Computing, Telecommunication and Control, 2025, Vol. 18, No. 1, Pp. 48–59.
Информатика, телекоммуникации и управление. 2025. Том 18, № 1. С. 48–59.

Research article
DOI: https://doi.org/10.18721/JCSTCS.18104
UDC 004.85

LEVERAGING NATURAL LANGUAGE PROCESSING TECHNIQUES FOR ENHANCED RECOMMENDER SYSTEMS

S.A. Shulgin ✉ , *E.N. Benderskaya*

Peter the Great St. Petersburg Polytechnic University,
St. Petersburg, Russian Federation

✉ shulginsergey0@gmail.com

Abstract. Recommender systems often use NLP methods primarily for content processing. In this study, we propose a new approach to building recommender systems, in which user interaction data with content is considered within the framework of a natural language model. Thus, the user preference vectorization model Pref2Vec is proposed as the basis for a hybrid recommender system. Moreover, a concept of a user embedding space (UES) is introduced, which represents a set of extended embeddings that capture end-user preferences. A new method of applying clustering analysis to the recommendation process is also proposed. The Pref2Vec model and the UES class were implemented in the Python programming language as an extension of the functionality of the Gensim library. The model was evaluated using Recall@k and NDCG@k metrics. The comparative analysis showed that the results obtained are comparable with the performance of the BPRMF, GRU4Rec and NextItRec models, which indicates the potential of the proposed model.

Keywords: recommender system, natural language processing, NLP methods, cluster analysis, 2Vec models, vectorization of user preferences, embedding

Citation: Shulgin S.A., Benderskaya E.N. Leveraging natural language processing techniques for enhanced recommender systems. Computing, Telecommunications and Control, 2025, Vol. 18, No. 1, Pp. 48–59. DOI: 10.18721/JCSTCS.18104

Научная статья
DOI: https://doi.org/10.18721/JCSTCS.18104
УДК 004.85

ИСПОЛЬЗОВАНИЕ МЕТОДОВ ОБРАБОТКИ ЕСТЕСТВЕННОГО ЯЗЫКА В ПРОДВИНУТЫХ РЕКОМЕНДАТЕЛЬНЫХ СИСТЕМАХ

С.А. Шульгин ✉ , *Е.Н. Бендерская*

Санкт-Петербургский политехнический университет Петра Великого,
Санкт-Петербург, Российская Федерация

✉ shulginsergey0@gmail.com

Аннотация. Многие рекомендательные системы используют методы NLP преимущественно в обработке контента. В работе предлагается новый подход к построению рекомендательных систем, в котором данные о пользовательских взаимодействиях с контентом рассматриваются в рамках модели естественного языка. Таким образом, предлагается модель векторизации пользовательских предпочтений Pref2Vec в качестве основы гибридной рекомендательной системы. Кроме того, предложена концепция пользовательского пространства эмбеддингов (UES), которая представляет собой набор расширенных эмбеддингов, отображающих предпочтения конечного пользователя. Также предложен новый способ применения задачи кластеризации в решении задачи построения рекомендаций. Модель Pref2Vec и класс UES были реализованы на языке программирования Python в качестве расширения функциональности библиотеки Gensim. Была произведена оценка модели при помощи метрик Recall@k и NDCG@k. Сравнительный анализ показал, что полученные результаты сравнимы с показателями моделей BPRMF, GRU4Rec и NextItRec, что говорит о перспективности предложенной модели.

Ключевые слова: рекомендательная система, обработка естественного языка, методы NLP, задача кластеризации, модели 2Vec, векторизация пользовательских предпочтений, эмбеддинг

Для цитирования: Shulgin S.A., Benderskaya E.N. Leveraging natural language processing techniques for enhanced recommender systems // Computing, Telecommunications and Control. 2025. Т. 18, № 1. С. 48–59. DOI: 10.18721/JCSTCS.18104

Introduction

The development of recommender systems (RS) is an actively developing area of IT. Natural language processing (NLP) methods allow to change the established paradigm of RS development – data about content, users and their sessions are considered within the framework of a natural language model. This approach opens new opportunities for interpreting a large amount of data.

The task of any RS is to extract and interpret information about user, content, and user's interactions with content to recommend them another content [1]. For this purpose, data mining methods have been used [2]. The different architectures of RS stem from specific information processing tasks: content recommendation, collaborative filtering, knowledge-based recommendation, and hybrid recommendation, the former being the most comprehensive one [1, 3].

