Session Similarity Based Approach for Alleviating Cold-start Session Problem in e-Commerce for Top-N Recommendations

Ramazan Esmeli, Mohamed Bader-El-Den & Hassana Abdullahi

Keywords: cold-start sessions, recommender systems, session-based recommender systems

Abstract: Cold-start problem is one of the main challenges for the recommender systems. There are many methods developed for traditional recommender systems to alleviate the drawback of cold-start user and item problems. However, to the best of our knowledge, in session based recommender systems cold-start session problem still needs to be investigated. In this paper, we propose a session similarity-based method to alleviate drawback of cold-start sessions in e-commerce domain, in which there are no interacted items in the sessions that can help to identify users' preferences. In the proposed method, product recommendations are given based on the most similar sessions that are found using session features such as session start time, location, etc. Computational experiments on two real-world datasets show that when the proposed method applied, there is a significant improvement on the performance of recommender systems in terms of recall and precision metrics comparing to random recommendations for cold-start sessions.

1 INTRODUCTION

Recommender Systems (RS) are widely used in many digital platforms including e-commerce, video and music since RS can help retrieve and recommend most relevant items from massive item options. Although RS are important, they suffer several setbacks. For example, in the e-commerce domain, when a user visits anonymously or visits without browsing any item, providing product recommendations will be challenging since there is little or no information about the user. According to a dataset released by DIGINETICA[1], it is found that 62.3% of the users are not registered in the system (Wu and Yan, 2017). This is known as cold-start problem (Son, 2016). There are two types of cold-start problems in RS (Son, 2016; Herce-Zelaya et al., 2020), namely, cold-start item problem and cold-start user problem. Cold-start item problem is when a new item added to a system or database has few or no user-new item interactions, while in cold-start user problem, a new user who is not registered to the system starts browsing items or is a newly registered user who has no previous history, making it difficult for RS algorithms to identify this user's preferences.

Generally, in cold-start item problems, content-based approaches are preferred (Lika et al., 2014; Anwaar et al., 2018). For example, using items' content features, similar items are ranked and recommended. On the other hand, since it is not easy to give specific product recommendations for the cold-start user, random or most popular items are recommended(Son, 2016). Therefore, in both cases, the accuracy of the RS can be low. There are several approaches that have been proposed to alleviate new user and item problems (Son, 2016; Herce-Zelaya et al., 2020; Son, 2016; Silva et al., 2019). Users' demographic data (Son, 2016; Silva et al., 2019) is the most commonly used data where similar users are found using their demographic information and items these users interacted are recommended for the new user. Also, in some works (Safoury and Salah, 2013; Bouadjenek et al., 2016), user demographic labels are matched with product features. For instance, the match between Movie's age group and user's age group products were recommended. Furthermore, users' account are associated with their social media profiles (Carrer-Neto et al., 2012; Rosli et al., 2015; Bouadjenek et al., 2016). In this case, their social media accounts can provide useful information about users preferences such as the pages, movies and books they like, and these details can be used to create better recommendations.

In this paper, we propose a method for solving cold-start user problem for Session based RS(SBRS) in e-commerce domain where no user prior information is available. Our proposed method uses session features such as the time session started, location details where users are connected to the website, the platform they use (mobile or desktop platform), etc. to find similar sessions. From these sessions, using their last interacted item, we create product recommendations. A final recommendation list is created

[1]http://diginetica.com/

applying ”majority voting” (Aledo et al., 2017) approach from recommendation pool created from similar sessions. Computational experiments show that our proposed approach has improved performance of RS when compared to random item RS. The main contributions of this paper are as follows:

1. a framework for cold-start session is introduced.
2. majority voting based recommendation creation process is designed.
3. proposed framework has been validated and compared with random item recommendation using two real-world datasets.

The remainder of the paper is organised as follows. In Section 2, we present a literature review of the relevant studies in e-commerce domain for solving cold-start problems. Section 3 introduces the proposed framework. Section 4 presents the comparative experiments of the proposed framework involving two real-world RS datasets. Finally, Section 5 draws conclusions and provides future research directions.

2 RELATED WORKS

In this section, we provide the most relevant works related to cold-start problem in e-commerce domain. As cold-start problems are the major problems of the RS, they have gained a lot of attention from the scientific community. Approaches for solving cold-start problems can be grouped into three categories based on the solution approach followed. These are: (i) content-based approaches such as demographic information or individual labels for items, (ii) approaches that extract and use user's external data (for example, information from social media) and (iii) approaches that use initial information provided by users.

