Paper: jc*--**-****:**

Similarity-Based Supervised User Session Segmentation Method for Behavior Logs

Yongzhi Jin*, Kazushi Okamoto*, Kei Harada*, Atsushi Shibata, and Koki Karube***

*Graduate School of Informatics and Engineering, The University of Electro-Communications
1-5-1 Chofugaoka, Chofu, Tokyo 182-8585, Japan
** Graduate School of Industrial Technology, Advanced Institute of Industrial Technology
1-10-40 Higashiooi, Shinagawa-ku, Tokyo 140-0011, Japan
E-mail: {jin-yongzhi, kazushi, harada, karubekoki}@uec.ac.jp, shibata-atsushi@aiit.ac.jp

[Received 00/00/00; accepted 00/00/00]

arXiv:2508.16106v1 [cs.IR] 22 Aug 2025

In information recommendation, a session refers to a sequence of user actions within a specific time frame. Session-based recommender systems aim to capture short-term preferences and generate relevant recommendations. However, user interests may shift even within a session, making appropriate segmentation essential for modeling dynamic behaviors. In this study, we propose a supervised session segmentation method based on similarity features derived from action embeddings and attributes. We compute the similarity scores between items within a fixed-size window around each candidate segmentation point, using four types of features: item co-occurrence embeddings, text embeddings of titles and brands, and price. These features are used to train supervised classifiers (LightGBM, XGBoost, CatBoost, support vector machine, and logistic regression) to predict the session boundaries. We construct a manually annotated dataset from real user browsing histories and evaluate the segmentation performance using F1-score, area under the precision-recall curve (PR-AUC), and area under the receiver operating characteristic curve. The LightGBM model achieves the best performance, with an F1-score of 0.806 and a PR-AUC of 0.831. These results demonstrate the effectiveness of the proposed method for session segmentation and its potential to capture dynamic user behaviors.

Keywords: e-commerce, behavior log, session data, segmentation, embedding, classification

1 . Introduction

Recommender systems (RSs) have been increasingly adopted across various online service domains, including video, music, and e-commerce platforms. In this regard, increasing focus has been devoted toward the algorithms behind RSs, with session-based RSs (SBRSs) being particularly effective in capturing recent user preferences [1]. A session refers to a sequence of user actions performed within a certain time frame while using an application or service. For instance, a user's browsing history is a typical session. An SBRS captures user preferences from session data and generates relevant item recommendations. However, user preferences are dynamic, and sessions often contain multiple topics. Using entire sessions for training may degrade prediction accuracy. Although session segmentation is crucial for the effective processing of session data, research on segmentation methods is limited.

Recently, several approaches have been explored for session segmentation. Zhang et al. [2] introduced a method for session segmentation based on the cosine similarity between item embeddings. Inspired by this idea, we performed a unified evaluation of session-segmentation methods, each constructed by combining behavioral and textual embeddings with supervised learning models [3]. However, these methods have several limitations. They fully exploit the rich information available in the dataset and rely solely on local features to determine session boundaries, ignoring broader contexts such as features derived from a larger window size.

In this study, we propose a supervised session segmentation method that uses similarity-based features derived from item embedding and additional attributes. Each potential segmentation point is evaluated by computing the similarities within a local window centered on that point. In addition, item-level information, such as price, is incorporated to enrich the representation. These features are then used in a supervised learning framework to predict whether each point corresponds to a session boundary. Specifically, in addition to the cosine similarities from various embeddings, we define a custom similarity metric based on item prices and include it as a feature. Motivated by the above, we address the following research questions (RQs):

RQ1: How effective is the proposed method in segmenting session data?

RQ2: Which supervised learning model is most effective for session segmentation?

To address the RQs, we use real user browsing history data from Amazon. We apply natural language processing (NLP)-based embedding techniques to each browsing item and segment user behavior based on the similarities between item representations. These similarities are used as features in machine learning (ML) models, including LightGBM [4], XGBoost [5], CatBoost [6], support vector machine (SVM) [7], and logistic regression (LR). We conducted comparative experiments against existing baselines to evaluate the effectiveness of the proposed method. Because the original dataset lacked segmentation labels, we recruited annotators to label the behavioral change points and construct the ground-truth data.

The remainder of this paper is organized as follows. Section 2 reviews related work. Section 3 introduces the proposed method. Section 4 describes the dataset used in this study and the annotation process. Section 5 details the experimental methodology. Section 6 presents the results and discussion. This study extends our previous study [3].

2 . Related Work

A session segmentation method aims to detect segment points in sequences of user-item interactions occurring within a continuous time period in the e-commerce domain. The concept of a session varies across different domains. For instance, in the web domain, sessions are typically represented as sequences of user queries or web page visits; in dialogue systems, they correspond to exchanges of utterances between users; and in the e-commerce domain, they are often defined by users' product browsing histories. This section provides a brief historical overview of how session segmentation has been approached across various domains, such as the web and dialogue systems, before focusing on its application and challenges in the e-commerce domain.

2.1 . Session Segmentation for Web

On the Web, session segmentation plays a crucial role. In search engine systems, dividing and analyzing user behavior at the session level enables a more accurate understanding of the user search context. This contextual information can be leveraged in user-oriented learning methods. Furthermore, one can predict the users' next search queries and provide more effective query suggestions by capturing topic transitions within sessions.

