Context Incorporation Techniques for Social Recommender Systems

Isam Mashhour Al Jawarneh, Paolo Bellavista , Antonio Corradi, Luca Foschini , Rebecca Montanari

Dipartimento di Informatica – Scienza e Ingegneria, University of Bologna
Viale Risorgimento 2, 40136 Bologna, Italy
{isam.aljawarneh3, paolo.bellavista, antonio.corradi, luca.foschini, rebecca.montanari}@unibo.it

Abstract— **The problem of information overloading is prevalent in recommendations websites and social networks. Users seek relevant recommendations from like-minded connections. User-item interactions (i.e., ratings) are prevalent in recommendation websites such as Netflix, whereas user-user connections are the interaction sought in social websites such as Twitter. Social recommender systems seek to generate recommendations for users based on similar preferences of their close friends. Because social networks do not normally contain user-item interactions, social recommender systems are typically hybridized with other recommenders (e.g., website recommenders such as Netflix) that provide such interaction. However, current systems are unaware of the user's additional contextual information when coupled with social counterparts. In this paper, we propose a context-aware deep learning-based recommender system, US-NCF, in support for social recommender systems. Our experiments show that US-NCF outperforms state-of-art counterparts by significant magnitudes.**

Keywords— context-aware, collaborative filtering, deep learning recommender system, social network

I. INTRODUCTION

Huge amounts of data are coming daily from various sources for supporting decision making in all aspects our life. Most importantly, the data that is useful for helping consumers make decisions by recommending appropriate items. Information overloading occurs because of the pace and diversity of data that is arriving to online systems. Recommender systems (RSs) offer viable solutions for the information overloading problem. They are designed to provide users with personalized recommendations that fit their preferences [1].

Traditional RSs utilize user interactions (either explicit on the form of 'user-item-rating' or implicit as 'user-item') to learn a prediction model for supporting future recommendations. However, data is often contextually tagged (enriched with contextual features that has an impact on the rating decisions) but the conventional RSs are not designed to process that kind of additional features.

Context-aware RSs are pivotal for improving recommendations in social networks, such as suggesting friendships and connections [2]. Recent literature suggests that social users trust suggestions from their social network connections only in certain contexts [2]. Having said that, being able to appropriately incorporate contextual information in Social RS (SRS for short) enables generating more accurate personalized recommendations in social networks.

This has encouraged the emergence of various context-aware recommender systems (known collectively as CARS). They generally work by incorporating contextual data using three types of methods: prefiltering, postfiltering and contextual modeling. Contextual pre-filtering uses contexts as filters to filter out irrelevant rating profiles and then applies a standard recommender to generate the recommendation list. Contextual post-filtering applies a standard recommender first and then uses contexts as filters to filter out irrelevant recommendations or re-rank the item recommendations. In contextual modeling approaches, contexts are utilized as parts of the learning process to develop the predictive models based on which recommendations are generated [3].

The most two predominant pre-filtering approaches are item-splitting [4, 5] and user-splitting (a.k.a. user micro-profiling [6]. Item-splitting is based on the idea that an item that receives different ratings in different contexts should be split and treated as if it was two items. For example, if a movie (say m) on Netflix is being rated differently based on the 'day of week' it has been watched (weekday, weekend), then two artificial IDs for this movie should be created (m1, m2 where m1 is the movie being watched in weekday, whereas m2 is the movie being watched in weekend). The importance of this contextual incorporation approach is that it acts as a prefiltering stage, thus enables applying conventional RSs untouched.

Deep learning (DL) has gained momentum in recent few years because of the promising improvement in results it provides over conventional machine learning approaches. However, most DL-based methods in the relevant literature focus on contextual modelling for incorporation [7]. In a previous work [7], we have designed a novel CARS method that hybridizes item-splitting with a traditional DL-based RS known as Neural Collaborative Filtering (NCF) [8]. We have termed the outcome algorithm as Context-Aware-NCF (CA-NCF).

At the time, we have relied on item-splitting as a pre-filtering stage that operates in front of the NCF. We have shown the potential of incorporating context that way. However, item-splitting is only one example of possible pre-filtering approaches that are applicable. In this paper, we have designed a novel method that alternatively hybridize a retrofitted version of a user-splitting approach with NCF. We term the novel method as US-NCF (for User-Splitting Neural Collaborative Filtering).

