

## RESEARCH ARTICLE

# CrossDomain Recommendation Based on MetaData Using Graph Convolution Networks

RABIA KHAN<sup>1</sup>, NAIMA ILTAF<sup>1</sup>, RABIA LATIF<sup>2</sup>, AND NOR SHAHIDA MOHD JAMAIL<sup>2</sup><sup>1</sup>Department of Computer Software Engineering, National University of Sciences and Technology, Islamabad 46000, Pakistan<sup>2</sup>Artificial Intelligence and Data Analytics Laboratory, College of Computer and Information Sciences (CCIS), Prince Sultan University, Riyadh 11586, Saudi Arabia

Corresponding author: Naima Iltaf (naima@mcs.edu.pk)


This work was supported by Prince Sultan University for paying the Article Processing Charges (APC).

**ABSTRACT** Recent advancements in the domain of recommender systems have stemmed from the inspiration of representing the user-item interaction into graphs. These heterogeneous graphs comprehensively capture the non-linear relationships between users and items alongwith features and emneddings. Graph Convolution Networks (GCNs) are state-of-the-art graph-based learning models that learn and represent the graph structures by recursively stacking layers of convolution and non-linear activation operations. GCNs are augmented with the strength of deep learning paradigms resulting in achieving better performance as compared to traditional CF (Collaborative Filtering) methods. Despite modern improvements in the domain of recommender systems, cold-start users are a daunting challenge in the design of recommender systems since the conventional recommendation services are based on solely one data source. During the recent years, cross-domain recommendation methods have gained popularity because of the availability of information in multiple domains for cold start users. We supplement this information by utilizing the data contained in the metadata of users alongwith the strength of modelling graphs using GCNs. Our proposed algorithm seams the strength of GCNs with cross-domain paradigm utilizing the richness in metadata in user's feedback to overcome the sparsity in user-item rating matrix. The combined advantages of GCNs and cross-domain approaches alleviated the issues of cold-start users by transferring user preferences from a source domain to a target domain.

**INDEX TERMS** Cross-domain recommendation, collaborative filtering, graph based representation, deep learning, coldstart users, graph convolution networks.

## I. INTRODUCTION

The volume of data introduced everyday into the world of information since the past few decades has made it a digital global village. Perhaps the most dramatic outcome of this digital revolution is the amount of data that's now available. The ever growing information has given birth to the need of Recommender Systems (RS). These systems help in filtering this information and recommending only a subset of choices pertaining to user's interest. Recommender system find its application in nearly all walks of practical life, for instance, in the field of educational data science,

The associate editor coordinating the review of this manuscript and approving it for publication was Fabrizio Messina .

making recommendations to tourists, suggesting literature to an online reader, recommending technical gadgets based on purchase history of a user, biomedical image analysis [12] or identifying deceptive images created by forgery [6]. There are many recommendation systems now that take demographics alongwith user's historical record to make recommendations. LAPTA [13] proposes location-aware personalized traveler assistance service based on user preferences and the global positioning system (GPS). Last decade has witnessed tremendous work in the world of recommender systems which includes traditional techniques like Collaborative Filtering (CF) and Content-based Filtering (CB) [9]. Traditional techniques are based on the principle of gauging user's interest by extrapolating user's preferences in the content he has been

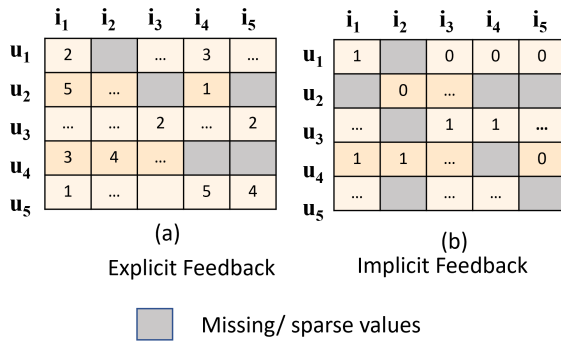


FIGURE 1. Sparsity in user-item rating matrix.

showing interest. CF has a strength of not depending on the content of the items or meta-data for recommendation, rather it operates on implicit or explicit feedback of the user. Implicit feedback means inferring user's preferences by user's interaction with the content. Number of page visits, number of clicks, observing user's browsing history, analyzing the record of online purchases made etc are few examples of implicit feedback sources. On the contrary, explicit feedback includes getting feedback or rating explicitly on an item or asking the user to rank a collection of items from most liked to least liked etc. CF can be perceived as building a user's profile according to its behaviour (either implicit or explicit). Despite promising results shown by CF and CB, they suffer from the issues of data sparsity, cold-start user, scalability etc. The exponential growth in the number of online users interacting with items makes it nearly impossible to acquire user's feedback explicitly on every item it comes across with which results in sparse user-item rating matrix as shown in Figure 1. To solve issue of cold-start, B. Hawashin [34] proposed to implicitly learn user's interests through patterns using machine learning instead of solely focusing on user-item ratings. A paradigm of techniques that learn personality traits to alleviate cold-start problem has attracted significant attention lately. Hu and Pu et.al [35] dealt with cold-start problem by incorporating personality traits with the traditional CF technique. The latest research by Han et.al. [36] proposed a personality interpretable model that uses machine learning algorithms to classify personalities in a semi-supervised manner.

CF finds its application in design of a lot of recommendation techniques when combined with neural network (e.g. RNN, CNN etc). CF and CB suffer degradation in performance due to sparsity and cold start. Since the world we exist in can be perceived as a real manifestation of a graph structure, therefore the learning of the associations between nodes can be learnt by applying graph learning techniques [18]. The users and items are the object nodes, and the items we show interest in are further linked to other users which can be visualized as a massive graph structure. Such characteristics are even more relevant in RS where the objects (users and items) are linked with each other through various connections [5].

This perception has led graph learning to emerge as one of the most distinctive domain in many recommendation scenarios owing to its great prospective in deriving knowledge embedded in graphs. Graph Learning-based Recommender Systems (GLRS) have been comprehensively explored in the recent times [20]. GCNs have particularly gained popularity due to their ability to iteratively aggregate feature information from the local neighbourhood of a node using neural networks [25]. This is graphically illustrated in Figure 2. However, this can result into over generalization due to overwhelming size of graph structure. An attention mechanism is a possible solution that weighs the significance of a node in a neighbourhood. Owing to the strengths of deep learning and GCN, we propose a crossdomain recommendation framework using MetaData based on Graph Convolution Networks (MD-GCN).

The major contributions of this paper can be summarized as:

- This proposed recommendation framework encodes similarity and interaction information as collaborative signal and further employs graph neural technique for embedding generation.
- This proposed model improvises the text embedding approach using *GloVe* embeddings by fusing the strengths of CNN and LSTM models thereby generating the features of metadata comprehensively.
- Augmenting the embedding function with attention mechanism for overlapping users in both domains to ensure data richness and diversity. The overlapping users serve as a bridge for both domains.

The rest of this paper is organized as follows. Section II describes the related work in this domain. Section III explains the details of our proposed recommendation. Experimental results are presented and analysed in Section IV. Finally, Section V concludes the paper.

## II. RELATED WORK

### A. CONVENTIONAL RECOMMENDATION

Conventional recommender systems can be mainly classified into two types: content-based filtering (CB) and collaborative filtering (CF) based recommender systems [20]. Evident from the name, the content based recommendation needs content of the choices made by the users. During the course of time, the choices made by the user consolidate the preferences he has in terms of content. This can be thought of as a profile of a user signifying his interests [20]. Later on, this profile is used in content based recommendation algorithms to make similar suggestions [30]. The formulation of this profile involves the application of many data mining and machine learning techniques. A contrasting technique focuses on finding the pattern based on the choices that the users makes. This paradigm of technique is called collaborative filtering [10]. It is guided by the motivation that a user's preference can be guessed from the collaborative preference of like-minded users. This like-mindedness is inferred from the ratings of the items that

the user records. Both CB and CF suffer from discrepancy in results when there is no data to formulate the profile of the user because of the introduction of a new user to the inventory. Such a user is termed as a ‘cold start user’. The most popular techniques of CF is matrix factorization (MF) [15], [24], in which latent embeddings are computed to model a pattern that generates ratings.

### B. DEEP LEARNING NEURAL NETWORKS FOR RECOMMENDATION

Recent advancements in machine learning (ML), specifically Deep Learning (DL), witnessed a paradigm shift in recommendation architectures, primarily due to seamless correspondence between natural language processing and recommender frameworks [3], [5]. The capability of identifying various patterns and occurrences renders DL the most suited for future applications in realm of personalized content and product recommendations. Complex problems are effectively solved using DL’s core principle of learning deep interpretations from data at multiple levels of representations and abstractions, for modelling non-linear interactions in the data, with non-linear activations. This is accomplished by discovering hidden patterns in the data and understanding the deep / intricate relationships between many interdependent variables. The strength of DL lies in it’s to identify and learn hidden patterns from the data, correlate them and enhance the rules for decisions.

Neural Networks have gained prominence with the enormous spread of social networks and knowledge bases since they have been in use for representation of graph-structured data. Neural Networks allow the application of DL techniques, to learn about the regular as well as irregular / arbitrary structure of the graphs, thus giving rise to the term **deep neural networks** [1], [2]. Numerous variants of deep neural networks (AE, CNN, RNN, GNN) are available to boost performance of recommendations by focussing on different areas. AE techniques are employed for unsupervised ML tasks, where the primary aim is to learn data approximations, representations, and encodings. CNNs are predominantly used for extraction of latent factors and features, especially from images and text due to their capability of reducing the number of parameters without loss of quality. This is achieved by their ability to share weights and local connections. RNNs are very effective for problems that are sequence-based in nature since they can process time series data and make predictions about the next output using patterns. Speech recognition and language modelling are common domains for these artificial intelligent techniques. PLSPL(Personalized Long Short-term Preference Learning) [37] proposed to use check-in history of users to recommend next Point-of-Interest(POI). The long-term dependencies are learnt using contextual features of POIs with attention, whereas short-term dependencies are learnt through two parallel LSTMs. Both are fused together to obtain final probabilities of candidate list of POI. AFRAM (Aspect-based Fashion Recommendation with Attention Mechanism) [39] proposed

to mine the user and item reviews to extract aspects. Each mining module was assembled with CNN and LSTM side-by-side with attention mechanism to trace local and global aspects from the reviews. These aspects were later fused to map preferences. Wang.et.al [38] exploit features of movies and demographics of users to propose a model that fuses CNN and LSTM to capture contextual dependencies. CNN focuses on the description data of movies while LSTM learns interdependencies of user’s features and ratings.