Regarding content-based approach (Safoury and Salah, 2013) designed a method that takes users' demographic information and movies' attributes into consideration to create a recommendation list based on the similarity of these features. They aimed to reduce the impact of cold-start movie problem that have little user interactions. Also, in (Lika et al., 2014), authors used classification algorithms using users' demographic information in order to identify similar users. Their proposed classification algorithms were integrated into pure Collaborative Filtering (CF) algorithms.

Moreover, in another work (Ralph et al., 2020) user and item descriptions are mined to find relationships between items and users. Relationships were created by text mining (item, user descriptions) and vectors for each document were created. These vectors were used to find similar products for the new user based on cosine similarity. Clustering is another method applied for alleviating cold-start user problem in CF (Pereira and Hruschka, 2015; Zhang et al., 2013). Zhang et al. (2013) applied bi-clustering that combines user and item features to associate like minded users with items. For the association, they predicted ratings for unrated items from like-minded user clusters and similar item clusters. Pereira and Hruschka (2015) designed a framework that combines simultaneous co-clustering learning with CF in order to alleviate new user problem. In their method, users' demographic information was used for clustering purpose.

Users' external data such as social media data is used to reduce the effect of cold-start problems (Sahebi and Cohen, 2011; Castillejo et al., 2012; Tian and Liang, 2017; Sedhain et al., 2014; Guy et al., 2010). For example, Guy et al. (2010) trained a model using Matrix Factorisation (MF) that learn latent factors for social tags. These factors were used to find relevant items for new users. Moreover, in (Tian and Liang, 2017) trust relationship in the social media is utilised, in which users' friends were categorised as trusted users and based on their preferences, new recommendations were created for the new user. Similarly, He and Chu (2010) presented a method that takes a user's social media preferences and influence from friends into account for alleviating cold-start user problem.

Lastly, using randomly collected user preferences for new items is another approach for reducing the effect of cold-start problem. An example of this approach can be seen in (Aharon et al., 2015). The authors designed an online method that randomly asks a group of user for their preferences for newly added items. Based on the collected preferences, new items are recommended to other like-minded users. Their method showed a significant performance improvement in terms of Root Mean Square Error (RMSE) evaluation metric.

2.1 Limitation of Previous Works and Proposed Contributions

Previous works showed improvement on alleviating cold-start problem when users and items have features (tags, preferences, social media). However, in e-commerce domain users can visit the website anonymously or without any data description. Consequently, SBRS suffers while providing product recommendations. In order to decrease the effect of cold-start session problem in the e-commerce domain, we propose a framework that can find most similar sessions based on the session attributes (user location, time user visited, device type, etc.) which does not require any user input.

The framework provides recommendations for the most similar sessions, and using majority voting strategy, final recommendation list is created for the cold-start session.

3 FRAMEWORK

In order to get more accurate product recommendations, a framework that utilises session features is designed (Figure 1). Since in cold-start sessions, there are no item interactions, identifying users' exact intention in the session is very challenging. However, product recommendations can be estimated from sessions that have similar features with cold-start session. This framework helps to find the most similar sessions and create a product recommendation pool using the item interactions of these sessions. Based on the majority voting method final recommendation list is created. The reason for choosing majority voting is to choose top n $n \in \{10,20,30\}$) high confidence (high frequency) recommended items from similar sessions.

The designed framework has three main steps. These are;

1. Data preparation, which includes session log collection and pre-processing. In pre-processing, the sessions have less than two item interactions are eliminated in order to have a meaningful RS models in the sense of creating item relation.
2. Second phase is designed for training the nearest neighbour model that helps to find the most 20 similar sessions from train dataset when a set of features is provided as input. Test dataset consists of session logs created in the last week of the whole duration of session logs. Also, in test dataset interacted items are hidden where these items are used for validation purpose. Also, the features of each test session are used to find the most similar sessions in train dataset.
3. Final phase of the framework is necessary for creating recommendation list using majority voting method, in which top n ($n \in \{10,20,30\}$) highest number of appeared items in the recommendations from each session are selected. In this phase recommendations from similar sessions are derived using different RS models. These are Item-Item KNN, Random and Session Popularity based recommendations. We use same parameter settings for these models, as explained in (Hidasi et al., 2015). In these models, next item recommendations are given based on the last clicked item. After collecting recommendations from similar sessions and selecting most frequented items, a final recommendation list is created. Then the final created recommendation list is evaluated with ground truth items in the test session. Evaluation metrics used in this paper are recall and precision, which are very common in the RS domain (Silva et al., 2019).

