# Reshaping Graph Recommendation with Edge Graph Collaborative Filtering and Customer Reviews

Vito Walter Anelli[1], Yashar Deldjoo[1], Tommaso Di Noia[1], Eugenio Di Sciascio[1], Antonio Ferrara[1], Daniele Malitesta[1,*] and Claudio Pomo[1,*]

[1]Politecnico di Bari, via Orabona, 4, 70126 Bari, Italy

Abstract

Graph collaborative filtering approaches learn refined users' and items' node representations by iteratively aggregating the informative content (called messages) coming from neighbor nodes into each ego node. Unfortunately, not all interactions (i.e., graph edges) may be equally important to the users and items involved. As this indiscriminate message aggregation leads to multi-hop representation errors, recent strategies have used attention mechanisms to weight neighbors' importance to the ego node. Despite their success, such solutions seem to disregard the potentially critical impact users' reviews may play on this weighting process. Reviews convey the multi-faceted user's opinion about items and provide a fundamental tool to group like-minded customers. In this work, we first formally show the causes of node error representation in graph collaborative filtering and demonstrate how existing neighborhood weighting procedures (e.g., attention mechanisms) may alleviate the issue at the expense of limited hop exploration. Second, we correct the representation error through an additional graph network where we enrich graph edge embeddings through opinion-aware review embeddings to smooth each neighbor node's importance on its ego node. We call our solution Edge Graph Collaborative Filtering (EGCF). Extensive experiments on three e-commerce datasets show that EGCF competes successfully with traditional, graph- and review-based approaches on accuracy and beyond-accuracy objectives, while a study on the number of explored hops justifies the adopted configuration for EGCF. Code and datasets are available at: https://github.com/sisinflab/Edge-Graph-Collaborative-Filtering.

**Keywords**

Collaborative Filtering, Recommendation, Graph Convolutional Networks, Reviews

## 1. Introduction

Recommender systems constitute the backbone of several online platforms (e.g., Amazon), offering consumers lists of products that might meet their needs and tastes. Recommendation algorithms are traditionally designed and trained to find preference patterns in user-item recorded interactions. Optionally, this learning process may be enriched through additional informative data constantly updated on those platforms, which may captivate customer's attention towards items' characteristics (e.g., product images) or provide a tool to share opinions about purchased items to guide other customers during their decision-making process (e.g., reviews).

Collaborative filtering (CF) [1], one of the most prominent recommendation paradigms in recent years, promotes the intuition of similar users interacting with similar items. CF-based models usually map users and items to embeddings in the latent space, and learn to predict user interactions by optimizing an objective function that combines these embeddings linearly (e.g., inner product [2]) or non-linearly (e.g., neural networks [3] and probabilistic models [4]). While focusing on improving the user-item prediction step, such techniques have long underestimated the importance of deriving informative features to describe users and items suitably.


**Figure 1:** A subset of users, items, and reviews users wrote about items, along with the expressed ratings (in the range 1-5). Despite being connected to the same items, users $u_1$-$u_2$, and users $u_1$-$u_3$ do not share similar opinions about the interacted items.

Recently, graph convolutional networks (GCNs) [5] have taken over CF-based recommendation, thanks to their capability of mining user-item high-order relationships. Unlike prior techniques, these models explicitly incorporate user and item relationships into their embedding representations. Concretely, the embedding of each node (defined as *ego* node) is refined by aggregating its

*DL4SR'22: Workshop on Deep Learning for Search and Recommendation, co-located with the 31st ACM International Conference on Information and Knowledge Management (CIKM), October 17-21, 2022, Atlanta, USA*

*Authors are listed in alphabetical order. Corresponding authors: Daniele Malitesta and Claudio Pomo.

daniele.malitesta@poliba.it (D. Malitesta);

claudio.pomo@poliba.it (C. Pomo)

neighbors' node embeddings (i.e., whose contribution is called *messages*). This step is repeated iteratively to propagate the **collaborative signal** over multiple hops. These models are becoming the de facto standard in personalized recommendation, reaching remarkable recommendation performance as in the pioneer works presented in [6, 7], and more recently, in the solutions [8, 9, 10].

The message-passing pattern, by design, may still present some limitations despite being successful. An argument could be made that not all user-item interactions (i.e., graph edges) have the same relative importance. To clarify this, consider the motivating scenario in Figure 1, where we depict a subset of users and items from a real-world e-commerce platform (i.e., the Amazon catalog) and enrich their interactions with ratings and reviews. Both user $u_1$ and $u_2$ interacted with item $i_1$, thus inferring that they might share similar interests and preferences. However, careful analysis of the corresponding reviews reveals that their **opinions** about item $i_1$ are opposite (the expressed ratings are 5 and 2, respectively). Following a similar reasoning schema, users $u_1$ and $u_3$ have both interacted with item $i_2$ but their **comments**, while being generally similar (the item is rated 3 and 5, respectively), show slight shades of disagreement (i.e., $u_1$ is not completely satisfied with the belt size). As the message-passing pattern works by indiscriminately aggregating the neighbor nodes **at multiple hops**, the node representation of $u_1$ is ultimately influenced by the representations of both $u_2$ and $u_3$ after two propagation hops. In the long term, such behavior may lead to what we could define as a **node representation error**.

Weighting the importance of neighborhood while aggregating the incoming messages into the ego node is among the prominent solutions to the abovementioned issue. Following the same direction path in [11], other popular and recent works in recommendation such as [12, 13, 14, 15] leverage attention mechanisms (i.e., a neural network) to perform the weighting procedure. Even if these models have widely demonstrated to provide superior accuracy recommendation performance, they are still affected by **oversmoothing**, the phenomenon according to which node embedded representations tend to get closer and closer in the latent space after multiple propagation hops, thus flattening the existing differences in the neighborhood [16, 17]. For this reason, attention-based approaches usually propagate messages for only one or two hops, but this does not help access wider portions of the user-item graph.

In this respect, we believe attention-based techniques generally disregard other potential sources of information (e.g., users' generated reviews) whose contribution may positively impact the neighborhood weighting process. Opinions and comments about interacted items constitute the basis on which like-minded users gather on online platforms, as they promote the discovery of novel and diverse items from the catalog. In this work, we first formally define the problem of nodes' representation error in graph collaborative filtering. After that, we show how existing weighting techniques (such as attention mechanisms) may alleviate the described issue at the expense of limiting the hop exploration depth to reduce the effect of oversmoothing. Thus, to address such drawback, we propose a lighter-weighting procedure that exploits the informative content extracted from reviews (i.e., opinions and comments about interacted items) to enhance graph edge representation. Such edge-enriched features are eventually used to derive the similarity between the ego node and its neighbors, which we re-interpret as the importance of the neighbor node on the ego node. Our proposed weighting procedure is applied to a GCN acting as the correction to another traditional (but error-affected) GCN. We call our solution **Edge Graph Collaborative Filtering (EGCF)**.