The model we have designed in this paper is geared toward context-aware social recommender systems. Our model then acts as a front stage for those systems. It basically aims at efficiently improving the contextual incorporation, and thereby assisting in the generation of more accurate personalized predictions in those domains. We have shown that US-NCF outperforms the plain NCF (so as with our previous work CA-NCF) and other latent factor models.

The framework we have designed in this work is general-purpose and can be applied to any latent factor model such as Matrix Factorization (MF) and its variants. It is also applied to social recommender systems [2].

In short, the main contributions of this paper are the following. We provide a general-purpose hybrid model that seamlessly and transparently incorporate a retrofitted version of user-splitting with a recent latent factor model known as NCF. This incorporation achieves plausible context-aware recommendations that can be passed then in a pipeline to a trust-based or OSN recommenders as a front-stage. Alternatively, the novel context-aware latent representations for users, which result from our retrofitted version of user-splitting, can then be passed as a new latent factor (replacing the stock version of latent factors of users) to an OSN or trust-aware recommender. This will assist in generating more accurate personalized context-aware trust and social-related predictions. Also, we have implemented our prototype over BigDL [9] , which is coined over Apache Spark [10]. Therefore, taking advantage of the distributed running of the training models.

The remainder of the paper is organized as follows. We first compare conventional and context-aware RSs. We then discuss the proposed framework including the proposed US-NCF method. In what follows, we showcase and discuss our results. We close the paper by conclusions and future research frontiers.

II. Background and Theoretical Foundations

In this section, we discuss the foundations of conventional and DL-based recommender systems.

A. Conventional Recommender Systems

RSs are systems designed specifically to solve the online information overloading problem by recommending 'items' to 'users', hoping that the former meets the preferences of the latter. Items are entities of an interest to users. For example, a 'YouTube video', a 'Spotify song', a 'flight operator on booking sites'. Users normally rate those items either explicitly in an interaction that is known as explicit feedback (user-item-rating interaction) or implicitly which is known as implicit feedback (user-item interaction) [3]. The distinction between implicit and explicit approaches is that, in the former ratings are obtained indirectly. For example, 'video watching time history', 'items purchased', 'number of clicks on a website link'. On the other hand, explicit feedbacks are obtained through users direct rating or thumbs-up (similarly 'like' buttons) interactions.

Several approaches and algorithms have been proposed for RSs, among which the conventional Collaborative Filtering (CF) and its variations remain competitive. The idea of CF presumes that friendship is a way for feedback similarity in recommendations. In a nutshell, CF-based methods work by recommending items to users based on a presumption that they match their preferences in a way that is similar to their friends [7].

Algorithms for RSs form a sub-domain of ML as they are considered either 'regression' or 'ranking' tasks. Working on a historical explicit feedback problem normally aims at predicting feedbacks (a.k.a. ratings) for unseen items, whereas algorithms receiving an implicit feedback input are considered ranking ML problems as they target recommending items based on user-item interaction history [7].

Conventional RSs are not aware of the contextual information that may be served as additional information with the input data. Various works in the relevant literature have shown that incorporating additional contextual information into recommendations decisions has a utility in improving the overall recommendation performance [3, 7, 11] in all domains, including, most importantly OSNs. In the next subsection we discuss various CARSs.

B. Context-Aware Recommender Systems

Context is an information that is surrounding the feedback decision in implicit and explicit feedback interactions [7]. For example, 'day of week' when a user has watched and ranked a movie where her feedback decision may depend on the so-called contextual condition (i.e., being 'weekend' or 'weekday', the rating may differ).