The details of the second and third phases of the framework are presented in following subsections.

3.1 Training Nearest Neighbour Model

This step consists of feature selection to train the nearest neighbour model and to find top n_s nearest sessions.

For the model training, we set $n = 100$ as neighbours, and other parameters have remained as default for the nearest neighbour algorithm as shown in [2].

Selected features from session specifications are:

1. day of the week: This feature indicates which day of the week the session has started.
2. hour of the day: Show the hour of the day the session has started.
3. day of the year: This feature represents which day of the year session started. This is important since users in a specific season can look for a particular group of items.
4. device type(mobile, desktop): Shows what kind of platform user uses in order to visit the e-commerce platform.
5. operator system: Shows the operating system the device used to connect to e-commerce platform uses such as Windows 10, IOS, Android, etc.
6. longitude: Shows longitude information of the location that the user is connected from.
7. latitude: Indicates latitude information of the location that the user is connected from.

Except for longitude, latitude, day of the year and hour of the day, other features are categorical. Therefore in order to represent these features, one-hot encoding is applied. After applying one-hot encoding, we have 34 features to represent each session. As seen in Algorithm 1, after creating nearest neighbour model, for each session we find $n_s \in \{10,20,30,40\}$ most similar sessions S. These sessions are used as

[2]https://scikit-learn.org/stable/modules/neighbors.html#unsupervised-nearest-neighbors

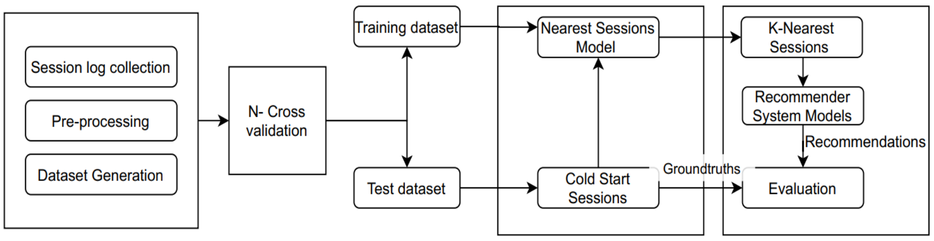

Figure 1: Framework for alleviating cold-start session problem

Algorithm 1 Hybrid approach for cold-start user session problem

```
dataset for training $D$.
dataset for testing $T$.
$RS_M = TrainModel(D)$    // Item-Item KNN and Session Pop based models
$NN = TrainNNModel(D)$    // Nearest Neighbour model to find similar sessions
**for** each $n_c$ in $N\ cross\ validation$ **do**
  $t = Selected\ Test\ Sessions$    // select 1000 sessions in each cross-validation iteration
  **for** each $s_s$ in $t$ **do**
    $GT_{s_s}$ // ground truth items of $s_s$
    $S = NN(s_s)$    // get $N_s \in \{10, 20, 30, 40\}$ similar sessions $S$ based on feature of $s_s$
    $r = \{\}$
    **for** each $s$ in $S$ **do**
      $r_s = RS_M(s)$
      $r = r \cup r_s$    // Collect recommended items $r_s$ from each similar session $s$ based on the last clicked item of $s$
    **end for**
    $r = Rank(r)$    // Rank recommendation List based on Majority voting
    $r_{final}$    // create final recommendation list by top $N \in \{10, 20, 30\}$ highest voted items.
    $Evaluate(r_{final}, GT_{s_s})$
    evaluate using recall and precision metrics.
  **end for**
**end for**
```

seed sessions in order to retrieve product recommendations r_s with the last interacted item of each session. This step is explained in detail in the next subsection.

3.2 Training Recommender System Models and Creating Final Recommendation List

In this paper, three different RS models are implemented. These are Item-Item KNN, Session-Based Popularity and Random RS models. In order to design these models, we apply a similar approach to (Hidasi et al., 2015). Also, the parameters for the models are kept the same as shown in (Hidasi et al., 2015). We give details of the parameters used in each RS model as follows;

1. Random Based RS: This recommender model does not get any special parameters.
2. Session Popularity Based RS: Session Popularity predictor that gives higher scores to items with higher number of occurrences in the session. Ties are broken up by adding the popularity score of the itemHidasi et al. (2015). We set $top_n = 100$ for items in each session with top 100 highest score.
3. Item-Item KNN based RS: This recommender model gives prediction scores for a selected set of items on how likely they are the next items in the session. Parameters for this recommender model are: $n_{sims} = 100$ which indicates the number of recommended items that only give back non-zero scores to the n_{sims} most similar items. $lmbd = 20$ is a regularisation parameter that exempts similar items with incidental co-occurrences. $alpha = 0.5$ is the cosine normalisation value that balances the similarity between two items.