After formalizing the theoretical basis for EGCF and its rationale, we assess its efficacy on three popular product categories from the Amazon catalog [18]. Given their similar intuitions and rationale to EGCF, we compare the method with four families of CF-based recommendation, i.e., traditional, review-based [19, 20], and graph-based approaches (both leveraging attention mechanisms and not). We seek to answer these research questions about our proposed approach:

- **RQ1.** Can the correction to the node error representation help EGCF produce more accurate recommendations than state-of-the-art baselines?
- **RQ2.** Considering the high impact that novel and diverse recommendation lists may have on both users and companies, how effective is EGCF when evaluated on beyond-accuracy metrics, given its strategy for neighborhood exploration?
- **RQ3.** What is the effect of changing the hop exploration number on recommendation performance, and how can we justify such behaviors for the adopted architecture?

The extensive experimental evaluation shows that the correction to the node representation error and the possibility of propagating messages across multiple hops permits EGCF to outperform state-of-the-art baselines on accuracy and beyond-accuracy metrics. Finally, the study on the hop propagation number proves the soundness of our proposed architectural configuration while shading interesting direction paths for future work.

## 2. Related Work

**Graph-based recommendation.** The approach proposed in [21] is the first attempt to address the recommendation task through a graph-based architecture. The

authors implement a graph autoencoder that labels its edges with users' ratings to perform link prediction. Ying et al. [6] design a graph convolutional network for a web-scale recommendation to produce high-quality image recommendations for the Pinterest platform, efficiently exploiting random walk and item's multimodal side information. Wang et al. [7] present neural graph collaborative filtering (NGCF), whose propagation layer aggregates the messages from the neighborhood considering the similarity between each neighbor node and its ego node. While providing higher performance to previous state-of-the-art solutions, NGCF (and GCN more generally) show limitations later addressed by He et al. [8]. Their idea is to lighten GCN's traditional layer structure and reach superior accuracy performance by removing non-linearities and node embedding transformation in the propagation layer (LightGCN). The latest approaches try to take a step further to the LightGCN strategy by allowing theoretically unlimited propagation layers [9] and revisiting the concept of graph convolution for recommendation and node embedding smoothness under the lens of graph signal processing [10].

While aggregating messages from neighbor nodes into the ego node, not all received contributions have the same importance. The pioneering work by Velickovic et al. [11], called graph attention network (GAT), takes advantage of attention mechanisms to weight the different influences of neighbor nodes on the ego node. Inspired by this rationale, several recent works in recommendation seek to assess the relative importance of interacted items on users involved in those interactions. In the last few years, recommendation tasks such as session-based recommendation [22, 23, 12] and sequential recommendation [13, 24] have been widely addressed by using attention mechanisms on graphs. Attention mechanisms may also be beneficial when the informative content conveyed by the bipartite user-item graph is enhanced by additional side information, like knowledge graphs [25], heterogeneous information networks [14], or multimodal items' content [26]. Exploiting attention to disentangle the aspects underlying node interactions may represent a fundamental step toward explainability [27]. Following this direction, the work by Wang et al. [15] named disentangled graph collaborative filtering (DGCF), and the method presented in Wu et al. [28], propose to disentangle user-item connections into possible user intents.

State-of-the-art attention-based approaches provide an efficient neighborhood weighting strategy. However, their multi-hop exploration is usually limited to prevent nodes in the neighborhood from getting too much similar in the latent space (see Section 3.2). Conversely, EGCF leverages additional information (i.e., reviews) whose extracted opinion-aware features do not flatten differences among nodes while easing the weighting process. Moreover, in contrast to prior works, EGCF enriches edges by representing them through the extracted embeddings.

**Review-based recommendation.** Reviews convey a rich source of information to access users' multi-faceted opinions about interacted items. For this reason, several existing works propose to extract valuable knowledge from them to produce better-tailored recommendations [19, 20]. Among the pioneer works, Wang et al. [29] adopt a stacked denoising autoencoder to approximate the user-item rating matrix starting from textual reviews, Almahairi et al. [30] introduce two neural network-based approaches built upon bag-of-words and recurrent neural networks, and Kim et al. [31] present convolutional matrix factorization (ConvMF), where a convolutional neural network is merged with probabilistic matrix factorization to learn the context of review documents.

Reviews are textual documents composed of words, which may further be grouped into sentences. To exploit such hierarchical structure, Zheng et al. [32] design a convolutional neural network on top of a factorization machine prediction model to extract from review's words a unique embedded representation for users and items. The adoption of attention mechanisms may help refine each review component's importance on the recommendation profile of users and items. In this respect, Liu et al. [33] improve the previous approach by weighting the importance of convolutionally-embedded reviews for both users and items for the sake of explanation. Similarly, Lu et al. [34] learn users' and items' attention features by exploring different review components such as words, sentences, and topics via a GRU-based network, while Liu et al. [33] (based upon the solution described in [35]) augment users' and items' collaborative latent factors through features extracted from their generated ratings and reviews. Wang et al. [36] leverage common review properties (e.g., how helpful the reviews were for other users) to assess its importance on users and items.

Only recently, very few works have injected the informative content of reviews into graph-based networks for recommendation. Wu et al. [37] propose a model named reviews meet graphs (RMG), a multi-view framework that learns users' and items' representation by considering the word- and sentence-level of reviews and exploring two hops of the user-item graphs to access also user-user and item-item relations. Gao et al. [38] present a three-structured architecture that catches the short- and long-term user preferences and item features, along with the collaborative information encoded in the bipartite user-item graph. Shi et al. [39] introduce a dual GCN model, where one extracts and propagates review aspects, and the other reuses the aspect for the graph.

Despite addressing recommendation through different strategies, the presented algorithms generally work by grouping reviews on both users and items profiles but, in fact, limiting the exploration of users and items neighbors at one hop (i.e., the nearest neighborhood).

Conversely, our proposed approach exploits reviews as edge side information to describe user-item interactions and propagate their informative content at multiple hops to overcome theoretical issues in the way graph-based recommender systems are usually designed (see later).

# 3. Methodology

The section presents and motivates our proposed method, Edge Graph Collaborative Filtering (EGCF). We first introduce some notation and preliminaries to graph models for collaborative filtering. Then, we highlight a potentially critical issue in the message-passing schema. Even if weighting the importance of each neighbor node may alleviate the problem, we discuss the insights and propose an enhanced application of the importance weighting.

## 3.1. Notation and preliminaries

Let $\mathcal{U} = \{u_1, u_2, \dots, u_N\}$ and $\mathcal{I} = \{i_1, i_2, \dots, i_M\}$ be the sets of $N$ users and $M$ items in the system, respectively. Then, let us consider a bipartite and undirected user-item graph that connects pairs of nodes when there exists a recorded interaction among them. User and item nodes are represented through embeddings in the latent space, i.e., $\mathbf{e}_u \in \mathbb{R}^d, \forall u \in \mathcal{U}$ and $\mathbf{e}_i \in \mathbb{R}^d, \forall i \in \mathcal{I}$.