Works of the related literature have relied on three approaches for incorporating contextual information into recommendation tasks: pre-filtering, post-filtering, and contextual modelling [1]. Pre-filtering approaches are dimension-based methods that basically work by filtering unsuitable rating profiles and afterwards passing the outcome to a conventional RS [7]. For example, in a RS with an explicit feedback input, if a user wants to watch a movie on Netflix in 'weekend' (a contextual condition in the 'day of week' context), then only historical rating profiles with a contextual condition value that corresponds to the user context ('weekend' in this case) are used for recommending a relevant movie. A breed of algorithms, that we refer to as *contextual-splitting approaches*, have emerged as predominant players for contextual pre-filtering. Those include item-splitting [4, 5] user-splitting [6], and UI-splitting [12]. Item-splitting is a simple, yet appealing, approach which is based on the idea that an item that may have a statistically significant difference in the rating depending on the contextual condition can be split into two items (i.e., artificial items). For example, imagine that we have an item 'I' which has rating values 1 and 4 on a scale from 1 to 5 depending on a ‘day of week’ context, such that the rating ‘1’ corresponds to the ‘weekday’ contextual condition, whereas the rating 4 corresponds to ‘weekend’. In this case item-splitting proposes to split ‘I’ into two artificial items ‘I1’ and ‘I2’, such that ‘I1’ is an item that corresponds to the ‘weekday’, whereas ‘I2’ corresponds to ‘weekend’. Having done that, the multidimensional problem plausibly resorts to a traditional two-dimensional problem because the context effect then is injected with the items. Thus, considering both aspects (dimensionality reduction and context-awareness) without their limitations. User-splitting is similar but is based on splitting users instead of items if same users show statistically significant feedback differences depending on various contextual conditions. UI-splitting relies on splitting based on the users and items altogether.

On the contrary, post-filtering works exactly the opposite way. It first applies the conventional recommendation model untouched with no awareness to the context fields. This results in a primary list of ratings, which will then be subjected to a post-filter that rules out records which do not correspond to the contextual target, or at least the list will be reranked to favor items that are more relevant to the target contextual condition. Contextual modelling approaches incorporates contexts as indispensable parts of the prediction function.

We focus in this paper on the pre-filtering approaches for several reasons. They are simple appealing approaches that enable applying conventional two-dimensional recommendation models untouched, thus utilizing a rich library from the relevant literature. Also, pre-filtering outperforms post-filtering when it comes to the operational costs [7].

Recently, the attention of the research community has been shifted toward applying context-awareness in trust-aware and OSNs [2]. Context-aware trust systems are based on the principle that trust is context-dependent where users trust recommendations from other users only in specific contexts [2]. For example, a trainee who is trying to lose weight in a diet center may trust the trainer about YouTube videos to watch for additional home exercises that may boost the weight loss. However, she may not trust the trainer when recommending other non-exercise related videos to watch during weekends. Contextual information is obviously essential for trust prediction in OSNs. However, most conventional OSN recommender systems are either designed without context-awareness or apply contextual modelling for incorporating contextual information. Having done that, they assume that user, items, and context features fall within the same latent space, which in reality is inconvenient and difficult to interpret.

III. RLETAED LITERATURE

RSs are traditionally context-free as they were unaware of additional contextual information that may potentially surround feedback decisions. Collaborative Filtering (CF) yet remains the most widely adopted traditional RS [7]. It basically depends on learning to predict active users’ ratings based on historical rating by similar users. Matrix Factorization (MF) extends CF by projecting items and users’ profiles to a shared space known as ‘latent factor space’.

It is well established among practitioners that context should be considered as a first-class citizen while generating recommendation lists. This is so because it has been proven that considering contextual information yields more accurate personalized recommendations [7]. This has caused the emergence of a group of algorithms that becomes to be known collectively as Context-Aware Recommender Systems (CARS for short). CARSs have redefine the classical user-item-rating interaction into a context-aware user-item-context-rating counterpart.

The adoption of Deep Learning (DL) has gained momentum in the last decade or so. This is attributed to the fact that it has shown promising performance that is mostly superior to traditional ML counterparts. However, the share of DL-based recommenders remains humble if compared with its applications in image and voice recognition [7].

One of the most widely accepted DL-based RSs is a method known as Neural Collaborative Filtering (abbreviated NCF) [8], which as a fine combination that hybridizes the benefits of CF and Neural Networks (NN), specifically Multi-Layer Perceptron (MLP), without their limitations (i.e., exploiting CF linearity and NN robustness). NCF is a robust DL-based RS algorithm which has outperformed several traditional counterparts. The downside however is that it is unaware of contextual features, thus losing a potential optimization that may otherwise improve the overall personalized recommendations.

To circumvent the problems of context-unawareness in the traditional NCF, we have, in a previous work [7], designed a context-aware counterpart that hybridized the item-splitting method as a prefiltering stage in front of the plain NCF. We have termed the novel algorithm as CA-NCF. However, item-splitting is only one pre-filtering approach. As such, we have decided to design a novel algorithm that basically incorporates a retrofitted version of user-splitting with the stock version of NCF, which is to be elaborated in the next section. We dub the new version as US-NCF.