One of the most fundamental session segmentation methods is the time interval-based approach proposed by He et al. [8]. Several conceptual clustering-based approaches have been proposed for session segmentation. For instance, the COBWEB-based method proposed by Hou et al. [9] uses a search engine dataset and employs features such as time intervals and query likelihoods to learn a session-segmentation model in an unsupervised manner. However, feature engineering in this approach remains basic and does not fully leverage the rich content information inherent in the data.

To address this limitation, Ustinovskiy et al. [10] proposed a supervised learning method that employed Gradient Boosting Decision Trees (GBDTs) to estimate the probability of two pages belonging to the same session. Their approach uses fully anonymized real browsing logs collected from a major search engine via a dedicated browser toolbar, and leverages a rich set of features, including URL features, textual features, and temporal distances between page visits. Inspired by this prior work, we formulated a session segmentation task as a binary classification problem, to predict whether a session boundary exists between two consecutive items. This formulation enables the application of various ML models, including GBDT models such as LightGBM, XGBoost, and CatBoost, as well as classical models such as SVM and LR.

2.2 . Session Segmentation for Dialogue

Semantic segmentation techniques have been applied to various tasks beyond the RS domain, such as dialogue and chat session analyses [11–14]. In particular, Lee et al. [13] utilized Doc2Vec [15] to encode user utterances during conversations. This approach enables session segmentation by detecting shifts in semantic similarity across consecutive utterances, allowing the system to effectively identify topic transitions. For targeted advertising, extracting and classifying blocks of utterances from instant messages that reflect a user's interests is essential. They performed dialogue session segmentation by vectorizing the utterances using Doc2Vec and dividing the sessions based on their semantic consistency, leveraging cosine similarity. The approach adopted by Lee et al. demonstrated the effectiveness of embedding-based representations in identifying topic shifts, which is highly relevant to segmentation tasks in various domains, including the present study. However, although most existing research has focused on dialogue or chat sessions, session segmentation methods in the e-commerce domain, particularly for tasks such as targeted advertising, remain largely unexplored.

2.3. Session Segmentation for E-commerce

In the e-commerce domain, an accurate understanding of user search behaviors is crucial for delivering sophisticated recommendations. Session-based behavioral analysis has been actively studied to better capture the user context [16,17]. However, session segmentation was typically performed using heuristic time thresholds that lacked a rigorous or data-driven basis.

An embedding approach to session segmentation was proposed by Zhang et al. [2]. They developed a method that segmented user sessions, such as browsing histories or item ratings, into multiple dummy sessions using behavior-based embedding and cosine similarity. We extended session segmentation approaches using not only Item2Vec item embeddings but also embeddings of item titles, and performed a comparative evaluation of supervised and unsupervised models [3]. The results demonstrated that utilizing the similarity between Item2Vec embeddings outperforms approaches that directly input embedding vectors into the model. However, this approach has several limitations. Specifically, it uses a narrow window size that considers only the immediate context around the segmentation point, and relies solely on embeddings as features, which fails to fully capture the relationships between items.

This study aimed to develop a more comprehensive session segmentation method by building on our previous method and extending its approach. Our earlier findings suggested that the cosine similarity between item embeddings effectively identified meaningful session boundaries. Based on this insight, we incorporated similarity scores as features for the segmentation models. We also used embedding-based similarity scores between the current item and its neighboring items as input features for the classification.

3. Proposed Model

In this study, we propose a supervised session segmentation method that predicts whether a given point in a sequence of user interactions corresponds to a session boundary. The proposed approach leverages similarity-based features computed from item embeddings and attributes, and applies ML models to perform segmentation. The proposed method comprises the following main components.

3.1. Item Embeddings

3.1.1. Behavior-based Embedding

First, we introduced an embedding approach based on Item2Vec [18], a method derived from Word2Vec [19]. Word2Vec is a neural embedding model originally developed for NLP tasks. It transforms words into dense vector representations based on their surrounding context, grounded in the distributional hypothesis. Item2Vec adapts this principle to behavioral data in e-commerce by treating user sessions as sequences of interactions, thereby learning the vector representations of items that reflect their contextual co-occurrence with other items.

In this study, Item2Vec was employed as a behavior-based embedding method to capture the relationships between items based on session-level co-occurrence patterns. We trained an Item2Vec model using the Word2Vec implementation provided by the Gensim library, where each session was treated analogously to a sentence in NLP. The used hyperparameters for training are shown in Appendix A.

3.1.2. Text-based Embedding

We incorporated text-based embeddings of item titles and brand names to capture the semantic relationships between items beyond user behavior. Traditionally, text embeddings are obtained by first tokenizing text into individual words, embedding each word using Word2Vec [19], and then averaging the resulting word vectors. This simple and efficient approach has been widely adopted in various NLP tasks [20]. However, such methods are limited in their ability to model complex contextual semantics.

In this study, we employed a text-embedding model to embed item titles and brand names to capture the relationships between items based on their textual attributes. Specifically, we utilized the text-embedding-3-small model [21], which is a state-of-the-art text-embedding model developed by OpenAI. This model produces high-quality vector representations of text and demonstrates improved performance compared with its predecessor, text-embedding-ada-002. It achieves a higher performance across various embedding tasks, particularly in multilingual and English-language benchmarks, making it well-suited for applications such as text retrieval, similarity analysis, and semantic search.