Inspired by popular approaches [5], current graph-based recommender systems refine users' and items' node embeddings by exploring their multi-hop interconnections represented in the graph. Let $u$ and $i$ be the nodes for a user and an item to be updated (i.e., the ego nodes), and let $\mathcal{N}(u)$ and $\mathcal{N}(i)$ be the sets of nodes at one hop from $u$ and $i$, respectively (i.e., their neighborhood). The ego node embeddings $\mathbf{e}_u$ and $\mathbf{e}_i$ are updated by aggregating their neighborhoods (i.e., messages):

$$\begin{aligned} \mathbf{e}_u^{(1)} &= \omega\left(\{\mathbf{e}_i, \forall i \in \mathcal{N}(u)\}\right) \\ \mathbf{e}_i^{(1)} &= \omega\left(\{\mathbf{e}_u, \forall u \in \mathcal{N}(i)\}\right) \end{aligned} \tag{1}$$

where $\mathbf{e}_u^{(1)}$ and $\mathbf{e}_i^{(1)}$ are the refined embedding versions of user $u$ and item $i$ after one hop, while $\omega(\cdot)$ indicates the aggregation function. This message-passing pattern may be iterated $L$ times, thus exploring wider and wider neighborhoods of the ego nodes. After two hops, the refined embeddings of user $u$ and item $i$ are:

$$\begin{aligned} \mathbf{e}_u^{(2)} &= \omega\left(\left\{\mathbf{e}_i^{(1)}, \forall i \in \mathcal{N}(u)\right\}\right) \\ \mathbf{e}_i^{(2)} &= \omega\left(\left\{\mathbf{e}_u^{(1)}, \forall u \in \mathcal{N}(i)\right\}\right) \end{aligned} \tag{2}$$

## 3.2. A limitation in the message-passing

The user formulation in Equation (2) can be expanded through Equation (1):

$$\mathbf{e}_u^{(2)} = \omega\left(\left\{\omega\left(\left\{\mathbf{e}_{u'}, \forall u' \in \mathcal{N}(i) \setminus \{u\}\right\}\right), \forall i \in \mathcal{N}(u)\right\}\right) \tag{3}$$

What emerges is that, by propagating messages at two hops, the node embedding of user $u$ is eventually refined through the contributions from other users who interacted with the same items as $u$. In other words, after two hops, **each user profile is influenced by the profiles of other users who rated the same items.**

Indeed, this assumption is aligned with the rationale behind collaborative filtering, i.e., similar users are likely to interact with the same items. However, not all user-item interactions (i.e., graph edges) may be equally important to the users and items involved. Thus, indiscriminately aggregating neighbor node embeddings into the ego node could, after multiple hops, harm the node updating process by bringing all contributions from the neighborhood, even the noisy ones. We interpret this as a **node representation error**, propagating with the exploration hops in the graph.

For this reason, contributions coming from each neighbor node are usually weighted before aggregating them into the ego nodes, modifying the presented formula:

$$\begin{aligned} \mathbf{e}_u^{(2)} = \omega\Big(\alpha_{i\to u}^{(2)}\Big\{\omega\Big(\Big\{\alpha_{u'\to i}^{(1)}\mathbf{e}_{u'}, \\ \forall u' \in \mathcal{N}(i) \setminus \{u\}\Big\}\Big), \forall i \in \mathcal{N}(u)\Big\}\Big) \end{aligned} \tag{4}$$

where $\alpha_{j\to k}^{(l)}$ stands for the importance that the neighbor node $j$ has on the ego node $k$ after $l$ hops. These weights are generally calculated by means of attention mechanisms, and depend on the embeddings of the neighbor and the ego nodes they refer to, e.g., $\alpha_{j\to k}^{(l)} = \varphi\Big(\mathbf{e}_j^{(l-1)}, \mathbf{e}_k^{(l-1)}\Big)$, where $\varphi(\cdot,\cdot)$ is a neural network:

$$\begin{aligned} \mathbf{e}_u^{(2)} = \omega\Big(\overbrace{\varphi\Big(\mathbf{e}_i^{(1)}, \mathbf{e}_u^{(1)}\Big)}^{(\square)}\Big\{\omega\Big(\Big\{\overbrace{\varphi(\mathbf{e}_{u'}, \mathbf{e}_i)}^{(\triangle)}\mathbf{e}_{u'}, \\ \forall u' \in \mathcal{N}(i) \setminus \{u\}\Big\}\Big), \forall i \in \mathcal{N}(u)\Big\}\Big) \end{aligned} \tag{5}$$

that is, $\mathbf{e}_u^{(2)}$ depends on $(\square)$ the importance each neighbor item node $i$ has on the ego user node $u$ after one hop, and $(\triangle)$ the importance all users interacting with the same items as $u$ have on their items. Note that $(\square)$ may be further expanded:

$$\begin{aligned} \varphi\Big(\mathbf{e}_i^{(1)}, \mathbf{e}_u^{(1)}\Big) &= \varphi\Big(\omega\Big(\Big\{\alpha_{u'\to i}^{(1)}\mathbf{e}_{u'}, \forall u' \in \mathcal{N}(i) \setminus \{u\}\Big\}\Big), \\ &\qquad \omega\Big(\Big\{\alpha_{i'\to u}^{(1)}\mathbf{e}_{i'}, \forall i' \in \mathcal{N}(u) \setminus \{i\}\Big\}\Big)\Big) \\ &= \varphi\Big(\omega\Big(\Big\{\varphi(\mathbf{e}_{u'}, \mathbf{e}_i)\mathbf{e}_{u'}, \forall u' \in \mathcal{N}(i) \setminus \{u\}\Big\}\Big), \\ &\qquad \omega\Big(\Big\{\varphi(\mathbf{e}_{i'}, \mathbf{e}_u)\mathbf{e}_{i'}, \forall i' \in \mathcal{N}(u) \setminus \{i\}\Big\}\Big)\Big) \end{aligned} \tag{6}$$

When merging Equation (5) and Equation (6):

$$\mathbf{e}_u^{(2)} = \omega\Big(\overbrace{\varphi\Big(\mathbf{e}_i^{(1)}, \mathbf{e}_u^{(1)}\Big)}^{(\triangle)} \Big\{\omega\Big(\Big\{\overbrace{\varphi(\mathbf{e}_{u'}, \mathbf{e}_i)}^{(\square)}\mathbf{e}_{u'}, \quad \forall u' \in \mathcal{N}(i) \setminus \{u\}\Big\}\Big), \forall i \in \mathcal{N}(u)\Big\}\Big) \tag{7}$$

The node embedding for user $u$ after two hops depends on ($\square$) the importance of all users interacting with the same items as $u$ on those items, and ($\triangle$) the importance of all items interacted by $u$ on user $u$. In other words, weighting the importance of each neighbor node on the ego node before the aggregation allows, after two propagation hops, to calculate **to what extent each user profile is influenced by the profiles of the other users who rated the same items**. Without loss of generality, a similar consideration could be made after a number of hops greater than two.

## 3.3. Enhancing neighborhood weighting through reviews

As known, graph-based models in machine learning are affected by oversmoothing [16, 17]. This phenomenon leads node embeddings, after multiple propagation hops, to become closer and closer in their representation in the latent space, eventually flattening their existing differences. As this behavior would profoundly weaken models' performance, exploration of the neighborhood generally tends to be constrained to very few hops (e.g., a maximum of two hops in attention-based weighting). **However, in recommendation scenarios, limiting the exploration of the user-item bipartite graph may represent an inconsistency to the idea of collaborative filtering**, where users are connected to share preferences and tastes for similar items.