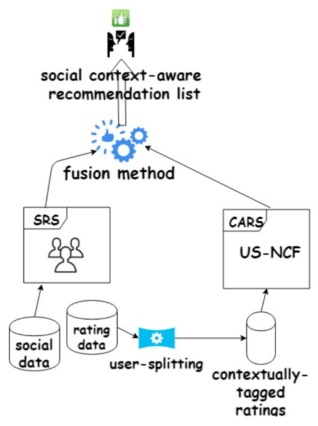
social context-aware recommendation list
fusion method
SRS
CARS
US-NCF
social data
rating data
user-splitting
contextually-tagged ratings

Fig. 1. Context-aware social recommender system overview

IV. User-Splitting Context-Aware Recommender System for Online Social Networks

We propose a relevant context incorporation technique for social context-aware and trust-aware recommender systems. It worth mentioning that in this paper we are not proposing a novel social context-aware method. Instead, we scope ourselves to proposing relevant cost-effective context pre-filtering-based context incorporation techniques that can benefit context-aware SRSs.

We are hybridizing a user-splitting method [6] with a deep-learning based CF recommender. Our context-aware recommender system can be used for supporting the operation of context-aware SRS as depicted in Fig. 1.

Raw explicit feedback data is coming from rating sources, which is then served to a retrofitted version of a user-splitting method. Our retrofitted version of user-splitting is analogous to a retrofitted version of item-splitting that we have designed in our previous work [7]. We first apply a context feature selection method that selects relevant contexts which has a significant statistical effect on the rating decisions. This method relies on the Analysis of Variance (ANOVA) test and Tukey HSD test as described in our previous work. Interested readers are referred to our previous work for further details on the method [7]. The output of the method is a list of relevant contextual features and relevant permutations of contextual conditions that can be used for user-splitting [6] which proceeds as follows. For every unique user in the ratings space, we divide ratings to two rating sub-lists depending on contextual pairs permutations. Thereafter, we apply a t-test. This procedure results in several t-test values, one for each possible contextual pair combination. If the maximum t-test among those is greater than a threshold (e.g., greater than 8), then we split the user into two users, one for each contextual condition in the contextual condition pair combination. For example, in a trip recommender system (e.g., TripAdvisor), the feature selection may select ‘trip type’ and ‘weather’ as the two significant contexts that affect rating decision. Thereafter, Tukey HSD suggests that relevant contextual conditions pairs are (‘business’, ‘vacation’) for ‘trip type’ and (‘warm’, ‘cold’) for ‘month of year’, respectively. This result is then passed to the stock version of user-splitting. Say that the t-test then suggests that users U1 and U2 can be split based on the following permutations, (‘business’, ‘vacation’) for U1 and (‘cold’, ‘warm’) for U2. Then U1 will be split into two artificial users (U11 and U12), where U11 is U1 when she rated the item (i.e., location she visited, say L1) when she was there for a ‘business’ trip, while U12 is U1 when she rated the same location differently while she was there for ‘vacation’. Similarly, U2 will be split into two users (U21 and U22) corresponding to ‘cold’ and ‘warm’ context conditions, respectively.

This is a dimensionality reduction context incorporation retrofitted pre-filtering approach that incorporates contextual factors with users’ latent factors seamlessly. The result is a context-aware users’ latent representation. Therefore, the output data is a new rating matrix on the user-item-rating form, but with the context conditions injected within the user latent representation. We then pass the intermediate result to the plain NCF. However, since we have a context-aware latent representation for users instead of the latent user vector, we redefine the plain prediction model of NCF as in (1).

$$\hat{r}_{uic} = f\left(P_{art}^{T} V_{uc}^{UC}, Q^{T} V_{i}^{I} \,\middle|\, P_{art}, Q, \Theta_f\right) \qquad (1)$$

The plain user latent vector V_u^U in NCF is substituted by a context-aware latent representation for users V_{uc}^{UC}, where UC represents artificial users based on the context conditions. $\hat{r}_{uic}$ is the predicted explicit feedback of user u on item i during a contextual condition c. Θ is a set of model parameters that we seek estimating. Q and P_{art} are latent factor matrices for items and artificial users, respectively. Θ_f is a set of model parameters for the interaction function f,