3.2. Point-wise Feature Extraction

The proposed method extracts four similarities between two items surrounding segmentation points. The details are summarized as follows:

- *Behavior similarity*: cosine similarity between item embedding vectors obtained from the Item2Vec model.
- *Brand similarity*: cosine similarity between brand embedding vectors generated by text-embedding-3-small.
- *Title similarity*: cosine similarity between product title embedding vectors generated by text-embedding-3-small.
- *Price similarity*: a score derived from the ratio of item prices.

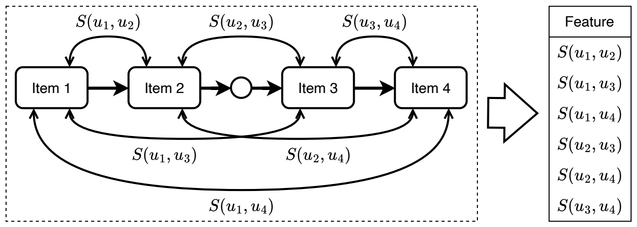


Fig. 1. : Example of feature extraction for window size $w = 2$

We define four similarity functions $S_t(u_i, u_j)$, $t \in \{\text{behavior}, \text{brand}, \text{title}, \text{price}\}$ between two items u_i and u_j to calculate the behavior, brand, title, and price similarities, respectively. The functions S_{behavior}, S_{brand}, and S_{title} are defined as follows:

$$S_t(u_i, u_j) = \frac{\boldsymbol{v}_{u_i}^T \cdot \boldsymbol{v}_{u_j}}{\|\boldsymbol{v}_{u_i}\| \|\boldsymbol{v}_{u_j}\|} \quad u_i, u_j \in W$$

where W represents the set of items within a window, and $\boldsymbol{v}$ denotes the behavior, brand, or title embeddings of an item. Furthermore, S_{price} is defined as follows:

$$S_{\text{price}}(u_i, u_j) = \exp\left(-\frac{|p_i - p_j|}{\min(p_{i+1}, p_{j+1})}\right) \quad u_i, u_j \in W,$$

where p represents the price of an item u.

As illustrated in Fig. 1, a sliding window was applied around each candidate segmentation point to extract contextual information. The window size w determines the range of items considered:

- If $w = 1$, only the immediately preceding and following items are included.
- If $w = 4$, the four items before and four after the segmentation point are included.

This resulted in $2w$ items being referenced, leading to $d = \binom{2w}{2}$ possible pairwise comparisons. All possible pairwise combinations within this window were used to compute similarity scores. These scores are aggregated to form feature vector $\boldsymbol{x} \in \mathbb{R}^{4d}$, which represents the local context.

3.3. Segmentation Point Classification

After feature extraction, we applied a supervised classifier $f_{\boldsymbol{\theta}} : \mathbb{R}^{4d} \longrightarrow \{0, 1\}$ to identify whether the input point $\boldsymbol{x}$ is the segmentation point, where $\boldsymbol{\theta}$ is a learned parameter vector. The parameter $\boldsymbol{\theta}$ is preliminarily determined using n-training samples $(y, \boldsymbol{x}) \in \{0, 1\} \times \mathbb{R}^{4d}$. Here, a label of 1 indicates a segmentation point, and 0 denotes a non-segmentation point. In this study, we employ five classifiers: LightGBM, XGBoost, CatBoost, SVM, and LR.

Table 1. : Statistical summary of session dataset (locale = JP)

Total Sessions	979,119
Total Items	389,888
Mean Session Length	5.48 items
Standard Deviation	4.08 items
Minimum Session Length	3 items
Median Session Length	4 items
Maximum Session Length	475 items

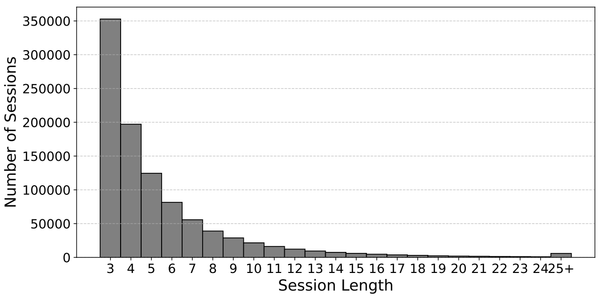


Fig. 2. : Distribution of session lengths

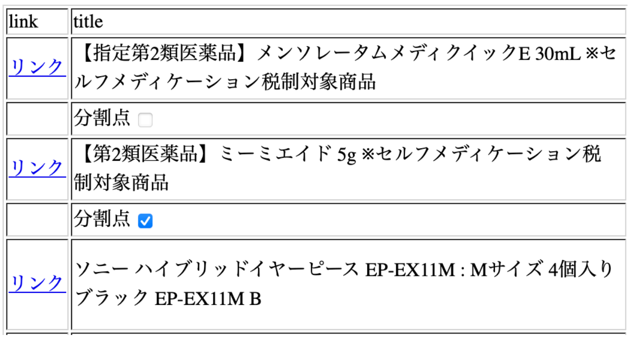
4. Session-Segmentation Experiment

This experiment evaluated the segmentation performance to investigate the effectiveness of the proposed method (RQ1) and the relative performance of various supervised learning models (RQ2). This section outlines the (1) dataset used in this study, (2) annotation workflow, and (3) evaluation procedure.