The prominence of graph neural networks (GNN) has resulted in many techniques that exploit the graph topology and node connectivity to improve recommendations [18]. It is achieved by focusing on preservation of network structure, when moving to low-dimensional space i.e. mapping the nodes with higher proximities in network closer to each other in low dimensional embedding space. DeepWalk employs techniques to learn representations of graph nodes based on randomwalk. Later, this neighbourhood of a node is used to learn feature representations of neighbouring nodes to make improved recommendations using DL approaches. The limitation of GNN is its inability to aggregate edge information while learning graph structure. This limitation is addressed by multiple improvements in GNN that has led to many variants namely Graph Convolution Networks (GCN) [25], Deep Convolutional Neural Networks (DCNN) [17], Graph Attention Networks (GAT) [19], [21], and Gated Graph Neural Networks (GGNN) [5] etc.

### C. GRAPH CONVOLUTION NETWORKS

The handicap of overwhelming computational power that comes with the large graph-structured datasets has set the foundation for research in the field of graph modelling [26], [27]. The motive behind this research is to preserve the structural and graphical essence of the graph while converting it into low dimensional space. MF techniques have been used to learn the graph representation by assuming that the input data exists in the low dimensional representation. However lately, owing to the inspiration from the CNN, a new paradigm of techniques has been explored to map the graph structure into low dimensional space preserving its characteristics [20]. Graph Convolution Networks (GCN) has increasingly been used on graph data as it has shown promising results in recommender systems [16], [29]. GCN improves a user’s embedding, based on the learned information through diffusion of its neighbourhood in a graph [4]. GCN has demonstrated encouraging results, in contrast to the conventional collaborative filtering techniques, as real-world data mostly resembles the graph structure [3]. GCN employs two basic operations - node embedding with convolutional neighbourhood aggregation; and non-linear conversion of embeddings transformed by a neural network as shown in Figure 2 [23]. Notable contributions of GCN are Social network analysis [41] and recommendation systems [21], that learn by iteratively stacking multiple layers of convolution aggregation operations and non-linear activation

operations. NGCF [8] and PinSage [20] implemented GCN to the user-item bipartite graph. STARGCN [28] addresses the cold start problem by reconstructing the embeddings for the new nodes. Furthermore, graph-based methods are improved by involving high-hop neighbours to model the embedding learning for sparse users [32], [33]. Examples of such models are GraphSage [18] and GAT [19] which redefine the target node's embedding by aggregating neighbour's embedding based on attention mechanism. KGAT [21] explicitly models the high-order connectivities in a knowledge graph and by linking items and users attribute information to effectively model users and items relationship. In spite of the considerable improvements shown in the results of these techniques, they disregard the heterogeneity of the graphs. Social networks is an example of such heterogeneous network. GraphRec [8] and DiffNet [7] take into account the heterogeneous nature of social network graph to learn user embeddings and item embeddings but till date scalability and coldstart problem remain unresolved in the crossdomain recommender systems.

### III. METADATA BASED CROSSDOMAIN FRAMEWORK USING GCN

#### A. MOTIVATION

To address the issue of sparsity, researchers have been exploring alternate sources to supplement the missing rating information. Cross-Domain Recommender Systems (CDRS) use multiple sources to supplement missing information in target domain by exploiting either user-overlap or item-overlap conditions or both. Pan et al. [32] proposed a Coordinate System Transfer (CST) technique which works on the condition when both source and target domains have homogeneous items but such homogeneity does not exist in real world. Lately, many recommendation methods have been employing graph neural networks in their recommendation frameworks. GCNs have particularly gained popularity among graph neural networks due to their ability to use local neighbourhood to collect and aggregate feature information [25]. KGAT [11] applies this technique by aggregating embeddings of a node's neighbours in a knowledge graph connected through relations of similar nature. Though this technique completely ignores the intra-relation among nodes during aggregation of embeddings which results in inadequacy in capturing collaborative signals through user-item graph. KGAT [21] addressed this issue by integrating user-item relations into the knowledge graph as a triple (user, Interact, item). This caused extensive modelling of sparse user-item relations generating extraneous attributes and failure in attentively modelling user-specific preferences on entities and relations. To address drawbacks of previous models discussed above, we propose a cross-domain recommendation framework that allows transfer of information among heterogeneous domains in a graphical setting.

#### B. SYSTEM DESCRIPTION

The motivation of crossdomain approach is to solve the problem of sparsity as source domain is dense in information as

compared to target domain (sparser domain). Preferences of coldstart users in source domain (richer domain) are formulated. With the help of mapping function among two domains, they are mapped onto the sparser domain. Even though both the domains are varied, their correlation makes it possible to use these domain's knowledge interchangeably. In the first step, metadata and rating matrices in separate domains are handled to learn domain specific features, whereas in the second step, linked users are used to learn a crossdomain mapping function based on neighbourhood among both domains. The knowledge graph serves as a basis of graph modelling for users and items embeddings. This heterogenous graph contains information regarding user-user similarity, item-item similarity and interaction information. The content-based similarity between users and items enables us to identify a neighbourhood graph which is fed into a multiple layered GCN. The embeddings of all nodes are propagated alongwith generating attention aware embeddings for overlapping users. Figure 3 graphically illustrates the system flow where after initial preprocessing of the input data, it is directed further for vector generation for the transformation of metadata into vectorized form. The vector's semantic similarity and interaction relationship defines the GCN graph. Crossdomain mapping function is learnt by the neural network during training of the model.

#### C. MATHEMATICAL FORMULATION OF THE PROBLEM

##### 1) GENERATING SEMANTIC SIMILARITY OF USERS

Let  $\mathcal{U} = \{u_1, u_2, \dots\}$  and  $\mathcal{V} = \{v_1, v_2, \dots\}$  be the user and item sets,  $\mathcal{R}^s$  and  $\mathcal{R}^t$  be two rating matrices from the source and target domains respectively, where  $\mathcal{R}_{uv}^s$  is the rating that user  $u$  gives to item  $v$  in the source domain and  $\mathcal{R}_{uv}^t$  is the corresponding rating in the target domain. Given two matrices,  $\mathcal{R}^s$  and  $\mathcal{R}^t$ , and cross-domain user and item sets,  $\mathcal{U}$  and  $\mathcal{V}$ , the aim is to make recommendation for those users and/or items in the target domain for whom the rating matrix is sparse.

$U_L$  signifies the set of linked users common to both domains. The training set is vector pair  $\{U_{v*}^s, U_{v*}^t\}_{v \in U_L}$  of the overlapping users for both domains. The proposed recommender task is to predict unknown ratings  $\mathcal{R}_{ux}$  for item  $x$  and user  $u$  in target domain based on known rating  $\mathcal{R}_{ui}$  of user  $u$  for item  $i$  in source domain. Let  $G$  be the plot description of a movie which is composed of words such as  $w_1, w_2, w_3, \dots, w_l$ .

$$G_{PlotDescription} \leftarrow \{w_1, w_2, w_3, \dots, w_l\}$$

From textual description of plots, a Corpus  $C$  is built, comprising of all movies and is represented as:

$$C_{CorpusGeneration} \leftarrow \{G_1, G_2, G_3, \dots, G_x\}$$

To generate profile of a user,  $\mathcal{P}_U$ , a certain threshold  $\theta \in [1 - 5]$  is set for all the items that receive a positively higher rating from the user. This profile can be thought of as a document containing salient tags pertaining to user and item



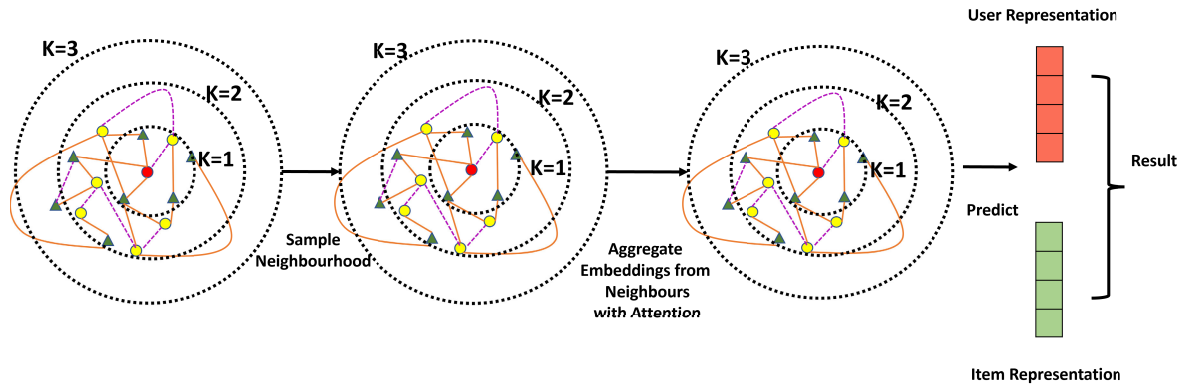


FIGURE 2. Embedding aggregation and propagation in GCN approach.

where the metadata of a user comprises of the tags associated with positively appraised items  $T_i$  and features from user’s feedback  $\mathcal{R}_u$ ,

$$\mathcal{P}_U = \{T_i, \mathcal{R}_U\}$$

A pre-trained *GloVe* embedding matrix is used to initialize the input matrix  $\mathcal{W}_g \in \mathcal{R}^{e \times n}$  where  $e$  and  $n$  are the embedding dimensions of both the CNN and the vectors. CNN proves viable in the field of text classification as it has the ability to learn dominant features in sentences along-with the relationships among words that are closer to each other in a sentence. The convolution windows in CNN can be perceived as filters that slide over a word along with it’s neighbouring words to output an embedding for that phrase. These convolution operations are window-based feature extractors that detect patterns in text. While CNN model focuses on learning spatial features, LSTM model is known to learn long-term relationships between word sequences. The combination of CNN and biLSTM enables the model to capture both long-term and short-term contextual information of the text. The strengths of both models result in condensing the text semantically in a computationally resource-efficient manner [40].

