Hypergraphs with Attention on Reviews for Explainable Recommendation

Theis E. Jendal (✉)[1], Trung-Hoang Le[2], Hady W. Lauw[2], Matteo Lissandrini[1],
Peter Dolog[1], and Katja Hose[1,3]

[1] Aalborg University, Denmark
{tjendal,matteo,dolog,khose}@cs.aau.dk
[2] Singapore Management University, Singapore
{thle.2017,hadywlauw}@smu.edu.sg
[3] Technische Universität Wien, Austria
katja.hose@tuwien.ac.at

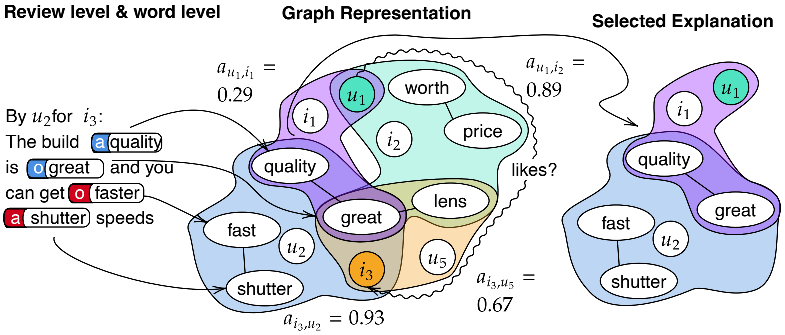
Abstract. Given a recommender system based on reviews, the challenges are how to effectively represent the review data and how to explain the produced recommendations. We propose a novel review-specific Hypergraph (HG) model, and further introduce a model-agnostic explainability module. The HG model captures high-order connections between users, items, aspects, and opinions while maintaining information about the review. The explainability module can use the HG model to explain a prediction generated by any model. We propose a path-restricted review-selection method biased by the user preference for item reviews and propose a novel explanation method based on a review graph. Experiments on real-world datasets confirm the ability of the HG model to capture appropriate explanations.

1 Introduction

Recommender Systems (RSs) utilize information about users' past interactions for recommendations, commonly referred to as Collaborative Filtering (CF) [9,18,49,22,19,50]. Often, these methods lack scrutability and users may not comprehend the reasons behind the recommendations [23]. Recent methods have exploited reviews for recommendation and explainability [27,9,52], typically by presenting the user with a given recommendation and an accompanying review that indicates what another user said about the item (see Figure 1). Yet, many review-level recommenders select reviews based solely on the item attention [9,27,52,35], disregarding the target user's preferences in selecting the explanation. Furthermore, some select the complete text, even when a review is verbose and filled with irrelevant information.

In this work, we extract aspects and opinions from reviews and represent them in a Hypergraph (HG) structure, where hyperedges connect sets of nodes representing aspects and opinions occurring together. A review-based HG captures the interdependencies of reviews by different users and items (e.g., *great*

Fig. 1. Review-level information connected to the item i_3 recommended to the user u_1 modelled as a hypergraph. Hyperedges are represented by the colored areas.



By $u_2$for i_3:
The build a quality is o great and you can get o faster a shutter speeds

quality in Figure 1), while capturing the intradependencies of aspects and opinions in a review (e.g., *fast shutter* and *great quality*). It also captures n-order correlations providing higher expressivity than binary relations in normal graphs.

Prior work focused on edge-labeled multigraph, which cannot easily model the high-order interdependencies between reviews mentioning different aspects and opinions [49,45,6]. They apply high-order graph convolutions, which can capture interdependencies across, but not within a review. Furthermore, the attention mechanism is often node-specific, as it does not take the user preferences into account when computing the attention of neighboring nodes [49,44,59].

Using a HG to represent reviews, we get a one-to-one mapping between edges and reviews while still capturing high-order interdependencies. Specifically, a review $r_{u,i} \in \mathcal{R}$ is represented as a set of triples consisting of an aspect, an opinion, and a sentiment, (a, o, s), where $a{\in}\mathcal{A}$, $o{\in}\mathcal{O}$, and $s{\in}\{-1, 1\}{=}\mathcal{S}$, often referred to as phrase-level opinions. The set of aspects $\mathcal{A}$ and opinions $\mathcal{O}$ are extracted from reviews, with the sentiment describing the polarity (e.g. *not* great). We opt for modeling reviews as hyperedges, being sets of nodes, in a graph consisting of users, items, aspects, and opinions. In Figure 1, we show an example of a review representation where a single edge connects the item to the user and related review's phrase-level opinions; thus, enabling us to capture ternary or higher intradependencies within the reviews. A HG consists of nodes $\mathcal{V}$ and hyperedges $\mathcal{E}{\subseteq}\mathcal{P}(\mathcal{V})\setminus\{\emptyset\}$; formally defined as $g{=}\langle\mathcal{V}, \mathcal{E}\rangle$. We define our HG, containing interactions, aspects, and opinions, as g, creating one edge for each review. Thus, we capture the global connections, i.e., interdependencies between different nodes and reviews, e.g., we can find that the aspect *quality* is important for the collaborative signal, and capture intradependencies within the individual reviews.

Our method, Hypergraph with Attention on Reviews (HypAR), takes into account both the learned attention about the user's reviews as well as the user's historical opinions about aspects when selecting a review as the explanation of a recommendation. Hence, given the predicted user preference for an item, we enable a better-informed attention mechanism that exploits connectivity between users, items, aspects, and opinions to provide an explanation. Furthermore, modeling the reviews in a graph allows us to generate graph-based explanations by

Table 1. Overview of methods. Review, Path, Graph Ex., and Word are the levels of the explanations. (*) Could not reproduce methods based on given information.

Methods	**Graph**	**Review**	**Path**	**Graph Ex.**	**Word**	**Source code**
R3* [35]	✗	✔	✗	✗	(✔)	✗
AENAR [61]	✗	✗	✗	✗	(✔)	✗
SGMC [8]	✗	✗	✗	✗	✗	✗
HAGERec* [59], **EIUM** [20], **KPRN** [51]	✔	✗	✔	✗	✗	✗
HRDR [27], **AHN** [12], **NARRE** [9]	✗	✔	✗	✗	✗	✗
KGCN(-LS) [46][45]	✔	✗	(✔)	✗	✗	✔
KTUP [6]	✔	✗	✔	(✔)	✗	✔
KGAT [49], **RuleRec** [31], **PGPR** [58], **RippleNet** [44]	✔	✗	✔	✗	✗	✔
RMG [53]	(✔)	✔	✗	✗	✔	✔
HUITA [52], **HANN** [11]	✗	✔	✗	✗	✔	✔
MTER [48]	✗	✗	✗	✗	✔	✔
TransNets [7]	✗	✔	✗	✗	✗	✔
TriRank [17]	✔	✗	✗	✗	(✔)	✗
HypAR	✔	✔	✔	✔	✔	✔

selecting relevant aspects and opinions; thus, we will present directly to the user the salient points from the review. For example, based on our review attention assigned to each review in Figure 1, we can select the most important reviews connecting user u_1 and item i_3 and to extrapolate which parts of the review text are most important, here the great quality of the lens. As such, the graph view captures directly both the path-based reasoning and a succinct and structured review representation as illustrated by the hyperedges.

