GAMMA: A Graph and Multi-view Memory Attention Mechanism for Top-N Heterogeneous Recommendation

M. Vijaikumar(✉), Shirish Shevade, and M. Narasimha Murty

Department of Computer Science and Automation, Indian Institute of Science, Bangalore, India
{vijaikumar,shirish,mnm}@iisc.ac.in

Abstract. Exploiting heterogeneous information networks (HIN) to top-N recommendation has been shown to alleviate the data sparsity problem present in recommendation systems. This requires careful effort in extracting relevant knowledge from HIN. However, existing models in this setting have the following shortcomings. Mainly, they are not end-to-end, which puts the burden on the system to first learn similarity or commuting matrix offline using some manually selected meta-paths before we train for the top-N recommendation objective. Further, they do not attentively extract user-specific information from HIN, which is essential for personalization. To address these challenges, we propose an end-to-end neural network model – GAMMA (Graph and Multi-view Memory Attention mechanism). We aim to replace the offline meta-path based similarity or commuting matrix computation with a graph attention mechanism. Besides, with different semantics of items in HIN, we propose a multi-view memory attention mechanism to learn more profound user-specific item views. Experiments on three real-world datasets demonstrate the effectiveness of our model for top-N recommendation setting.

Keywords: Heterogeneous information network · Recommendation systems · Memory attention network

1 Introduction

Due to exponential growth in the quantum of information available on the web, recommendation systems become inevitable in our day to day life. The objective of a top-N recommendation system is to come up with a ranked list of highly probable items that the user will interact in the future. Collaborative filtering (CF) techniques have been successful in modeling the top-N recommendation problem. Matrix factorization (MF) models [6,7,11] and the recently proposed neural network models such as [4,19] are a few instances of CF techniques.

However, a core strength of the CF models – *users' preferences are obtained from like-minded users' preferences through their historical records* – is also a major drawback. This happens because the users interact with a very few items as compared to available items in the system. This phenomenon leads to *data sparsity problem* in recommendation systems.

Several works have been proposed to alleviate this data sparsity problem by exploiting information coming from external sources. In particular, leveraging knowledge coming from heterogeneous information networks is gaining more attention recently.

Definition 1. ***Heterogeneous Information Network (HIN)*** *[14]. An information network is defined by a directed graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ with an entity type function $f_e : \mathcal{V} \rightarrow \mathcal{O}$ and relation type function $f_r : \mathcal{E} \rightarrow \mathcal{R}$, where $\mathcal{O}$ and $\mathcal{R}$ denote entity (object) type and relation (edge) type, respectively. Here, $\mathcal{V}$ denotes the set of entities and $\mathcal{E}$ denotes the set of relations between the entities. This Graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$ is called Heterogeneous Information Network (HIN) if $|\mathcal{O}| + |\mathcal{R}| > 2$.*

An example of a HIN is a network consisting of movies connected with 'actor' and 'genre' entities, and 'has' and 'belong to' relationships between them, respectively. In this work, we study top-N recommendation systems where items are involved in an external heterogeneous information network. We call the above setup top-N heterogeneous recommendation. Formally, we define this as

Problem Formulation: Top-N Heterogeneous Recommendation. Let $y_{(u,j)}$ be the rating that exists between user u and item j. Here, we consider the implicit rating setting, that is,

$$y_{(u,j)} = \begin{cases} 1, \text{ if } (u, j) \in \Omega \\ 0, \text{ otherwise} \end{cases}$$

where $\Omega = \{(u, j) : \text{user } u \text{ interacts with item } j\}$. In addition, a subset of items are involved in HIN, that is, $\mathcal{I} \cap \mathcal{V} \neq \phi$, where $\mathcal{I}$ denotes the set of items. The problem of top-N heterogeneous recommendation (short for top-N recommendation with heterogeneous information network) is to come up with a ranked list of highly probable items for each user by utilizing the associated HIN.

Recent advancements in deep neural networks have led to several models proposed for HIN based recommendation systems [3,5,12,22]. However, these models have the following shortcomings. First, they are not end-to-end – they rely on the tedious process of manual selection of meta-paths or meta-graphs, followed by offline embedding construction [12], similarity [3] or commuting matrix computation [22] for the entities in the HIN. Further, the proposed model in [3] uses similarity matrices which are inefficient in both memory and computation. Second, each user may look for different attributes of the items, for example, a user may decide to watch movies based on either the director or cast. We refer to subgraphs of a HIN associated with different attributes of items as different *views*. Therefore, the relevant information from the HIN should be extracted 'view-wise', according to each user's individual preferences. Third, users may look for more deeper characteristics such as *movies starring academy award-winning actors* (we call them *components*). In such cases, the model should not only be able to focus on the actors of the movie but also focus on whether they are academy award winners. This requires extraction of knowledge from the HIN 'component-wise' and such knowledge can potentially be used to explain why the recommendation system suggests a particular list of movies.