4.1. Dataset

In this experiment, we used the Amazon M2 dataset [22], released for the Amazon KDD Cup '23. This dataset provides customer session data and is notable for supporting six locales—English, German, Japanese, French, Italian, and Spanish—unlike traditional session datasets. However, we focus exclusively on the Japanese data (locale = JP). Each item within a session includes a locale, an ID, and several attributes, such as title, price, brand, color, size, model, material, author, and description. Among these, we primarily utilized three representative attributes—title, price, and brand—as they are considered to capture the essential characteristics of the item.

The dataset comprises of two components: *prev_items*, representing items viewed by the user, and *next_item*, indicating items actually purchased. We merge these components and treat them as a single session. Table 1 presents a statistical summary of the session dataset with a locale set to JP. The distribution of session lengths is shown in Fig. 2. As illustrated, sessions of length three were the most frequent, with the frequency decreasing as the session length increased.



link	title
リンク	【指定第2類医薬品】メンソレータムメディクイックE 30mL ※セルフメディケーション税制対象商品
	分割点 ☐
リンク	【第2類医薬品】ミーミエイド 5g ※セルフメディケーション税制対象商品
	分割点 ☑
リンク	ソニー ハイブリッドイヤーピース EP-EX11M : Mサイズ 4個入り ブラック EP-EX11M B

Fig. 3. : Example of annotation form

Table 2. : Distribution of segmentation points labeled in each session by the annotator

Annotator	0	1	2	3+
1	0.577	0.287	0.097	0.040
2	0.600	0.300	0.070	0.030
3	0.650	0.273	0.063	0.013
4	0.590	0.330	0.063	0.017
5	0.573	0.333	0.053	0.040
6	0.620	0.240	0.103	0.037
7	0.640	0.287	0.060	0.013
8	0.753	0.220	0.020	0.007

4.2. Annotation

Because the Amazon M2 dataset did not include segmentation point labels, we performed an annotation process with eight participants from the authors' laboratory. All the tasks were performed using a custom-developed form implemented in PHP and operated on an Apache-based server. The annotation interface is shown in Fig. 3. Session data for annotation were randomly sampled from the complete set of available sessions. Consequently, a total of 2,400 sessions were annotated. Among these, 10,904 segmentation points were identified, including 1,230 positive examples (segmentation points) and 9,674 negative examples (non-segmentation points). Hereafter, we define the annotated data as follows: segmentation points are labeled as 1 and non-segmentation points as 0.

Table 2 summarizes the number of session segmentation points labeled within the annotated sessions, along with the distribution of segmentation points per session. Each row corresponds to the segmentation results provided by a different annotator. The results indicate that approximately 40% of the sessions required segmentation. Furthermore, the small variation among the annotators suggested that the annotations were consistent and reliable.

Finally, we analyze the number of segmentation points per session. We defined sessions with five or more items as *long sessions*, and those with fewer than five items as *short sessions*. Table 3 presents a distribution of the annotation results. The first row corresponds to long sessions, whereas the second row represents short sessions. Each column indicates the number of segmentation points within a session, and the numerical values indicate the number of sessions for each combination. As summarized in Table 3, long sessions tended to contain more segmentation points. The distribution of sessions requiring segmentation was 32.4% for short sessions and increasing to 44.1% for long sessions. This result aligns with intuitive expectations.

Table 3. : Distribution of segment numbers and session length types

Length	0	1	2	3+
Short session	923	379	60	4
Long session	578	302	99	55

4.3. Evaluation Procedure

4.3.1. Overview of the Segmentation Task

In this experiment, we address session segmentation as a binary classification task. The objective was to identify whether a given point in a sequence of user interactions corresponds to a session boundary. The overall experimental pipeline is illustrated in Fig. 4.

We used all the session data summarized in Table 2 to train an Item2Vec model that generated item embeddings based on item co-occurrence patterns within sessions. To prevent data leakage, annotated sessions were excluded from the training set, and only the remaining unlabeled sessions were used for training. Simultaneously, item titles and brands stored in the database were embedded using the text-embedding-3-small model to capture semantic similarities.

For each candidate segmentation point in a session, four types of similarity scores, $S_{\text{behavior}}(u_i, u_j)$, $S_{\text{brand}}(u_i, u_j)$, $S_{\text{title}}(u_i, u_j)$, and $S_{\text{price}}(u_i, u_j)$, were computed for all item combinations (u_i, u_j) such that $u_i, u_j \in W$ and $u_i \neq u_j$, where W denotes the set of items within a local window. These scores were aggregated to form input vectors for supervised classification. The annotated segmentation labels were then used to train the classification models, which output a probability score indicating the likelihood that each point is a session boundary.

To assess the segmentation performance, we used standard classification metrics: F1-score, area under the precision-recall curve (PR-AUC), and area under the receiver operating characteristic curve (ROC-AUC). These metrics were used to evaluate the proposed method (RQ1) and compare the performance of various classification models (RQ2).