Contributions. We summarize our contribution as follows: (i) We propose a novel review representation using the HG structure and an accompanying architecture that applies graph convolutions to incorporate sentiment polarity and opinions. (ii) We provide a *dual-view* explanation: review-to-graph, which results in a novel graph explanation taking user preferences w.r.t. items, aspects, and opinions into account. (iii) We construct a framework for explainable recommendations that is agnostic to the preference module. (iv) We define simple quantitative evaluation measures for explainable graph recommendation in the problem formulation. Through studies on four real-world datasets in different domains, we show HypAR's improvements upon baselines. Therefore, HypAR is the first model with the ability to make both review-level and graph-based explanations for its recommendations, providing ad-hoc explanations that are integral to the recommendation process instead of weak post-hoc explanations.

2 Related Work

In recent years, Matrix Factorization (MF) methods [59,49,46,45,44] and graph-based methods have become popular for CF [18,50,17]. Yet, limited work focus on representing reviews as graphs. In the following, we will introduce recent works using graphs and reviews as explanations, as well as related HG architectures.

Explainability. Multiple types of explanations have been proposed, from identifying areas of interest in product images [10] to finding relevant users or items [28]. Previous works has used the attention mechanims in multiple research areas [10,2,41], particularly some use it to select the most important review as textual explanations [9,27,52]. In Table 1, we show various explainability options, as reviews written by users, paths selected in a graph, showing a complete graph, and showing individual words. *The first column is graph-based* *recommenders, clearly showing no graph-based recommendation methods provide review explanations*, with the exception of RMG [53]. However, RMG uses the graph structure as the preference module and not for review-representation.

Reviews contain subjective opinions on different aspects and have been used for explanations [27,12,52,9,53,7]. Specifically, some use review-level attention mechanism [9,27,52] for explanations; however, these are unable to capture high-order relationships between users and items. The attention mechanism used is often item-specific, meaning non-personalized explanations, while those that utilize a user's preference, i.e. personalized explainations [12,11,7], are ill-suited for ranking, requiring distinct unique computations for each user-item pair. *Therefore, instead of increasing the ranking complexity, we propose a novel review selection method utilizing the HG structure based on the non-personalized attention mechanisms.* In the experiment section, we show our selection strategy outperforms the non-personalized methodology.

Knowledge Graphs (KGs) [20,58,59,49,45] can supplement MF methods when ratings are sparse. Information is either propagated inwards using Graph Neural Networks (GNNs) [49,6] or outwards using ripples [44] to capture high-order connectivities. Current KG based methods only exploit factual (instead of opinion-based) explanations, e.g., a path would only describe a product, not any user opinions. To provide an explanation that would match the subjective judgment the user may have about a product, methods have explicitly extracted aspects and opinions from reviews to generate opinionated explanations [48] or aspect-level explanations [4,17,56,63], but they cannot capture high-order connections.

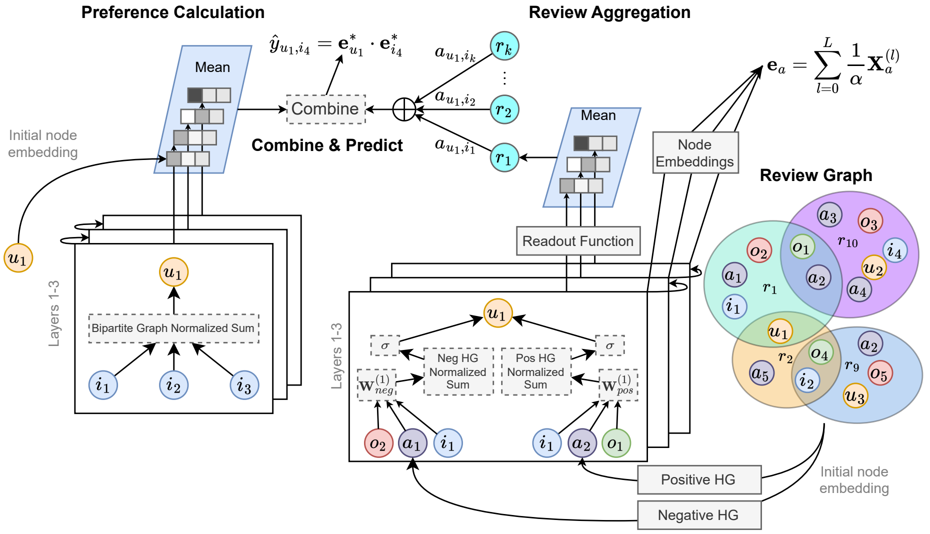
Hypergraph. The Hypergraph Neural Network (HGNN) [14] uses a GNN on HGs; however as a HG's edges are sets of nodes, the HGNN first aggregates nodes ocurring in the edges and then the aggregates the edges a node occurs in. HGNNs variants have been used for multiple tasks [47,55,16,54], often differing in the HG construction methodology [42,16,60]. However, none of these methods are explainable [24,38], or, if explainably, use a naïve explanation like producing k-most similar items [8]. Yet, such 'explanations' are still opaque, as they not explain why items are similar. *Instead, we set out to select reviews (hyperedges, not nodes) as explanations, which provide relationships between items and aspects.*

3 Methodology

Our method HypAR (Figure 2) consists of four modules: (i) review representation, which computes the embeddings for each review; (ii) review aggregation, which aggregates the reviews' embedding generating an opinion vector; (iii) preference computation, computing the user and item preference vectors; and finally (iv) combine and predict module, which combines the vectors for ranking. In the following, we define our problem and then expand on each module.

Problem formulation. We define the interaction matrix, given a set of users $\mathcal{U}$ and items $\mathcal{I}$, as $\mathbf{I} \in \{0,1\}^{|\mathcal{U}|\times|\mathcal{I}|}$, s.t. $\mathbf{I}_{u,i}$=1 if the user u has interacted with the item i; otherwise 0. Similarly, for all interactions, we have a corresponding

Fig. 2. Illustration of the embedding process of user u_1.



review $r_{u,i}$ if $\mathbf{I}_{u,i} = 1$. For each interaction, we extract aspects mentioned in the reviews and the given opinion, such as *fast* or *worth*, along with the sentiment.

The objective of our method is: (i) to rank items according to a user's preference and (ii) to ensure that the ranking of items is explainable. Regarding the recommendation objective, we model our task as a top-k recommendation problem, s.t., given a learned model Θ, we are able to rank the items $\mathcal{I}$ based on their likelihood of being liked by the user. For explainability, we predict both whether the user will like/buy the product as well as the reason behind the choice. In our model, we assume reviews to be concrete manifestations of the reasoning behind the user preference. Thus, here we assume an explanation to be either a given review or to be comparable to a review, i.e., a set $\varepsilon \subset \mathcal{A} \times \mathcal{O} \times \mathcal{S}$ of aspects, opinion, sentiment triples that justify the (predicted) user choice.

Based on this, we can identify *two new forms of explainability metrics* describing a good explanation ε: (i) ε is the set of aspect, opinion, sentiment triples that constitute the actual review the user will write; or (ii) ε can be used to deterministically separate items that will be ranked higher by the user. The first case can intuitively be understood as a strong correlation between a given set of aspects (e.g., *fast shutter*) and if a user would generally prefer items described with those aspects to items selected based on another disjoint set of random aspects (e.g., *long cable*). For the second objective, we are interested in knowing if the generated explanation matches the method's ranking, i.e., if the user prefers the item due to the explanation (*fast shutter*), we assume items matching the explanation would be ranked higher than another random ranking.