Contributions. To address the above challenges, we propose **GAMMA** – a Graph And Multi-view Memory Attention mechanism for the top-N heterogeneous recommendation problem. This is illustrated in Fig. 1. The novelty of our approach lies in making the model end-to-end – our approach does not require any offline similarity, commuting matrix computations or random-walk based embedding construction. Additionally, no manual selection of meta-paths is required. We achieve this by using graph attention networks – one for each view of the HIN. Further, we propose a multi-view multi-head memory attention layer, henceforth the *M3-layer*. The responsibility of this layer is to extract component-wise user-specific information. For example, it could extract information such as *actors who are academy-award winners*. Furthermore, we propose to use an attention mechanism to aggregate the knowledge coming from different views according to their influence. We conduct experiments on three real-world datasets and demonstrate the effectiveness of our model against several state-of-the-art models for top-N recommendation setting. Our implementation is available at https://github.com/mvijaikumar/GAMMA.

2 The Proposed Model

2.1 GAMMA

In this section, we persent our proposed model – GAMMA. The overall architecture is illustrated in Fig. 1. GAMMA has five building blocks. In what follows, we discuss them one by one in detail.

Embedding Construction from User-Item Interaction Matrix. This block is responsible for constructing embeddings for users and items from their interaction matrix. Let $P \in \mathbb{R}^{|\mathcal{U}| \times d}$ be the user embedding matrix and $Q \in \mathbb{R}^{|\mathcal{I}| \times d}$ be the item embedding matrix, where $|\mathcal{U}|$ and $|\mathcal{I}|$ denote number of users and items, and d denotes embedding dimension, respectively. Further, assume $x_u \in \mathbb{R}^{|\mathcal{U}|}$ and $z_j \in \mathbb{R}^{|\mathcal{I}|}$ be the one-hot encoding representations for user u and item j, respectively. We obtain the user embedding (p_u) and the item embedding (q_j) from P and Q as follows.

$$p_u = P^T x_u \text{ and } q_j = Q^T z_j. \tag{1}$$

In our proposed model, user embedding p_u has two roles to play. First, it learns the necessary user representations from user-item interactions. Second, it is used in further blocks as a query vector which extracts user-specific information from the HIN at both component- and view-levels.

Item-View Embedding Constructions via Graph Attention Mechanism (GAT Layers). This block is responsible for constructing initial view-wise embeddings for items from the HIN, each representing different views. To do this, we first extract different sub-graphs involving items with different entity types. For example, movies can be associated with both actors and genres. So, in this case, we construct two item-view sub-graphs – one with movie and actor nodes and the other with movie and genre nodes, as illustrated in Fig. 1.