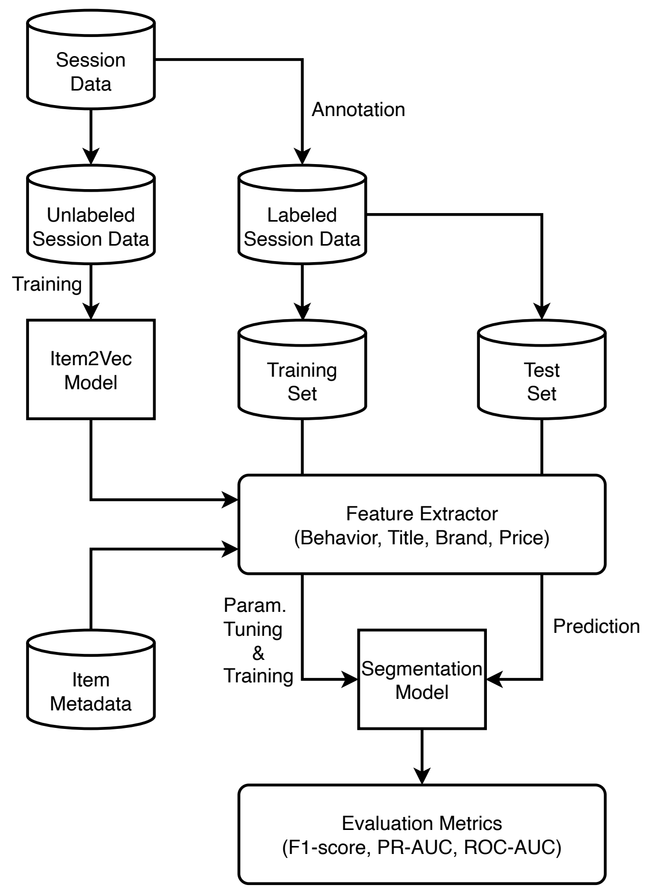


Fig. 4. : Experiment flow

4.3.2. Data Partitioning and Hyperparameter Tuning

The annotated dataset $D = \{(y_i, \boldsymbol{x}_i)\}_{i=1}^{n}$ was split into training and test sets, denoted as D_{train} and D_{test}, respectively, in a 4:1 ratio. In this experiment, we used D_{train} to train the classification models, and evaluated the segmentation performance on D_{test}. The hyperparameters of the classification models were optimized using five-fold cross-validation (CV) on D_{train}, in conjunction with Optuna [23]. Details of the models and their corresponding tunable parameters are summarized in Appendix B. The optimized hyperparameters were then used to evaluate the classification models on D_{test}

A key consideration in this CV process is the avoidance of data leakage. The input features were extracted from points surrounding the segmentation boundaries. However, if segmentation points from the same session appear in both the training and validation sets, the model may learn session-specific patterns instead of generalizable features, resulting in overestimated performance. To mitigate this issue, we employed a Group K-Fold CV, treating each session as a distinct group. Finally, we performed hyperparameter optimization using Optuna, with the F1-score as the objective metric.

4.3.3. Padding and Missing Value Handling

We refer to the features of the surrounding items when identifying whether a point is a segmentation boundary. For each candidate point, a fixed-size window is applied to extract contextual items from both the preceding and following sides. However, if the window size exceeds the session boundaries (e.g., at the beginning or end of a session), padding is applied by repeating the nearest available item. For instance, given a session $[u_1, u_2, u_3, u_4]$ and a candidate point between u_1 and u_2, a window size of two results in the padded context $[u_1, u_1, u_2, u_3]$, where the leftmost item is duplicated to meet the required window size. This ensures that the input vector maintains a consistent dimensionality, regardless of the position of the candidate point in the session.

As described in Section 3.1, we train the Item2Vec model using session data. Annotated sessions are excluded from the training set to avoid potential data leakage, annotated sessions are excluded from the training set. This exclusion leads to a situation in which some items appear only in the test set and therefore lack corresponding Item2Vec embeddings during testing. Consequently, the training of models such as SVM and LR becomes difficult because the similarity scores involving such items cannot be computed. To mitigate this issue, we assigned a similarity score of zero if the embedding of either item in the item pair was unavailable.

5 . Results and Discussion

This section reports the session segmentation performance of both the baseline and proposed methods on test set D_{test}. Table 4 summarizes the evaluation results (F1-scores calculated using a threshold of 0.5). In Table 4, the columns represent the classification methods, whereas the rows are grouped by evaluation metrics, and show the results for different window sizes. The highest score for each metric is highlighted in bold.

As a baseline, we used the cosine similarity-based segmentation method, which achieved the highest accuracy in the experiments reported in [3]. This unsupervised approach computes the cosine similarity between two items adjacent to each candidate segmentation point. If the similarity exceeded a predefined threshold, the point was classified as a segmentation point.

5.1 . Effectiveness of the Proposed Method (RQ1)

As summarized in Table 4, the proposed method consistently outperformed the baseline across all supervised learning models in terms of F1-score and PR-AUC. This performance advantage holds for different window sizes, indicating the robustness of the

Table 4. : Session segmentation performance

Metric	w	LightGBM	XGBoost	CatBoost	SVM	LR	Cosine Similarity
F1-score	1	0.793	0.791	0.796	0.785	0.774	0.728
	2	0.802	0.793	0.798	0.793	0.788	–
	3	**0.806**	0.794	0.792	0.790	0.782	–
	4	0.795	0.790	0.798	0.783	0.792	–
PR-AUC	1	0.814	0.814	0.816	0.810	0.800	0.637
	2	0.823	0.814	0.821	0.814	0.810	–
	3	**0.831**	0.816	0.815	0.810	0.801	–
	4	0.817	0.812	0.824	0.801	0.812	–
ROC-AUC	1	0.860	0.857	0.863	0.850	0.844	**0.932**
	2	0.864	0.860	0.858	0.860	0.856	–
	3	0.859	0.860	0.855	0.859	0.858	–
	4	0.859	0.856	0.855	0.860	0.861	–

proposed method. In particular, increasing the window size beyond one improves the segmentation performance in most metrics. LightGBM serves as a representative example, achieving the highest overall scores. For instance, its F1-score improved from 0.793 to 0.806 and the PR-AUC from 0.814 to 0.831 when the window size increased from one to three, demonstrating the benefit of incorporating a broader context.