A total of  $e$  words are transformed into vectors where each word is expressed in  $n$  dimensional vector. *GloVe* generates feature matrix based on feature-feature co-occurrence.

$$X = \begin{bmatrix} w_{10} & w_{11} & w_{13} & \dots & \dots & w_{1n} \\ w_{20} & w_{21} & w_{23} & \dots & \dots & w_{2n} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ w_{i0} & w_{i1} & w_{i3} & \dots & \dots & w_{in} \\ \dots & \dots & \dots & \dots & \dots & \dots \\ w_{e0} & w_{e1} & w_{e3} & \dots & \dots & w_{en} \end{bmatrix}^{e \times n}$$

After applying a pre-trained *GloVe* embedding model, multiple convolution layers with *ReLU* activation function are applied to extract local features  $f_i$  from this profile,

$$f_i = \text{ReLU}(W_h \otimes X(:, w_{i:i+h-1})) + b \quad (1)$$

where  $\otimes$  signifies a convolution operation.  $W_h$  is a convolutional kernel of size  $h \times s$ . The kernel size is adjusted by -1 so

as to ensure that it does not match/exceed the dimensions of word vector. This kernel is applied on multiple possible size of word vectors in a sentence to produce contextual feature maps.

$$f = [f_1, f_2, \dots, f_{n-h+1}] \quad (2)$$

Now that the word embeddings are generated that characterizes vector for each word in our dataset, these pretrained *GloVe* embeddings are fed into sentence encoder which utilizes bi-LSTM with maxpooling. The motivation behind using bi-LSTM is to effectively capture the context of a word by reading the sentence in forward and reverse order. Given a sequence of word vectors,  $w_{10}, w_{11}, w_{12}$ , the bi-LSTM reads the sentence in forward and backward order and later concatenates the two vectors as shown in the equation.

$$\begin{aligned} \vec{h}_t &= \overrightarrow{\text{LSTM}}_t(w_1, \dots, w_T) \\ \overleftarrow{h}_t &= \overleftarrow{\text{LSTM}}_t(w_1, \dots, w_T) \\ h_t &= [\vec{h}_t, \overleftarrow{h}_t] \end{aligned}$$

Later, the maxpooling operation creates the vectors of same dimension  $h_t$  but delivers the maximum (most salient) value over the hidden units for each dimension. The dominant feature  $X_f$  after max-pooling is given by,

$$X_f = \max_{0 < i < s-h} \{h_i\} \quad (3)$$

where,

$i = 0, 1, 2, \dots, s - h$  and  $S_{i:j}$  represent a sub-matrix of  $S$  from row  $i$  to  $j$

$S$  is a sentence containing  $w$  words.

Higher the similarity score among semantic vectors of two users, higher the likelihood of similar preferences in future. This semantic similarity score between any two feature vectors  $V_{G_u}$  and  $V_{G_a}$  of users  $u$  and  $a$  respectively is calculated by cosine similarity,

$$S_{ua} = \frac{\sum_{z=1}^d (\bar{X}_{G_{uz}} - \hat{X}_{G_u}) \cdot (\bar{X}_{G_{az}} - \hat{X}_{G_a})}{\sqrt{\sum_{z=1}^d (\bar{X}_{G_{uz}} - \hat{X}_{G_u})^2} \sqrt{\sum_{z=1}^d (\bar{X}_{G_{az}} - \hat{X}_{G_a})^2}} \quad (4)$$

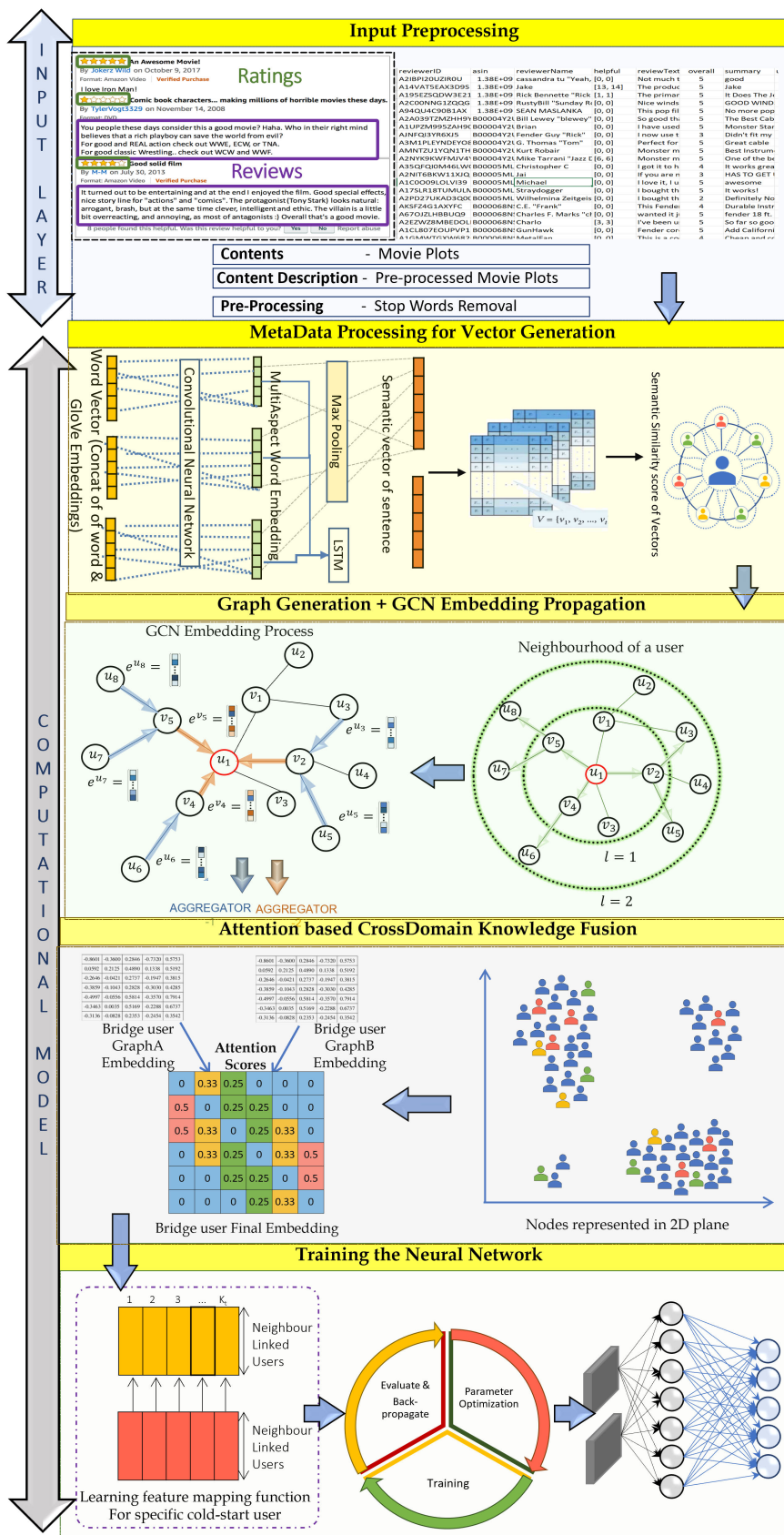


FIGURE 3. System flow of proposed MD-GCN architecture.

A graph  $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$  is a non empty finite set of  $\mathcal{V}$  vertices  $\mathcal{E}$  edges that signify the relationships between nodes based on node-node similarity and user-item interaction. If user  $u_i$  has a rating record for item  $v_j$ , then the ratings are normalized by  $\mathcal{R}_{ij} / \max \mathcal{R}$ . The ratings are normalized to cater for the fact that some users tend to rate too high or low overall instead of giving a range of ratings. Apart from the user-item interaction graph, a similarity graph  $S \in \mathbb{R}^{m \times m}$  is constructed using semantic similarity among users based on threshold  $\theta$ , where,  $S_{ij} = 1$  if user  $u_i$  is similar to user  $u_j$  and zero otherwise. This graph is employed to graphically place users who have similar preferences closer in proximity since for a node with sparse data, propagating embeddings from neighbourhood of likeminded users is a plausible approach.