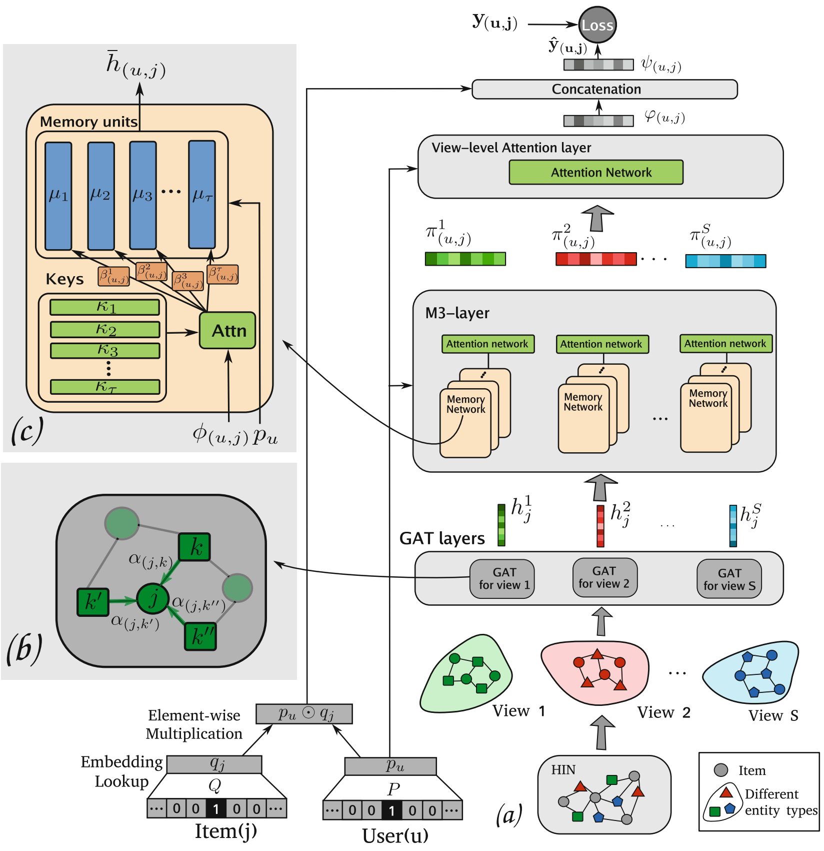

Fig. 1. The illustration of GAMMA architecture for top-N heterogeneous recommendation. Here, (a) provides overall architecture. The main components – (b) GAT layer, (c) memory network are given separately for more clarity. (Best viewed in colour.) (Color figure online)

Definition 2. ***Item-view sub-graph.*** *Formally, we define item-view sub-graph for each view (s) as* $\mathcal{G}_s(\mathcal{V}_s, \mathcal{E}_s)$ *where* $\mathcal{V}_s \subseteq \mathcal{V}$ *and* $\mathcal{E}_s \subseteq \mathcal{E}$. *Here,* $\mathcal{I} \cap \mathcal{V}_s \neq \phi, \forall s$.

Further, individually on these sub-graphs, we employ a multi-layer graph attention mechanism (GAT [16]) to construct first-level view-wise embeddings for the items. Let W_G^i and c_G^i be the weight matrix and vector associated with layer i and $\mathcal{N}(j)$ be a set of neighbors for the node j. The representations (f_j^i for item j at layer i) for the items involved in the sub-graphs are learned as follows.

$$f_j^i = \sum_{k \in \mathcal{N}(j)} \alpha_{(j,k)}^i W_G^i f_k^{i-1},$$
$$\text{where} \quad \alpha_{(j,k)}^i = \frac{\exp(a(c_G^i \cdot [W_G^i f_j^{i-1} \| W_G^i f_k^{i-1}]))}{\sum_{k' \in \mathcal{N}(j)} \exp(a(c_G^i \cdot [W_G^i f_j^{i-1} \| W_G^i f_{k'}^{i-1}]))}. \tag{2}$$

Here, $\|$ denotes concatenation operation, $f_k^0 = z_k$ (one-hot item representation), and $\alpha_{(j,k)}^i$ denotes influence value for node k on node j at layer i. At each GAT layer, we have multiple attention heads acting on the node representations. Then, we concatenate the embeddings coming from different heads. For ease of explanation, we represent the final embedding from GAT layers by h_j^s – the embedding associated with view s for item j.

Therefore, we obtain multiple embeddings for the items – one from each sub-graph. These provide us different views $\{h_j^s\}_{s=1}^S$ for the item j where S denotes the total number of views. We compactly represent the embeddings obtained for all the nodes for single view as $\mathbf{h}^s = [h_1^s, h_2^s, ..., h_n^s]$.

Multi-view Multi-head Memory Attention Layer (M3-layer). M3-layer takes embedding $\{\mathbf{h}^s\}_{s=1}^S$ obtained from the previous GAT layers as input, extracts attentively user specific components from each view s, individually. As we discussed earlier, the user may prefer to watch movies acted by an academy award-winning actor. Hence, keeping that into consideration, the goal of this layer is to capture such information using multiple memory attention networks followed by aggregating such information using the attention network. This is done for each view separately.

During this process, we use multiple memory networks for each view to capture the different notions of influential components. This can be intuitively thought of as different filters acting on the different parts of the input. Further, we utilize attention mechanisms to aggregate these constructed user-specific item embeddings for each view, as illustrated in Fig. 1. We first explain what happens in one view and one memory attention network. The same procedure is followed in other views and different memory attention heads, respectively.