Under this assumption, we believe the neighborhood weighting process could be further enhanced by exploiting other sources of information that are not usually taken into account. In the majority of popular online platforms for e-commerce (e.g., Amazon), **reviews** are fundamental tools to share **opinions** and **comments** about interacted items, as they convey the multi-faceted aspects that drove a user to interact with an item. Leveraging such side information on the connections existing among users and items in the bipartite graph (i.e., graph edges) can improve the learning of the importance weights by reducing the oversmoothing effect because each user/item node embedding is conditioned on the opinion conveyed by the review.

Let $\mathcal{W}_{ui} = \{w_1, w_2, \ldots, w_R\}$ be the set of $R$ words that compose the review written by user $u$ about item $i$. After an initial tokenization step, the sets of tokens for $\mathcal{W}_{ui}$ is defined as $\mathcal{T}_{ui} = \{t_1, t_2, \ldots, t_T\}$. Tokens are mapped to word embeddings, which are injected into an opinion-based model pretrained to predict the rating expressed by the user through specific terms in the review. While the output model carries the single information about the predicted review score, the activation of a hidden layer would unveil a richer source of textual features (i.e., an embedding) which drove the opinion-based model to predict that score. High-level features extracted from pretrained deep learning models can boost the recommendation performance of recommender systems leveraging items' side information (e.g., visual-based recommender systems [40, 41]). We deem these textual features to deserve a pivotal role in this weighting process.

Let $\mathbf{r}_{ui} \in \mathbb{R}^f$ be the textual embedding extracted from the review of user $u$ about item $i$ through the pretrained opinion-based model. First, we project $\mathbf{r}_{ui} \in \mathbb{R}^f$ to the same latent space as $\mathbf{e}_u \in \mathbb{R}^d$ and $\mathbf{e}_i \in \mathbb{R}^d$ with a one-layer neural network:

$$\mathbf{p}_{ui} = \text{LeakyReLU}\left(\mathbf{W}\mathbf{r}_{ui} + \mathbf{b}\right) \tag{8}$$

where $\mathbf{p}_{ui} \in \mathbb{R}^d$ is the projected review embedding, while $\mathbf{W} \in \mathbb{R}^{f \times d}$ and $\mathbf{b} \in \mathbb{R}^d$ are the projection matrix and the bias, respectively. We seek to retain only those textual features of review $\mathbf{r}_{ui}$ **which can be significant to later calculate the interdependence between this embedding and user/item ones**.

Then, we propose to enhance the neighborhood weighting procedure at hop $l$ by conditioning the importance weights also on the projected embedding of the review connecting user $u$ and item $i$. For instance, the importance of the neighbor item node $i$ on the ego user node $u$ after $l$ hops is calculated as:

$$\alpha_{i \to u}^{(l)} = \varphi\left(\mathbf{e}_i^{(l-1)}, \mathbf{e}_u^{(l-1)}, \mathbf{p}_{ui}\right) \tag{9}$$

Note that, **since $\mathbf{p}_{ui}$ cannot increase the impact of the oversmoothing effect (because it is not dependent on the hop $l$), its usage in the importance weight formula becomes even more beneficial**. Let us focus on the weighting function $\varphi(\cdot, \cdot, \cdot)$. Many approaches from the literature propose to leverage attention mechanisms, usually implemented as a neural network trained in the downstream task to predict the importance of the neighbor node on the ego node. In our solution, we opt for a simplified and lightweight formulation that seeks to calculate the similarity between the neighbor and the ego nodes, **conditioned on the opinion embedding of the review connecting them**. Specifically:

$$\alpha_{i \to u}^{(l)} = \cos\left(\mathbf{e}_i^{(l-1)} \odot \mathbf{p}_{ui}, \mathbf{e}_u^{(l-1)} \odot \mathbf{p}_{ui}\right) \tag{10}$$

where $\odot$ is the element-wise multiplication, and $\cos(\cdot, \cdot)$ is the cosine similarity. Note that we suppress negative similarities to zero as such weights are usually non-negative. Multiplying both node embeddings by the review opinion embedding provides the interplay between

each node feature and the opinion features, **thus producing a modified version of the node representation that conveys a richer source of information.** No trainable projection weight is learned in the presented formulation since the contribution of the review embedding is meaningful enough.

### 3.4. A double message-passing schema

The proposed neighborhood weighting procedure can help correct the representation error generated in the traditional message-passing schema. However, the idea is not to completely replace it, as several recent works from the literature have demonstrated its efficacy, especially in producing accurate recommendations [8]. The proposed approach involves a double message-passing schema, where two graph models are trained to refine **their own user/item node representations**. While the first one aggregates the contributions coming from the neighbor nodes into the ego nodes by weighting the neighborhood importance on the ego node **statically**, the second one aggregates the neighborhood's messages which are also weighted through the **opinion** embeddings from reviews.

We define the two graph convolutional networks as $\text{GCN}_e$ (*error-affected*) and $\text{GCN}_c$ (*correction*) and assign the node embeddings $\mathbf{e}_*$ to $\text{GCN}_e$, and the node embeddings $\mathbf{c}_*$ to $\text{GCN}_c$. As for the aggregation function, in both cases, we sum the weighted messages coming from the neighbor nodes. As such, the update of the user node embedding $u$ after $l$ hops is calculated as:

$$\begin{aligned}
\mathbf{e}_u^{(l)} &= \sum_{i \in \mathcal{N}(u)} \alpha_{i \to u} \mathbf{e}_i^{(l-1)} = \sum_{i \in \mathcal{N}(u)} \frac{\mathbf{e}_i^{(l-1)}}{\sqrt{|\mathcal{N}(u)|}\sqrt{|\mathcal{N}(i)|}} \\
\mathbf{c}_u^{(l)} &= \sum_{i \in \mathcal{N}(u)} \alpha_{i \to u} \alpha_{i \to u}^{(l)} \mathbf{c}_i^{(l-1)} = \\
&= \sum_{i \in \mathcal{N}(u)} \frac{\cos\left(\mathbf{e}_i^{(l-1)} \odot \mathbf{p}_{ui}, \mathbf{e}_u^{(l-1)} \odot \mathbf{p}_{ui}\right)}{\sqrt{|\mathcal{N}(u)|}\sqrt{|\mathcal{N}(i)|}} \mathbf{c}_i^{(l-1)}
\end{aligned} \tag{11}$$

Note that $\alpha_{i \to u}$ is static and only depends on the topology of the bipartite graph, while $\alpha_{i \to u}^{(l)}$ varies along with the exploration hop and depends on the embeddings of ego/neighbor nodes, and the opinion review embedding. After $L$ propagation hops, the final embedding representation is obtained as:

$$\begin{aligned}
\bar{\mathbf{e}}_u &= \sum_{l=0}^{L} \frac{1}{1+l} \mathbf{e}_u^{(l)}, \quad \bar{\mathbf{e}}_i = \sum_{l=0}^{L} \frac{1}{1+l} \mathbf{e}_i^{(l)} \\
\bar{\mathbf{c}}_u &= \sum_{l=0}^{L} \frac{1}{1+l} \mathbf{c}_u^{(l)}, \quad \bar{\mathbf{c}}_i = \sum_{l=0}^{L} \frac{1}{1+l} \mathbf{c}_i^{(l)}
\end{aligned} \tag{12}$$

where we apply the scaling factor $1/(1+l)$ to further alleviate the oversmoothing problem. A schematic overview of the node refining algorithm proposed for EGCF is displayed in Figure 2.

