second step is the data preprocessing. In this stage, we have removed duplication and dropped unwanted features. A pre-trained English-based word embedding is used. We have unified the articles' languages using a translation API instead of dropping non-English papers to obtain more data and better train models. As shown in (Figure 3), most of the texts are translated from Portuguese (pt) to English (en-pt).

TABLE I
Data Description

Dateset features	Description
Article attributes	Title, Article URL, content's plain text presented in Portuguese and English.
Logged users	The users needed to log to the platform and provided long-term tracking of their preferences.
Contextual information	Users' visits context like the client (mobile native app or browser), date/time, and geolocation.
Rich implicit feedback	Variant interaction types were logged allowing the inference of the user's interest level in the articles (e.g., likes, comments, views).

TABLE II
Associated Interaction Weights

Interaction	Weights
view	1
like	2
comment	2.5
follow	3
bookmark	4

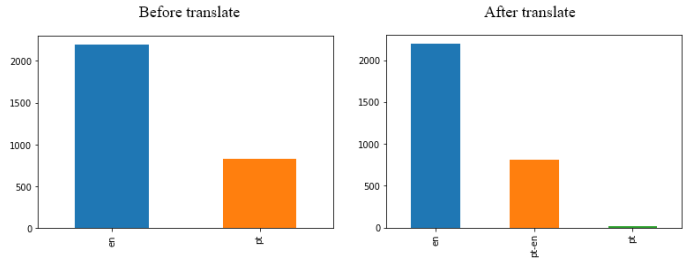

Fig. 3. Unify The Articles Languages

B. Evaluation Metrics

To test the used models performance, the research done in the field gave a collection of various evaluation metrics. Recall@k and Precision@k are among the most used measures of recommendation. In RS, the most significant user recommendations are top-N items. So, it becomes clear to compute recall and precision metrics in the first N items instead of whole items because each user generally rated a few number of items regarding the number of all the items in the dataset. Also, in the test set, relevant items may be lower than all items in the dataset. Therefore, in recommender systems, recall and precision rely on the number of rated items per user [28]. Recommended and relevant are two significant terms used to mesure RS.

- True positive (TP): It denotes a relevant item recommended.
- True negative (TN): It is a non-relevant item not recommended.
- False negative (FN): It is a relevant item recommended.

- False-positive (FP): is a non-relevant item recommended.

where, in the dataset, relevant items are already known, whereas, recommended items are produced by the models [28].

- **Precision@k:** The Precision measures the rate of the user-relevant and recommended items (TP) to the entire recommended items (TP + FP) [28]:

$$Precision@k = \frac{recommended \cap relevenat}{recommended} = \frac{TP}{TP + FP} \quad (7)$$

- **Recall@k:** The Recall measures the rate of the user-relevant and recommended items (TP) to the entire number of user-relevant items (TP and FN) [28]:

$$Recall@k = \frac{recommended \cap relevenat}{relevenat} = \frac{TP}{TP + FN} \quad (8)$$

- **F1 measure:** The F1 measure is the harmonic mean of the Recall and the Precision [28]:

$$F1 - measure = \frac{2 * precision * recall}{precision + recall} \quad (9)$$

- **Accuracy:** The Accuracy is the part of predictions the model got right:

$$Accuracy = \frac{Numberofcorrectpredictioins}{Totalnumberofpredictions} = \frac{TP + TN}{TP + TN + FP + FN} \quad (10)$$

The performances of our approaches are evaluted using the holdout strategy, which is a variant of the cross-validation approach. A random sample from the dataset is held out as training data, whereas the remaining data is the testing data. We have different propositions to split data to train and test (80 % for training, 20% for testing in our case).

To rank all items by the user in the dataset is very time-consuming. So, we used a popular strategy that randomly samples the topK items not interacted by the user (in our case: 100 items). Our RS ranks the leave-out items among these topK items list and calculates the accuracy metrics for this user and interacted item from the recommendation ranked list. Our RS system used Recall@10 and Precision@10 to measure recommendations' accuracy. Also, we used cosine similarity to calculate the distance between the word embeddings of items and users.

C. System Configuration

Our experiments are performed on DELL Intel(CORE i5). Processor: Intel(R) Core(TM) i5-4310U, CPU: @2.00 GHz 2.60 GHz, and RAM: 4.00 GB.