Let $\mu = \{\mu_1, \mu_2, ..., \mu_\tau\}$, where $\mu_i \in \mathbb{R}^d$ and $\kappa = \{\kappa_1, \kappa_2, ..., \kappa_\tau\}$, where $\kappa_i \in \mathbb{R}^d$ be memory components and the corresponding keys, respectively. Here, τ denotes the total number of memory units. First, we get normalized user and item-view representation ($\phi_{(u,j)}$) as,

$$\phi_{(u,j)} = \frac{p_u \odot h_j}{\|p_u\| \|h_j\|}, \tag{3}$$

where $\phi_{(u,j)} \in \mathbb{R}^d$ and $\odot$ denotes Hadamard product. This is then combined with the keys (κ) to provide the required influential (attention) values,

$$\beta^t_{(u,j)} = \frac{\exp(\phi_{(u,j)} \cdot \kappa_t)}{\sum_{t'=1}^{\tau} \exp(\phi_{(u,j)} \cdot \kappa_{t'})}. \tag{4}$$

These influence values provide the influence score of how much each component contributes to the user's interest. They are then used along with the memory units that provide user-specific item views. This is done as follows. First, we extract the required information ($\bar{\mu}^t_j$) from item views using memory units and then we obtain the item representation ($\bar{h}_{(u,j)}$) as,

$$\bar{h}_{(u,j)} = \sum_{t=1}^{\tau} \beta^t_{(u,j)} \bar{\mu}^t_j, \text{ where } \bar{\mu}^t_j = \mu_t \odot h_j. \tag{5}$$

Here $\bar{h}_{(u,j)} \in \mathbb{R}^d$ denotes the user-specific item-view representation from a single memory network. We get multiple such representations from multiple views and multiple heads (indexed by γ) given as $\bar{h}^{(s,\gamma)}_{(u,j)}$. We then aggregate the representations view-wise (into $\pi^s_{(u,j)}$) as follows,

$$\pi^s_{(u,j)} = \mathcal{A}(\{\bar{h}^{(s,\gamma)}_{(u,j)}\}^{\Gamma}_{\gamma=1}, p_u), \tag{6}$$

where Γ denotes the total number of memory networks and $\mathcal{A}(\cdot)$ denotes attention network [9]. This is defined as,

$$\begin{gathered} \mathcal{A}(p_u, \{x^\gamma\}^{\Gamma}_{\gamma=1}) = \sum_{\gamma=1}^{\Gamma} \zeta_{u\gamma} x_{u\gamma}, \text{ where} \\ \zeta_{u\gamma} = \frac{\exp(\text{score}(p_u, x^\gamma))}{\sum_{\gamma'=1}^{\Gamma} \exp(\text{score}(x^{\gamma'}, p_u))} \text{ and } \text{score}(p_u, x^\gamma) = p_u^T W_{\mathcal{A}} x^\gamma. \end{gathered} \tag{7}$$

Here, x^γ is a vector, $\zeta_{u\gamma}$ denotes influence value of γ on user u and $W_{\mathcal{A}}$ is weight matrix associated with the attention network. Note that, memory units and keys are initialized randomly and the weights are shared across the inputs.

View-Level Attention Layer. The previous layer provides the item embeddings ($\{\pi^s_{(u,j)}\}^S_{s=1}$) concerning different views, consisting of important user-specific information extracted, components-wise. The purpose of this layer is to aggregate the information present in item embeddings, view-wise. Different users get influenced by different views of items, for example, the genre of the movie or actors in the movie. Hence, we combine such different views attentively as follows,

$$\varphi_{(u,j)} = \mathcal{A}(p_u, \{\pi^s_{(u,j)}\}^S_{s=1}), \tag{8}$$

where $\varphi_{(u,j)}$ is a resultant item-view representation extracted from HIN.

Prediction Layer. This layer is responsible for gathering information coming from the interaction matrix and HIN network and provides the predicted probability that a user may interact with each item in the future. It first concatenates representations coming from the user-item interactions and the HIN. We then pass these representations through a sequence of fully connected layers and finally, we obtain the predictive probability as

$$\begin{aligned} \hat{y}_{(u,j)} &= g(\psi_{(u,j)}), \\ \text{where } \psi_{(u,j)} &= (p_u \odot q_j) \| (p_u \odot \varphi_{(u,j)}). \end{aligned} \tag{9}$$

Here $g(x) = a(\cdots a(W_2 a(W_1 x + b_1) + b_2) \cdots)$ denotes a fully connected network, $a(\cdot)$ is an activation function and W_k and b_k are weight matrices and bias vectors, respectively.