Train dataset is used in order to create RS models. For each session in the test dataset, most similar sessions are found. One hundred recommended items are retrieved and added to the recommendation list pool by using last interacted item in each similar session. Final recommendation is created using majority voting method, in which highest frequent N($N \in (10, 20, 30)$) items are selected as final recommendation list.

4 Experiments and Results

We conduct a set of experiments to validate the performance of our proposed framework on cold-start sessions. For evaluation and comparison of our method, we use two performance metrics based on

Item-Item KNN, and Random and Session Popularity based models.

4.1 Dataset Specifications

In this paper, we use two real world datasets collected from e-commerce platforms by a UK based personalisation company. For pre-processing, sessions which have less than two item interactions are eliminated to have better correlation between items. Table 1 shows the number of sessions, items and interactions of datasets before and after pre-processing.

Table 1: Statistics about datasets

Dataset	#session	#item	#interaction
ds 1	2490591	10811	4892053
ds 2	1190897	2316	2211056
After Pre-processing			
ds 1	461627	9605	2479139
ds 2	186239	2260	1052636

Since we use session features (time, day of the week, operating system, etc.) in order to create session similarities, we provide statistics about these features.

Figure 2 shows the number of the user interactions with e-commerce platform by the day of week for both dataset. Monday is the highest number of user interactions for ds 1, while Thursday and Saturday are the highest number of user interactions days for the ds 2. Moreover, Figure 3 presents statistical

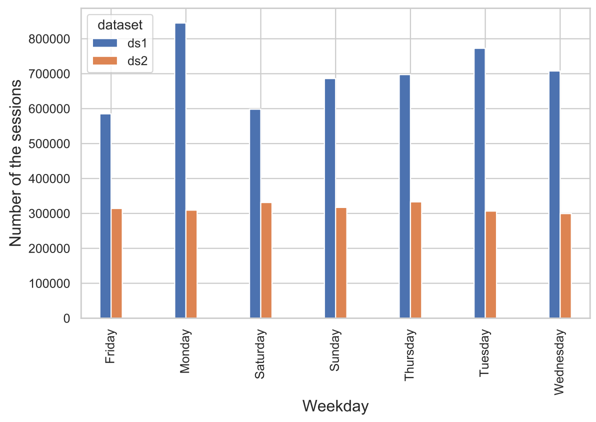

Figure 2: User interaction statistics by day of week

analysis of the number of user interactions by used platform by users. It can be seen that both datasets have similar trend for the platforms.

Figure 4 shows the number of user interactions by operating systems used. Similar to Figure 3, Figure 4 also shows similar trends for number of user interactions by used operating systems.