Review Representation. Given an initial representation of nodes $\mathbf{X}^0 \in \mathbb{R}^{|\mathcal{V}| \times d}$ and our HG g, we define the incidence matrix $\mathbf{H} \in \{0,1\}^{|\mathcal{V}| \times |\mathcal{E}|}$, indicating whether $v \in \mathcal{V}$ occurs in $e \in \mathcal{E}$, the diagonal matrices of edge degrees $\mathbf{D}_e$ and nodes $\mathbf{D}_v$, where the node degree is defined as $d(v) = \sum_{e \in \mathcal{E}} \mathbf{H}_{v,e}$, and the edge degree as $d(e) = \sum_{v \in \mathcal{V}} \mathbf{H}_{v,e}$ [14]. To capture the semantics of each word we use Word2Vec [33], as in [9,35], to learn the word embeddings and apply 2 layers of Multi-Layer Perceptrons (MLPs) with `tanh` activation to transform the initial aspect and opinion vectors from the word-embedding space into $\mathbf{X}^0$.

We employ sentiment-specific linear transformation matrices and split our HG into two HGs: one containing only positive phrase-level opinions and one containing only negative phrase-level opinions (see Figure 2), allowing our model to differentiate between positive and negative phrases. As such, we are able to capture both the sentiment and the high-order connections, using:

$$\mathbf{X}^{(l)} = \tfrac{1}{2} \sum_{s \in \mathcal{S}} \mathbf{D}_{vs}^{-\frac{1}{2}} \mathbf{H}_s \mathbf{D}_{es}^{-1} \mathbf{H}_s^\top \mathbf{D}_{vs}^{-\frac{1}{2}} \mathbf{X}^{(l-1)} \mathbf{W}_s^(l) \tag{1}$$

where H_s, D_{vs}, and D_{es} are the sentiment-specific incidence, node degree, and edge degree matrices, respectively, and $\mathbf{W}_s^l \in \mathbb{R}^{d \times d}$ is the sentiment specific transformation. The HG convolution is very similar to Graph Convolutional Network (GCN) layers with an extra normalization using edge degrees. Yet, we are interested in capturing review-specific occurrences; therefore, we define an edge-wise readout function using the mean aggregator. Thus, the review representation at layer l is $\mathbf{r}_{u,i}^{(l)} = \frac{1}{|r_{u,i}|} \sum_{v \in r_{u,i}} \mathbf{X}_v^{(l)}$. To better capture the connectivities in the HG, we propagate the initial embeddings through the convolutional layer L times and aggregate the layers using the mean, leaving the study of other aggretators as future work, and define the vector as $\mathbf{r}_{u,i} = \sum_{l=0}^{L} \alpha_l \, \mathbf{r}_{u,i}^{(l)}$, where $\alpha_l = \frac{1}{L+1}$.

Attention-Based Review Aggregation. We employ an attention-specific user and item representation vector to capture the quality of the reviews written by the user and about the item [9], defined as:

$$a_{i,u}^* = \mathbf{h}^\top \mathrm{ReLU}(\mathbf{W}_a(\mathbf{r}_{u,i} \| \mathbf{q}_u) + \mathbf{b}_1) + b_2 \tag{2}$$

where $\mathbf{W}_a \in \mathbb{R}^{d' \times 2d}$ transforms the review representation into the attention space taking the quality of the user into account, $\mathbf{h} \in \mathbb{R}^{d'}$ is the attention vector, $\mathbf{b}_1$ and b_2 are learned biases, ReLU [34] is a non-linear activation function, and $\|$ is the concatenation operation. We can calculate the user-specific review attention $a_{u,i}$ by substituting $\mathbf{q}_u$ with $\mathbf{q}_i$. The aggregation for a user is illustrated in Figure 2 under review aggregation. The current attention score is unbounded; we normalize the attention weight w.r.t. all other reviews of the item using softmax $a_{i,u} = \frac{\exp(a_{i,u}^*)}{\sum_{(u',i) \in R} \exp(a_{i,u'}^*)}$. Given the attention mechanism, we calculate the weighted mean of the item-specific review embeddings. Our method selects the more "important" reviews for the item as $\mathbf{r}_i = \sum_{(u,i) \in R} a_{i,u} \mathbf{r}_{u,i}$. As a single attention kernel may not be sufficient to capture complex explanations, we learn multiple kernels, taking the average embedding over all kernels' final output.

Preference Computation. We note that our review aggregation module is agnostic to the preference computation module (see Figure 2), i.e., our architecture that learns to select explanations does not impose any restriction on the module that predicts which item to recommend to the user. Therefore, our architecture learns in parallel both to provide recommendations as well as to explain them. Aggregating information across high-order connectivity has been found to greatly increase performance for top-n recommendation in the preference computation module [49,18,46,45]. To show the effectiveness of representing the extracted aspects and opinions in a HG, we adopt two preference computation modules: (i) using MF, as in NARRE [9] and HRDR [27], allowing for direct comparison of performance, simply representing users and items in a latent space $X_p^{(0)} \in \mathbb{R}^{|\mathcal{V}|\times d}$, s.t. a user u and item i each have a unique row in $X_p^{(0)}$, defined as $\mathbf{e}_u$ and $\mathbf{e}_i$. (ii) using the well-performing LightGCN [18], where users and items are represented as the average embedding of neighbors, through multiple layers of GNN convolutions, as: $\mathbf{e}_u^{(l)} = \sum_{i\in\mathcal{N}_u} \frac{1}{\sqrt{|\mathcal{N}_u||\mathcal{N}_i|}}\mathbf{e}_i^{(l-1)}$, $\mathbf{e}_i^{(l)} = \sum_{u\in\mathcal{N}_i} \frac{1}{\sqrt{|\mathcal{N}_u||\mathcal{N}_i|}}\mathbf{e}_u^{(l-1)}$, where $e_i^{(0)}$ is the item i's row in the learned preference matrix $\mathbf{X}_p^{(0)}$.

Combine and Predict. There are multiple ways of combining the review and aggregation modules, such as adding them [9,27]. Yet, simply combining the embeddings using addition may lead to subpar performance. We, therefore, investigate three different combination methods: (i) **addition** combines the preference and review embeddings using element-wise addition as $\mathcal{C}_{add}(u) = \mathbf{e}_u + \mathbf{r}_u$; (ii) **multiplication** uses element-wise multiplication, ($\odot$), thereby capturing the affinity between the two module representations as $\mathcal{C}_{mul}(u) = \mathbf{e}_u \odot \mathbf{r}_u$; and (iii) **concatenation** creates a vector of new dimension as it concatenates the two embeddings as $\mathcal{C}_{cat}(u) = \mathbf{e}_u \| \mathbf{r}_u$. In this case, if the output embeddings are of unequal size, then the larger embedding would have a higher weight in the final prediction step. Yet, this method is also the only one proposed here that handles the case where preference and review modules produce output embeddings of unequal size. As such, using any of the combination functions $\mathcal{C}$, we can compute the embeddings of the users and items as: $\mathbf{e}_u^* = \mathcal{C}(u), \quad \mathbf{e}_i^* = \mathcal{C}(i)$.