D. Parameter Settings

For our approach, a pre-trained word2vec model is used. This model is owned by Google. It incorporate word vectors to a vocabulary having 3 million phrases and words. This last is trained on Google News dataset having around 100 billion words. The length of the vector is 300 features. We decided to use Google news Word2vec for those three reasons: Google is reliable, the content of the google news dataset is close to the content of the shared articles dataset, and training a word2vec model is time-consuming and domain-independent itself. We also used two dimensionality reduction techniques: t-Distributed Stochastic Neighbor Embedding (t-SNE) and Principal Component Analysis to reduce the dimensions of users and items vectors.

E. Competing Approaches

We have performed two other approaches: a popularity-based and a content-based approach using the TF-IDF technique to validate our approach. We have implemented each one separately and made a comparison between all of them. Our achieved performance of our proposed approach is compared to these two approaches. Precision, recall, F-score, and accuracy metrics are used to better evaluate the recommendations.

F. Discussion

After building users and items word embedding, cosine similarity is used to calculate distances between items and users vectors. According to the sampled user in (Table: III), the system determined that the user likes to read about the computer science domain, and the top three items are proposed with an average similarity degree of (0.80285), which indicates that (80%) of similarity between the item and user embedding vectors (i.e., a high level of user preference): pr(u, i) = {u, i, 0.80}. On the other hand, after reducing the dimensionality of users and items embedding using (t-SNE). The plot in (Figure: 4) illustrates the distribution of articles and users where articles about blockchain are pointed in green.

TABLE III
SIMILAR ARTICLES TO A SAMPLED USER

personId	articles titles	similarity degree
6999578934585823267	Hello, TensorFlow!	0.81351
	SyntaxNet in context: Understanding Google's new TensorFlow NLP model	0.80368
	Machine Learning is Fun! Part 2	0.79136

Last, using the word2vec technique in our content-based approach outperforms the second approach. This latter used TF-IDF which is only limited to the term frequency. However, a global recall of (0.43) is scored in our system. In the test set, this model ranks a placeholder about (43%) of interacted articles among the top ten items. And a precision of (0.42) showing that 42% of recommended items are relevant to the users. (Table: IV) shows the precision@10 and recall@10 details of some users with a global precision of (0,20).

According to Figure 5 representing the evaluation metrics of the implemented models, we conclude that our CB_W2VEC surpasses the Popularity and the CB_TFIDF approaches. It is the best for the recall, precision, and accuracy metrics. Our content based CB_W2VEC model was better in predicting the articles recommended. It reached 92% for the accuracy@5, 88% for the accuracy@10, and 80% for the accuracy@20.