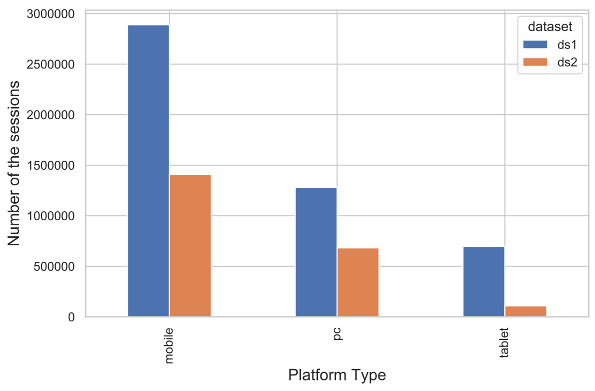

Figure 3: User interaction statistics by platform

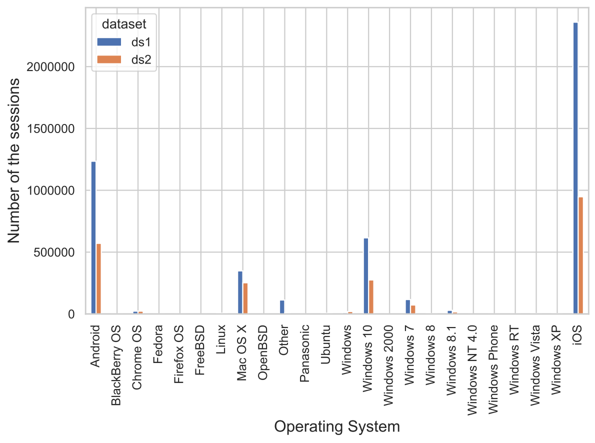

Figure 4: User interaction statistics by platform

4.2 Evaluation metrics

To assess the performance of the proposed framework, precision and recall metrics are used. In RS domain, *recall@n* is the ratio between the length of correct predicted items within top n recommendations and the length of test items (ground truth) and *precision@n* is the ratio between the length of correct predicted items within top n recommendations and the length of total recommended items. They are computed as follows:

$$recall@n = \frac{|relevant\ recommendations|}{|relevant\ items|} \tag{1}$$

$$precision@n = \frac{|relevant\ recommendations|}{|total\ recommendations|} \tag{2}$$

After having final ranked recommended items, we evaluate the proposed models with above mentioned metrics for top n recommendations where $n \in \{10, 20, 30\}$ since users are more interested in the recommendations in top of the recommendation list.

Table 2: Performance comparison of RS models on proposed framework using two datasets with different N_s parameter

	Models	**p@10**	**r@10**	**p@20**	**r@20**	**p@30**	**r@30**	**N_s**
ds 1	KNN	0.028	0.278	0.018	0.363	0.014	0.418	
	Session Pop	0.005	0.045	0.003	0.068	0.003	0.090	
	Random RS	0.000	0.000	0.000	0.000	0.000	0.001	
	KNN cold-start	0.001	0.007	0.001	0.013	0.001	0.021	n=10
	Session Pop cold-start	0.001	0.013	0.001	0.022	0.001	0.030	
	KNN cold-start	0.002	0.021	0.002	0.033	0.001	0.038	n=20
	Session Pop cold-start	0.001	0.007	0.001	0.016	0.001	0.037	
	KNN cold-start	**0.002**	0.021	**0.002**	**0.041**	**0.002**	**0.062**	n=30
	Session Pop cold-start	0.001	0.011	0.001	0.018	0.001	0.027	
	KNN cold-start	0.002	**0.023**	0.002	0.036	0.002	0.05	n=40
	Session Pop cold-start	0.001	0.011	0.001	0.019	0.001	0.026	
ds 2	KNN	0.038	0.382	0.025	0.491	0.019	0.558	
	Session Pop	0.007	0.073	0.010	0.202	0.008	0.236	
	Random RS	0.001	0.006	0.000	0.009	0.000	0.013	
	KNN cold-start	0.004	0.044	0.004	0.082	0.004	0.116	n=10
	Session Pop cold-start	0.003	0.026	0.002	0.041	0.003	0.075	
	KNN cold-start	0.007	0.067	0.006	0.109	0.005	0.147	n=20
	Session Pop cold-start	0.004	0.044	0.004	0.071	0.004	0.112	
	KNN cold-start	0.006	0.064	0.006	0.120	**0.005**	**0.164**	n=30
	Session Pop cold-start	0.002	0.023	0.002	0.041	0.002	0.061	
	KNN cold-start	**0.007**	**0.069**	**0.006**	**0.125**	0.005	0.164	n=40
	Session Pop cold-start	0.002	0.018	0.002	0.041	0.002	0.062	

4.3 Experiments

The main aim of the experiments is to measure the accuracy of the recommendations of the proposed framework for cold-start sessions comparing to random item recommendations. The fundamental idea of our approach is that similar sessions which have similar features can share similar user preferences.

Nearest neighbour and RS models are created using train dataset. Moreover, for each $n_s \in \{10, 20, 30, 40\}$ value, a new nearest neighbour model is created. For each created nearest neighbour model, final recommendation list is created using two different RS model.

For each similar session $s \in s_1, s_2 ... s_{n_s}$, we derive 100 product recommendations from each RS model and create a final recommendation list based on majority voting method. We illustrate how majority voting works in small example; lets say recommended items from session s_1 are $\{i_1, i_2, i_{100}, i_{10}\}$, and recommendations from s_2 are $\{i_1, i_7, i_{100}, i_{34}\}$. So, items $\{i_1 \& i_{100}\}$ are the same recommended times found in both sessions $\{s_1, s_2 ... s_{n_s}\}$. Therefore, items i_1 and i_{100} are selected and added to the final recommendation list. The final recommendation list is ranked based on the the number of vote each item gets. For instance, if item i_1 appeared 10 times in the recommendations while i_3 appeared 6 times, then i_1 is listed before i_3. After the recommendation list is ranked based on the votes, we evaluate the RS accuracy by selecting top n items from the ranked list using recall and precision metrics.