The interest in research and development of RS is still wide and the issues are still challenging despite many advances in this area [4]. These issues arise at various stages of RS design and are typically formulated as five core tasks:

- *Evaluation* of RS performance and availability of datasets, including the evaluative criteria and appropriate indicators choice and calculation;
- *Sparsity overcoming*, which arises both from a large volume of input data and users' reluctance to evaluate content;

• *Scalability* of RS, which assumes that all the algorithms used should undergo performance tests on large amount of data;

• *Contextual dependence* that permits an RS to obtain information about users' environment. It improves the accuracy of recommendations;

• *The «gray sheep» problem*, which is the presence of users whose opinions do not coincide with the opinions of the group they belong to.

Machine learning methods are the one way to overcome the above problems. It enables RS to recognize regularities and patterns in amounts of data with the purpose of accuracy improvement of data representation and interpretation. While considering the main methods used in the development of RS [2], it was decided to focus on 2Vec models (Word2Vec, Paragraph2Vec, and Doc2Vec). These models are NLP models that use a neural network to represent entities (words, paragraphs, documents, etc.) as embeddings based on the context of their usage in a corpus [4, 5]. The software implementation of 2Vec models is presented in the Gensim library [6].

Thus, the user preference vectorization model (Pref2Vec) is proposed in this paper as a basis for a hybrid recommender system. The model uses the Doc2Vec model. It extends the functionality of the Gensim library. In addition, the class of user embedding space (UES) is proposed. It is used in constructing the Pref2Vec model. Further, it allows to vectorize end-user preferences by applying a new method of vectorizing text data, which considers the type of user's interaction with the content.

Vectorization of user preferences

Vectorization (2Vec models) converts text into embeddings that reflect text semantics. Handling embeddings enables to use simple, effective and intuitive algorithms in development. For example, the k-nearest neighbors (KNN) algorithm can be used within the search for the nearest embeddings in the embedding space [7]. Almost all known 2Vec models are put into practice in isolation from RS development, i.e. the models process the text only [5]. In this way, it seems appropriate to develop a model for vectorizing text information that considers the type of user's interaction with the processed content (text). Pref2Vec is a new approach to RS building, which solves the problem of implementation context dependence into the core of a RS by moving to a higher level of abstraction. It means that recommendations will be generated depending on the preferences of a specific user. This approach also permits to follow the principles of constructing a hybrid recommendation, since both static (content) and dynamic (user preferences) data are subjects to the analysis.

Let us impose constraints on the data processed by the RS core. The content will be presented as text. User's interaction with the content will be formalized as differentiated content assessment. It helps to understand the degree of user's interest in a specific content. Based on these constraints, it was decided to utilize the dataset from the GoodReads website [8]. A more detailed exploration of the input data is presented in Sec. «Evaluation of the proposed model».

The Pref2Vec model is based on Doc2Vec, which is used for the initial processing of the content, i.e. for the initial embeddings formation. The model converts documents (text fragments) to fixed-length feature vectors using the PV-DM model (Distributed Memory model of Paragraph Vector) [9]. The algorithm is built on the Word2Vec model: the task of word vectorization consists of its statistical analysis (CBoW or Skip-gram models) relative to other words by researching the context of their use. Vectorization is reached by using a neural network with stochastic gradient descent (SGD) and backpropagation [10]. Further vectorization of documents consists of considering the vectors of all words used in the document using the PV-DM model and the neural network. The optimal embedding size for Doc2Vec depends on the specific application and the size of the data. The general range of embedding sizes is from 100 to 300. In this regard, it was decided to form 200-dimensional embeddings.

It is necessary to overcome the contextual dependence problem. One of the solutions to this problem is to include a type of user's interaction semantics into an initial embedding (a vector representation

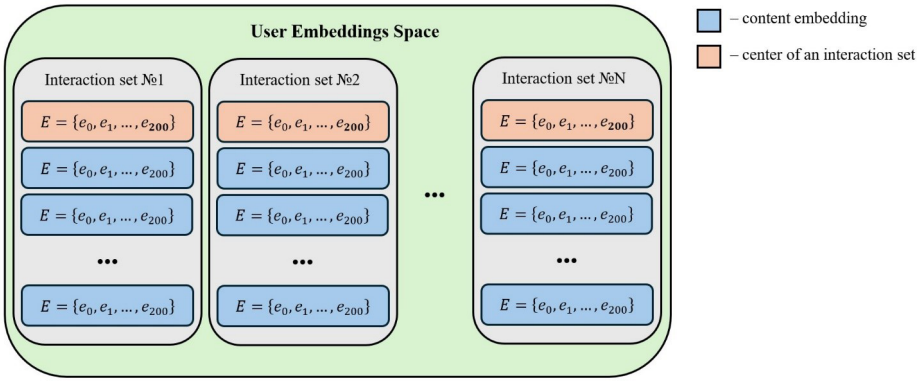


Fig. 1. Scheme of the UES content

of a text formed by Doc2Vec). Initial embeddings will be expanded by adding another one dimension that characterizes the type of user's interaction with content as the subsequent phase of the processing. Expansion of initial embeddings plays the same role as tone in tonal languages. For instance, different tones in the Chinese interpret the same orthography in different ways. In this way, a type of user's interaction with content determines the context of use.