## 2) ATTENTION BASED CROSS-DOMAIN KNOWLEDGE FUSION

Based on the heterogenous graph of users and items, the overlapping user's embeddings are combined to generate all-domain consolidated embedding that has a contribution from all domains. As mentioned earlier, bridge users controls the transfer knowledge among both domains, therefore, this transfer knowledge is guided by the cross-information function for overlapping users. Mathematically,

$$\hat{U} = \gamma_s U_s + (1 - \gamma_s) U_t \quad (5)$$

where,  $\hat{U}_s$ , is the combined embedding of the bridge user,  $\gamma_s$ , controls the contribution of latent factors of bridge user from both domains, source domain  $U_s$  and target domain  $U_t$ .  $\gamma_s$  therefore is the weight matrix that controls contribution from both domains and is learnt through jointly align and translate task that models an encoder-decoder model with attention [14] that employs a feed-forward neural network which is trainable together with whole sequence to sequence model. It identifies the parts of embedding that have maximum influence on the combined embedding. The attention mechanism lines up the input and output sequences with an alignment score trained by a feedforward neural network. It picks up most salient features of input to get a certain output. In this context, the attention layer rearranges the attention weights to pick the most salient feature in the source embeddings to achieve a certain target embedding. The learning of attention score is supervised by autoencoder recurrent neural network based on previous target embedding and previous decoder's hidden state  $s_{i-1}$  and the hidden state of the input embedding,

$$e_{ij} = a(s_{i-1}, h_j)$$

where, inputs around position  $j$  and the output at position  $i$  match and  $a$  is the feedforward neural network. The softmax activation function is applied to the attention score to obtain attention weight as,

$$\gamma_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})} \quad (6)$$

---

### Algorithm 1 Training the MD-GCN Framework

---

**Input:**  $R_s, R_t, U_s, U_t, V_s, V_t$

**Output:** Trained MD-GCN Framework

#### Phase 1: Textual Semantic Similarity

Obtain metadata for the users & items in the source & target domains  $D = \{d_1, d_2, \dots, d_{m+n}\}$

Obtain dense user & item vectors  $U_C^*, V_C^*$  for both domains using pre-trained *GloVe* model

Obtain the **user-user** similarity matrix  $S_U^A, S_U^T$  in the source domain & target domain respectively via cosine similarity

Obtain the **item-item** similarity matrix  $S_V^A, S_V^T$  in the source domain & target domain respectively via cosine similarity

Generate user-item graph  $G_A$  and  $G_T$  for source & target domain using  $S_U^A, S_U^T, S_V^A, S_V^T, R_s, R_t$

Initialize GCN with *GloVe* embeddings, for node  $i$ ,  $h_i^0 = e_i$

Update and propagate embeddings of a node  $i$  at level  $k$  based on neighbourhood  $N(i)$ ,

$$h_i^k = \text{ReLU} \left( W \cdot \left\{ \sum_{o \in N(i)} \frac{1}{|N(i)|} h_o^{k-1} \right\} + b \right)$$

#### Phase 2: Latent Embedding for Bridge Users

**for** each bridge user  $u$  in  $G_A$  and  $G_T$  **do**

an attention weight is normalized using softmax activation function

Calculate attention weight of output sequence given input embedding

**end while**

**end for**

Compute combined embedding  $\hat{U}_s$  for bridge user using attention weight  $\gamma$

#### Phase 3: Neural Learning for Cross-Domain Recommendation

**for** each user  $u$  in Training set  $T_u$

Obtain user-item rating  $y$

Initialize the weights in the MLP

the objective loss function of MLP does not converge over  $T_u$  **do**

Calculate relevance of user embedding vector and item embedding vector

Calculate predicted rating  $\hat{y}$

Compute the current loss of MLP using actual rating

Train the parameters of the deep neural network

Back-propagate errors, compute the weights & update parameters

**end while**

**end for**

---

This will give probabilities to all the input embeddings (that will sum up to 1) with maximum probability assigned to most informative domain embedding among  $U_s$  and  $U_t$ .

## 3) EMBEDDING THE GRAPH USING GCN

Preceding layers produce a heterogeneous graph (fusing the interaction relationship from adjacency matrix and similarity relationship from semantic similarity matrix  $S$  discussed in Section III-C1) where each node has an embedding

associated with it. Given this graph, GCN takes in two kinds of input:

- Input embedding matrix  $N \times e$  matrix, where  $N$  is the number of nodes and  $e$  is the number of latent embeddings for each node coming from the preceding layer.
- $N \times N$  adjacency matrix of the graph structure  $\mathcal{G}$

This initial embedding is further fed into the GCN for aggregation and convolution operation through multihop neighbours as shown in Figure 4. The connections in the graph signify two types of relationships, similarity relationship among similar type of nodes (based on node's metadata), and interaction relationship (from adjacency matrix).

Embedding broadcast and aggregation is performed where each node propagates its embedding in its neighbourhood while accumulating the embeddings it receives to update its embedding. This broadcast and aggregation using  $L$  hop neighbours based on the knowledge graph is the essence of GCN framework. The knowledge graph exploits the higher-order connectivity of a node's graphical neighbourhood. Since a user's preferences are inspired by their  $L$ -hop neighbours, graph convolution operators are used to aggregate the embeddings of a user's neighbourhood. The input of this graph convolution network are the embedding matrix learnt in the last phase and the adjacency matrix. The embeddings are the representations of a node's neighbours that are transformed through dense neural network and later fed to an aggregator function which results in a vector representation of a node. The adjacency matrix is inferred from user-item rating matrix. Populating the graph with embeddings is focussed at mapping the nodes to low-dimensional space while sustaining the structure of the graph.

Let  $Y \in \mathbb{R}^{(M+N) \times D}$  be the embedding matrix of users and items. By feeding the embedding matrix  $Y$  into GCN with graph  $\mathcal{G} < \mathcal{U} \cup \mathcal{V}, A >$  where  $A$  is constructed from the rating matrix  $\mathcal{R}$  as:

$$\forall i \in \mathcal{U} \cup \mathcal{V}, \quad h_i^0 = e_i \quad (7)$$

The GCN iteratively carries out the embedding propagation and non-linear transformation as shown in the eq 10 and 11. Each node's embedding in the heterogenous graph is updated in the iterative process. Therefore, the final embedding  $H^{(K)}$  explicitly injects upto  $K$ -th order collective connections between users and items.

Following this, aggregation of node's embedding is carried out in order to aggregate the embedding of neighbouring nodes from the user-item adjacency relationship and similarity relationship in one heterogeneous graph. For example, the neighbourhood based user latent factor of node  $i$ , is given by  $h_i$ , after aggregating user's embeddings of its neighbours  $\mathcal{N}(i)$ , as follows:

$$h_i^k = \sigma \left( W \cdot \text{Aggre}_{neighbours} \left( \left\{ h_o^k, \forall o \in \mathcal{N}(i) \right\} \right) + b \right) \quad (8)$$

where  $\sigma$  is a nonlinear activation function and  $\text{Aggre}$  is an aggregation function. One natural aggregation function for  $\text{Aggre}_{neighbours}$  is the mean operator which takes the

element-wise mean of the vectors in  $\{h_o^k, \forall o \in \mathcal{N}(i)\}$  as shown mathematically in the following equation,

$$h_i^k = \sigma \left( W \cdot \left\{ \sum_{o \in \mathcal{N}(i)} \beta_i h_o^k \right\} + b \right) \quad \forall o \in \mathcal{N}(i) \quad (9)$$

Since all neighbours contribute equally to the representation of a target node in a mean-based aggregator,  $\beta_i$  is set to  $\frac{1}{|\mathcal{N}(i)|}$ . Furthermore, multiple layer GCN, with convolutional network layer  $h_i^l$  for node  $i$  is given by,

$$h_i^k = \text{ReLU} \left( W \cdot \left\{ \sum_{o \in \mathcal{N}(i)} \frac{1}{|\mathcal{N}(i)|} h_o^{k-1} \right\} + b \right) \quad (10)$$

where  $h_i^k$  denotes the  $i^{\text{th}}$  node's current representation at  $k$  step in the forward propagation. First, each node's  $o \in \mathcal{N}(i)$  representations are aggregated. Preceding representations (i.e  $k-1$ ) step are accumulated to define a node's embedding at  $k$ th step, whereas, the  $k=0$  ("base case") representations are defined as the node's input embeddings. Even though, in order to fully exploit the higher order connectivities in a GCN, we need to stack multiple graph convolutional network layers, however, stacking multiple layers results into considerable drop in the performance due to gradient vanishing or over smoothing. Hence,  $K=2$  is most optimized depth for optimized performance results for GCNs [25]. Therefore, final embedding at layer ' $k$ ' after GCN aggregation from nodes's neighbours and concatenating node's current embedding is given by,

$$h_i^k = \sigma \left( W \cdot \left[ h_i^{k-1} \oplus n_i^{k-1} \right] \right) \quad (11)$$

where  $n_i^{k-1}$  is aggregated neighbourhood embedding vector and  $\sigma$  is the activation function  $\text{ReLU}$  in this case). Above equation calculates the current layer embedding for a node by concatenating its preceding convolutional layer embedding with the aggregated embedding from all its neighbours.

#### 4) TRAINING THE NEURAL NETWORK AND OUTPUT LAYERS

After propagating and calculating the embeddings for each node through GCN layers, a neural network is adopted to model user-item interactions and learn the prediction score. The embeddings of a node are calculated through aggregation operators applied on local neighbourhood of the node. The user-item rating is normalized and interaction matrix  $\mathcal{R} \in \mathbb{R}^{m \times n}$  contains the normalized ratings.

The output  $\hat{\mathcal{R}}_{ui}$  represents the relevance between user  $u$  and item  $i$ . The input embedding matrices consists of joint embeddings for overlapping users and embeddings for distinct users. Let  $Y_{in}^a$  be the input embedding matrix for domain  $a$ . Then,  $Y^a = U^j : U^a$  where  $U^j$  is the joint embeddings for overlapping users inferred from both domains and  $U^a$  is the embedding for distinct users in domain  $a$ . Similarly,  $Z_{in}^a = V^a$  is the input embedding for items of domain  $a$ . These input embedding matrices are fed into MLP with network weights



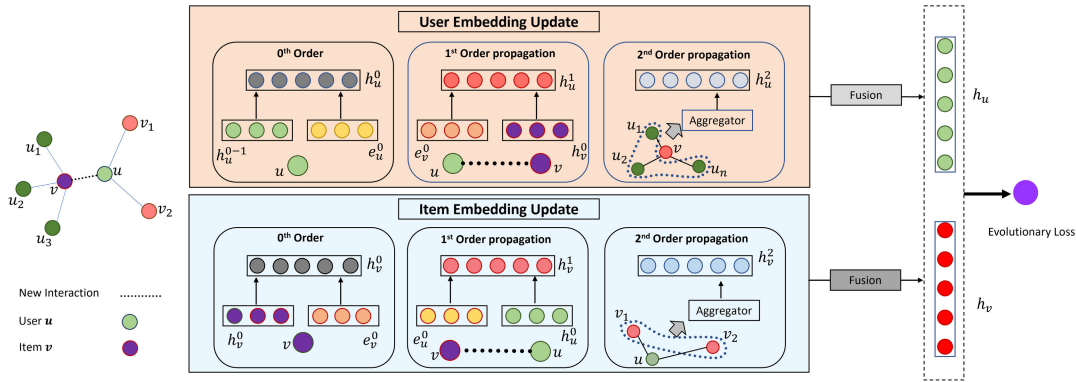


FIGURE 4. Illustration of user and item embedding propagation.