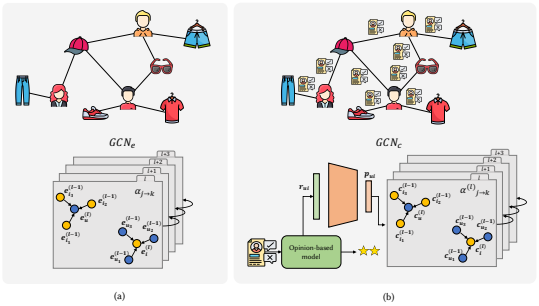



**Figure 2:** Overview of the node refining algorithm proposed for EGCF. A statically-weighted GCN network affected by node representation error (a) is corrected through another GCN network (b), where an opinion-based embedding is extracted from each review as edge side information to weight the importance of the neighbor nodes on their ego nodes.

**Table 1**
Statistics of the tested datasets.

| Datasets | #Users | #Items | #Interactions | Density | Average interactions per user |
|---|---|---|---|---|---|
| Baby | 4,669 | 5,435 | 29,214 | 0.00115 | 6.3 |
| Boys & Girls | 8,806 | 4,165 | 57,928 | 0.00158 | 6.6 |
| Men | 3,218 | 7,605 | 60,299 | 0.00246 | 18.7 |

Given the learned *error-affected* and *correction* embeddings from above, EGCF predicts if a user $u$ may interact with item $i$ through the following formulation:

$$\hat{y}_{ui} = \underbrace{\bar{\mathbf{e}}_u^\top \bar{\mathbf{e}}_i}_{\text{error-affected}} + \underbrace{\bar{\mathbf{c}}_u^\top \bar{\mathbf{c}}_i}_{\text{correction}} \tag{13}$$

Thus, we apply the error correction to the user/item embedding representation only when predicting the user/item interaction. We optimize EGCF with the state-of-the-art Bayesian Personalized Ranking (BPR) [42].

## 4. Experiments and Discussion

### 4.1. Experimental Setup

**Datasets.** We use three popular [43, 44] datasets from Amazon's Baby, Boys & Girls, and Men categories [18] which contain historical user-item interactions and reviews. We retain only interactions with non-empty reviews, then keep the 20k and 10k most popular items for Baby and Boys & Girls/Men, respectively. Finally, we apply the 5- and 15-core on items and users on Baby/Boys & Girls and Men, respectively. Statistics are in Table 1.
**Baselines.** We compare our approach with eight state-of-the-art models spanning several families: (i) *traditional CF* (BPRMF [42] and MultiVAE [4]); (ii) *review-based*

*CF* (ConvMF [31] and RMG [37]); (iii) *graph-based CF* (NGCF [7] and LightGCN [8]); (iv) *graph-based CF with attention* (GAT [11] and DGCF [15]).

**Reproducibility.** We adopt the temporal leave-one-out to split the datasets, where the last two recorded interactions are included in the validation and test. We tune hyper-parameters with [45] and follow the baselines papers, and fix the batch size to 256 and epochs to 400. As for EGCF, we extract review embeddings through a popular pre-trained model[1]. Datasets and codes are publicly available[2]. All models are implemented in Elliot [46].

**Evaluation protocol.** We measure the model accuracy by adopting the recall ($Recall@k$), the normalized discounted cumulative gain ($nDCG@k$), and the mean average recall ($MAR@k$) [8, 15]. Additionally, considering the influence of novel and diverse recommendation lists [47, 48] on both user's and business's interests, we also assess beyond-accuracy metrics such as the expected popularity complement ($EPC@k$) and the expected free discovery ($EFD@k$), along with indices measuring concentration and coverage, i.e., the 1's complement of the Gini ($Gini@k$), the Shannon entropy ($SE@k$), and the item coverage ($iCov@k$). Specifically, the $EPC@k$ and the $EFD@k$ refer to long-tail items and stand for the expected number of recommended unknown items which are also relevant, and the expected number of recommended known items which are also relevant, respectively. Furthermore, the $Gini@k$ and the $SE@k$ are used to assess items' distributional inequality, i.e., how unequally a recommender system shows different items to users, and the $iCov@k$ quantifies the number of items that the model recommends. For all metrics, higher values mean better performance. We leave the assessment of complexity measures for the proposed model in future extensions of the work.

## 4.2. Results and Discussion

**Recommendation accuracy (RQ1).** Table 2 reports the results for accuracy measures on the top-10 recommendation lists. Surprisingly, the sole introduction of reviews does not seem to produce a consistent accuracy boost. For instance, the strongest review-based method (i.e., RMG) surpasses BPRMF only for the $nDCG$ and the $MAR$ on Baby (i.e., 0.0911 vs. 0.0785 and 0.1059 vs. 0.0980, respectively). Contrarily, adopting a graph model can increase the accuracy to traditional CF. When comparing LightGCN with MultiVAE, which obtain the best performance in their respective recommendation families, we observe that the former improves, on Baby, the $Recall$ of 7% and the $MAR$ of 9%. However, the observed difference even reverts on Men for the $nDCG$ and the $MAR$. The application of attention mechanisms to weight the importance of neighbor nodes is rewarded in Baby and Boys & Girls, where GAT always outperforms NGCF, reaching remarkable results such as the $Recall$ on Baby (i.e., 0.1595 vs. 0.1411) and the $MAR$ on Boys & Girls (i.e., 0.1846 vs. 0.1783). Disentangling users' intents on interacted items (i.e., DGCF) produces even more accurate recommendations to NGCF on all datasets. Nevertheless, LightGCN always performs better than DGCF apart from very few cases (i.e., $nDCG$ and $MAR$ on Men), even though DGCF's calculated accuracy values do not substantially differ from LightGCN's ones (e.g., see the $MAR$ on Baby). Noticeably, the proposed model (i.e., EGCF) outperforms the other baselines under all settings and datasets, with near 100% statistical hypothesis tests (i.e., paired t-test) showing that the results significantly differ. This finding further motivates the goodness of the solution. While we observe a substantial accuracy improvement in traditional and review-based approaches (e.g., +12% to MultiVAE for the $MAR$ on Boys & Girls and +53% to RMG for the $Recall$ on Baby), introducing an additional GCN-like network guided by users' reviews is even more beneficial to correct the representation error observable in unweighted graph approaches. Particularly, results show that such correction may lead to small accuracy improvements in some cases (e.g., see the $Recall$ on Boys & Girls when correcting LightGCN) but also larger ones in other cases (e.g., see the $nDCG$ on Men when correcting LightGCN). Such outcomes suggest that *while keeping the error-affected contribution in the final prediction formula is useful to preserve the superior performance of graph-based models to traditional and review-based approaches, the introduced correction term is useful to gain even more accurate preference predictions than unweighted graph architectures.*