In contrast, the baseline achieved the highest ROC-AUC of 0.932 at a window size of one, suggesting that the cosine similarity is highly effective in distinguishing positive and negative samples. However, the F1-score and PR-AUC were lower, indicating that it may be less reliable for precise session segmentation than supervised learning-based methods. A possible reason for the lower ROC-AUC of the proposed method is that hyperparameter tuning was performed using the F1-score, which emphasizes recall. This likely caused the model to favor positive predictions, increasing false positives. Given the class imbalance in the test samples (10.82% positive vs. 89.18% negative), this bias may have degraded the performance in the negative class, reducing the ROC-AUC.

In summary, these results support RQ1 by showing that the proposed method outperforms the baseline in key metrics such as the F1-score and PR-AUC, and benefits from a broader contextual window.

5.2. Best Supervised Learning Model (RQ2)

This section compares the segmentation performance of each model to identify the best-performing model. We also analyzed the features emphasized by the top-performing model and discussed the important features that are commonly shared across models.

5.2.1. Model Performance Comparison and Insights

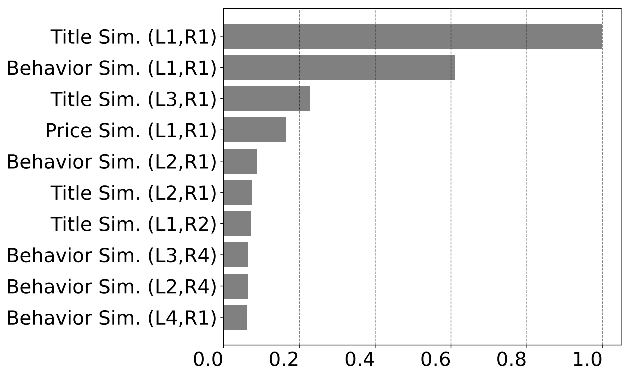
As summarized in Table 4, LightGBM achieved the highest performance, with an F1-score of 0.806 and a PR-AUC of 0.831 when the window size was set to three, which were the best scores among all the models. These results suggest that LightGBM effectively identifies session boundaries while maintaining a well-balanced tradeoff between precision and recall. In addition, Table 4 reveals several notable trends across models and metrics. The tree-based models (LightGBM, XGBoost, and CatBoost) consistently outperformed the linear models (SVM and LR) in terms of the F1-score and PR-AUC, indicating their advantage in capturing complex interactions among features.

Another key observation concerns the window size w. Several models improve the performance as w increases from one to two or three, but some show that the performance decreases performance drops at $w = 3$ or $w = 4$. This suggests that although incorporating contextual information is helpful, excessive context or long-range dependencies may introduce noise and degrade segmentation accuracy.

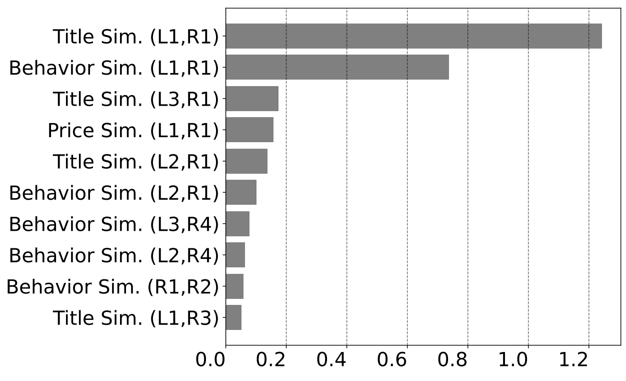
In summary, LightGBM proved to be the most effective model under the proposed method, offering the best overall balance across the evaluation metrics and providing a clear answer to RQ2.

5.2.2. Feature Importance Analysis

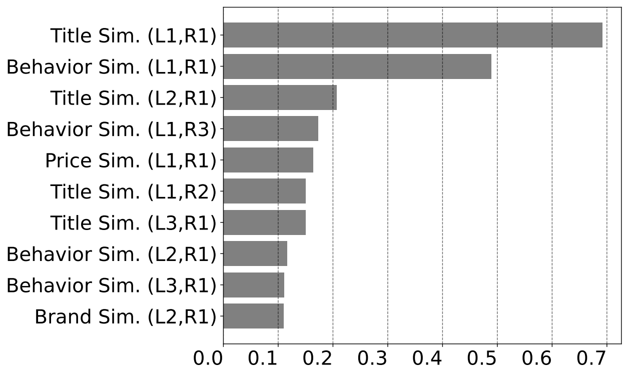
We assessed feature contributions to session segmentation using SHapley Additive exPlanations (SHAP) values, with the window size set to four. Although the best segmentation performance was achieved at a window size of three, we used four to examine feature interactions in a broader context. The feature importance was measured using the average magnitude of the SHAP values across the test samples. For clarity, the feature pairs are denoted as (L_i, R_j), where L_i and R_j indicate the i-th item to