$W^a$  and activation function  $ReLU$ , output embeddings are calculated as,

$$Y_i = \alpha(\dots \alpha(Y_{in}^a \cdot W^a), W^a)$$

$$Z_j = \alpha(\dots \alpha(Z_{in}^a \cdot W^a), W^a)$$

where,

$\alpha$  is the activation function,  $ReLU$

Later, the loss is calculated and backpropagated as,

$$\hat{\mathcal{R}}_{ui} = \cos(Y_i^a, Z_j^a) = \frac{Y_i^a \cdot Z_j^a}{\|Y_i^a\| \cdot \|Z_j^a\|} \quad (12)$$

$$loss(\mathcal{R}_{ui}, \hat{\mathcal{R}}_{ui}) = \mathcal{R}_{ui} \log \hat{\mathcal{R}}_{ui} + (1 - \mathcal{R}_{ui}) \log (1 - \hat{\mathcal{R}}_{ui}) \quad (13)$$

## IV. EXPERIMENTS

### A. DATASETS

The proposed technique is evaluated on benchmark domains of Amazon dataset, MoviesandTV, Books, DigitalMusic. The source domain is required to be richer in information therefore, Amazon-Movie dataset is chosen as source domain and Amazon-Books and Amazon-Music as target domain for our set of experiments.

The sparsity of the Amazon dataset domains varies between  $\approx 93.7\%$  to  $\approx 99.46\%$  which makes them a good selection for solving coldstart problems in RSs [4]. Since the technique requires considerable metadata pertaining to a user therefore users who have atleast rated and reviewed 8 movies are chosen. The Amazon-Book dataset consists of 9,876 explicit ratings  $R_{ui} \in \{1, 2, 3, 4, 5\}$  given by 51,800 users on approximately 21,453 books. Amazon-Music rated and reviewed items 3658 by 10,262 users. Moreover Amazon-MoviesandTV dataset has users 78,500 who rated 25,645 with a total of 42,877 user-item ratings. The 5-core dataset provided by McAuley.et.al [30] where each user and each item has at least 5 associated fields of metadata. These datasets vary in sparsity. The statistics of these datasets are summarized in Table 1.

TABLE 1. Statistics of experimental dataset.

Dataset	Domain	Users	Items	Rating & Meta Data	Rating Scale
Amazon	Movies	78,500	25,645	42,877	1-5
	Books	51,800	21,453	9,876	1-5
	Music	10,262	5,237	3658	1-5

### B. DATASET PREPROCESSING

The Amazon5-Core dataset [30] contains product’s ratings, reviews, and metadata. Since Amazon-Movies domain is used as a source domain in the proposed technique, therefore, to address sparsity in remaining movies, data was supplemented with descriptions elicited by [41]. The dataset is later fed through many preprocessing steps such as lemmatization, tokenization, stopwords removal etc. Length of each movie description is set to 300. Meaningful words are picked using TF-IDF technique and threshold is set heuristically as shown in Figure 5. Grid Search cross validation technique with an estimator is employed which while ‘fitting’ it on a dataset, evaluates all the possible combinations of parameters and retains the best combination [42]. The threshold is ranged from 0.25 through till 0.75 to analyse the performance of TF-IDF on three domains of Amazon datasets, Amazon-Movies, Amazon-Music and Amazon-Books. As evident from the figure, the precision of Amazon-Movies grows till TF-IDF threshold is ranged from 0.20 till 0.5 and drops later. Similar trend is observed for domain Amazon-Music where the graph steadies from threshold=0.35 till 0.55 and drops later. Amazon-Books performs well for threshold 0.52. The threshold is therefore averaged and found to give maximum precision at an average of 0.5. Top 1000 words are selected to form a vocabulary size. The source domain consists of a database that consists of users with minimum 5 fields of metadata and ratings and items (movies) with descriptions. This preprocessed information is used in the next step to generate vectors for the users and items.

To generate vector representations of the metadata of the items  $GloVe300d$  is used. The user’s metadata, tags and item’s description and tags turn into a vector signifying the latent

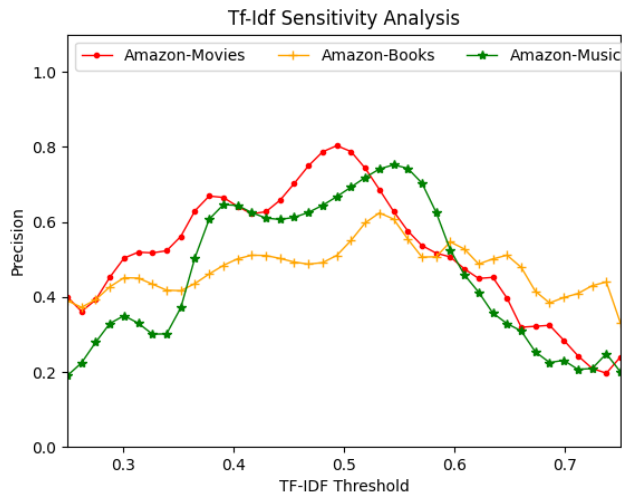


FIGURE 5. Sensitivity analysis of TF-IDF vs precision.

embedding of the user's and item's. Since the model is a pre-trained model, for a new word that does not exist in the *GloVe* embedding, a new vector equal to embedding size is created with zeros initialized. These are fed through series of 1D convolution and pooling operations to learn most significant feature. LSTM network is employed to encode sentences to compute their semantic similarity. This network uses hidden vectors of the sentences to work out the Manhattan distance between them to gauge the similarity between the two sentence pairs.

### C. EXPERIMENTAL SETUP

To evaluate the performance of the proposed technique against other recommendation techniques, the data is divided into training, test set and validation set by randomly splitting the data into a training set (60%), a test set (20%) and a validation set (20%) to tune the hyperparameters. The results stated constitute the average of 10 experiments conducted and the training lasts for 100 epochs. The performance of the algorithm is tested for varying values of top-K neighbours results of which are discussed in Section IV-F.

To produce vectors, a pretrained *GloVe* embedding model is employed with input length of word set to 30. It encodes each word into a 300 element vector. These vectors are fed to a 1 layer-CNN having 300 filters and activation function *ReLU*, later succeeded by one maxpooling layer and one LSTM layer. This network helps transform word vectors into sentence embedding with parameters as shown in Figure 6.

GCN is implemented using PyTorch, with batchsize=128 and learning rate=0.001. The model is optimized using Adam optimizer to update all parameters during training phase. The maximum number of node neighbours in GCN is set to 25 and similarity threshold is limited to 0.8. This proves to be a reasonable tradeoff and relegates the neighbourhood that results in a computational efficiency. To incorporate higher hop neighbours, the depth of neighbourhood is set to 2. The

Layer (type)	Output Shape	Param #
embedding_8 (Embedding)	(None, 450, 300)	300000
conv1d_8 (Conv1D)	(None, 443, 32)	76832
max_pooling1d_8 (MaxPooling 1D)	(None, 221, 32)	0
lstm_7 (LSTM)	(None, 128)	82432
=====		
Total params: 459,264		
Trainable params: 459,264		
Non-trainable params: 0		

FIGURE 6. Model parameters using *GloVe* embeddings.

size of the hidden layer is equal to embedding size with *ReLU* as activation function for each layer.

For training of neural network, the model parameters are initialized using Gaussian distribution. The embedding size is varied through the values of [8, 16, 32, 64, 128]. The size of the hidden layer is the same as that of embedding dimension with *ReLU* as activation function. The learning rate is set to 0.001, the regularization coefficient  $\lambda$  is 0.002, and the batch size is 128.

An observed interaction is treated as a positive sample and an absence of an interaction as a negative sample. For training purposes, one negative instance for 5 positive instance is sampled. The training is stopped when the RMSE continues to increase for 5 successive epochs.

To simulate cold-start scenario, 50% of bridge users are simulated to be cold-start users (specifically, 30% are set for test users and 20% are set for validation of the results). The remaining 50% is used for training purposes. Since the cross-domain mapping function is to be learnt on overlapping users therefore their interactions in target domain are masked. To create an even distribution, 50% of training set bridge users are sampled randomly. Splitting the overlapping users into training, testing and validation set by the ratio of 50%-30%-20% and random sampling makes a fair distribution for training and testing. To further evaluate the performance, we simulate a variable cold-rate, which is used to describe the level of cold-start scenario (the larger the value, the higher the degree of cold-start). The fraction of these items is varied through 5%, 10%, 20% and 30% of the total data records and sampled repeatedly 10 times to generate different cold-start users sets.

### D. BASELINES

To make a reasonable and fair comparison, the effectiveness of the proposed technique is established with the following renowned research baselines in the paradigm of cross-domain RSs. These baselines implement deep neural network and matrix factorization techniques as their strengths to design recommendation algorithm.

- **EMCDR [16]** : Embedding and Mapping framework for Cross-Domain Recommendation is a cross-domain

TABLE 2. HR scores @ varying holdout percentages on Amazon datasets using proposed algorithm.