Sets of embeddings divided by interaction types can be interpreted as clusters. The proposed application of clustering algorithms is to build recommendations analyzing information about the interaction clusters of different users. Considering the large dimensionality of the expanded embeddings only key information about the clusters should be used while building recommendations. We can say that the centers of the clusters reflect an average user preference within a specific type of interaction. Hence, let us take the centers as the key information for each user. This application of the cluster analysis permits to reduce the amount of data involved in computations. In addition, it reduces the impact of outliers in interactions clusters given the presence of "gray sheep" in the system. Therefore, there are solutions to the problem of input data sparsity and the "gray sheep" problem. This non-classical application of the clustering analysis (by computing the centers of interaction clusters) is a helpful method in designing the Pref2Vec model. It was decided to use the k-medoids [11] algorithm as the implementation of the cluster analysis. The algorithm is generally similar to k-means [12], but the centers of the clusters are medoids (elements of the clusters). This feature of the algorithm also helps to reduce the impact of the outliers on the process of building recommendations, because the effect of cluster elements distant from the majority decreases while computing the center of its medoids.

Result of the data processing for one user (space of extended embeddings) is referred to as the UES. It represents sets of extended embeddings divided by type of user's interactions with the content. The UES should likewise contain the computed centers for each set. Fig. 1 shows the scheme of the UES content.

Based on the above, the process of converting content and users' data is as follows:

1) Texts are transformed into 200-dimensional embeddings (initial embeddings) using the Doc2Vec model;

2) Embeddings are modified by adding an extra dimension, which will allow reflecting content's belonging to a specific type of user's interaction with the content. Subsequently, a UES is constructed;

3) Interaction clusters (sets) are formed as a result of constructing the UES. Afterwards, the clusters need to be analyzed by computing for their centers using the k-medoids algorithm.

The scheme of the content processing of one category of user's interactions is shown in Fig. 2.

The UES was implemented in Python using the Doc2Vec model from the Gensim library and k-medoids algorithm from the Scikit-Learn library. The Doc2Vec model was trained on annotations corpus

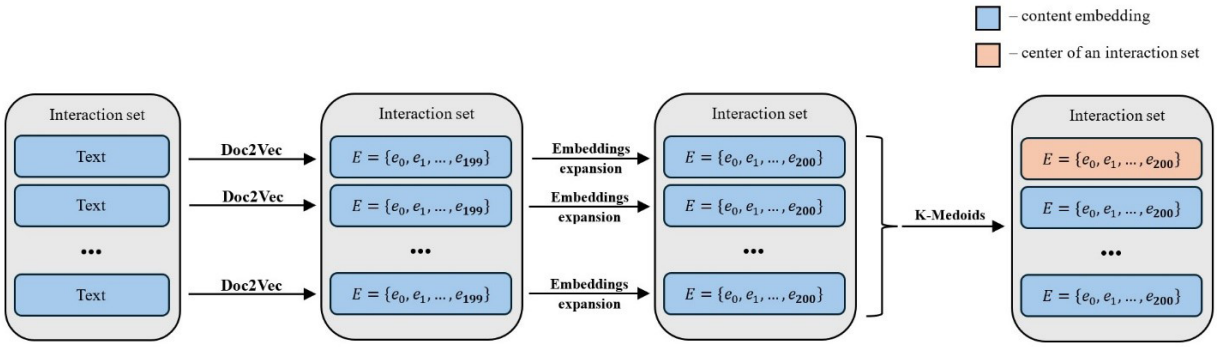


Fig. 2. Scheme of the content processing of one interaction set

from the GoodReads dataset [8]. The t-SNE method was also used to reduce the dimensionality of the embeddings for the purpose of visualization. Fig. 3 illustrates an example of the UES for one user described in the dataset. We can see that cluster partitioning of the content into interaction sets is achieved by expansion of the embeddings. This application of the cluster analysis will further search for users with similar preferences.

Mechanism of recommendation building

It is proposed to build the core of an RS at a higher level of abstraction – recommendations will be built not only depending on a content similarity, but also depending on the preferences of a specific user, i.e. on their UES. There is an opportunity to vectorize all users' interactions with content as users' preferences by the UES concept. The prospect of searching for similar UES and subsequently building recommendations based on the data obtained opens up. Therefore, it is necessary to determine a method for similar UES searching.