Prediction. Our framework is agnostic to the prediction method. We here describe to possibilitites. The learned similarity, $f_{NARRE}(u,i) = \mathbf{W}_p(\mathbf{e}_u^* \odot \mathbf{e}_i^*) + b_u + b_i + \mu$ [9], is a linear transformation of the affinity between the users and items representation, where $\mathbf{W}_p \in \mathbb{R}^{1\times d'}$ and b_u, b_i, and μ denotes user, item and global biases, respectively. Here, the weight matrix $\mathbf{W}_p$ is able to select which features are most important for ranking user and item affinities. The other proposed prediction method is the inner product ($\cdot$) between a user and an item, which in certain settings outperform the learned [37], as $f_{dot}(u,i) = \mathbf{e}_u^* \cdot \mathbf{e}_i^*$.

Optimization. We use Bayesian Personalized Ranking (BPR) as the collaborative loss function [36] as $L_{CF} = \sum_{(u,i,j)\in\{(u,i,j)|\mathbf{I}_{u,i}=1,\mathbf{I}_{u,j}=0\}} -\ln\sigma\left(\hat{y}_{u,i} - \hat{y}_{u,j}\right)$, item i is preferred over item j; σ is the sigmoid function; and $\hat{y}_{u,i}$ is the output of either f_{NARRE} or f_{dot}. To accommodate the quantitative explainability reasoning, we develop an explainability-specific loss function. Intuitively, if a user mentions some aspects and opinions about an item, we assume them to be important. We, therefore, propose a TransR [26]-like similarity function augmented for AOS triple as $f_{TR}(u,i,a,o,s) = \left(\mathbf{W}_s^1(\mathbf{e}_u^* \| \mathbf{e}_i^*) + \mathbf{s}\right)^\top \mathbf{W}_s^2(\mathbf{e}_a \| \mathbf{e}_o)$, where $\mathbf{W}_s^1, \mathbf{W}_s^2 \in \mathcal{R}^{d''\times d'}$ are sentiment-specific weight matrices, such that we can rank aspect and opinion of both positive and negative sentiment. Here $\mathbf{e}_a$ and $\mathbf{e}_o$ are the average node representations of the aspect and opinion, computed using the HG convolutions calculated (see Figure 2); and $\mathbf{s} \in \mathcal{R}^{d''}$ is a sentiment-specific relation vector. We maximizes the similarity using BPR, minimizing:

$$L_{AOS} = \sum_{(u,i,a,o,s,\bar{a},\bar{o},\bar{s})\in\mathcal{B}} -\ln\sigma\left(f_{TR}(u,i,a,o,s) - f_{TR}(u,i,\bar{a},\bar{o},\bar{s})\right) \quad (3)$$

where $\mathcal{B}$=$\{(u,i,a,o,s,\bar{a},\bar{o},\bar{s})|\ \mathbf{I}_{u,i}$=$1, (a,o,s)$∈$r_{u,i}, (\bar{a},\bar{o},\bar{s})$∉$r_{u,i}\}$.

The final loss function includes Θ, the set of all learnable parameters, including $\mathbf{X}^{(0)}$ and $\mathbf{X}_p^{(0)}$ as $L = L_{CF} + \gamma L_{AOS} + \lambda\|\Theta\|_2^2$, where γ and λ are parameters for tuning the L_{AOS} loss and L_2 regalurization, respectively. When training, we exclude the interactions we are ranking from the preference module and the accompanying reviews from the review aggregation module, ensuring the method does not have a bias towards already seen items. For each excluded review, we sample an AOS triple for the explainability loss such that we can optimize both the preference and review aggregation modules, concurrently. In practice, we optimize using AdamW with decoupled weight decay. Weight decay is equivalent to L_2 when using SGD, but L_2 does not scale properly with adaptive gradients [29].

Generating Explanations. Our method produces explanations that can then be employed at three different levels (see Figure 1): (i) at review level, selecting a review that can provide information that is *user specific*; (ii) at word level, highlighting important aspects and opinions; and (iii) at graph level, explanations are paths connecting the user to the item via aspects-opinion pairs and items in other reviews. The graph review level, allows a user to quickly understand why an item is recommended, by giving an easy-to-understand connection between previous purchases and the item. However, this view lacks the context of the extracted phrases, which is often necessary for purchase decisions. Thus, the underlying review text can be used for context, and the extracted phrase-level words are highlighted to draw attention to important areas. Yet, the reviews selected in previous works [9,27] are not user-specific, meaning the same review is given to all users when selecting an item. Such review selection is suboptimal, as it does not consider the user's preferences. Instead, we propose a path-based restriction of the reviews we can use as explanations for an item towards a user.

In Figure 1, there are multiple paths from the user to the item. However, connecting on the opinion word is most likely uninformative, as reviews may use *great* in very different contexts. Instead, we define the matching criterion as matching on aspects and opinion pairs, creating a path from r_{u_1,i_1} to r_{u_2,i_3}, via the pair (*quality, great*). While in Figure 1, there is a direct link between a user review and the selected item review, longer paths are possible. In such cases, paths such as $i \rightarrow u$ or $i \rightarrow v \rightarrow u$ are possible, where v is the connecting user and $\rightarrow$ indicates a connection through an aspect-opinion pair. We limit to a max of one intermediary user for reduced graph sizes. Intuitively, if a user writes about similar aspects and we have learned that these are important for the respective user, their reviews about a possible recommendation might also be important. The undirected labeled graph g_m used to compute the explanation contains only weighted paths from user to item satisfying the connectivity constraints. We define the graph as $g_m = \{\{u, r_{u,i}, \{a, o\}\}|r_{u,i} \in \mathcal{R}, (a,o,s) \in r_{u,i}\}$.

Table 2. Data statistics

Dataset	#User	#Product	#Aspect	#Opinion	#Review
Computer	19,818	8,431	5,046	4,017	92,761
Camera	4,770	2,612	2,182	2,218	21,122
Toy	2,672	1,919	780	1,186	16,070
Cellphone	2,340	1,350	817	1,196	11,134

We are interested in limiting the possible reviews to a single review for review explanations and to a set of reviews for graph-recommendation. Based on the graph constructed either directly between a user and an item or through multiple hops, we will use the user's attention on reviews along the path, except for the last hop, where we use the item's attention, to select the best path. Intuitively, we are interested in knowing how important users find the reviews, and by extension, the aspects and opinions, but also which reviews are important for the item we are recommending. For example, in Figure 1, we could select either the review $r_{u_1,i_2}\rightarrow r_{u_5,i_3}$ or $r_{u_1,i_1}\rightarrow r_{u_2,i_3}$. If we greedily select starting from the user, we would select the path $r_{u_1,i_2}\rightarrow r_{u_5,i_3}$, leading to the review level explanation of r_{u_5,i_3}, while greedily selecting, starting from the item would lead to the same selection as shown in Figure 1. Starting from the user, would create more diverse, user-specific explanations but could disregard the attention score on the item side, while greedily selecting reviews from the item side, could lead to less diversity. Based on these limitations, we propose three selection methods which we will study in the experimental section: (i) greedily selecting reviews, starting from the user until we find the item; (ii) greedily selecting reviews, starting from the item until we find the user; and (iii) finding the path with maximum weight.