Amazon Datasets												
$D^s \rightarrow D^T$	$Movies \rightarrow Books$				$Books \rightarrow Movies$				$Movies \rightarrow Music$			
Holdout%	$TR_{50}$	$TR_{40}$	$TR_{30}$	$TR_{20}$	$TR_{50}$	$TR_{40}$	$TR_{30}$	$TR_{20}$	$TR_{50}$	$TR_{40}$	$TR_{30}$	$TR_{20}$
k=8	0.377	0.451	0.598	0.542	0.366	0.455	0.522	0.566	0.398	0.508	0.531	0.598
k=16	0.455	0.450	0.627	0.661	0.412	0.523	0.515	0.622	0.409	0.576	0.563	0.644
k=32	0.577	0.652	0.656	0.672	0.579	0.612	0.638	0.653	0.586	0.648	0.655	0.673
k=64	0.645	0.652	0.723	0.766	0.679	0.657	0.655	0.722	0.659	0.692	0.699	0.721
k=128	0.769	0.770	0.771	0.776	0.715	0.719	0.722	0.726	0.702	0.714	0.716	0.728

recommendation model that employs a Linear Matrix Translation (LIN) and then a Multilayer Perceptron (MLP) to learn latent space mapping through vector pairs of bridge users. This results in two variations of this technique, MF\_EMCDR\_LIN that uses MF as its latent factor model and a linear cross-domain mapping function and MF\_EMCDR\_MLP, that adopts MF as its latent factor model, but has an MLP based cross-domain mapping function.

- **GCCA-ISSM [23]**: Cross-domain recommender system using generalized canonical correlation analysis exploits user's demographic data, ratings and reviews to build a user profile. User's latent features are learnt using canonical correlation analysis to provide better recommendations for new users.
- **DHCD [22]**: Deep Hybrid Cross-Domain Model generates user and items latent representation by modelling user's ratings and reviews. These domain independent latent representations are then fused with domain specific representations to transfer knowledge across domains which will serve as a function to predict preferences for new users.
- **CDLFM [29]**: Cross-Domain recommendation [21] for cold-start users via Latent Feature Mapping uses the three kinds of similarities to identify neighborhood of a user. These neighbours are later used to learn a mapping function for a coldstart user through MLP and GBT (Gradient Boosting Trees).

### E. PERFORMANCE METRICS

The proposed technique is evaluated using Root Mean Square Error (RMSE) and Mean Absolute Error (MAE). These two metrics are conventionally used to evaluate RSs [1]. RMSE signifies how closely the predicted rating matches with the actual rating. Mathematically, it is given by:

$$RMSE = \sqrt{\sum_{y_{ui} \in T_E} \frac{(y_{ui} - \hat{y}_{ui})^2}{|T_E|}} \quad (14)$$

where,  $y_{ui}$  is the actual rating for a test record belonging to test dataset  $T_E$  whereas  $\hat{y}_{ui}$  is the predicted rating and  $|T_E|$  is the total number of test ratings. MAE is the average of all the absolute differences of the actual ratings to the forecasted ratings. The lower the MAE score, the closer the predicted ratings are to the actual ratings. MAE is computed using the

following formula:

$$MAE = \frac{\sum_{i=1}^n |y_{ui} - \hat{y}_{ui}|}{n} \quad (15)$$

where,  $y_{ui}$  is the actual rating for an item  $i$  whereas  $\hat{y}_{ui}$  is the predicted rating and  $n$  is the total number of items being evaluated. Even though both the terms signify the strength of recommendation accuracy of a recommendation technique, the RMSE penalizes the term more than MAE when the error is high.

The RMSE and MAE analyse the results from the perspective of rating accuracy considering the historical data, but they do not give an insight into whether right preferences were made for the user or not. Alternative metrics that reflect effectiveness of choices made suiting to the user are HitRatio@K (HR) and Normalized Discounted Cumulative Gain (NDCG@K). HR signifies if a target item was present or not in the ranked list of items 'K', that the recommendation model produced or not. The presence marks a 'hit' whereas absence marks a 'miss'. Mathematically, represented as,

$$HitRatio @ K == \frac{Number\ of\ hits\ in\ K}{|T|} \quad (16)$$

where,  $K$  is the ranked list of items and  $T$  denotes the number of interactions in the test set. Since HR only checks the presence of an item in the ranked list irrespective of its rank therefore nDCG@K is measured to assess the rank of the 'hit'. It is a rank-sensitive quality metric that assigns higher scores to hits that are top ranked items in the list. nDCG can be perceived as a relevance score for items that made to the recommendation list. nDCG is a ratio of Cumulative Gain (CG) to Ideal Cumulative Gain where CG is a sum of the total relevance scores of all the items in the recommendation list and DCG takes into account the rank of the result controlled in logarithmic factor.

$$CD_k = \sum_{i=1}^k rel_i$$

$$DCG_k = \sum_{i=1}^k \frac{2rel_{i-1}}{\log_2(i+1)}$$

$$nDCG@k = \frac{DCG_k}{IDCG_K} \quad (17)$$

where,  $IDCG$  is the maximum possible  $DCG$  for a given set of results in the test set. Both  $HR$  and  $nDCG$  requires generating top  $k$  items for a user. Top  $K$  recommendations are made by

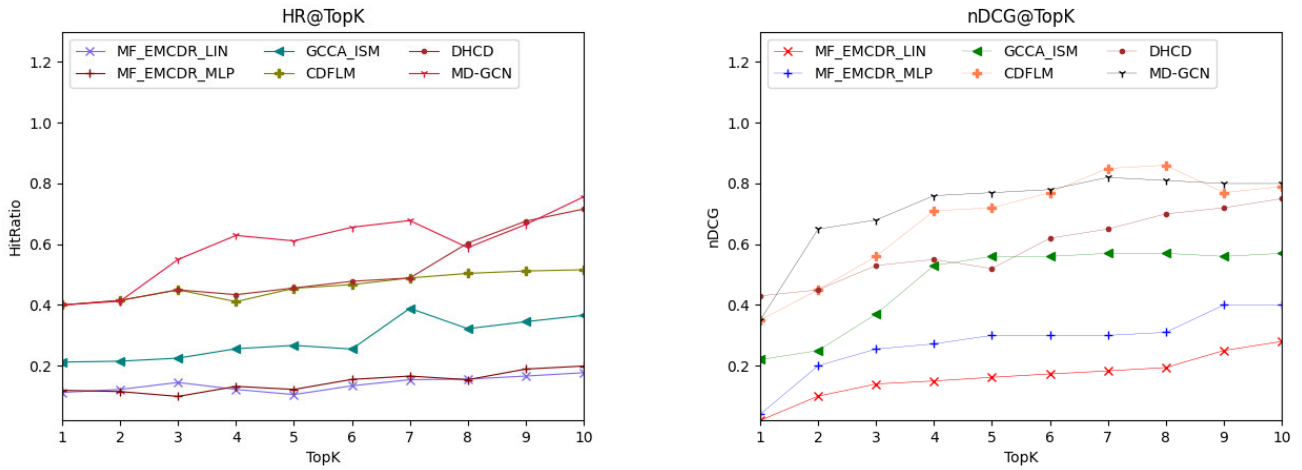


FIGURE 7. Top K performance of all methods on Amazon Datasets w.r.t hitRatio@K and nDCG@K.

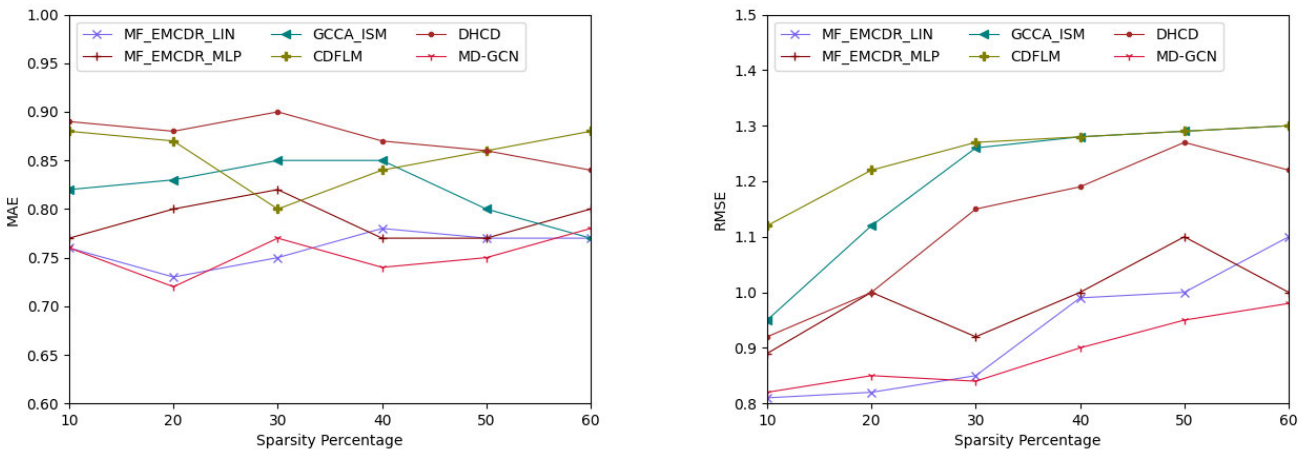


FIGURE 8. MAE and RMSE performance comparison with baselines using Amazon Datasets.

intentionally removing an item for a user in training set and during testing phase checking if that item is present in the recommendation list or not.

Precision signifies the proportion of positive identifications that were actually correct. Precision is given by,

$$Precision = \frac{TP_K}{TP_K + FP_K} \quad (18)$$

where  $TP_K$ ,  $FP_K$  are true positive and false positive items in the Top-K recommendation list respectively.