It has been said before that each UES has a collection of interaction centers. We can say that each collection reflects the main direction of user interests at every interaction type. Consequently, all other embeddings except centers can be removed from further calculations. We will reduce the number of embeddings considered in interaction sets. Thus, similar UES searching will be done through a comparison of interaction centers. This approach will permit to avoid a large number of calculations. It also will permit to improve the level of accuracy in determining similar UES. In this case, one should resort to calculating the cosine similarity between centers of considering UES. It is a measure of similarity between two vectors of a Hausdorff pre-Hilbert space, which is used to measure the cosine of the angle between them [13]. The measure is also actively used in 2Vec models [9]. For two vectors A and B, the cosine similarity $\cos(\theta)$ can be represented as follows:

$$\cos(\theta) = \frac{A \cdot B}{\|A\| \cdot \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}.$$

The Pref2Vec model is assumed to have a users' data corpus, i.e. a corpus of calculated UES. Next, we obtain a list of the UES, closest to a target UES in cosine similarity, when forming a recommendation for a target user. The contents of each UES from the list are potential recommendations, since these are contents that users with similar preferences to a target user interacted with. Due to this, it is necessary to rank the contents of similar UES to build a recommendation. The cosine similarity is also used to solve this problem – embeddings of the nearest UES content will be compared with content embeddings that a

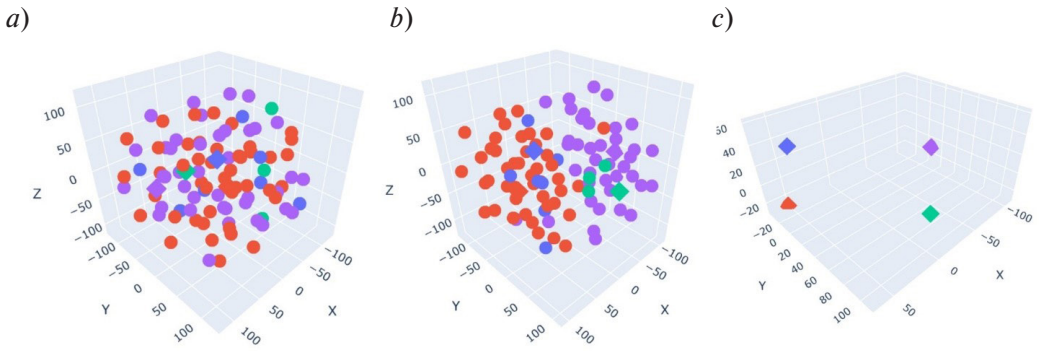


Fig. 3. Visualization of the transforming content and user data into an UES:
(*a*) embeddings formation, (*b*) embeddings expansion, and (*c*) computing centers of interaction sets

target user interacted with. Based on the calculations, we will interpret a result list of the closest content as a list of Top-K recommendations.

As a consequence, the algorithm of recommendation building is as follows:

1) Centers of interaction sets are calculated for a target user (their UES);

2) Further, a search for the nearest UES to the target is performed on the model corpus based on the calculated centers. In other words, it is a search for users with similar preferences in the corpus;

3) Elements of similar UES are compared with elements of the target UES and ranked according to a degree of similarity;

4) Ranked list of the content is a result list of Top-K recommendations.

The above-described algorithm for recommendation building by the Pref2Vec model is presented as a scheme in Fig. 4.

The Pref2Vec model was also implemented in Python. The model computes a corpus of UES by the UES class previously developed in Python (see Sec. "Vectorization of user preferences"). It furthermore allows to investigate the corpus in order to build recommendations for a target user. The developed model has a *depth* parameter, which characterizes the depth of the search for similar UES, i.e. the number of UES considered in the search. There is also a *target_interactions* parameter, which reflects the types of interactions used in the recommendations building. An example of a prediction for one of the users from the GoodReads dataset is shown in Fig. 5. We can see that the recommended books (Fig. 5, *a*) can be semantically correlated with the word clouds of the target UES (Fig. 5, *c*) based on the titles and annotations of the books (Fig. 5, *b*). The word clouds represent statistics of words occurrence for content a user interacted with [14] (the content is divided on types of interaction). A more detailed model evaluation will be made below.

Evaluation of the Pref2Vec model

For evaluation of the model, it was decided to use the GoodReads website dataset [8], since the dataset meets the requirements for the input data (see Sec. "Vectorization of user preferences"). The website provides free access to an extensive database of books and their metadata. Hence, the dataset is a collection of information obtained from the website: book data, user bookshelves data (anonymized information about user interactions with books) and various user reviews [8, 15]. Content of user bookshelves can be used to determine whether a user has read a book on their shelf and how quickly he/she did it, whether a user gave a book a differential assessing (rating), and whether he/she wrote a book review. The main interest is text data (book annotations) and user interactions data within the evaluation of the core of a RS. The interactions data will allow to categorize the text data by interaction types. The collections are also divided into different genre categories. We will use the data from the categories