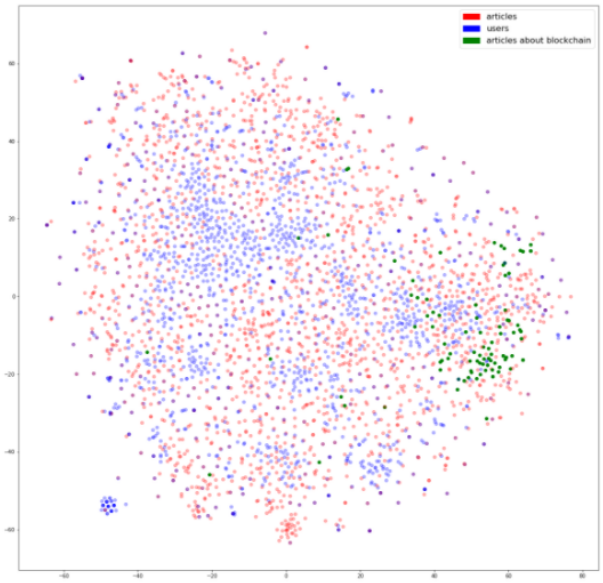

Fig. 4. Users and Items vectors plot

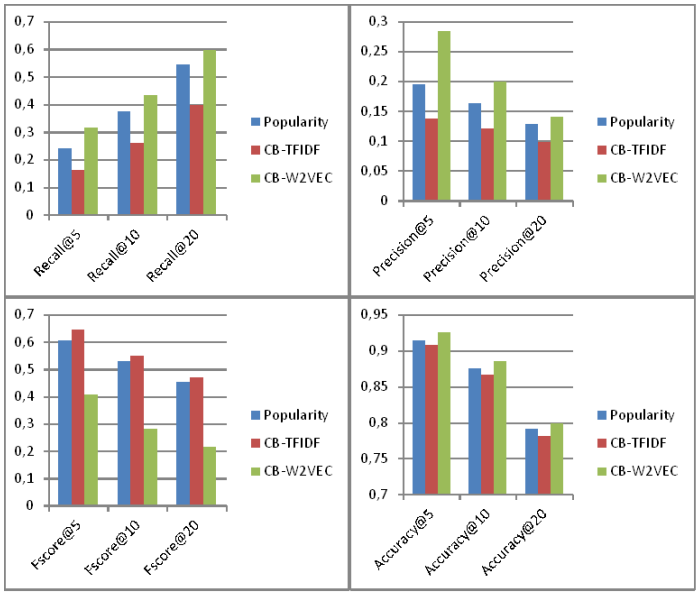

Fig. 5. Recommendations measures for Popularity, TF-IDF content-based and Word2vec content-based

TABLE IV
W2VEC_CB MODEL RECALL AND PRECISION SCORES

personId	hits@10	interacted	recommended	R@10	P@10
3636910968448833585	10	57	20	0.17	0.50
2416280733544962613	17	60	26	0.28	0.65
3609194402293569455	30	138	40	0.21	0.75

VI. Conclusion

Recommender systems have become essential for any website and online platform, like a search engine. In this research, we have developped a content-based recommender system using deep learning techniques and compared multiple models to identify the best-performing one. We have used word2vec technology to comprehend the semantics of words of online resources. We have evaluated the models using four important metrics: recall, precision, F1-score, and accuracy. The precision helped assess the system's accuracy in recommending relevant items, while recall measured the number of successfully recommended relevant items. The F1 score provided a balanced assessment by considering both precision and recall. Our models exhibited good performance on real-world datasets and can be applied to various online resources. As future work, we plan to suggest more advanced deep learning models and expand our findings to encompass additional datasets, aiming for greater efficacy and broader applicability.

References

[1] F.O. Isinkaye, Y.O. Folajimi, and B.A. Ojokoh, "Recommendation systems: Principles, methods and evaluation", Egyptian Informatics Journal, Vol. 16, no.3, pp. 261-273, 2015. https://doi.org/10.1016/j.eij.2015.06.005.

[2] E. Çano and M. Morisio, "Hybrid Recommender Systems: A Systematic Literature Review", Intelligent Data Analysis, Vol. 21, no. 6, pp. 1487-1524, 2017. DOI: 10.3233/IDA-163209

[3] C. Kwan, M.Q. Koh, and M.B. Jasser, "A Comparison Study Between Content-Based and Popularity-Based Filtering via Implementing a Book Recommendation System", International Journal of Advanced Research in Engineering & Technology, Vol. 11, no. 12, pp. 1121-1135, 2020. DOI: 10.34218/IJARET.11.12.2020.109

[4] N. Jonnalagedda, S. Gauch, K. Labille, and S. Alfarhood, "Incorporating popularity in apersonalized news recommender system", PeerJ Computer Science, Vol. 2:e63, 2016. https://doi.org/10.7717/peerj-cs.63

[5] A. Aziz and M. Fayyaz, "Comparison of Content Based and Collaborative Filtering in Recommendation Systems", International Conference on Multimedia Information Technology and Applications, January 2021, Vietnam, 2021.

[6] A. Nurcahya and Supriyanto, "Content-based recommender system architecture for similar ecommerce products", Jurnal Informatika, Vol. 14, no. 3, pp. 90-101, 2020.

[7] J.B. Schafer, D. Frankowski, J. Herlocker, and S. Sen, "Collaborative Filtering Recommender Systems", In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds) The Adaptive Web, Lecture Notes in Computer Science, Vol. 4321, Springer, Berlin, Heidelberg. 2007. https://doi.org/10.1007/978-3-540-72079-9_9

[8] K.D. Gupta, "A Survey on Recommender System", International Journal of Applied Engineering Research ISSN 0973-4562, Vol. 14, no. 14, pp. 3274-3277, 2019.