**Table 2**
Accuracy metrics, i.e., $Recall$, $nDCG$, and $MAR$, for top-10 lists. Best value is in **bold**, while second-to-best is underlined.

| Models | Baby | | | Boys & Girls | | | Men | | |
|---|---|---|---|---|---|---|---|---|---|
| | $Recall$ | $nDCG$ | $MAR$ | $Recall$ | $nDCG$ | $MAR$ | $Recall$ | $nDCG$ | $MAR$ |
| MostPop | 0.0940 | 0.0520 | 0.0627 | 0.1195 | 0.0647 | 0.0776 | 0.0702 | 0.0590 | 0.0672 |
| BPRMF | 0.1377 | 0.0785 | 0.0980 | 0.1821 | 0.1446 | 0.1666 | 0.1662 | 0.1314 | 0.1527 |
| MultiVAE | 0.1768 | 0.1262 | 0.1455 | 0.2224 | 0.1695 | 0.1990 | 0.2091 | 0.1656 | 0.1898 |
| ConvMF | 0.1230 | 0.0647 | 0.0800 | 0.1146 | 0.0831 | 0.0972 | 0.0838 | 0.0524 | 0.0584 |
| RMG | 0.1272 | 0.0911 | 0.1059 | 0.1512 | 0.1065 | 0.1325 | 0.1067 | 0.0727 | 0.0867 |
| NGCF | 0.1411 | 0.0916 | 0.1092 | 0.2006 | 0.1523 | 0.1783 | 0.1969 | 0.1461 | 0.1722 |
| LightGCN | 0.1892 | 0.1362 | 0.1590 | 0.2305 | 0.1743 | 0.2054 | 0.2124 | 0.1605 | 0.1882 |
| GAT | 0.1595 | 0.1051 | 0.1233 | 0.2069 | 0.1573 | 0.1846 | 0.1695 | 0.1254 | 0.1476 |
| DGCF | 0.1874 | 0.1352 | 0.1558 | 0.2249 | 0.1716 | 0.2023 | 0.2070 | 0.1554 | 0.1823 |
| **EGCF** | **0.1944**$^*$ | **0.1402**$^*$ | **0.1623**$^*$ | **0.2325** | **0.1792**$^*$ | **0.2089**$^*$ | **0.2195**$^*$ | **0.1703**$^*$ | **0.1988**$^*$ |

$^*$*statistically significant differences (p-value* $\leq 0.05$*).*

**Recommendation novelty and diversity (RQ2).** We also assess how novel and diverse recommendation lists are. The two novelty metrics in Table 3 (i.e., the $EPC@k$ and the $EFD@k$, left side) are discussed with concentration and coverage indices (i.e., the $Gini@k$, the $SE@k$, and the $iCov@k$, right side) as in an ideal recommender system, a loosely concentrated and large set of recom-

[1] Please refer to our GitHub repository.
[2] https://github.com/sisinflab/Edge-Graph-Collaborative-Filtering.

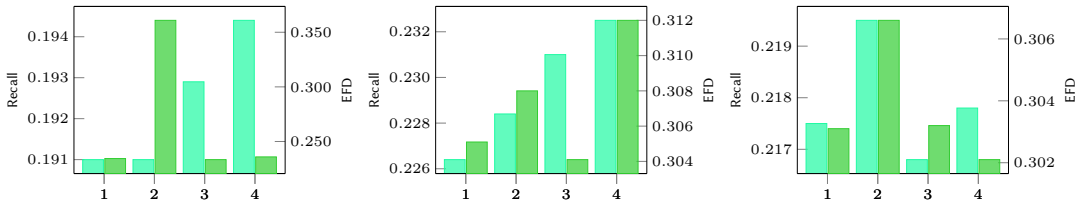
**Table 3**
Calculated novelty metrics, i.e., $EPC$ and $EFD$, on the left side, and diversity indices, i.e., $Gini$, $SE$, and $iCov$, on the right side, for top-10 lists. Best value is in **bold**, while second-to-best is underlined.

| Models | Baby | | Boys & Girls | | Men | |
|---|---|---|---|---|---|---|
| | $EPC$ | $EFD$ | $EPC$ | $EFD$ | $EPC$ | $EFD$ |
| MostPop | 0.0108 | 0.0728 | 0.0135 | 0.0913 | 0.0112 | 0.0904 |
| BPRMF | 0.0164 | 0.1153 | 0.0306 | 0.2282 | 0.0259 | 0.2167 |
| MultiVAE | 0.0268 | 0.2088 | 0.0360 | 0.2874 | 0.0333 | 0.2912 |
| ConvMF | 0.0135 | 0.0930 | 0.0174 | 0.1219 | 0.0102 | 0.0857 |
| RMG | 0.0193 | 0.1488 | 0.0226 | 0.1787 | 0.0144 | 0.1226 |
| NGCF | 0.0194 | 0.1463 | 0.0323 | 0.2510 | 0.0292 | 0.2531 |
| LightGCN | 0.0289 | 0.2271 | 0.0371 | 0.3012 | 0.0323 | 0.2856 |
| GAT | 0.0223 | 0.1708 | 0.0334 | 0.2616 | 0.0248 | 0.2106 |
| DGCF | 0.0287 | 0.2228 | 0.0365 | 0.2945 | 0.0311 | 0.2734 |
| **EGCF** | **0.0298**$^*$ | **0.2359**$^*$ | **0.0382**$^*$ | **0.3120**$^*$ | **0.0343**$^*$ | **0.3066**$^*$ |

$^*$*statistically significant differences (p-value $\leq 0.05$)*

| Models | Baby | | | Boys & Girls | | | Men | | |
|---|---|---|---|---|---|---|---|---|---|
| | $Gini$ | $SE$ | $iCov$ | $Gini$ | $SE$ | $iCov$ | $Gini$ | $SE$ | $iCov$ |
| MostPop | 0.0018 | 3.5313 | 18 | 0.0023 | 3.5724 | 18 | 0.0015 | 3.9332 | 32 |
| BPRMF | 0.0019 | 3.7819 | 40 | 0.0031 | 4.0921 | 203 | 0.0037 | 5.2991 | 192 |
| MultiVAE | 0.2139 | **9.9160** | 4,143 | 0.2671 | 10.2463 | 3,824 | 0.1085 | 9.8988 | 3,014 |
| ConvMF | 0.0018 | 3.5933 | 18 | 0.0030 | 3.9745 | 220 | 0.0029 | 4.6783 | 265 |
| RMG | 0.1059 | 9.4892 | 2,130 | 0.1567 | 9.7193 | 2,538 | 0.1146 | 10.0344 | 2,549 |
| NGCF | 0.0948 | 8.8700 | 2,641 | 0.3031 | **10.5595** | 3,668 | 0.1749 | 10.7116 | 3,651 |
| LightGCN | 0.1405 | 9.3105 | 3,417 | 0.2398 | 10.1586 | 3,647 | 0.2051 | 10.8815 | 4,384 |
| GAT | 0.1370 | 9.2024 | 3,102 | 0.2496 | 10.2821 | 3,449 | 0.1235 | 9.7802 | 3,530 |
| DGCF | 0.0673 | 8.3193 | 2,325 | 0.1800 | 9.7617 | 3,208 | 0.1304 | 10.2011 | 3,378 |
| **EGCF** | **0.2294** | 9.8535 | **4,490** | **0.3037** | 10.4545 | **4,030** | **0.2208** | **10.8876** | **4,920** |