### F. RESULTS AND ANALYSIS

The proposed technique is evaluated on the domains of benchmark dataset Amazon using different holdout percentages (10%, 20%, 30%, 40% and 50%). The richer domain Amazon-Movies is chosen as source domain while keeping sparser domains Amazon-Books and Amazon-Music as target domains. For example, we can recommend a category of a book to a user by judging his preference from the genre

of a movie he prefers. If a user shows preferences for sci-fi movies in movies domain and rates them high, we recommend similar category books to the user even if his taste in books is sparse. This transfer of knowledge among domains to judge choices serves to improve recommendation accuracy. Table 2 records the HitRatio results of the proposed technique with varying holdout percentages ranging from 50%, 40%, 30% and 20% and different embedding dimensions  $k$  for 8, 16, 32, 64 and 128. A 20% holdout percentage means 20% data is heldout while rest 80% percent of data is used for training and validation. This 80% can be used in ratio of 60% (train) and 20%(validation) or 64%(train) and 16%(validation) etc. Lesser values of holdout percentages results in better HR scores within the same embedding size and the performance decreases with increase in the holdout data percentage. This is because as the percentage of holdout data increases, the chunk of data available for training and validation decreases. The proposed technique shows promising results for 20%, 30% and 40% holdout data percentage



**TABLE 3.** MAE performance comparison among baselines with varying sparsity.

DataSet	Model	Sparsity				
		10%	20%	30%	40%	50%
Amazon ( <i>Movies</i> → <i>Books</i> )	MF_EMCDR_LIN [16]	0.8784	0.8725	0.8648	0.8481	0.8298
	MF_EMCDR_MLP [16]	0.8650	0.8672	0.8197	0.8271	0.8046
	GCCA-ISSM [23]	0.8711	0.8701	0.8665	0.7854	0.7986
	CDLFM [29]	0.7809	0.7598	0.7638	0.7588	0.7688
	DHCD [22]	0.8162	0.821	0.8511	0.8112	0.7921
	MD-GCN (ours)	0.7854	<b>0.748</b>	<b>0.759</b>	<b>0.743</b>	<b>0.7650</b>
Amazon ( <i>Movies</i> → <i>Music</i> )	MF_EMCDR_LIN [16]	0.7645	0.7610	0.7783	0.7651	0.7619
	MF_EMCDR_MLP [16]	0.7781	0.7810	0.7749	0.7682	0.7817
	GCCA-ISSM [23]	0.7712	0.7690	0.7704	0.7650	0.7780
	CDLFM [29]	0.7855	0.7799	0.7750	0.771	0.7789
	DHCD [22]	0.7850	0.7788	0.774	0.766	0.780
	MD-GCN (ours)	<b>0.758</b>	<b>0.745</b>	<b>0.7550</b>	<b>0.7611</b>	<b>0.7616</b>

**TABLE 4.** RMSE performance comparison among baselines with varying sparsity.

DataSet	2Model	Sparsity				
		10%	20%	30%	40%	50%
Amazon ( <i>Movies</i> → <i>Books</i> )	MF_EMCDR_LIN [16]	0.91	0.95	0.975	0.977	1.0
	MF_EMCDR_MLP [16]	0.8952	0.989	1.003	1.014	1.052
	GCCA-ISSM [23]	0.896	0.963	1.0054	1.025	1.266
	CDLFM [29]	0.952	1.069	1.155	1.169	1.126
	DHCD [22]	0.996	1.099	1.198	1.212	1.278
	MD-GCN (ours)	<b>0.852</b>	0.989	<b>1.002</b>	<b>1.009</b>	<b>1.001</b>
Amazon ( <i>Movies</i> → <i>Music</i> )	MF_EMCDR_LIN [16]	0.873	0.996	1.029	1.055	1.078
	MF_EMCDR_MLP [16]	0.856	1.066	1.199	1.204	1.236
	GCCA-ISSM [23]	0.843	0.993	1.099	1.106	1.145
	CDLFM [29]	0.851	0.998	1.079	1.188	1.198
	DHCD [22]	0.925	0.989	1.029	1.030	1.036
	MD-GCN (ours)	<b>0.844</b>	<b>0.978</b>	<b>1.001</b>	<b>1.1043</b>	<b>1.102</b>

since more data is available for training and validation. For  $k=128$  neural embedding layer, the technique trains well and learns more details of latent factors hence achieving improved performance. To draw comparison with other neural based recommendation baselines, the hyperparameters are set to default setting i.e.  $k=128$ , minibatch size of 1024, the parameter of the neural network are set to Gaussian distribution with default learning rate=0.001, the regularization parameter=0.001 and epoch=100. These settings were kept constant to gauge HR@K and nDCG@K performance with baselines graphically shown in Figure 7. These two metrics are gauged with the size of the recommendation list. Compared with other baselines, it is observed that our proposed technique outperforms EMCDR, GCCA\_ISSM, CDLFM, and DHCD. The size of the recommendation list varies from 1 to 10. The proposed technique is observed to improve HR@Top10 of DHCD by 8.52% on Amazon Dataset. The NDCG of our method at Top10 outperforms DHCD with 13.13% improvement. Compared with the CDLFM model, the HR of our method at Top10 is improved by 6.69% on Amazon Dataset. The NDCG of our method at Top10 outperforms CDLFM with 11.48% improvement. Figure 8 graphically illustrates the MAE and RMSE performance in comparison with other baselines.

Table 3 and Table 4 tabulate the comparison of recommendation accuracy in terms of MAE and RMSE of the proposed technique against other cross-domain models stated in literature so far. The MAE comparison of the proposed technique with other crossdomain techniques are drawn keeping sparsity values ranging from 10, 20, 30, 40 and 50 as shown in Figure 8. The overall comparison with other cross-domain recommendation model exhibits that our approach outperforms all baselines by assessing all benchmark metrics. This establishes the superiority of our technique for sparse scenarios. A series of experiments were conducted to study the impact of neighbourhood size on the performance of the proposed mode. The value of K is ranged from 10 till 50. It has been observed that increasing the number of neighbours in the user-item graph resulted in increased HR and nDCG showing that similar historical choices continue resulting in similar future preferences. The best results are observed at  $K=40$ .

## V. CONCLUSION

The proposed model employs a graph neural technique to make rating predictions while utilizing metadata from multiple domains. This addresses the problem of cold-start in target domain since a richer domain supplements the data

adding to the richness of information. MD-GCN empirically investigates the advantage of fusing metadata through graph convolution network for recommendation tasks. Particularly, we have fused the metadata and interaction information jointly to model a graphical structure. This helps in learning a user's representation through hierarchical graph attention model that also incorporates preferences of likeminded users. To model user's preferences, the proposed technique enforces an attention based mechanism for overlapping users from both domains for knowledge transfer. This approach implements GCN framework alongwith utilizing the dense information in the metadata from both domains to generate the recommendation list. This considerably reduced the cumulative prediction error. In addition, the proposed method is verified to perform well on the coldstart problem. Our extensive experimental results showed that our technique outperformed the baselines in terms of the MAE and RMSE metrics.