2.2 Loss Function

Since our ratings are implicit (binary), one can use point-wise loss functions such as cross-entropy [3,4] or pair-wise loss functions such as BPR-loss [11]. In this work, we use cross-entropy loss function and the final optimization problem is,

$$\min_{\mathcal{W}} \ \mathcal{L}(\mathcal{W}) = - \sum_{(u,j) \in \mathcal{D}} y_{(u,j)} \ln \hat{y}_{(u,j)} + (1 - y_{(u,j)}) \ln(1 - \hat{y}_{(u,j)}) + \lambda \ \mathcal{R}(\mathcal{W}), \tag{10}$$

where $\mathcal{W}$ consists of all the model parameters, $\mathcal{R}(\cdot)$ is a regularizer and λ is a non-negative hyperparameter. We employ negative sampling strategy during training. Here, $\mathcal{D} = \mathcal{D}^+ \cup \mathcal{D}^-_{samp}$ where $\mathcal{D}^+ := \{(u,j) \in \Omega\}$ and $\mathcal{D}^-_{samp} \subset \{(u,j') \notin \Omega\}$, and $\mathcal{D}^-_{samp}$ is obtained using a negative sampling procedure [10].

3 Experiments

3.1 Experimental Settings

Datasets. We use three real-world datasets – Amazon, Yelp and MovieLens – for our experiments.[1] Besides the user-item interactions, the datasets contain HINs where items are involved. In particular, the HIN associated with the Amazon dataset has a co-viewed network, brand and category information for the products, the HIN associated with the Yelp dataset has category and city information for the businesses and the HIN associated with MovieLens dataset has actor, director, country and genre information for the movies, respectively. The MovieLens dataset has ratings in the range [0.5–5] and the other datasets have ratings in the range [1–5], respectively. We do the following pre-processing, as

[1] https://github.com/librahu/HIN-Datasets-for-Recommendation-and-Network-Embedding.

done in [3,4]. That is, (1) we retain the ratings more than 3 for the Amazon and Yelp datasets and 3.5 for the MovieLens dataset, and treat them as positive interactions, and (2) we retain users and items having more than 10 ratings for the Amazon and MovieLens datasets and more than 5 for the Yelp dataset, respectively. The statistics of the datasets are given in Table 1.

Metrics and Evaluation Procedure. We adopt two top-N recommendation metrics (ranking metrics) – hit ratio (HR@N) and normalized discounted cumulative gain (NDCG@N) for the performance comparison. Further, following [3,4], we split the dataset into train, validation and test sets where we hold-out one randomly selected item for each user for validation and test set, respectively. Since it is difficult to rank all the available items for each user during the evaluation, we sample 50 non-interacted items for each user to compare with the hold-out item in the validation and test set. We repeat this procedure five times and obtain five different splits. We report the mean and standard deviation of these five splits as the final result.

Table 1. Dataset statistics

Dataset	# Users	# Items	# Entities in HIN	# Connections in HIN	# Ratings
Amazon	4365	2617	5625	10324	131167
MovieLens	1761	1036	4301	13661	112616
Yelp	4511	3862	4406	15556	96356

Comparison Models. We evaluate our proposed model with the following models. Note that BPR, MF, GMF and NeuMF are rating-only models, and linear and non-linear variants of HERec and NeuACF are HIN based models for recommendation task.

- **NeuACF** [3] is a state-of-the-art model for top-N heterogeneous recommendation setting. NeuACF is a two-stage approach. In the first stage, it computes user-user and item-item similarity matrices offline using some manually selected meta-paths. In the second stage, it utilizes these similarity matrices along with the user-item rating matrix for prediction.
- **HERec** [12] is a recently proposed two-stage approach for heterogeneous recommendation setting. In the first stage, it computes user and item embeddings offline using a meta-path based random walk and skip-gram technique [10]. In the second stage, it leverages embeddings learned for users and items for the recommendation tasks. We include the two proposed variants of HERec – HERec (linear) and HERec (non-linear) for comparison.
- **NeuMF** [4] is one of the state-of-the-art models for rating-only top-N recommendation setting. NeuMF is a fusion between MF and a neural network approach. Further, **GMF** is proposed as a part of NeuMF.

- **MF** [6] is a standard and well-known baseline for the recommendation task.
- **BPR** [11] is a standard baseline for the top-N recommendation setting.

FMG [22] is another recently proposed model that utilizes commuting matrices learned from meta-graph based strategy. Since it has been shown in [3] that NeuACF outperforms FMG, we omit this from the comparison.