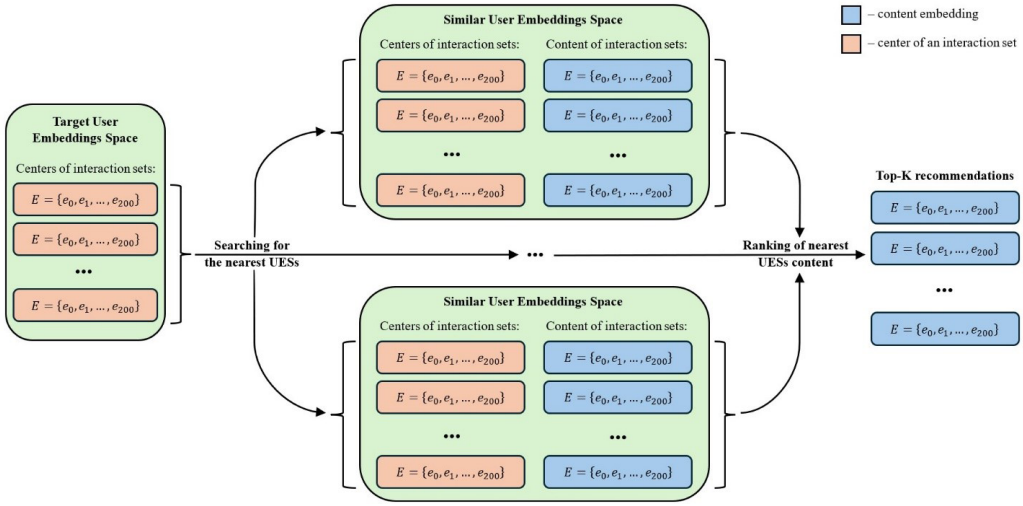


Fig. 4. Scheme of recommendation building by Pref2Vec model

"Children" and "Comics & Graphic". In total, the collections contain information on 213493 books and 17406979 user interactions.

Book annotations were preprocessed and vectorized using the Gensim library [6]. In addition, a dataset of user preferences was formed by storing book identifiers into different interaction sets (*read*, *shelved*, *rating_0*, *rating_1*, *rating_2*, *rating_3*, *rating_4*, *rating_5*) for each user. The resulting dataset was noise filtered – users whose set *rating_5* contained less than 40 elements were removed. The Pref2Vec model corpus was composed of 80% of user interactions from the dataset (sample X). The remaining 20% of interactions (sample y) were used to assess the quality of recommendation building. Elements of sample y were ranked by the maximum value of cosine similarity with sample X elements. This approach opens an opportunity to estimate not only a percentage of an inclusion of recommendation elements in the reference sample, but also to evaluate the quality of the ranking ability of the model.

Significantly, it is quite difficult to evaluate the performance of a RS due to the non-trivial nature of the recommendation task. For this reason, the range of metrics used is individual for each developed system [16]. The documentation of the RecPack metrics module [17] was discussed in that context. The module contains a large number of baseline and metrics commonly used to evaluate state-of-the-art recommendation algorithms. The Pref2Vec model generates a list of Top-K recommendations, so in this way it is necessary to refer to the metrics for evaluating the ranking quality. The following evaluation methodologies were chosen to achieve comparability with the performance of relevant models [18].

Table 1

Listwise metrics utilized in the model evaluation

Metric	Formula	Description
Recall@k	$Recall(u) = \frac{\sum_{i \in Top-N(u)} y_{u,i}^{true}}{\sum_{j \in I} y_{u,j}^{true}}$	Computes the fraction of true interactions that made it into the Top-K recommendations
NDCG@k	$NDCG(u) = \frac{DCG(u)}{IDCG(u)}$ $IDCG(u) = \sum_{j=1}^{\min\left(K, \left\|y_n^{true}\right\|\right)} \frac{1}{\log_2(j+1)}$	Computes the normalized sum of gains of all items in a recommendation list (NCDG@k – Normalized Discounted Cumulative Gain, ICDG@k – Ideal Discounted Cumulative Gain)

a)

```
Recommendations for user #e9104aee79c530cef254ee043bbfc859:
1) Book #['73800'] - similarity is 0.9994449887081361
2) Book #['157993'] - similarity is 0.9994219263956959
3) Book #['125640'] - similarity is 0.9993917947056836
4) Book #['3594968'] - similarity is 0.9993915428704876
5) Book #['20530933'] - similarity is 0.9993905111392447
```

b)

book_id	description	title
157993	Moral allegory and spiritual autobiography, Th...	The Little Prince
20530933	"I bought the milk," said my father. "I walked...	Fortunately, the Milk
73800	Come deep into the forest if you dare...but be...	The Minpins
125640	What could be scarier than Slappy, the evil du...	Slappy's Nightmare (Goosebumps Series 2000, #23)
3594968	A thoroughly subversive and hilarious twist on...	Don't Cook Cinderella

c)

Fig. 5. Result of Top-K recommendation building for a target user: (*a*) recommendations for the user, (*b*) recommended content, (*c*) word clouds of the target UES

Building a Top-K recommendation for one user by the Pref2Vec model is a rather long process. Due to this limitation, the evaluation of the RS core was performed for 200 users. The model corpus consisted of interaction data about 100000 users (UES objects). As a result of the evaluation, the values presented in Table 2 were obtained.