Complexity analysis. HypAR is bounded by the HG convolutions. Thus, to estimate the time complexity of our method, it is sufficient to study the complexity of the HGNN. The convolutions can be described as four operations applied sequentially: (i) feature transformation, with complexity $O(|\mathcal{V}|d^2)$; (ii) two node degree normalizations, of $O(2|\mathcal{V}|^3)$; (iii) transformations from nodes to edges and back again, with $O(2|\mathcal{V}||\mathcal{R}|^2)$; and (iv) neighborhood aggregation, being $O(|\mathcal{V}|^2d)$. Of these, the node to edge transformations are of highest complexity, as $|\mathcal{R}|\gg|\mathcal{V}|\gg d$. The complexity of the HGNN is $O(|\mathcal{V}||\mathcal{R}|^2)$; however, this naïvely assumes that we utilize dense Matrix Multiplication (MM) instead of Sparse MM (SPMM). Using SPMM we reduce the complexity of (iii) to $O(\|\mathbf{H}\|_1|\mathcal{R}|)$, which can be rewritten using the average number of edges $\mathcal{E}_a$, as $O(|\mathcal{V}|\mathcal{E}_a|\mathcal{R}|)$. In most cases $\mathcal{E}_a\ll|\mathcal{V}|$, greatly reducing the complexity. Furthermore, as operations (ii) and (iii) are computed once they have no influence on time complexity during forward propagation, and HypAR is thus bounded by operation (iv).

4 Experiments

The experimental objectives revolve around how our HG-based model compare to state-of-the-art models at providing high-quality recommendations; explanation quality, whether our model is better suited for providing high-quality explanations; and how well the model upholds our explanation objectives.

Datasets. We utilize data based on the four public datasets of 2014 Amazon review dataset [32]: *Computer and Accessories* (Computer), *Camera and Photo*

(Camera), *Toys and Games* (Toy), and *Cell Phones and Accessories* (Cellphone), for which aspects, opinions and sentiments have been extracted [23]. However, due to space restrictions, we only show our explainability experiments on the smallest and largest datasets, being Cellphone and Computer, respectively, having similar results on the other datasets withheld. We filter users and items with fewer than five ratings and split the datasets with the ratio 0.6:0.2:0.2, for each user based on time. Furthermore, we use a porter-stemmed version of all aspects and opinions by applying Gensim's text preprocessor[4].

Baselines. As shown in Table 1, there are two major categories of explanations, either based on reviews or paths. We have selected NARRE [9] and HRDR [27] for the first group of explainable recommenders. Both use Convolutional Neural Networks on word vectors to represent a review, with attention mechanisms for review selection as explanation. The most prominent method using GNN for recommendation that also produces some form of explanation exploiting connections in the graphs is KGAT [49]. KGAT utilizes TransR [26] to learn weights between entities in a KG. For word level, we use TriRank [17], using a tripartite graph of user, items, and aspects for ranking smoothing. We further compare to MF [36] and LightGCN [18], as we model our preference module after them. We have implemented all methods in the Cornac [40] framework as it supports multi-modal information, such as aspects and reviews[5].

Evaluation Metrics. For each user in the test set, we rank all items not interacted with in the train and validation sets [49,18,50]. To evaluate the ranking quality, we measure AUC [15], MAP [5], and NDCG [21]. To evaluate the methods' explainability, we opt for three different methodologies: (i) compare the selected review with the review written by the user; (ii) select a graph (or path) as the explanation and compare it to the ground truth graph constructed from the user's review; and (iii) given a set of aspects and opinions assumed to describe the user's preferences, we study the quality of the approximate ranking obtained by ranking higher items matching them (described in Section 3).

To compare a selected review with a ground truth review, we adopt five different sentence similarity metrics for evaluation: BERTScore [62], being based on textual embedding similarities; and BLEU [57], METEOR with alpha=0.9 [3], and ROUGE [25] which are based on n-gram overlaps. For graph overlap, we use Precision [43], Recall [43], F1-measure, overlap-coefficient [30], and Diversity [1]; measuring the methods' ability to retrieve correct and unique aspects and opinions. For fairness, we allow KGAT to sample an unlimited number of paths until it has chosen (close to) as many nodes as HypAR to study KGAT's ability to select a diverse and relevant subset of nodes.

Recommendation Performance. The results are shown in Table 3, with the default HypAR using LightGCN as the preference module, concatenation as the combiner, and dot product for prediction. Our experiments show concatenation to outperform addition and multiplication (not detailed here for brevity).

[4] `https://radimrehurek.com/gensim/parsing/preprocessing.html#gensim.parsing.preprocessing.stem_text`

[5] All methods are available at `https://github.com/PreferredAI/cornac`

Table 3. Recommendation performance on different datasets. **Bold** is the best performing and $\underline{\text{underline}}$ is the second best.

Method	**Cellphone**			**Toy**			**Camera**			**Computer**		
	AUC	**MAP**	**NDCG**	**AUC**	**MAP**	**NDCG**	**AUC**	**MAP**	**NDCG**	**AUC**	**MAP**	**NDCG**
MostPop	0.5370	0.0314	0.1709	0.4412	0.0047	0.1338	0.5916	0.0171	0.1528	0.6604	$\underline{0.0148}$	**0.1390**
BPR-MF [36]	0.5662	$\underline{0.0317}$	0.1711	0.4729	0.0050	0.1337	0.6190	$\underline{0.0174}$	$\underline{0.1527}$	$\underline{0.6785}$	**0.0150**	**0.1390**
NARRE [9]	0.5123	0.0077	0.1385	0.5135	0.0053	0.1357	0.4983	0.0030	0.1217	0.5020	0.0011	0.1036
HRDR [27]	0.5317	0.0206	0.1541	0.4964	0.0053	0.1346	0.5088	0.0060	0.1272	0.4522	0.0059	0.1120
HypAR-MF	**0.6440**	**0.0334**	**0.1789**	**0.6462**	**0.0128**	**0.1544**	**0.6635**	**0.0177**	**0.1549**	**0.6890**	0.0124	$\underline{0.1355}$
HypAR-MF$_{NARRE}$	$\underline{0.6422}$	0.0272	$\underline{0.1732}$	$\underline{0.6302}$	$\underline{0.0116}$	$\underline{0.1524}$	$\underline{0.6314}$	0.0145	0.1484	0.6684	0.0110	0.1330
TriRank [17]	0.6965	0.0248	0.1769	0.6666	0.0136	0.1585	0.6887	0.0117	0.1507	0.7054	0.0048	0.1247
NGCF [50]	0.7430	0.0365	0.1900	0.7138	0.0158	0.1622	0.7122	0.0216	0.1631	0.6978	0.0130	0.1358
KGAT [49]	0.7295	0.0500	0.2017	0.6830	0.0155	0.1599	0.6928	0.0202	0.1602	0.7105	0.0113	0.1352
LightGCN [18]	0.7448	$\underline{0.0507}$	0.2037	0.7129	0.0184	0.1648	0.7294	$\underline{0.0293}$	$\underline{0.1741}$	0.7181	0.0187	0.1458
HypAR	**0.7533**	**0.0517**	**0.2054**	$\underline{0.7169}$	0.0199	$\underline{0.1663}$	$\underline{0.7325}$	0.0286	0.1734	0.7278	$\underline{0.0194}$	**0.1473**
HypAR ($\gamma = .1$)	$\underline{0.7515}$	$\underline{0.0507}$	$\underline{0.2045}$	**0.7172**	$\underline{0.0200}$	**0.1665**	**0.7348**	**0.0297**	**0.1747**	$\underline{0.7280}$	**0.0196**	0.1471
HypAR$_{NARRE}$	0.7293	0.0501	0.2029	0.6826	**0.0201**	0.1656	0.7207	0.0278	0.1718	**0.7308**	0.0191	$\underline{0.1472}$

Table 4. The results of the review selection.