Hyperparameter Setting and Reproducibility. We implement our model using Python and TensorFlow 1.14. We tune the hyperparameters using the validation set and report the corresponding test set performance. From the validation set performance, we set the embedding dimension (d) to 32 for the Amazon and Yelp datasets, and 64 for the MovieLens dataset. We set the learning rate to 0.002, the number of negative samples to 3, the mini-batch size to 2048 with Xavier initialization, use an RMSprop optimizer, and a dropout of 0.5. We use a 2-layer GAT network with 6 and 4 attention heads in the first and second layers, respectively. Further, we set the number of heads (Γ) in M3-layer to 4, and the number of memory units (T) to 8, respectively. We tune and set the hyperparameters for the baselines according to the respective papers.

Table 2. Overall performance of different models on three real-world datasets – Amazon, MovieLens and Yelp (given in **HR@5**).

Model	Amazon	MovieLens	Yelp
MF [6]	0.4976 ± 0.0056	0.6579 ± 0.0045	0.5109 ± 0.0102
BPR [11]	0.5079 ± 0.0071	0.6516 ± 0.0046	0.5107 ± 0.0064
GMF [4]	0.5235 ± 0.0049	0.6757 ± 0.0078	0.5307 ± 0.0079
NeuMF [4]	0.5366 ± 0.0063	0.7025 ± 0.0056	0.5396 ± 0.0060
HERec (linear) [12]	0.5359 ± 0.0079	0.6927 ± 0.0040	0.5279 ± 0.0040
HERec (non-linear) [12]	0.5280 ± 0.0066	0.6947 ± 0.0072	0.5306 ± 0.0070
NeuACF [3]	0.5520 ± 0.0042	0.7054 ± 0.0108	**0.5678 ± 0.0124**
GAMMA (ours)	**0.5593 ± 0.0060**	**0.7171 ± 0.0117**	0.5579 ± 0.0067

Table 3. Overall performance of different models on three real-world datasets – Amazon, MovieLens and Yelp (given in **NDCG@5**).

Model	Amazon	MovieLens	Yelp
MF [6]	0.3467 ± 0.0052	0.4717 ± 0.0039	0.3568 ± 0.0073
BPR [11]	0.3585 ± 0.0022	0.4765 ± 0.0033	0.3565 ± 0.0035
GMF [4]	0.3664 ± 0.0040	0.5062 ± 0.0062	0.3768 ± 0.0056
NeuMF [4]	0.3782 ± 0.0057	0.5233 ± 0.0031	0.3808 ± 0.0028
HERec (linear) [12]	0.3816 ± 0.0071	0.5081 ± 0.0051	0.3741 ± 0.0013
HERec (non-linear) [12]	0.3760 ± 0.0054	0.5116 ± 0.0057	0.3764 ± 0.0049
NeuACF [3]	0.3992 ± 0.0041	0.5350 ± 0.0061	**0.4061 ± 0.0089**
GAMMA (ours)	**0.4049 ± 0.0028**	**0.5461 ± 0.0056**	0.3979 ± 0.0052

3.2 Experimental Results and Discussion

Overall Performance. We present the overall performance of our model – GAMMA in Table 2 and Table 3. Here, we conduct paired t-test and the improvements obtained here are statistically significant with $p < 0.01$. Note that HERec and NeuACF heavily utilize the embeddings constructed offline, while GAMMA does not use any such offline embeddings. As we can see from Table 2 and Table 3, our model outperforms the HIN based recommendation models as well as the rating-only models on the Amazon and the MovieLens datasets and performs comparably with NeuACF on the Yelp dataset. From this, we conclude that the GAT mechanism along with user-specific knowledge extraction from the HIN can effectively replace the tedious offline embedding construction strategies.

Performance Against Different Sparsity Levels. Here, we experiment to study the performance of the models for different sparsity levels. For this, we take

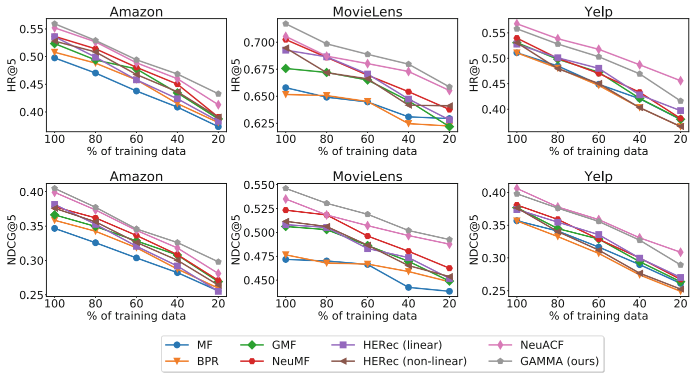

Fig. 2. Performance comparison of different models against different sparsity levels on the datasets: Amazon, MovieLens and Yelp. Here, the mean values obtained from the five different splits for each sparsity level are reported.