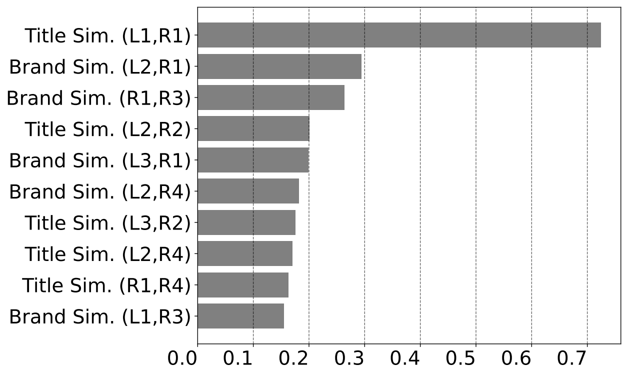
(a) LightGBM importance



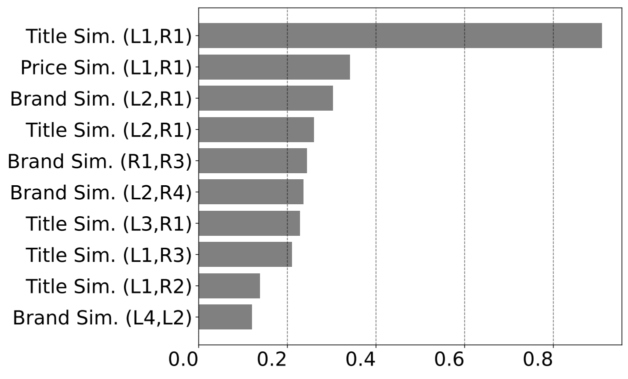
(b) XGBoost Importance



(c) CatBoost importance



(d) SVM Importance



(e) LR importance

Fig. 5. : Feature importance comparison across models

the left and j-th item to the right of a potential segmentation point, respectively. For instance, (L_1, R_1) represents the feature pair between the last item before and first item after the segmentation point.

The results are shown in Fig. 5. As shown in Figs. 5a, 5b, and 5c, item title similarity has the strongest impact on the predictions across all models. Tree-based models (LightGBM, XGBoost, and CatBoost) rely heavily on behavior-based embeddings, in addition to title embeddings. By contrast, SVM and LR place significant emphasis not only on title embeddings but also on brand similarity.

6 . Conclusion

In this study, we proposed a supervised session segmentation method that uses similarity-based features derived from item embeddings and additional item attributes. This method evaluates each candidate segmentation point using features computed within a fixed-size window, capturing the contextual relationships between items. We trained several classifiers, including LightGBM, XGBoost, CatBoost, SVM, and LR, and evaluated their segmentation performance using a manually annotated dataset constructed for this study. The segmentation accuracy was assessed using the F1-score, ROC-AUC, and PR-AUC as evaluation metrics.

The experimental results demonstrated that the LightGBM-based model achieved the highest performance, with an F1-score of 0.806 and a PR-AUC of 0.831. Compared with the baseline based on cosine similarity with Item2Vec embeddings, the proposed method demonstrated improvements of 10.71% in the F1-score and 30.61% in the PR-AUC, confirming the effectiveness of incorporating contextual similarity features and supervised learning. These findings suggest that the proposed method can serve as a reliable preprocessing step for downstream tasks such as session-based recommendation, user modeling, or behavioral pattern mining. Moreover, the modularity of the approach allows easy integration with different types of embeddings and additional item-level metadata.

Despite these promising results, several limitations remain. First, the method depends on manually annotated segmentation labels, which are expensive and time-consuming to obtain at scale. Second, because the dataset did not include user identifiers, the segmentation model applied a global criterion across all sessions, limiting its ability to capture personalized session boundaries and learn user-specific behavioral patterns.

In future work, we plan to explore semi-supervised or weakly supervised methods to reduce the dependence on manual annotation. In addition, we intend to evaluate the practical impact of the improved segmentation on downstream tasks, such as click prediction and personalized recommendations.

Acknowledgements

This work was supported by JSPS KAKENHI Grant Numbers JP23K21724, JP23K24953, JP24K21410.

References:

[1] S. Wang, L. Cao, Y. Wang, Q. Z. Sheng, M. A. Orgun, and D. Lian, "A Survey on Session-based Recommender Systems", ACM Computing Surveys, 54(154):1–38, 2021.

[2] Y. Zhang, N. Gao, and J. Chen, "A Practical Defense against Attribute Inference Attacks in Session-based Recommendations", In Proceeding of 2020 IEEE International Conference on Web Services, pp. 355–363, 2020.

[3] Y. Jin and K. Okamoto, "Evaluation of Session Segmentation Methods Using Behavior and Text Embeddings", In The 11th International Symposium on Computational Intelligence and Industrial Applications and the 15th China-Japan International Workshop on Information Technology and Control Applications (ISCIIA&ITCA 2024), 2024.

[4] G. Ke, Q. Meng, T. Finley, T. Wang, W. Chen, W. Ma, Q. Ye, and T.-Y. Liu, "LightGBM: A Highly Efficient Gradient Boosting Decision Tree", In Advances in Neural Information Processing Systems, volume 30. Curran Associates, Inc., 2017.

[5] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System", In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794. Association for Computing Machinery, 2016.

[6] L. Prokhorenkova, G. Gusev, A. Vorobev, A. V. Dorogush, and A. Gulin, "CatBoost: unbiased boosting with categorical features", In Proceedings of the 32nd International Conference on Neural Information Processing Systems, pp. 6639–6649, 2018.