Table 2

NDCG@k and Recall@k metrics performance of the Pref2Vec model

Dataset	Recall@10	Recall@15	Recall@20	NDCG@10	NDCG@15	NDCG@20
GoodReads-Children	0.0904	0.1056	0.1585	0.0996	0.0769	0.0628
GoodReads-Comics	0.0542	0.1033	0.1627	0.1140	0.1099	0.0933

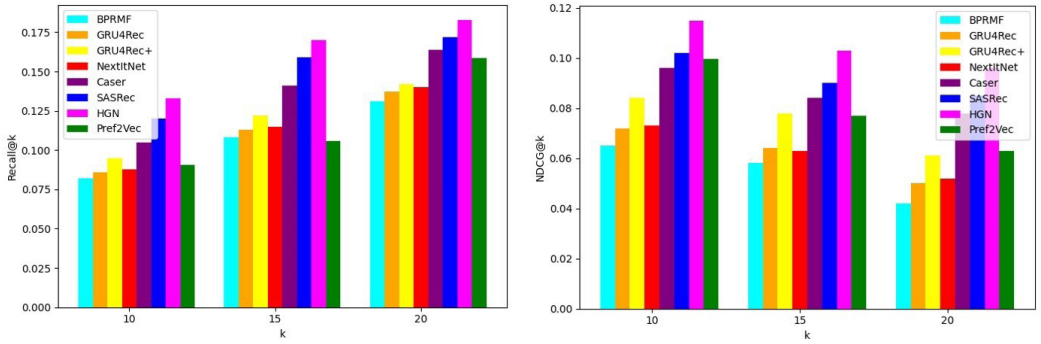
The obtained values were compared with the performance of the BPRMF, GRU4Rec, GRU4Rec+, NextItRec, Caser, SASRec and HGN models [18]. The results of the comparison are presented in Fig. 6 and 7. In addition, there is a more detailed comparison of the values for the NDCG@10 and Recall@10 metrics in Tables 3 and 4.

The Recall@k metric values of the Pref2Vec evaluation were not as high as those of some considering recommendation algorithms: HGN, SASRec, and Caser. However, at the same time, the metric values are approximately at the same level as other models: BPRMF, GRU4Rec, and NextItRec. It is also evident that the values of the Recall@k have a higher growth rate with the growth of the parameter k than most of the existing models. On the contrary, the values of the NDCG@k metrics were relatively high. This indicates a good ability of the Pref2Vec to rank recommendations. We can say that the results of the model evaluation seem to be adequate to meet expectations considering the introduction of a new approach to processing user data and the time limitations of the Pref2Vec model.

Discussion

We can say that the values obtained are approximately at the same level as some of the relevant models in the field of modern RS design. Let us consider how this model can be improved:

– The performance of the model is definitely affected by its corpus size. Because of time and resource constraints, a corpus of 100000 users data was used during the evaluation, although the GoodReads dataset has data on 876145 users in total [8]. It is likely that a larger corpus size would increase the performance.



(a) Recall@k on GoodReads-Children **(b) NDCG@k on GoodReads-Children**

Fig. 6. Comparison of the Recall@k (*a*) and NDCG@k (*b*) metrics on the «GoodReads-Children» dataset

Table 3

Comparison of the NDCG@10 and Recall@10 metrics values of the Pref2Vec model with the performance of relevant RSs (part 1)

Dataset	BPRMF	GRU4Rec	GRU4Rec+	NextItRec	Pref2Vec
Recall@10					
GoodReads-Children	0.0814	0.0857	0.0978	0.0879	**0.0904**
GoodReads-Comics	0.0788	0.0958	0.1288	0.1078	**0.0542**
NDCG@10					
GoodReads-Children	0.0664	0.0715	0.0821	0.0720	**0.0996**
GoodReads-Comics	0.0713	0.0912	0.1328	0.1171	**0.1140**

– The *depth* hyperparameter impact on the efficiency of the model was not investigated in the paper. Presumably, increasing the value of this hyperparameter will have a positive effect on the metrics, since more content will be examined during the recommendation building.

– The quality of content vectorization directly affects the quality of the recommendation building. It is possible to increase performance by varying the Doc2Vec model hyperparameters, which are used to generate the initial embeddings.