Method	**Cellphone**				**Computer**			
	BERTScore	**BLEU**	**METEOR**	**ROUGEL**	**BERTScore**	**BLEU**	**METEOR**	**ROUGEL**
HRDR	0.8390	0.0143	0.0680	0.0943	0.8341	0.0106	0.0561	0.0793
NARRE	**0.8408**	0.0220	0.0920	0.1049	**0.8341**	0.0070	0.0429	0.0727
HypAR$_i$	0.8389	0.0291	0.1090	0.1082	$\underline{0.8328}$	0.0250	0.0993	0.0968
HypAR$_{gi}$	0.8392	0.0360	0.1141	$\underline{0.1141}$	0.8321	$\underline{0.0282}$	0.0990	**0.0993**
HypAR$_{gu}$	0.8389	0.0295	0.1114	0.1085	0.8324	0.0258	$\underline{0.1015}$	0.0971
HypAR$_w$	0.8389	0.0295	0.1106	0.1088	0.8323	0.0259	**0.1019**	0.0969
HypAR$_{gi}$($\gamma = .05$)	$\underline{0.8396}$	**0.0374**	**0.1152**	**0.1154**	0.8322	**0.0283**	0.0991	**0.0993**
HypAR$_{gi}$($\gamma = .1$)	0.8391	$\underline{0.0362}$	$\underline{0.1143}$	0.1138	0.8321	**0.0283**	0.0993	$\underline{0.0991}$
Improv %	-0.15*	70.23*	25.19*	10.05*	-0.16*	165.87*	81.62*	25.24*

HypAR$_e$ refers to the method with the explainability loss, HypAR$_{NARRE}$ to experiments with f_{NARRE}, and HypAR-MF with MF as the preference module.

We outperform both review-level recommenders, NARRE and HRDR, with both MF and LightGCN as preference modules, having up to 255% increase in performance using MAP. Using high-order connections is crucial for recommendation performance. While TriRank outperforms all non-graph-based methods, yet HypAR, NGCF, KGAT, and LightGCN outperform this method on all metrics. Thus, as the number of users and items increases, so does the running time and memory footprint, contrary to all other graph-based methods chosen here.

Of all methods, both LightGCN and HypAR consistently perform well across datasets. However, LigthGCN cannot explain its recommendations, motivating our model-agnostic explainability module, i.e., *we aim to maintain the prediction power of LigthGCN while also learning to explain its recommendations.* We see HypAR performing consistently better or similarly to LightGCN.

Explanation Quality. In Table 5, we have HypAR where gi, gu, w, i being the selection strategies greedy user, greedy item, weighted, and naïve selection using only item attention. Furthermore, γ is the AOS loss weight. The experiments on the quality of the explanations (Table 4) show that HypAR selects most often reviews of higher quality than both NARRE and HRDR. We often see a statistically significant increase in performance when using t-test over users (p-value of 0.05). Using explainability loss (Equation 3), we have a general increase in explainability performance but also recommendation. Given that our method outperforms the two baselines without the path restriction methodology, indicates that the HG structure is essential for the review explainablity. We perform

Table 5. The results of the graph selection.

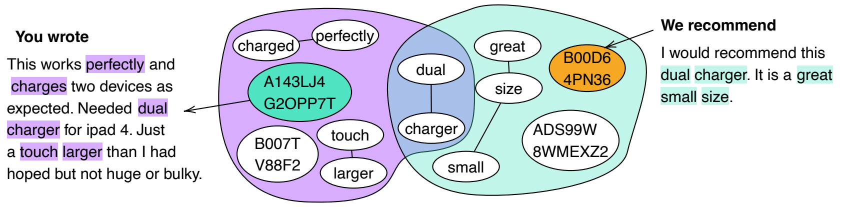
Method	Cellphone					Computer				
	Div	**F1**	**OC**	**Prec.**	**Recall**	**Div**	**F1**	**OC**	**Prec.**	**Recall**
KGAT	0.4851	0.1863	0.3610	0.1346	0.3598	0.4874	0.1612	0.3513	0.1141	0.3500
HypAR$_i$	0.3167	0.2627	0.3420	**0.2595**	0.3050	0.3163	0.2349	0.3183	**0.2325**	0.2829
HypAR$_{gi}$	0.5573	0.2893	0.5487	0.2110	0.5470	0.5274	0.2417	0.5169	0.1733	0.5145
HypAR$_{gu}$	0.5672	0.2892	0.5469	0.2111	0.5452	0.5634	0.2448	0.5081	0.1768	0.5056
HypAR$_w$	**0.5702**	0.2900	0.5483	0.2116	0.5465	**0.5707**	**0.2451**	0.5106	0.1767	0.5081
HypAR$_{gi}$ ($\gamma = .05$)	0.5644	**0.2916**	0.5495	0.2136	0.5472	0.5271	0.2417	**0.5171**	0.1730	**0.5149**
HypAR$_{gi}$ ($\gamma = .1$)	0.5568	0.2911	**0.5527**	0.2122	**0.5508**	0.5258	0.2421	0.5169	0.1735	0.5146
Improv %	17.53	56.50*	53.08*	92.77*	53.08*	17.10	52.11*	47.21*	103.75*	47.10*

slightly worse with the BERTScore metric; however, BERTScore suffers from the antonym embedding problem, where antonyms have similar embeddings due to similar contextual information [39,13].

We are outperforming KGAT (Table 5) on all metrics with statistical significance. KGAT selects a smaller subset of nodes, as seen with the diversity metric, even when selecting as many nodes as HypAR. The nodes selected are of less importance for the user, as KGAT only selects half as many relevant nodes as HypAR, which can be extrapolated from its recall. We see the path selection algorithm affect the performance; particularly, the performance between diversity, precision, and recall. Which to use therefore depends on the specific explainability scenario. When selecting a single review, we reduce the number of selected nodes and thereby increasing the precision at the cost of diversity and recall. Furthermore, in both explainability settings, using our path restriction methodology increases performance over the naïve attention mechanism.

Explainability Criteria. We test our method using the two explainability criteria defined in Section 3. Specifically, based on the top-5 recommended items for each user, we find the explanation ε of each item and all other items matching the explanation ε. We then conduct two studies comparing to: (i) a random ordering of items and (ii) items ranked using a different explanation of similar size. This evaluation works as a *litmus test* where we inspect the ability of the explanation to carry some information about the user preference beyond the specific item for which it was generated. For example, if we recommend item i_3 and we provide *great quality* instead of *fast shutter* as explanation, we would expect then that items with *great quality* are, on average, preferred to items with *fast shutter*. Any method not passing these criteria would not be explainable as the explanations would be indistinguishable from random explanation. This test is designed specifically to validate the explanation informativeness for set-based techniques as ours. Thus, these metrics and the results are not comparable to methods that select explanations based on other criteria, e.g., MostPop. On all datasets, we see a statistically significant increase over both the random ordering and the random explanation when using the average rank of the selected items. For example, for Cellphone, we have an average rank (lower is better) of 543.8, with random ordering having 674.5 and random explanation having 628.0 leading to a p-value less than < 0.01. As such, our method is, firstly, able to select a set of nodes that correlate to the actual ordering of items by our method, and secondly, the set of selected items correlates with our method's understanding of the user's preferences. Since our method outperforms the baselines, w.r.t.