[7] C. Cortes and V. Vapnik, "Support-Vector Networks", Mach. Learn., 20(3):273–297, 1995.

[8] D. He and A. Göker, "Detecting Session Boundaries from Web User Logs", In Proceedings of the 22nd Annual Colloquium on IR Research IRSG 2000, pp. 57–66, 2000.

[9] Z. Hou, M. Cui, P. Li, L. Wei, W. Ying, and W. Zuo, "Session Segmentation Method Based on COBWEB", In Proceeding of 2012 IEEE 2nd International Conference on Cloud Computing and Intelligence Systems, volume 01, pp. 148–153, 2012.

[10] Y. Ustinovskiy, A. Mazur, and P. Serdyukov, "Intent-Based Browse Activity Segmentation", In Advances in Information Retrieval, pp. 242–253. Springer Berlin Heidelberg, 2013.

[11] Y. Song, L. Mou, R. Yan, L. Yi, Z. Zhu, X. Hu, and M. Zhang, "Dialogue Session Segmentation by Embedding-Enhanced TextTiling", In Proceeding of the Interspeech 2016, pp. 2706–2710, 2016.

[12] S. Xiong, J. Guo, H. Xie, and Y. Liu, "Long Dialogue Segmentation Based on Memory Similarity Algorithm", In 2024 4th International Symposium on Artificial Intelligence and Intelligent Manufacturing, pp. 948–951, 2024.

[13] H. Lee and Y. Yoon, "Interest Recognition from Online Instant Messaging Sessions Using Text Segmentation and Document Embedding Techniques", In Proceeding of 2018 IEEE International Conference on Cognitive Computing, pp. 126–129, 2018.

[14] H. Gao, R. Wang, T.-E. Lin, Y. Wu, M. Yang, F. Huang, and Y. Li, "Unsupervised Dialogue Topic Segmentation with Topic-aware Contrastive Learning", In Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 2481–2485, 2023.

[15] Q. Le and T. Mikolov, "Distributed representations of sentences and documents", In Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32, ICML'14, pp. II–1188–II–1196, 2014.

[16] Y. Ma, B. M. Narayanaswamy, H. Lin, and H. Ding, "Temporal-Contextual Recommendation in Real-Time", In Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, KDD '20, pp. 2291–2299, 2020.

[17] J. J. Seol, Y. Ko, and S.-g. Lee, "Exploiting Session Information in BERT-based Session-aware Sequential Recommendation", In Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '22, pp. 2639–2644. Association for Computing Machinery, 2022.

[18] O. Barkan and N. Koenigstein, "ITEM2VEC: Neural Item Embedding for Collaborative Filtering", In Proceeding of 2016 IEEE 26th International Workshop on Machine Learning for Signal Processing (MLSP), pp. 1–6, 2016.

[19] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and Their Compositionality", In Proceeding of the 26th International Conference on Neural Information Processing Systems, volume 02, pp. 3111–3119, 2013.

[20] I. Yamada, M. Suzuki, K. Yamada, and L. Li, "Introduction to Large language Models", Gijutsu-Hyohron Co., Ltd, 2023.

[21] OpenAI. "New embedding models and API updates". `https://openai.com/ja-JP/index/new-embedding-models-and-api-updates/`, 2024. Accessed:14-Feb-2025.

[22] W. Jin, H. Mao, Z. Li, H. Jiang, C. Luo, H. Wen, H. Han, H. Lu, Z. Wang, R. Li, Z. Li, M. X. Cheng, R. Goutam, H. Zhang, K. Subbian, S. Wang, Y. Sun, J. Tang, B. Yin, and X. Tang, "Amazon-M2: A Multilingual Multi-locale Shopping Session Dataset for Recommendation and Text Generation", arXiv preprint arXiv:2307.09688, 2023.

[23] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A Next-generation Hyperparameter Optimization Framework", In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 2623–2631, 2019.

Appendix A . Hyperparameters of the Item2Vec Model Training

- vector_size: 200
- sample: 10^{-3}
- negative: 1
- window: 6
- min_count: 1
- epochs: 100
- sg: 1

Appendix B . Parameter Search Ranges for Classifiers

B.1 . LightGBM

- learning_rate: LogFloat[0.0001, 0.1]
- feature_fraction: Float[0.5, 1]
- lambda_l2: LogFloat[0.1, 10]
- num_leaves: Int[4, 768]
- min_sum_hessian_in_leaf: LogFloat[0.0001, 100]
- bagging_fraction: Float[0.5, 1]

B.2 . XGBoost

- learning_rate: LogFloat[0.0001, 0.1]
- colsample_bytree: Float[0.5, 1.0]
- gamma: LogFloat[0.001, 100]
- lambda: LogFloat[0.1, 10]
- max_depth: Int[3, 14]
- min_child_weight: LogFloat[0.0001, 100]
- subsample: Float[0.5, 1]

B.3 . CatBoost

- learning_rate: LogFloat[0.0001,0.1]
- depth: Int[3, 10]
- random_strength: Int[0, 100]
- bagging_temperature: LogFloat[0.01, 100]

B.4 . SVM

- C: LogFloat[0.0001, 10]
- gamma: LogFloat[0.0001, 1]

B.5 . LR

- C: LogFloat[0.0001, 10]