– In addition, the model performance is affected by the method of searching for interaction cluster centers. The use of methods such as Affinity Propagation [19] or Hierarchical Agglomerative Clustering [20] can have a positive impact on the values of the metrics.

– In addition, it is possible to increase the model's performance by implementing spatial computing methods from the SciPy library [21] other than cosine similarity: for example, computing the Euclidean distance [22] or the Mahalanobis distance [23].

Conclusion

In this paper, the user preferences vectorization model Pref2Vec was proposed as the core of a hybrid RS. The model is a new approach to RS design, which solves the problem of introducing context dependence into the RS core. We managed to achieve it by moving to a higher level of abstraction, where recommendations are built depending on the preferences of a specific user. The model also permits adhering to the principles of hybrid recommendation, which is the most relevant approach in RS development. Furthermore, the concept of the UES was proposed as the result of end-user data processing. It is a set of extended embeddings that display the semantics of vectorized content and the semantics

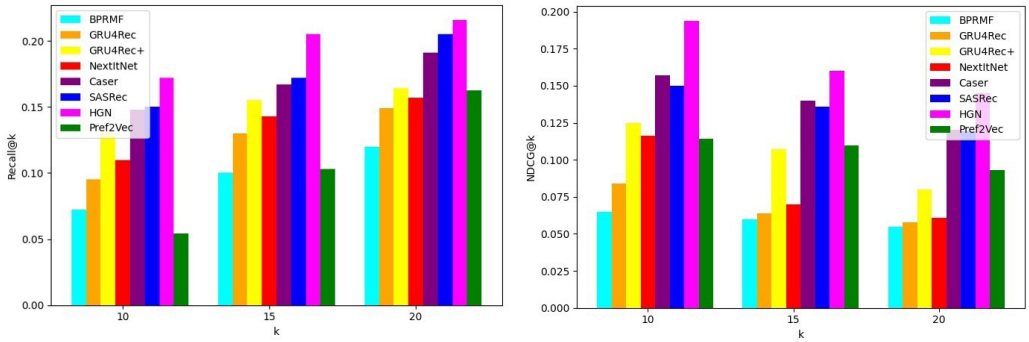


Fig. 7. Comparison of the Recall@k (*a*) and NDCG@k (*b*) metrics on the «GoodReads-Comics» dataset

Table 4

Comparison of the NDCG@10 and Recall@10 metrics values of the Pref2Vec model with the performance of relevant RSs (part 2)

Dataset	Caser	SASRec	HGN	Pref2Vec
Recall@10				
GoodReads-Children	0.1060	0.1165	0.1263	**0.0904**
GoodReads-Comics	0.1473	0.1494	0.1743	**0.0542**
NDCG@10				
GoodReads-Children	0.0943	0.1007	0.1130	**0.0996**
GoodReads-Comics	0.1629	0.1592	0.1927	**0.1140**

of user's interaction with a content. Clustering helps to tackle data sparsity, improves RS scalability, and manages the problem of “gray sheep”.

The Pref2Vec model and the UES class were implemented in Python as an extension of the Gensim library functionality. Afterwards, the model was evaluated using the Recall@k and NDCG@k metrics. Comparative analysis showed that the results obtained are comparable with the performance of the BPRMF, GRU4Rec and NextItRec models. Values of the Recall@k metric of the Pref2Vec model have a higher growth rate with the growth of the k parameter than most of the existing models. In addition, values of the NDCG@k metric showed that the Pref2Vec model has a high quality of ranking recommendations.

REFERENCES

1. **Jannach D., Zanker M., Felfernig A., Friedrich G.** *Recommender Systems: An Introduction*. Cambridge: Cambridge University Press, 2010, 352 p. DOI: 10.1017/CBO9780511763113

2. **Karatzoglou A., Hidasi B.** Deep learning for recommender systems. *11th ACM Conference on Recommender Systems* (*RecSys ’17*), 2017, Pp. 396–397. DOI: 10.1145/3109859.3109933

3. **Jannach D., Zanker M., Friedrich G.** Tutorial: Recommender Systems. *International Joint Conference on Artificial Intelligence* (*IJCAI-13*), 2013, pp. 221–236.