Context-aware recommendation list can then be fused with recommendations resulted from an SRS using a linear weighting method such as the one in (2):

$$\hat{r}_{uics} = p\,(v_i = r_c)\,.\,\hat{r}_{uic} + (1 - p\,(v_i = r_c)\,)\,.\,\hat{r}_{uis} \qquad (2)$$

Where $\hat{r}_{uics}$ is the predicted rating for item i in context condition c by user u within her social network circle s. $\hat{r}_{uic}$ is the rating for the same item by the same user in that context condition without considering the social ties. On the contrary, $\hat{r}_{uis}$ is the context-free predicted rating considering only the social ties. $p\,(v_i = r_c)$ is the probability that user u has rated item i with rating r_c in context condition c. It is calculated by dividing the total number of times user’s friends rated item i in context condition c over total number of times user’s friends rated item i in all context conditions. This mechanism is equivalent to devaluing commonly liked items (inter-context) as they do not serve enough information about user’s preferences in specific context factors.

By being able to appropriately incorporate contexts into the recommender model, more accurate results can be achieved in context-aware SRSs. This is so because, users in SNs trust recommendations only in certain contexts [2]. This is reciprocally true as context-aware connection recommendations outperform context-free counterparts.

The intuition that allows such a hybridization is that users strongly connected in social networks normally have similar feedbacks for items [13]. Another possible hybridization for a shared training from both domains (i.e., websites recommenders and OSNs) is to learn the user embeddings for the users of the OSN who are not having item ratings in the website recommendations. This can be accomplished by propagating the embeddings of users who have ratings (in a website RS) to social users who do not have ratings. This enables learning representations for social users from the same latent space which represents users who have ratings.

V. Evaluation

In this section, we summarize experimental settings, including the datasets, baseline methods and evaluation metrics.

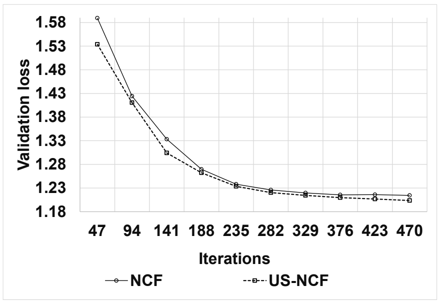

Fig. 2. Validation loss of US-NCF Vs. NCF against 'number of iterations' on MovieLens 1M dataset.

A. Experimental setup

Datasets. We rely on two explicit feedback rating datasets for benchmarking as follows. Movie rating dataset Movielens 1M [14], and trip planning website TripAdvisor [15] . Movielens dataset has roughly one million ratings done by 6040 users for circa 3706 items (i.e., movies). For TripAdvisor dataset has 14175 ratings, 2731 users and 2269 items ('hotels' in this case). A *timestamp* in MovieLens is an implicit source of contextual information [1]. From every timestamp, we have got the following context factors: 'day of week', 'month', 'year'. 'trip type' is the only appropriate context feature in TripAdvisor data.

Baselines. We have compared our method (US-NCF) with the following baselines:

- The stock version of NCF [4] , s an explicit feedback version, which is considered as state-of-art context-free DL-based model for RSs.
- Biased MF [28] , a conventional context-free benchmark.
- Context-Aware NCF (CA-NCF) [29], which is a state-of-art method for incorporating contextual-information into DL-based recommender model known as NCF. It utilizes item-splitting for incorporating context.

Evaluation methods. To We divide ratings data into 80% training and 20% testing, then we apply a train-test evaluation Sparse-categorical-cross-entropy has been used as a loss function. Most common CF tasks for modelling user preferences include ranking and ratings prediction. To measure the performance of rating prediction, we apply error metrics that include average Mean Absolute Error (MAE) and validation loss. For ranking (i.e., top-N), we adopt an accuracy measure known as precision-in-top-N [26]. We specifically have adopted 'top-one-accuracy' (a.k.a. P@1) [1], measuring the counts when top element in the ranking list matches the target. P@1 was selected because users normally check highest ranked items [26]. SoftMax was the activation function used in the output layer, whereas ReLU was used for hidden layers.