4.4 Results

Table 2 shows the results of the proposed framework applied to two different RS models. We report RS models' performance on top n ($n \in \{10, 20, 30\}$) recommendations using recall@n (r), precision@n (p) metrics. N_s shows the selected number of similar sessions for each test session, which are derived from the nearest neighbour model based on test session features. In our experiments, we use random RS model as the baseline to evaluate the performance of our proposed framework. Random RS models are commonly used in the case where there is a lack of prior information (e.g. item interactions, user preferences, associated social media account, etc.) about the user (Negre et al., 2013; Rohani et al., 2014; Castillejo et al., 2012).

Experiments results show that regardless of the N_s parameter, when the proposed framework applied, RS models show better performance than Random RS for both datasets. It can be seen from the results that when RS does not have any information about the new session (cold-start session), the performance of RS is low. However, where a session has interacted items, the performance of the RS models improves significantly. For example, when sessions have interacted items, experiment results show that Item-Item KNN based RS can show around 41% and 55 % recall scores in ds 1 and ds 2 respectively.

On the other hand, when the proposed framework is applied to two well-known RS models, it can be seen that the recall score can increase significantly in comparison with random RS model in cold-start sessions. In the results table, we also show the impact of N_s parameter on RS performance. The results indicate that increasing N_s parameter after a certain point does not help improve the results. We found that $N_s = 30$ is the best value for achieving the best RS accuracy score. Moreover, in each N_s parameter and dataset, the experiment results showed that Item-Item KNN based RS is better than Session Popularity based RS model.

4.5 Conclusion

Cold-start problems are major problems of the RS. In this paper, we introduced a novel framework that can deal with the cold-start problem for the new sessions that have no prior user information such as item interactions. The proposed framework can find similar sessions based on session features such as time session started, users' location, users' platform, etc. and create recommendations from these sessions. Computational experiments using two datasets proved that the proposed framework significantly improved recommendation accuracy and alleviate cold-start user session problem. We applied the proposed framework on Item-Item KNN and Session Popularity based RS models. Nevertheless, the proposed framework is a general framework that can be applied to different kinds of RS model, including Deep-Learning based models (Hidasi et al., 2015; Gabriel De Souza et al., 2019; Ludewig et al., 2019).

For future work, other session-based datasets could be tested on the framework to substantiate the robustness of the proposed framework. Also, clustering approach (Zhang et al., 2013; Yanxiang et al., 2013) can be applied to find similar sessions. For example, in the first step, sessions can be clustered into groups, and based on new session features the cluster group that session belongs to can be identified then most similar sessions from that cluster can be used in order to create a recommendation list. Also, other available session features can be included for nearest neighbour model creation to find similar sessions that can represent new sessions more accurately.

REFERENCES

Aharon, M., Anava, O., Avigdor-Elgrabli, N., Drachsler-Cohen, D., Golan, S., and Somekh, O. (2015). Excuseme: Asking users to help in item cold-start recommendations. In *Proceedings of the 9th ACM Conference on Recommender Systems*, pages 83–90.

Aledo, J. A., Gámez, J. A., and Molina, D. (2017). Tackling the supervised label ranking problem by bagging weak learners. *Information Fusion*, 35:38–50.

Anwaar, F., Iltaf, N., Afzal, H., and Nawaz, R. (2018). Hrs-ce: A hybrid framework to integrate content embeddings in recommender systems for cold start items. *Journal of computational science*, 29:9–18.

Bouadjenek, M. R., Hacid, H., and Bouzeghoub, M. (2016). Social networks and information retrieval, how are they converging? a survey, a taxonomy and an analysis of social information retrieval approaches and platforms. *Information Systems*, 56:1–18.

Carrer-Neto, W., Hernández-Alcaraz, M. L., Valencia-García, R., and García-Sánchez, F. (2012). Social knowledge-based recommender system. application to the movies domain. *Expert Systems with applications*, 39(12):10990–11000.