Fig. 3. Real example from the Cellphone dataset.



You wrote

This works perfectly and charges two devices as expected. Needed dual charger for ipad 4. Just a touch larger than I had hoped but not huge or bulky.

We recommend

I would recommend this dual charger. It is a great small size.

the evaluation metrics in Table 4 and Table 5, and the explanations selected correlate with the HypAR's learned user preferences; we have strong evidence that the graph selection strategy is sound.

Case study. We present here an illustrative case study (Figure 3) by randomly selecting a user (`A143LJ4G2OPP7T`) and providing an explanation for the highest-ranked item (`B00D64PN36`). The extrapolated explanation would be that both products are dual-chargers based on the intersection; however, `B00D`... has a smaller size. This is due to the user's displeasure of `B007`...'s larger size, which makes `B00D`... likely preferable. As such, our method selects the relevant information for the user, and this can then be adopted for word-level highlighting of the parts of text in review of the recommended item.

5 Conclusion

In this work, we propose a novel model-agnostic review-based HG architecture for explainable recommendation. Our new graph model, HypAR, is based on review-induced hyperedges and illustrate its possible use cases. We demonstrate both the recommendation abilities of our method and the power of its explanations compared to existing review-based explanation methods. We show that HypAR either improves or maintains the performance of the underlying recommendation method we provide explanations for while improving the explanation quality compared to state-of-the-art methods; concluding that more attention is required for graph-based review explanations, as existing methods underperform. However, if the number of phrase-level sentences explodes, the graph view may become indigestible. Future work could therefore focus on pruning and highlighting the graph view for easy understanding and digestibility.

Acknowledgements. Katja Hose and Theis Jendal are supported by the Poul Due Jensen Foundation and the Independent Research Fund Denmark (DFF) under grant agreement no. DFF-8048-00051B. Hady W. Lauw and Trung-Hoang Le acknowledge that this research/project is supported by the National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG2-RP-2021-020).

Disclosure of interests. The authors have no competing interests to declare.

References

1. Adomavicius, G., Kwon, Y.: Improving aggregate recommendation diversity using ranking-based techniques. TKDE'12 **24**(5), 896–911 (2012)
2. Al-Taie, M.Z., Kadry, S.: Visualization of explanations in recommender systems. Journal of Advanced Management Science Vol **2**(2), 140–144 (2014)
3. Banerjee, S., Lavie, A.: METEOR: an automatic metric for MT evaluation with improved correlation with human judgments. In: WIEEMMTS'05. pp. 65–72 (2005)
4. Bauman, K., Liu, B., Tuzhilin, A.: Aspect based recommendations: Recommending items with the most valuable aspects based on user reviews. In: KDD'17
5. Beitzel, S.M., Jensen, E.C., Frieder, O.: MAP. In: Encyclopedia of Database Systems, Second Edition (2018)
6. Cao, Y., Wang, X., He, X., Hu, Z., Chua, T.: Unifying knowledge graph learning and recommendation: Towards a better understanding of user preferences. In: WWW'19. pp. 151–161 (2019)
7. Catherine, R., Cohen, W.W.: Transnets: Learning to transform for recommendation. In: Cremonesi, P., Ricci, F., Berkovsky, S., Tuzhilin, A. (eds.) Proceedings of the Eleventh ACM Conference on Recommender Systems, RecSys 2017, Como, Italy, August 27-31, 2017. pp. 288–296. ACM (2017)
8. Chen, C., Li, D., Yan, J., Huang, H., Yang, X.: Scalable and explainable 1-bit matrix completion via graph signal learning. In: AAAI'21. pp. 7011–7019 (2021)
9. Chen, C., Zhang, M., Liu, Y., Ma, S.: Neural attentional rating regression with review-level explanations. In: WWW'18. pp. 1583–1592 (2018)
10. Chen, X., Chen, H., Xu, H., Zhang, Y., Cao, Y., Qin, Z., Zha, H.: Personalized fashion recommendation with visual explanations based on multimodal attention network: Towards visually explainable recommendation. In: SIGIR'19. pp. 765–774
11. Cong, D., Zhao, Y., Qin, B., Han, Y., Zhang, M., Liu, A., Chen, N.: Hierarchical attention based neural network for explainable recommendation. In: ICMR'19
12. Dong, X., Ni, J., Cheng, W., Chen, Z., Zong, B., Song, D., Liu, Y., Chen, H., de Melo, G.: Asymmetrical hierarchical networks with attentive interactions for interpretable review-based recommendation. In: AAAI'20. pp. 7667–7674 (2020)
13. Etcheverry, M., Wonsever, D.: Unraveling antonym's word vectors through a siamese-like network. In: Korhonen, A., Traum, D.R., Màrquez, L. (eds.) Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers. pp. 3297–3307. Association for Computational Linguistics (2019). https://doi.org/10.18653/V1/P19-1319, `https://doi.org/10.18653/v1/p19-1319`
14. Feng, Y., You, H., Zhang, Z., Ji, R., Gao, Y.: Hypergraph neural networks. In: AAAI'19. pp. 3558–3565 (2019)
15. Flach, P.A.: ROC analysis. In: Encyclopedia of Machine Learning and Data Mining, pp. 1109–1116 (2017)
16. Gao, Y., Feng, Y., Ji, S., Ji, R.: Hgnn$^+$: General hypergraph neural networks. IEEE Trans. Pattern Anal. Mach. Intell. **45**(3), 3181–3199 (2023)
17. He, X., Chen, T., Kan, M., Chen, X.: Trirank: Review-aware explainable recommendation by modeling aspects. In: CIKM'15. pp. 1661–1670 (2015)
18. He, X., Deng, K., Wang, X., Li, Y., Zhang, Y., Wang, M.: Lightgcn: Simplifying and powering graph convolution network for recommendation. In: SIGIR'20. pp. 639–648 (2020)

19. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.: Neural collaborative filtering. In: WWW'17. pp. 173–182 (2017)
20. Huang, X., Fang, Q., Qian, S., Sang, J., Li, Y., Xu, C.: Explainable interaction-driven user modeling over knowledge graph for sequential recommendation. In: MM'19. pp. 548–556 (2019)
21. Järvelin, K., Kekäläinen, J.: Discounted cumulated gain. In: Encyclopedia of Database Systems, Second Edition (2018)
22. Koren, Y., Bell, R.M., Volinsky, C.: Matrix factorization techniques for recommender systems. Computer **42**(8), 30–37 (2009)
23. Le, T., Lauw, H.W.: Synthesizing aspect-driven recommendation explanations from reviews. In: IJCAI'20. pp. 2427–2434 (2020)
24. Li, Y., Chen, H., Sun, X., Sun, Z., Li, L., Cui, L., Yu, P.S., Xu, G.: Hyperbolic hypergraphs for sequential recommendation. In: CIKM'21. pp. 988–997 (2021)
25. Lin, C.Y.: ROUGE: A package for automatic evaluation of summaries. In: Text Summarization Branches Out. pp. 74–81 (Jul 2004)
26. Lin, Y., Liu, Z., Sun, M., Liu, Y., Zhu, X.: Learning entity and relation embeddings for knowledge graph completion. In: AAAI'15. pp. 2181–2187 (2015)
27. Liu, H., Wang, Y., Peng, Q., Wu, F., Gan, L., Pan, L., Jiao, P.: Hybrid neural recommendation with joint deep representation learning of ratings and reviews. Neurocomputing **374**, 77–85 (2020)
28. Liu, H., Wen, J., Jing, L., Yu, J., Zhang, X., Zhang, M.: In2rec: Influence-based interpretable recommendation. In: CIKM'19. pp. 1803–1812 (2019)
29. Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: ICLR'19
30. M K, V., K, K.: A survey on similarity measures in text mining. Machine Learning and Applications: An International Journal **3**, 19–28 (2016)
31. Ma, W., Zhang, M., Cao, Y., Jin, W., Wang, C., Liu, Y., Ma, S., Ren, X.: Jointly learning explainable rules for recommendation with knowledge graph. In: WWW'19. pp. 1210–1221 (2019)
32. McAuley, J.J., Targett, C., Shi, Q., van den Hengel, A.: Image-based recommendations on styles and substitutes. In: SIGIR'15. pp. 43–52 (2015)
33. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. In: ICLR'13 (2013)
34. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: ICML-10. pp. 807–814 (2010)
35. Pan, S., Li, D., Gu, H., Lu, T., Luo, X., Gu, N.: Accurate and explainable recommendation via review rationalization. In: WWW '22 (2022)
36. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: BPR: bayesian personalized ranking from implicit feedback. In: UAI'09. pp. 452–461 (2009)
37. Rendle, S., Krichene, W., Zhang, L., Anderson, J.R.: Neural collaborative filtering vs. matrix factorization revisited. In: RecSys'20. pp. 240–248 (2020)
38. Rong, G., Zhang, Y., Yang, L., Zhang, F., Kuang, H., Zhang, H.: Modeling review history for reviewer recommendation: A hypergraph approach. In: ICSE'22 (2022)
39. Saadany, H., Orăsan, C.: Bleu, meteor, bertscore: Evaluation of metrics performance in assessing critical translation errors in sentiment-oriented text. In: TRITON' 20'21 (2021)
40. Salah, A., Truong, Q., Lauw, H.W.: Cornac: A comparative framework for multimodal recommender systems. J. Mach. Learn. Res. **21**, 95:1–95:5 (2020)
41. Sánchez, L.Q., Sauer, C., Recio-García, J.A., Díaz-Agudo, B.: Make it personal: A social explanation system applied to group recommendations. Expert Syst. Appl. **76**, 36–48 (2017)

42. Sun, X., Yin, H., Liu, B., Chen, H., Cao, J., Shao, Y., Hung, N.Q.V.: Heterogeneous hypergraph embedding for graph classification. In: WSDM'21. pp. 725–733 (2021)
43. Ting, K.M.: Precision and recall. In: Encyclopedia of Machine Learning and Data Mining, pp. 990–991 (2017)
44. Wang, H., Zhang, F., Wang, J., Zhao, M., Li, W., Xie, X., Guo, M.: Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In: CIKM'18. pp. 417–426 (2018)
45. Wang, H., Zhang, F., Zhang, M., Leskovec, J., Zhao, M., Li, W., Wang, Z.: Knowledge-aware graph neural networks with label smoothness regularization for recommender systems. In: SIGKDD'19. pp. 968–977 (2019)
46. Wang, H., Zhao, M., Xie, X., Li, W., Guo, M.: Knowledge graph convolutional networks for recommender systems. In: WWW'19. pp. 3307–3313 (2019)
47. Wang, J., Zhang, Y., Wang, L., Hu, Y., Piao, X., Yin, B.: Multitask hypergraph convolutional networks: A heterogeneous traffic prediction framework. IEEE Trans. Intell. Transp. Syst. **23**(10), 18557–18567 (2022)
48. Wang, N., Wang, H., Jia, Y., Yin, Y.: Explainable recommendation via multi-task learning in opinionated text data. In: SIGIR'18. pp. 165–174 (2018)
49. Wang, X., He, X., Cao, Y., Liu, M., Chua, T.: KGAT: knowledge graph attention network for recommendation. In: SIGKDD'19. pp. 950–958 (2019)
50. Wang, X., He, X., Wang, M., Feng, F., Chua, T.: Neural graph collaborative filtering. In: SIGIR'19. pp. 165–174 (2019)
51. Wang, X., Wang, D., Xu, C., He, X., Cao, Y., Chua, T.: Explainable reasoning over knowledge graphs for recommendation. In: AAAI'19. pp. 5329–5336 (2019)
52. Wu, C., Wu, F., Liu, J., Huang, Y.: Hierarchical user and item representation with three-tier attention for recommendation. In: NAACL-HLT'19. pp. 1818–1826
53. Wu, C., Wu, F., Qi, T., Ge, S., Huang, Y., Xie, X.: Reviews meet graphs: Enhancing user and item representations for recommendation with hierarchical attentive graph neural network. In: EMNLP-IJCNLP'19 (2019)
54. Wu, L., Wang, D., Song, K., Feng, S., Zhang, Y., Yu, G.: Dual-view hypergraph neural networks for attributed graph learning. Knowl. Based Syst. (2021)
55. Wu, X., Chen, Q., Li, W., Xiao, Y., Hu, B.: Adahgnn: Adaptive hypergraph neural networks for multi-label image classification. In: MM'20. pp. 284–293 (2020)
56. Wu, Y., Ester, M.: FLAME: A probabilistic model combining aspect based opinion mining and collaborative filtering. In: WSDM'2015. pp. 199–208 (2015)
57. Wu, Y., Schuster, M., Chen, Z., Le, Q.V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Kaiser, L., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., Kurian, G., Patil, N., Wang, W., Young, C., Smith, J., Riesa, J., Rudnick, A., Vinyals, O., Corrado, G., Hughes, M., Dean, J.: Google's neural machine translation system: Bridging the gap between human and machine translation. CoRR (2016)
58. Xian, Y., Fu, Z., Muthukrishnan, S., de Melo, G., Zhang, Y.: Reinforcement knowledge graph reasoning for explainable recommendation. In: SIGIR'19
59. Yang, Z., Dong, S.: Hagerec: Hierarchical attention graph convolutional network incorporating knowledge graph for explainable recommendation. Knowl. Based Syst. **204**, 106194 (2020)
60. Yu, J., Yin, H., Li, J., Wang, Q., Hung, N.Q.V., Zhang, X.: Self-supervised multi-channel hypergraph convolutional network for social recommendation. In: WWW'21. pp. 413–424 (2021)
61. Zhang, T., Sun, C., Cheng, Z., Dong, X.: AENAR: an aspect-aware explainable neural attentional recommender model for rating predication. Expert Syst. Appl. **198**, 116717 (2022)

62. Zhang, T., Kishore, V., Wu, F., Weinberger, K.Q., Artzi, Y.: Bertscore: Evaluating text generation with BERT. In: ICLR (2020)
63. Zhang, Y., Lai, G., Zhang, M., Zhang, Y., Liu, Y., Ma, S.: Explicit factor models for explainable recommendation based on phrase-level sentiment analysis. In: SIGIR'14. pp. 83–92 (2014)