## REFERENCES

- [1] P. Cremonesi, A. Tripodi, and R. Turrin, "Cross-domain recommender systems," in *Proc. IEEE 11th Int. Conf. Data Mining Workshops*, Dec. 2011, pp. 496–503.
- [2] W. Pan, N. N. Liu, E. W. Xiang, and Q. Yang, "Transfer learning to predict missing ratings via heterogeneous user feedbacks," *Proc. Int. Joint Conf. Artif. Intell. (IJCAI)*, vol. 22, Jul. 2011, p. 2318.
- [3] B. Li, Q. Yang, and X. Xue, "Transfer learning for collaborative filtering via a rating-matrix generative model," in *Proc. 26th Annu. Int. Conf. Mach. Learn.*, Jun. 2009, pp. 617–624.
- [4] J. Bobadilla, F. Ortega, and A. Hernando, "Recommender systems survey," *Knowl.-Based Syst.*, vol. 46, pp. 109–132, Jul. 2013.
- [5] X. Wang, X. He, M. Wang, F. Feng, and T.-S. Chua, "Neural graph collaborative filtering," in *Proc. 42nd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2019, pp. 1–10, doi: [10.1145/3331184.3331267](https://doi.org/10.1145/3331184.3331267).
- [6] M. A. Elaskily, M. M. Dessouky, O. S. Faragallah, and A. Sedik, "A survey on traditional and deep learning copy move forgery detection (CMFD) techniques," in *Multimedia Tools and Applications*. Cham, Switzerland: Springer, Mar. 2023, doi: [10.1007/s11042-023-14424-y](https://doi.org/10.1007/s11042-023-14424-y).
- [7] L. Wu, J. Li, P. Sun, R. Hong, Y. Ge, and M. Wang, "DiffNet++: A neural influence and interest diffusion network for social recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 10, pp. 4753–4766, Oct. 2022.
- [8] W. Fan, Y. Ma, Q. Li, Y. He, E. Zhao, J. Tang, and D. Yin, "Graph neural networks for social recommendation," 2019, *arXiv:1902.07243*.
- [9] R. M. D'Addio and M. G. Manzato, "A collaborative filtering approach based on user's reviews," in *Proc. Brazilian Conf. Intell. Syst.*, Oct. 2014, pp. 204–209.
- [10] N. Pappas and A. Popescu-Belis, "Adaptive sentiment-aware one-class collaborative filtering," *Exp. Syst. Appl.*, vol. 43, pp. 23–41, Jan. 2016.
- [11] L. Zheng, V. Noroozi, and P. S. Yu, "Joint deep modeling of users and items using reviews for recommendation," in *Proc. 10th ACM Int. Conf. Web Search Data Mining*, Feb. 2017, pp. 425–434.
- [12] M. Sajjad, F. Ramzan, M. U. G. Khan, A. Rehman, M. Kolivand, S. M. Fati, and S. A. Bahaj, "Deep convolutional generative adversarial network for Alzheimer's disease classification using positron emission tomography (PET) and synthetic data augmentation," *Microsc. Res. Technique*, vol. 82, pp. 3023–3034, Jul. 2021, doi: [10.1002/jemt.23861](https://doi.org/10.1002/jemt.23861).
- [13] M. Al-Ghobari, A. Muneer, and S. M. Fati, "Location-aware personalized traveler recommender system (LAPTA) using collaborative filtering KNN," *Comput., Mater. Continua*, vol. 69, no. 2, pp. 1553–1570, 2021.
- [14] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," 2014, *arXiv:1409.0473*.
- [15] I. Fernández-Tobías and I. Cantador, "Exploiting social tags in matrix factorization models for crossdomain collaborative filtering," in *Proc. CEUR Workshop*, vol. 1245, 2014, pp. 34–40.
- [16] T. Man, H. Shen, X. Jin, and X. Cheng, "Cross-domain recommendation: An embedding and mapping approach," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 2464–2470.
- [17] R. van den Berg, T. N. Kipf, and M. Welling, "Graph convolutional matrix completion," 2017, *arXiv:1706.02263*.
- [18] W. L. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst. (NeurIPS)*, 2017, pp. 1025–1035.
- [19] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Liò, and Y. Bengio, "Graph attention networks," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2018, pp. 1–12. [Online]. Available: <https://openreview.net/forum?id=rJXMpikCZ>
- [20] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for web-scale recommender systems," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2019, pp. 974–983.
- [21] X. Wang, X. He, Y. Cao, M. Liu, and T.-S. Chua, "KGAT: Knowledge graph attention network for recommendation," in *Proc. 25th ACM Int. Conf. Knowl. Discovery Data Mining (SIGKDD)*, 2019, pp. 950–958.
- [22] H. Wang, F. Zhang, M. Zhang, J. Leskovec, M. Zhao, W. Li, and Z. Wang, "Knowledge-aware graph neural networks with label smoothness regularization for recommender systems," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 968–971.
- [23] S. M. Hashemi and M. Rahmati, "Cross-domain recommender system using generalized canonical correlation analysis," *Knowl. Inf. Syst.*, vol. 62, pp. 4625–4651, Sep. 2020, doi: [10.1007/s10115-020-01499-4](https://doi.org/10.1007/s10115-020-01499-4).
- [24] H. Guo, R. Tang, Y. Ye, Z. Li, and X. He, "DeepFM: A factorization-machine based neural network for CTR prediction," 2017, *arXiv:1703.04247*.
- [25] X. Zhang, H. Liu, Q. Li, and X.-M. Wu, "Attributed graph clustering via adaptive graph convolution," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 4327–4333.
- [26] B. Li, Q. Yang, and X. Xue, "Can movies and books collaborate? Cross-domain collaborative filtering for sparsity reduction," *Proc. 21st Int. Joint Conf. Artif. Intell. (IJCAI)*, 2009, pp. 2052–2057.
- [27] M. Enrich, M. Braunhofer, and F. Ricci, "Cold-start management with cross-domain collaborative filtering and tags," in *Proc. 14th Int. Conf. e-Commerce Web Technol.* Heidelberg, Germany: Springer, vol. 152, 2013, pp. 101–112, doi: [10.1007/978-3-642-39878-0\\_10](https://doi.org/10.1007/978-3-642-39878-0_10).
- [28] J. Zhang, X. Shi, S. Zhao, and I. King, "STAR-GCN: Stacked and reconstructed graph convolutional networks for recommender systems," 2019, *arXiv:1905.13129*.
- [29] X. Wang, Z. Peng, S. Wang, P. S. Yu, W. Fu, X. Xu, and X. Hong, "CDLFM: Cross-domain recommendation for cold-start users via latent feature mapping," *Knowl. Inf. Syst.*, vol. 62, no. 5, pp. 1723–1750, May 2020.
- [30] J. Ni, J. Li, and J. McAuley, "Justifying recommendations using distantly-labeled reviews and fined-grained aspects," in *Proc. Empirical Methods Natural Lang. Process. (EMNLP)*, 2019, pp. 188–197, doi: [10.18653/v1/D19-1018](https://doi.org/10.18653/v1/D19-1018).
- [31] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, and T.-S. Chua, "Neural collaborative filtering," in *Proc. 26th Int. Conf. World Wide Web*, Switzerland, Apr. 2017, pp. 173–182, doi: [10.1145/3038912.3052569](https://doi.org/10.1145/3038912.3052569).
- [32] C. Zhao, C. Li, R. Xiao, H. Deng, and A. Sun, "CATN: Cross-domain recommendation for cold-start users via aspect transfer network," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Jul. 2020, pp. 229–238, doi: [10.1145/3397271.3401169](https://doi.org/10.1145/3397271.3401169).
- [33] X. Wang, D. Wang, C. Xu, X. He, Y. Cao, and T.-S. Chua, "Explainable reasoning over knowledge graphs for recommendation," in *Proc. Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2019, pp. 950–958.
- [34] B. Hawashin, S. Alzubi, A. Mughaid, F. Fotouhi, and A. Abusukhon, "An efficient cold start solution for recommender systems based on machine learning and user interests," in *Proc. 7th Int. Conf. Softw. Defined Syst. (SDS)*, France, Apr. 2020, pp. 220–225, doi: [10.1109/SDS49854.2020.9143953](https://doi.org/10.1109/SDS49854.2020.9143953).
- [35] R. Hu and P. Pu, "Enhancing collaborative filtering systems with personality information," in *Proc. 5th ACM Conf. Recommender Syst.*, Oct. 2011, pp. 197–204, doi: [10.1145/2043932.2043969](https://doi.org/10.1145/2043932.2043969).
- [36] H. Wang, Y. Zuo, H. Li, and J. Wu, "Cross-domain recommendation with user personality," *Knowl.-Based Syst.*, vol. 213, Feb. 2021, Art. no. 106664, doi: [10.1016/j.knsys.2020.106664](https://doi.org/10.1016/j.knsys.2020.106664).
- [37] Y. Wu, K. Li, G. Zhao, and X. Qian, "Personalized long- and short-term preference learning for next POI recommendation," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 4, pp. 1944–1957, Apr. 2022, doi: [10.1109/TKDE.2020.3002531](https://doi.org/10.1109/TKDE.2020.3002531).

- [38] W. Wang, C. Ye, P. Yang, and Z. Miao, "Research on movie recommendation model based on LSTM and CNN," in *Proc. 5th Int. Conf. Comput. Intell. Appl. (ICCIA)*, China, Jun. 2020, pp. 28–32, doi: [10.1109/ICCIA49625.2020.00013](https://doi.org/10.1109/ICCIA49625.2020.00013).
- [39] W. Li and B. Xu, "Aspect-based fashion recommendation with attention mechanism," *IEEE Access*, vol. 8, pp. 141814–141823, 2020, doi: [10.1109/ACCESS.2020.3013639](https://doi.org/10.1109/ACCESS.2020.3013639).
- [40] J. Deng, L. Cheng, and Z. Wang, "Attention-based BiLSTM fused CNN with gating mechanism model for Chinese long text classification," *Comput. Speech Lang.*, vol. 68, Jul. 2021, Art. no. 101182, doi: [10.1016/j.csl.2020.101182](https://doi.org/10.1016/j.csl.2020.101182).
- [41] N. A. Osman and S. A. M. Noah, "Sentiment-based model for recommender systems," in *Proc. 4th Int. Conf. Inf. Retr. Knowl. Manage. (CAMP)*, Malaysia, Mar. 2018, pp. 1–6, doi: [10.1109/INFRKM.2018.8464694](https://doi.org/10.1109/INFRKM.2018.8464694).
- [42] P. Liashchynskiy and P. Liashchynskiy, "Grid search, random search, genetic algorithm: A big comparison for NAS," 2019, *arXiv:1912.06059*.

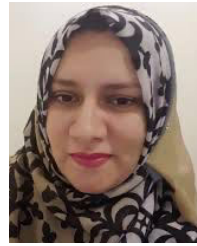


**RABIA KHAN** received the M.S. degree in software engineering from the National University of Sciences and Technology (NUST), Pakistan, in 2009, where she is currently pursuing the Ph.D. degree. From 2009 to 2014, she was a Faculty Member with NUST. She has more than eight research publications in international journals and conferences to her credit. Her research interests include data mining, natural language processing, and machine learning.

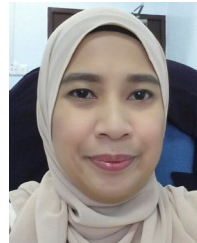


**NAIMA ILTAF** received the Ph.D. degree in software engineering from the National University of Sciences and Technology (NUST), Pakistan, in 2013. She is currently an Associate Professor with the Department of Computer Software Engineering, NUST. She is author or coauthor of more than 40 articles published in international journals and conferences. She is engaged with a few academic and industrial research projects as a PI and Co-I. Her research interests include data mining,

text mining, natural language processing, and recommender systems.



**RABIA LATIF** received the M.S. and Ph.D. degrees in information security from the National University of Sciences and Technology (NUST), Pakistan, in 2010 and 2016, respectively. She is currently an Assistant Professor with the College of Computer and Information Sciences, Prince Sultan University, Riyadh, Saudi Arabia. She is an active member of the Artificial Intelligence Data Analytics Research Laboratory, Prince Sultan University, Riyadh. She earned various security certifications, including CEH, CND, ECSA, VMware's Data Center Virtualization and Network Virtualization, and IBM Train the Trainer Security Workshop. Her research interests include cloud computing security, healthcare data security, web security, cyber security, and network security. Her professional career consists of activities ranging from the conference publication chair, the technical program committee chair, and a reviewer for several international journals and conferences.



**NOR SHAHIDA MOHD JAMAIL** received the Ph.D. degree in software engineering from Universiti Putra Malaysia. She is currently an Assistant Professor with Prince Sultan University, Riyadh, Saudi Arabia. She is very committed in the academic career. She was involved in the Machine Learning Research Group, Prince Sultan University, and also involved in big research project which collaborated with other universities. Her specialized are purely in software engineering, software process modeling, software testing, and cloud computing services. She involved in lecturing and handling classes to Diploma, bachelor's, and master's students. She is really active in research, where she had published chapter in books, journal publications, and conference papers. Her current research interests include cloud computing, quality in software engineering and software modeling, big data, machine learning, and more on commercialization of research products. Besides, she is the co-chair, the track chair, and a committee member in several international conference published by IEEE, Scopus indexed.

• • •