B. Experimental Results

Hyper-Parameter configurations. We apply Adam optimizer. Batch sizes of 128 and 256 have been used with TripAdvisor dataset, whereas 16k was used for MovieLens 1M dataset. Tested learning rates range from 0.01 to 0.0001 and 0.05 to 0.0005. An impurity criterion [23] was applied to split users. T_{mean} was applied for measuring the statistical significance between means of pairs of rating lists, such that every list contains ratings in a specific context factor.

Fig. 2 illustrates the validation loss we obtain, averaged from 100 running sessions. On average, we got roughly 1.2 % loss gain because of applying US-NCF instead of the stock version NCF.

Fig. 3 depicts that our model US-NCF significantly surpasses several baselines. On average, a gain that equals 1.8 % was obtained compared to plain NCF, slightly better that that obtained when applying the state of art CA-NCF. A higher gain is obtained when comparing US-NCF with conventional context-free model, specifically BiasedMF, where we obtain, on average, a gain that equals roughly 13.3 %.
Results that have obtained support that context-aware recommenders yield more accurate rating prediction results. For testing the other conventional CF task (i.e., ranking), we apply the top1Accuracy measure. We have obtained the results shown in Fig. 4. It is again obvious that our novel method US-NCF compares favorably to the baselines. We roughly obtain 4 % and 90 % when comparing US-NCF to the context-free plain NCF and BiasedMF, respectively. This signifies the importance of incorporating contexts in RSs. Also, it suggests that context-free deep-learning based RSs perform better than traditional counterparts. The novel method US-NCF performs similarly when comparing it with the item-based state-of-art context-aware RS (CA-NCF).
It worth mentioning that despite the novel method US-NCF performs similar to the state-of-art context-aware RS (CA-NCF) when it comes to skills against the two tasks (prediction and ranking), US-NCF is favorable over CA-NCF for social

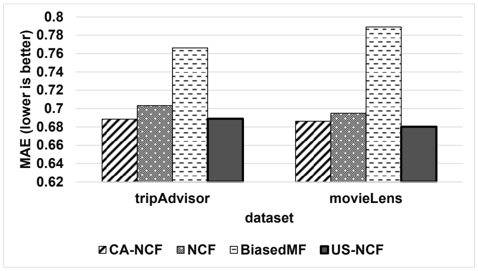

Fig. 3 MAE of US-NCF Vs. baselines for all datasets

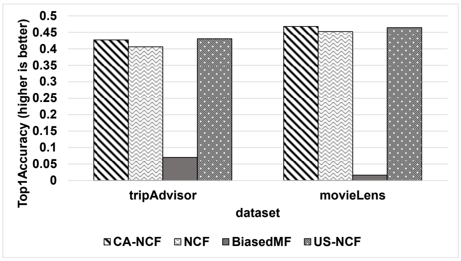

Fig. 4 Top1Accuracy US-NCF against baselines on all datasets

recommender systems. This is so because US-NCF is designed to model user's contexts, whereas CA-NCF was designed to model item's context. For SRSs, it is the relationships between the users that is the center of the analysis, not between items. Having said that, being able to seamlessly generate context-aware user latent factors is more significant than that of generating context-aware item latent factors (which was the case with the state-of-art DL method CA-NCF). Also, being able to do so reassures that the embedding learning of like-mind social users is guided by the embedding of users coming out from the CARS. The distinction is similar to the following conceptualization. The state-of-art method CA-NCF incorporates contexts with items of the plain NCF, thus recovering an item-based NCF, whereas the novel method US-NCF incorporates context into users, thus recovering a user-based version of NCF.

VI. Conclusions and Future Works

In an online world that is overwhelmed by information overloading, people normally trust a 'word-of-mouth' recommendation from strong social ties on OSNs. However, studies have shown that people trust recommendations from their social ties only in specific contexts. Because OSNs normally model user-user interactions, they do not serve user-item interactions. The latter is rather collectable from merchandising websites that seek to market items to users (such as movies websites, Netflix for example). Such information is highly valuable for enriching social recommender systems.

Only few works from the relevant literature have focused on bringing context-aware and social recommender systems together. More than often, limited by their inability to model contextual information interactions appropriately.

Of a special interest is projecting same users that has accounts in both domains (social sites and e-commerce sites) to the same latent space. Most importantly, being able to model contextual features appropriately before merging the two domains. To close this void, in this paper we have designed a purpose-built deep learning-based context-aware recommender system (US-NCF) that sufficiently solve the previously mentioned problems.