*Statistical significance is not reported since it is calculated only on user level.*




(a) Baby (b) Boys & Girls (c) Men

**Figure 3:** Recommendation performance of EGCF, i.e., $Recall@k$ (histogram bars in teal blue) and $EFD@k$ (histogram bars in lime green), on top-10 recommendation lists, when varying the number of explored hops from 1 to 4.

mended items should equally span different ranges of popularity. As previously observed, EGCF is again the best or second-to-best technique. While NGCF is not as capable as LightGCN of proposing long-tail items on Boys & Girls (e.g., 0.2510 vs. 0.3012 for the $EFD$), the former surpasses the latter for the concentration indices on the same dataset (e.g., 10.5595 vs. 10.1586 for the $SE$). Since NGCF adopts an ego-neighbor interaction component, the concentration of explored and recommended near items gets loose. Moreover, neighborhood weighting leads to recommend items from the long tail (e.g., comparing GAT with NGCF, we observe a +17% for the $EFD$ on Baby). However, such a finding is not consistent with the trend recognized for the concentration and coverage indices (e.g., when comparing LightGCN with DGCF, we notice 0.1304 vs. 0.2051 for the $Gini$ on Men), as the neighborhood weighting procedure comes at the expense of a limited hop exploration, not allowing such models to explore wider catalog portions. Conversely, injecting user-generated reviews brings new informative content (e.g., RMG recommends a broader and less concentrated range of items from the catalog than DGCF on the Baby dataset). Finally, weighting the neighborhood importance and exploring long-distant user-item interactions through reviews-enriched content (i.e., EGCF) allows to retrieve larger portions of heterogeneous items (e.g., EGCF outperforms LightGCN for the $Gini$ by +63% on Baby and DGCF for the $SE$ by +7% on Boys & Girls), without retaining less popular items from the long-tail (observing the same models, +3% for the $EPC$ on Baby and +6% for the $EFD$ on Boys & Girls). Such outcomes demonstrate that *the content enrichment brought by the extracted review features (injected into the representation error correction) allows to explore user-item interactions at multiple hops, leading to more heterogeneous recommendation lists which also include items from the long-tail.*

**Effect of hop exploration number (RQ3).** Figure 3 displays, for EGCF, the $Recall@k$ and $EFD@k$ performance variation on top-10 recommendation lists when exploring a number of hops in the range 1-4, where even numbers stand for same node type connections (e.g., user-user), while odd numbers refer to opposite node type connections (i.e., user-item). As evident from the histograms of Baby and Boys & Girls, the $Recall@k$ consistently increases from 1 to 4 hops (this is why we adopt four hop explorations for EGCF on those datasets). The same trend is not observable for Men, where two explored hops seem to provide the highest accuracy boost, motivating the adoption of 2 hop explorations for EGCF on the same dataset. Such behavior could be due to the average number of users' interacted items in Men (approximately 19, see Table 1). The node refining probably does not require a broad exploration of its neighborhood. As for the $EFD@k$, the Baby and the Men datasets seem to agree on two exploration hops to produce the most diverse item lists of recommendations because they leverage (as

previously recalled) user-user and item-item interconnections (and similarities). The trend is also aligned with the Boys & Girls dataset, where user-user and item-item links are exploited even at a higher depth (i.e., four exploration hops). The emerged insights shed light on two main contributions: *(i) with the modified neighborhood weighting process, which makes use of reviews to enhance the informative content carried by user-item interactions, EGCF is less limited in the hop exploration, thus providing more accurate recommendations, and (ii) user-user and item-item connections are the keystones on which building more diverse item recommendation lists.*

## 5. Conclusion and Future Work

This work proposes Edge Graph Collaborative Filtering (EGCF), which incorporates users' opinions extracted from reviews into the edges of a GCN to weight the neighborhood importance on the ego node. Extensive experimental evaluation shows that EGCF outperforms traditional, review- and graph-based models. The work complements with an analysis of beyond-accuracy performance and an extensive study on the number of layers. Leveraging the importance of graph edges through node-node side information (e.g., users' reviews) opens to future directions, namely: (i) study the impact of this re-weighting by making it a hyper-parameter, and (ii) analyze the possible application of the proposed technique to different tasks other than recommendation.

## Acknowledgments

The authors acknowledge partial support from the projects PASSEPARTOUT, ServiziLocali2.0, Smart Rights Management Platform, BIO-D, and ERP4.0.

## References

[1] M. D. Ekstrand, J. Riedl, J. A. Konstan, Collaborative filtering recommender systems, Found. Trends Hum. Comput. Interact. 4 (2011) 175–243.

[2] Y. Koren, R. M. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer 42 (2009) 30–37.

[3] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, T. Chua, Neural collaborative filtering, in: WWW, ACM, 2017, pp. 173–182.

[4] D. Liang, R. G. Krishnan, M. D. Hoffman, T. Jebara, Variational autoencoders for collaborative filtering, in: WWW, ACM, 2018, pp. 689–698.

[5] T. N. Kipf, M. Welling, Semi-supervised classification with graph convolutional networks, in: ICLR (Poster), OpenReview.net, 2017.

[6] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, J. Leskovec, Graph convolutional neural networks for web-scale recommender systems, in: KDD, ACM, 2018, pp. 974–983.

[7] X. Wang, X. He, M. Wang, F. Feng, T. Chua, Neural graph collaborative filtering, in: SIGIR, ACM, 2019, pp. 165–174.

[8] X. He, K. Deng, X. Wang, Y. Li, Y. Zhang, M. Wang, Lightgcn: Simplifying and powering graph convolution network for recommendation, in: SIGIR, ACM, 2020, pp. 639–648.

[9] K. Mao, J. Zhu, X. Xiao, B. Lu, Z. Wang, X. He, Ultragcn: Ultra simplification of graph convolutional networks for recommendation, in: CIKM, ACM, 2021, pp. 1253–1262.

[10] Y. Shen, Y. Wu, Y. Zhang, C. Shan, J. Zhang, K. B. Letaief, D. Li, How powerful is graph convolution for recommendation?, in: CIKM, ACM, 2021, pp. 1619–1629.

[11] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, Y. Bengio, Graph attention networks, in: ICLR (Poster), OpenReview.net, 2018.

[12] Y. Xie, Z. Li, T. Qin, F. Tseng, J. Kristinsson, S. Qiu, Y. L. Murphey, Personalized session-based recommendation using graph attention networks, in: IJCNN, IEEE, 2021, pp. 1–8.

[13] M. Zhang, C. Guo, J. Jin, M. Pan, J. Fang, Sequential recommendation with context-aware collaborative graph attention networks, in: IJCNN, IEEE, 2021, pp. 1–8.

[14] Y. Wang, S. Tang, Y. Lei, W. Song, S. Wang, M. Zhang, Disenhan: Disentangled heterogeneous graph attention network for recommendation, in: CIKM, ACM, 2020, pp. 1605–1614.