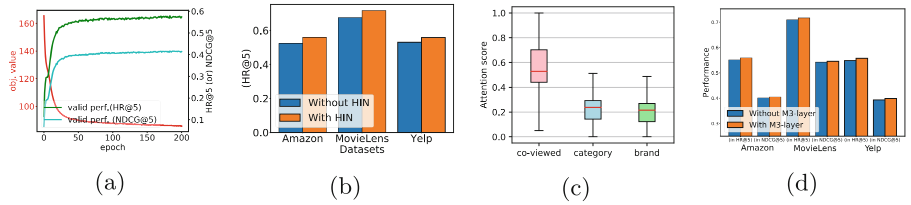

Fig. 3. (a) Objective function value, performance of GAMMA on validation set (in HR@5 and NDCG@5) against number of training epochs. (b) Performance comparison when HIN is ignored vs HIN is used. (c) Distribution of attention scores learned by GAMMA for different views on the Amazon dataset. Here, red-colored line inside the box signifies the median. (d) Performance comparison when M3-layer is not included vs M3-layer is included. (Best viewed in color.) (Color figure online)

the full training set, and at each level, we remove 20% of the interactions. This is illustrated in Fig. 2. Here, the x-axis denotes the percentage of the training data used, and the y-axis denotes the performance in HR@5 or NDCG@5. From this study, we observe that despite the varying sparsity levels, the performance of our model is consistent across different datasets.

Figure 3(a) shows the objective function value and the validation set performance (in HR@5 and NDCG@5) against the number of training epochs. Further, Fig. 3(b) illustrates the performance comparison of GAMMA when the HIN is ignored vs the HIN is included. The performance improvement shows that our proposed approach effectively utilizes the HIN to improve the top-N recommendations. Figure 3(c) demonstrates the attention score distribution for different views on the Amazon dataset. This indicates that the influence of different views on the recommendation tasks varies, and attentively selecting information coming from different views is essential. Further, Fig. 3(d) illustrates the performance of GAMMA when the M3-layer is not included vs the M3-layer is included. From this, we observe that incorporating the M3-layer indeed helps in improving the performance.

4 Related Work

Early CF techniques are mainly based on matrix factorization models – BPR [11] and MF [6] – and their extensions [7]. In recent years, due to their rich representation capabilities, several neural networks and deep learning models are developed for top-N recommendation settings [21]. For instance, NeuMF [4] combines MF with a multi-layer perceptron to learn better representations for users and items through MF as well as neural networks. Further, autoencoders and variational autoencoders [8] and graph neural network-based models [19] have also been proposed for recommendation systems. Nevertheless, since most of these models entirely rely on past interactions between users and items, their performance is mostly affected when sparsity increases. Recently, to mitigate the sparsity issues, there is an increasing interest in leveraging knowledge from the HIN. Our work falls under this category.

Early work on heterogeneous recommendation incorporate knowledge extracted from the HIN to MF models. For instance, HeteRec [20] uses meta-path based similarity construction followed by the Bayesian ranking optimization technique for the top-N recommendation. Besides, SemRec [13] employs weighted meta-paths to prioritize and personalize user preferences on different paths. Recently, due to its ability to effectively extract relevant information, employing attention mechanism [9,18] to recommendation systems is gaining ground [2,15]. Further, the memory attention network has been employed in the context of top-N social [1] and multi-media [2] recommendation tasks. In terms of utilizing multiple views (aspects) for the items from the HIN, our work is related to NeuACF [3], MCRec [5], HERec [12], KGAT [17] and FMG [22]. For instance, NeuACF [3] follows a two-stage approach – in the first stage, it constructs similarity matrices. In the second stage, these similarity matrices are used for learning deeper representations of the users and items. KGAT [17] incorporates information from the knowledge graph using graph neural networks. Further, Hu *et al.* [5] proposes a three-way neural interaction model with co-attention for the top-N heterogeneous recommendation. In place of meta-paths, FMG [22] proposes meta-graphs for extracting knowledge from HIN. Further, Shi *et al.* [12] propose HERec that fuses meta-path based embeddings into extended MF model.