Castillejo, E., Almeida, A., and López-De-Ipiña, D. (2012). Alleviating cold-user start problem with users' social network data in recommendation systems. In *Workshop on Preference Learning: Problems and Applications in AI (PL-12) at ECAI*, pages 28–33.

Gabriel De Souza, P. M., Jannach, D., and Da Cunha, A. M. (2019). Contextual hybrid session-based news recommendation with recurrent neural networks. *IEEE Access*, 7:169185–169203.

Guy, I., Zwerdling, N., Ronen, I., Carmel, D., and Uziel, E. (2010). Social media recommendation based on people and tags. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 194–201.

He, J. and Chu, W. W. (2010). A social network-based recommender system (snrs). In *Data mining for social network data*, pages 47–74. Springer.

Herce-Zelaya, J., Porcel, C., Bernabé-Moreno, J., Tejeda-Lorente, A., and Herrera-Viedma, E. (2020). New technique to alleviate the cold start problem in recommender systems using information from social media and random decision forests. *Information Sciences*.

Hidasi, B., Karatzoglou, A., Baltrunas, L., and Tikk, D. (2015). Session-based recommendations with recurrent neural networks. *arXiv preprint arXiv:1511.06939*.

Lika, B., Kolomvatsos, K., and Hadjiefthymiades, S. (2014). Facing the cold start problem in recommender systems. *Expert Systems with Applications*, 41(4):2065–2073.

Ludewig, M., Mauro, N., Latifi, S., and Jannach, D. (2019). Performance comparison of neural and non-neural approaches to session-based recommendation. In *Proceedings of the 13th ACM Conference on Recommender Systems*, pages 462–466.

Negre, E., Ravat, F., Teste, O., and Tournier, R. (2013). Cold-start recommender system problem within a multidimensional data warehouse. In *IEEE 7th International Conference on Research Challenges in Information Science (RCIS)*, pages 1–8. IEEE.

Pereira, A. L. V. and Hruschka, E. R. (2015). Simultaneous co-clustering and learning to address the cold start problem in recommender systems. *Knowledge-Based Systems*, 82:11–19.

Ralph, D., Li, Y., Wills, G., and Green, N. G. (2020). Recommendations from cold starts in big data. *Computing*, pages 1–22.

Rohani, V. A., Kasirun, Z. M., Kumar, S., and Shamshirband, S. (2014). An effective recommender algorithm for cold-start problem in academic social networks. *Mathematical Problems in Engineering*, 2014.

Rosli, A. N., You, T., Ha, I., Chung, K.-Y., and Jo, G.-S. (2015). Alleviating the cold-start problem by incorporating movies facebook pages. *Cluster Computing*, 18(1):187–197.

Safoury, L. and Salah, A. (2013). Exploiting user demographic attributes for solving cold-start problem in recommender system. *Lecture Notes on Software Engineering*, 1(3):303–307.

Sahebi, S. and Cohen, W. W. (2011). Community-based recommendations: a solution to the cold start problem. In *Workshop on recommender systems and the social web, RSWEB*, page 60.

Sedhain, S., Sanner, S., Braziunas, D., Xie, L., and Christensen, J. (2014). Social collaborative filtering for cold-start recommendations. In *Proceedings of the 8th ACM Conference on Recommender systems*, pages 345–348.

Silva, N., Carvalho, D., Pereira, A. C., Mourao, F., and Rocha, L. (2019). The pure cold-start problem: A deep study about how to conquer first-time users in recommendations domains. *Information Systems*, 80:1–12.

Son, L. H. (2016). Dealing with the new user cold-start problem in recommender systems: A comparative review. *Information Systems*, 58:87–104.

Tian, H. and Liang, P. (2017). Personalized service recommendation based on trust relationship. *Scientific Programming*, 2017.

Wu, C. and Yan, M. (2017). Session-aware information embedding for e-commerce product recommendation. In *Proceedings of the 2017 ACM on conference on information and knowledge management*, pages 2379–2382. ACM.

Yanxiang, L., Deke, G., Fei, C., and Honghui, C. (2013). User-based clustering with top-n recommendation on cold-start problem. In *2013 third international conference on intelligent system design and engineering applications*, pages 1585–1589. IEEE.

Zhang, D., Hsu, C.-H., Chen, M., Chen, Q., Xiong, N., and Lloret, J. (2013). Cold-start recommendation using bi-clustering and fusion for large-scale social recommender systems. *IEEE Transactions on Emerging Topics in Computing*, 2(2):239–250.