[15] X. Wang, H. Jin, A. Zhang, X. He, T. Xu, T. Chua, Disentangled graph collaborative filtering, in: SIGIR, ACM, 2020, pp. 1001–1010.

[16] D. Chen, Y. Lin, W. Li, P. Li, J. Zhou, X. Sun, Measuring and relieving the over-smoothing problem for graph neural networks from the topological view, in: AAAI, AAAI Press, 2020, pp. 3438–3445.

[17] K. Zhou, X. Huang, Y. Li, D. Zha, R. Chen, X. Hu, Towards deeper graph neural networks with differentiable group normalization, in: NeurIPS, 2020.

[18] J. Ni, J. Li, J. J. McAuley, Justifying recommendations using distantly-labeled reviews and fine-grained aspects, in: EMNLP/IJCNLP (1), Association for Computational Linguistics, 2019, pp. 188–197.

[19] L. Chen, G. Chen, F. Wang, Recommender systems based on user reviews: the state of the art, User Model. User Adapt. Interact. 25 (2015) 99–154.

[20] M. Srifi, A. Oussous, A. A. Lahcen, S. Mouline, Recommender systems based on collaborative filtering using review texts - A survey, Inf. 11 (2020) 317.

[21] R. van den Berg, T. N. Kipf, M. Welling, Graph convolutional matrix completion, CoRR abs/1706.02263 (2017).

[22] W. Song, Z. Xiao, Y. Wang, L. Charlin, M. Zhang, J. Tang, Session-based social recommendation via dynamic graph attention networks, in: WSDM, ACM, 2019, pp. 555–563.

[23] C. Xu, P. Zhao, Y. Liu, V. S. Sheng, J. Xu, F. Zhuang, J. Fang, X. Zhou, Graph contextualized self-attention network for session-based recommendation, in: IJCAI, ijcai.org, 2019, pp. 3940–3946.

[24] Y. Wu, J. Yang, Dual sequential recommendation integrating high-order collaborative relations via graph attention networks, in: IJCNN, IEEE, 2021, pp. 1–8.

[25] X. Wang, X. He, Y. Cao, M. Liu, T. Chua, KGAT: knowledge graph attention network for recommendation, in: KDD, ACM, 2019, pp. 950–958.

[26] Z. Tao, Y. Wei, X. Wang, X. He, X. Huang, T. Chua, MGAT: multimodal graph attention network for recommendation, Inf. Process. Manag. 57 (2020) 102277.

[27] J. Ma, P. Cui, K. Kuang, X. Wang, W. Zhu, Disentangled graph convolutional networks, in: ICML, volume 97 of *Proceedings of Machine Learning Research*, PMLR, 2019, pp. 4212–4221.

[28] J. Wu, W. Shi, X. Cao, J. Chen, W. Lei, F. Zhang, W. Wu, X. He, Disenkgat: Knowledge graph embedding with disentangled graph attention network, in: CIKM, ACM, 2021, pp. 2140–2149.

[29] H. Wang, N. Wang, D. Yeung, Collaborative deep learning for recommender systems, in: KDD, ACM, 2015, pp. 1235–1244.

[30] A. Almahairi, K. Kastner, K. Cho, A. C. Courville, Learning distributed representations from reviews for collaborative filtering, in: RecSys, ACM, 2015, pp. 147–154.

[31] D. H. Kim, C. Park, J. Oh, S. Lee, H. Yu, Convolutional matrix factorization for document context-aware recommendation, in: RecSys, ACM, 2016, pp. 233–240.

[32] L. Zheng, V. Noroozi, P. S. Yu, Joint deep modeling of users and items using reviews for recommendation, in: WSDM, ACM, 2017, pp. 425–434.

[33] H. Liu, Y. Wang, Q. Peng, F. Wu, L. Gan, L. Pan, P. Jiao, Hybrid neural recommendation with joint deep representation learning of ratings and reviews, Neurocomputing 374 (2020) 77–85.

[34] Y. Lu, R. Dong, B. Smyth, Coevolutionary recommendation model: Mutual learning between ratings and reviews, in: WWW, ACM, 2018, pp. 773–782.

[35] H. Liu, F. Wu, W. Wang, X. Wang, P. Jiao, C. Wu, X. Xie, NRPA: neural recommendation with personalized attention, in: SIGIR, ACM, 2019, pp. 1233–1236.

[36] X. Wang, I. Ounis, C. Macdonald, Leveraging review properties for effective recommendation, in: WWW, ACM / IW3C2, 2021, pp. 2209–2219.

[37] C. Wu, F. Wu, T. Qi, S. Ge, Y. Huang, X. Xie, Reviews meet graphs: Enhancing user and item representations for recommendation with hierarchical attentive graph neural network, in: EMNLP/IJCNLP (1), Association for Computational Linguistics, 2019, pp. 4883–4892.

[38] J. Gao, Y. Lin, Y. Wang, X. Wang, Z. Yang, Y. He, X. Chu, Set-sequence-graph: A multi-view approach towards exploiting reviews for recommendation, in: CIKM, ACM, 2020, pp. 395–404.

[39] L. Shi, W. Wu, W. Hu, J. Zhou, J. Chen, W. Zheng, L. He, Dualgcn: An aspect-aware dual graph convolutional network for review-based recommender, Knowl. Based Syst. 242 (2022) 108359.

[40] R. He, J. J. McAuley, VBPR: visual bayesian personalized ranking from implicit feedback, in: AAAI, AAAI Press, 2016, pp. 144–150.

[41] Y. Deldjoo, T. D. Noia, D. Malitesta, F. A. Merra, Leveraging content-style item representation for visual recommendation, in: ECIR (2), volume 13186 of *Lecture Notes in Computer Science*, Springer, 2022, pp. 84–92.

[42] S. Rendle, C. Freudenthaler, Z. Gantner, L. Schmidt-Thieme, BPR: bayesian personalized ranking from implicit feedback, in: UAI, AUAI Press, 2009, pp. 452–461.

[43] X. Chen, H. Chen, H. Xu, Y. Zhang, Y. Cao, Z. Qin, H. Zha, Personalized fashion recommendation with visual explanations based on multimodal attention network: Towards visually explainable recommendation, in: SIGIR, ACM, 2019, pp. 765–774.

[44] Z. Wang, W. Ye, X. Chen, W. Zhang, Z. Wang, L. Zou, W. Liu, Generative session-based recommendation, in: WWW, ACM, 2022, pp. 2227–2235.

[45] J. Bergstra, R. Bardenet, Y. Bengio, B. Kégl, Algorithms for hyper-parameter optimization, in: NIPS, 2011, pp. 2546–2554.

[46] V. W. Anelli, A. Bellogín, A. Ferrara, D. Malitesta, F. A. Merra, C. Pomo, F. M. Donini, T. D. Noia, Elliot: A comprehensive and rigorous framework for reproducible recommender systems evaluation, in: SIGIR, ACM, 2021, pp. 2405–2414.

[47] S. Vargas, Novelty and diversity enhancement and evaluation in recommender systems and information retrieval, in: SIGIR, ACM, 2014, p. 1281.

[48] S. Vargas, P. Castells, Rank and relevance in novelty and diversity metrics for recommender systems, in: RecSys, ACM, 2011, pp. 109–116.