5 Conclusion

In this work, we proposed a graph and memory attention-based neural network model – GAMMA for top-N recommendation systems. The proposed technique replaces the tedious process of computing similarity or commuting matrix offline with a graph attention mechanism. This makes the whole procedure end-to-end. Further, we proposed a multi-view multi-head memory attention layer to extract fine-grained user-specific information using a memory attention network. The proposed model is general, and it can easily be extended to scenarios where both users and items are involved in the HIN. Extensive experiments on three real-world datasets demonstrated the effectiveness of our model over state-of-the-art top-N recommendation models.

References

1. Chen, C., Zhang, M., Liu, Y., Ma, S.: Social attentional memory network: modeling aspect-and friend-level differences in recommendation. In: WSDM. ACM (2019)
2. Chen, J., Zhang, H., He, X., Nie, L., Liu, W., Chua, T.S.: Attentive collaborative filtering: multimedia recommendation with item-and component-level attention. In: SIGIR, pp. 335–344. ACM (2017)
3. Han, X., Shi, C., Wang, S., Philip, S.Y., Song, L.: Aspect-level deep collaborative filtering via heterogeneous information networks. In: IJCAI, pp. 3393–3399 (2018)
4. He, X., Liao, L., Zhang, H., Nie, L., Hu, X., Chua, T.S.: Neural collaborative filtering. In: WWW, pp. 173–182 (2017)
5. Hu, B., Shi, C., Zhao, W.X., Yu, P.S.: Leveraging meta-path based context for top-n recommendation with a neural co-attention model. In: SIGKDD. ACM (2018)
6. Hu, Y., Koren, Y., Volinsky, C.: Collaborative filtering for implicit feedback datasets. In: ICDM, pp. 263–272. IEEE (2008)
7. Koren, Y., Bell, R.: Advances in collaborative filtering. In: Ricci, F., Rokach, L., Shapira, B. (eds.) Recommender Systems Handbook, pp. 77–118. Springer, Boston (2015). https://doi.org/10.1007/978-1-4899-7637-6_3
8. Liang, D., Krishnan, R.G., Hoffman, M.D., Jebara, T.: Variational autoencoders for collaborative filtering. In: WWW, pp. 689–698 (2018)
9. Luong, M.T., Pham, H., Manning, C.D.: Effective approaches to attention-based neural machine translation. In: EMNLP, pp. 1412–1421. ACL (2015)
10. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in neural information processing systems, pp. 3111–3119 (2013)

11. Rendle, S., Freudenthaler, C., Gantner, Z., Schmidt-Thieme, L.: Bpr: bayesian personalized ranking from implicit feedback. In: UAI, pp. 452–461. AUAI (2009)
12. Shi, C., Hu, B., Zhao, W.X., Philip, S.Y.: Heterogeneous information network embedding for recommendation. In: TKDE. vol. 31, pp. 357–370. IEEE (2018)
13. Shi, C., Zhang, Z., Luo, P., Yu, P.S., Yue, Y., Wu, B.: Semantic path based personalized recommendation on weighted heterogeneous information networks. In: CIKM, pp. 453–462. ACM (2015)
14. Sun, Y., Han, J.: Mining heterogeneous information networks: a structural analysis approach. ACM SIGKDD Explor. Newslett. **14**(2), 20–28 (2013)
15. Tay, Y., Luu, A.T., Hui, S.C.: Multi-pointer co-attention networks for recommendation. In: SIGKDD, pp. 2309–2318. ACM (2018)
16. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. In: ICLR (2018)
17. Wang, X., He, X., Cao, Y., Liu, M., Chua, T.S.: Kgat: knowledge graph attention network for recommendation. In: SIGKDD, pp. 950–958 (2019)
18. Wang, X., et al.: Heterogeneous graph attention network. In: WWW, pp. 2022–2032 (2019)
19. XWang, X., He, X., Wang, M., Feng, F., Chua, T.: Neural graph collaborative filtering. In: SIGIR, pp. 165–174. ACM (2019)
20. Yu, X., et al.: Recommendation in heterogeneous information networks with implicit user feedback. In: RecSys, pp. 347–350. ACM (2013)
21. Zhang, S., Yao, L., Sun, A., Tay, Y.: Deep learning based recommender system: a survey and new perspectives. ACM Comput. Surv. (CSUR) **52**(1), 5 (2019)
22. Zhao, H., Yao, Q., Li, J., Song, Y., Lee, D.L.: Meta-graph based recommendation fusion over heterogeneous information networks. In: SIGKDD. ACM (2017)