[9] Y. Mediani, M. Gharzouli, and C. Mediani, "A Hybrid Recommender System for Pedagogical Resources". The International Conference on Digital Technologies and Applications, Morocco, 29-30 January. 2022.

[10] R. Ramesh and S. Vijayalakshmi, "Improvement to Recommendation system using Hybrid techniques," 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE), Greater Noida, India, pp. 778-782, 2022. doi: 10.1109/ICACITE53722.2022.9823879.

[11] R. Burke, "Hybrid recommender systems: Survey and experiments", User Model. User-Adapted Interact, Vol. 12, no. 4, pp. 331–370, 2002. doi: 10.1023/A:1021240730564.

[12] Q. Zhang, J. Lu, G. Zhang, "Recommender Systems in E-learning", Journal of Smart Environments and Green Computing, Vol. 1, no. 2, pp. 76-89. 2021. http://dx.doi.org/10.20517/jsegc.2020.06

[13] R. Mu, "A survey of recommender systems based on deep learning", IEEE Access, vol. 6, pp. 69009–69022, 2018. doi: 10.1109/ACCESS.2018.2880197.

[14] C. Mediani, "Interactive Hybrid Recommendation of Pedagogical Resources", Ingénierie des Systèmes d'Information, Vol. 27, no. 05, pp. 695-704, 2022.

[15] D. Kim, C. Park, J. Oh, and S. Lee, "Convolutional Matrix Factorization for Document Context-Aware Recommendation", Proceedings of the 2018 ACM International Conference on Information and Knowledge Management, 2018.
[16] C. Musto, G. Semeraro, M. Degemmis, and P. Lops, "Word Embedding techniques for Content-based Recommender Systems: an empirical evaluation", RecSys 2015 Proceedings, September 16-20, Vienna, Austria, 2015. https://ceur-ws.org/Vol-1441/recsys2015_poster23.pdf
[17] J. Shu, X. Shen, H. Liu, B. Yi, and Z. Zhang, "A content-based recommendation algorithm for learning resources", Multimedia Systems, Vol. 24, no. 2, pp. 163-173, 2017.
[18] S. Nguyen and B.D. Tran, "Long Short-Term Memory Based Movie Recommendation", Science & Technology Development Journal Engineering and Technology, Vol. 3(SI1), pp. SI1-SI9, 2020.
[19] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T. Chua, "Neural Collaborative Filtering", Proceedings of the 26th International Conference on World Wide Web, Perth, Australia, Vol. 10, pp. 173–182, 2017. https://doi.org/10.1145/3038912.3052569
[20] A. Samih, A. Ghadi, and A. Fennan, "ExMrec2vec: Explainable Movie Recommender System based on Word2vec", International Journal of Advanced Computer Science and Applications (IJACSA), Vol. 12, no. 8, pp. 653–660, 2021.
[21] Z.S. Harris, Distributional structure, Word, Vol.10, no. 2-3, pp. 146–162, 1954.
[22] O. Levy and Y. Goldberg, "Dependency-based word embeddings', Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, Vol. 2, pp. 302-308, 2014.
[23] M. Kusner, Y. Sun, N. Kolkin, and K. Weinberger, "From word embeddings to document distances", Proceedings of the 32 nd International conference on machine learning, Lille, France, In. JMLR: W&CP, Vol. 37, pp. 957-966, 2015.
[24] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representations in Vector Space", J. Computer Science, 2013.
[25] P. Liu, X. Qiu, and X. Huang, "Learning Context-Sensitive Word Embeddings with Neural Tensor Skip-Gram Model", Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI), pp. 1284-1290, 2015.
[26] A. Tixier, M. Vazirgiannis, and M.R. Hallowell, "Word Embeddings for the Construction Domain", ArXiv, 2016.
[27] C. Mediani and M.H. Abel, "Semantic recommendation of pedagogical resources within learning ecosystems", The IEEE International Conference on Industrial Informatics and Computer Systems (CIICS), 13-15 March, Sharjah, United Arab Emirates, 2016.
[28] M. Kuanr and P. Mohapatra, "Assessment Methods for Evaluation of Recommender Systems: A Survey", Foundations of Computing and Decision Sciences, Vol. 46, no. 4, 2021. DOI: 10.2478/fcds-2021-0023.