4. **Khusro S., Ali Z., Ullah I.** Recommender systems: Issues, challenges, and research opportunities. *Information Science and Applications* (*ICISA*), 2016, Vol. 376, Pp. 1179–1189. DOI: 10.1007/978-981-10-0557-2_112

5. **Barkan O., Koenigstein N.** ITEM2VEC: Neural item embedding for collaborative filtering. *2016 IEEE 26th International Workshop on Machine Learning for Signal Processing* (*MLSP*), 2016, Pp. 1–6. DOI: 10.1109/MLSP.2016.7738886

6. **Řehůřek R., Sojka P.** Software framework for topic modelling with large corpora. *LREC 2010 Workshop on New Challenges for NLP Frameworks*, 2010, pp. 45–50. DOI: 10.13140/2.1.2393.1847

7. **Gong C., Shen G., Guo L., Tallent N., Zhao D.** OPDR: Order-Preserving Dimension Reduction for semantic embedding of multimodal scientific data. *arXiv:2408.10264*, 2024. DOI: 10.48550/arXiv.2408.10264

8. **Wan M., McAuley J.** Item recommendation on monotonic behavior chains. *12th ACM Conference on Recommender Systems* (*RecSys '18*), 2018, Pp. 86–94. DOI: 10.1145/3240323.3240369

9. **Le Q.V., Mikolov T.** Distributed representations of sentences and documents. *arXiv:1405.4053*, 2014. DOI: 10.48550/arXiv.1405.4053

10. **Mikolov T., Chen K., Corrado G., Dean J.** Efficient estimation of word representations in vector space. *arXiv:1301.3781*, 2013. DOI: 10.48550/arXiv.1301.3781

11. **Schubert E., Rousseeuw P.J.** Fast and eager k-medoids clustering: $O(k)$ runtime improvement of the PAM, CLARA, and CLARANS algorithms. *Information Systems*, 2021, Vol. 101, Art. no. 101804. DOI: 10.1016/j.is.2021.101804

12. **Camilli G., Suter L.** NLP cluster analysis of common core state standards and NAEP item specifications. *arXiv:2412.04482*, 2024. DOI: 10.48550/arXiv.2412.04482

13. **Manning C.D., Raghavan P., Schütze H.** *An Introduction to Information Retrieval*. Cambridge: Cambridge University Press, 2008, 544 p. DOI: 10.1017/CBO9780511809071

14. **Mueller A.C.** Wordcloud (1.9.4). *Zenodo*, 2024. DOI: 10.5281/zenodo.14062883

15. **Wan M., Misra R., Nakashole N., McAuley J.** Fine-grained spoiler detection from large-scale review corpora. *57th Conference of the Association for Computational Linguistics* (*ACL*), 2019, Pp. 2605–2610. DOI: 10.18653/v1/P19-1248

16. **Schröder G., Thiele M., Lehner W.** Setting goals and choosing metrics for recommender system evaluations. *2nd Workshop on User-Centric Evaluation of Recommender Systems and Their Interfaces* (*UCERSTI2*), 2011, Pp. 23–53.

17. **Michiels L., Verachtert R., Goethals B.** RecPack: An(other) experimentation toolkit for Top-N recommendation using implicit feedback data. *16th ACM Conference on Recommender Systems* (*RecSys '16*), 2022, Pp. 648–651. DOI: 10.1145/3523227.3551472

18. **Ma C., Kang P., Liu X.** Hierarchical gating networks for sequential recommendation. *25th ACM SIGKDD Conference on Knowledge Discovery & Data Mining* (*KDD '19*), 2019, Pp. 825–833. DOI: 10.1145/3292500.3330984

19. **Hämäläinen M., Rueter J., Alnajjar K.** Analyzing Pokémon and Mario streamers' twitch chat with LLM-based user embeddings. *4th International Conference on Natural Language Processing for Digital Humanities* (*NLP4DH*), 2024, Pp. 499–503. DOI: 10.18653/v1/2024.nlp4dh-1.48

20. **Aufschläger R., Wilhelm S., Heigl M., Schramm M.** ClustEm4Ano: Clustering text embeddings of nominal textual attributes for microdata anonymization. *arXiv:2412.12649*, 2024. DOI: 10.48550/arXiv.2412.12649

21. **Virtanen P., Gommers R., Oliphant T.E. et al.** SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature Methods*, 2020, Vol. 17, Pp. 261–272. DOI: 10.1038/s41592-019-0686-2

22. **Lo K.I., Sadrzadeh M., Mansfield S.** Quantum-like contextuality in large language models. *arXiv:2412.16806*, 2024. DOI: 10.48550/arXiv.2412.16806

23. **Hart S.N., Tavolara T.E.** Measuring what matters: Intrinsic distance preservation as a robust metric for embedding quality. *arXiv:2407.21590*, 2024. DOI: 10.48550/arXiv.2407.21590

INFORMATION ABOUT AUTHORS / СВЕДЕНИЯ ОБ АВТОРАХ

Shulgin Sergey A.
Шульгин Сергей Александрович
E-mail: shulginsergey0@gmail.com
ORCID: https://orcid.org/0009-0009-1102-7333

Benderskaya Elena N.
Бендерская Елена Николаевна
E-mail: helen.bend@gmail.com

Submitted: 29.12.2024; Approved: 18.02.2025; Accepted: 04.03.2025.
Поступила: 29.12.2024; Одобрена: 18.02.2025; Принята: 04.03.2025.