Our model basically can incorporate additional contextual features into user's representations so that any social recommender system will be able to benefit of such incorporation. This is simply achieved by propagating the users from both domains into the same latent space, thus appropriately modelling the user-item-context-social interaction.

We restricted ourselves in this paper to generating context incorporation techniques and recommendations. In a future work, we will further investigate the effectiveness of integration with OSNs data by, for example, applying a graph semi-supervised learning method. Also, to our knowledge, the literature does not have a benchmark data for the two domains altogether (OSNs and e-commerce sites) such that there are users who have accounts in both domains. To close this gap, we aim in the future to generate relevant data for such cross-domain recommender systems.

Acknowledgment

This research was supported by the IDEHA project funded by PON "RICERCA E INNOVAZIONE" 2014-2020 (no. J46C18000440008).

References

[1] F. Ricci, L. Rokach and B. Shapira, "Introduction to recommender systems handbook," in *Recommender Systems Handbook*Anonymous Springer, 2011, pp. 1-35.

[2] S. M. Ghafari, A. Joshi, A. Beheshti, C. Paris, S. Yakhchi and M. Orgun, "DCAT: A deep context-aware trust prediction approach for online social networks," in *Proceedings of the 17th International Conference on Advances in Mobile Computing & Multimedia,* 2019, pp. 20-27.

[3] Y. Zheng and B. Mobasher, "Context-aware recommendations," in *Collaborative Recommendations: Algorithms, Practical Challenges and Applications,* 2019, pp. 173-202.

[4] L. Baltrunas and F. Ricci, "Context-based splitting of item ratings in collaborative filtering," in *Proceedings of the Third ACM Conference on Recommender Systems,* 2009, pp. 245-248.

[5] L. Baltrunas and F. Ricci, "Experimental evaluation of context-dependent collaborative filtering using item splitting," *User Modeling and User-Adapted Interaction,* vol. 24, *(1-2),* pp. 7-34, 2014.

[6] L. Baltrunas and X. Amatriain, "Towards time-dependant recommendation based on implicit feedback," in *Workshop on Context-Aware Recommender Systems (CARS'09),* 2009, pp. 25-30.

[7] I. M. Al Jawarneh, P. Bellavista, A. Corradi, L. Foschini, R. Montanari, J. Berrocal and J. M. Murillo, "A Pre-Filtering Approach for Incorporating Contextual Information Into Deep Learning Based Recommender Systems," *IEEE Access,* vol. 8, pp. 40485-40498, 2020.

[8] X. He, L. Liao, H. Zhang, L. Nie, X. Hu and T. Chua, "Neural collaborative filtering," in *Proceedings of the 26th International Conference on World Wide Web,* 2017, pp. 173-182.

[9] J. Dai, Y. Wang, X. Qiu, D. Ding, Y. Zhang, Y. Wang, X. Jia, C. Zhang, Y. Wan and Z. Li, "BigDL: A distributed deep learning framework for big data," *arXiv Preprint arXiv:1804.05839,* 2018.

[10] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker and I. Stoica, "Spark: Cluster computing with working sets." *HotCloud,* vol. 10, *(10-10),* pp. 95, 2010.

[11] Y. Zheng, B. Mobasher and R. Burke, "Incorporating context correlation into context-aware matrix factorization," in *Proceedings of the 2015 International Conference on Constraints and Preferences for Configuration and Recommendation and Intelligent Techniques for Web Personalization-Volume 1440,* 2015, pp. 21-27.

[12] Y. Zheng, R. Burke and B. Mobasher, "Splitting approaches for context-aware recommendation: An empirical study," in *Proceedings of the 29th Annual ACM Symposium on Applied Computing,* 2014, pp. 274-279.

[13] X. Wang, X. He, L. Nie and T. Chua, "Item silk road: Recommending items from information domains to social users," in *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval,* 2017, pp. 185-194.

[14] F. M. Harper and J. A. Konstan, "The movielens datasets: History and context," *Acm Transactions on Interactive Intelligent Systems (Tiis),* vol. 5, *(4),* pp. 19, 2016.

[15] Y. Zheng, B. Mobasher and R. Burke, "Context recommendation using multi-label classification," in *Proceedings of the 2014 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)-Volume 02,* 2014, pp. 288-295.