Generalized Cross-Multi Domains Representations using User Sentiments

Aabhaas Batra
aabhaas19001@iiitd.ac.in

Eeshaan Ravi Tivari
eeshaan19465@iiitd.ac.in

Jahnvi Kumari
jahnvi19469@iiitd.ac.in

Rishi Singhal
rishi19194@iiitd.ac.in

Soham Das
soham19477@iiitd.ac.in

Udit Narang
udit19120@iiitd.ac.in

1 Introduction & Motivation

In today's day and age, people spend more time on social media and streaming apps like netflix, instagram, spotify, etc. than they do with their social circles. With over 5 billion users of e-commerce websites, social media websites and streaming platforms, providing the best user experience is key to these companies. Hence, to keep the user retention high, these companies need to continuously innovate and optimize their recommendation systems. The goal of a recommender system is to provide personalized recommendations to users by predicting what they are most likely to prefer among a large set of items. Recommendation systems often rely on user-item interactions to make recommendations. But data sparsity in recommendation systems is a common problem. It makes it challenging for the recommendation system to accurately predict the user's preferences and make relevant recommendations. This may arise due to a user interacting with an item (say, buying a dress) in a domain different from what they usually interact in (say, buying books). However, cross-domain representations can be used to transfer the knowledge of user-item interaction in interacted domains to a sparse domain. While work has been done on bi-domain models, generalizing them to k domains can help find the optimal number of domains to consider for representation learning. By considering multiple domains and user sentiments, the model can learn to generalize across a wider range of user-item interactions and make more informed recommendations. This method can help improve the accuracy and relevance of recommendations, even in sparser domains with limited user-item interaction data.

2 Problem Statement

The main objective of the project is to carry out the task of generating generalized multi-cross-domains representations while taking into account user reviews and sentiments which would be useful in a cross-domain recommendation system. Since most of the cross-domain work has been done across two domains. Thus, in the proposed task, given an adjacency list T of items (from multiple K domains where $K \geq 2$) and common users across k domains, along with their metadata M about the user-item interactions such as comments, ratings, and item descriptions, we would generate cross-domain representations for recommender systems to capture the essence of user-item interaction in a better manner. To do so, we intend to make use of a weighted graph that would represent the user-item interactions along with their sentiment scores.

The problem statement we are trying to work on is novel since there's not much-existing literature on generalized multi-cross-domain representations. Also, adding user sentiments as a weighted graph would help provide a deeper insight into cross-domain recommendations.

3 Literature Survey

As part of existing work, various different techniques have been proposed to build an efficient cross-domain recommendation system based on user-item interactions.

Cao et al. [1] proposes a model to create disentangled cross-domain representations for the user given two different domains, a shared user set, and an adjacency list to determine user-item interaction. Their model, disenCDR, disentangles shared user information and domain specific information for the two domains. For this purpose, they use a variational bipartite graph encoder and carefully formulated mutual information based regularizers. The paper carries out its experiments on Amazon review dataset.

Liu et al. [6] have proposed a Joint Spectral Convolutional Network(JSCN) to capture the high-order connectivity information in the field of cross-domain recommendation. The

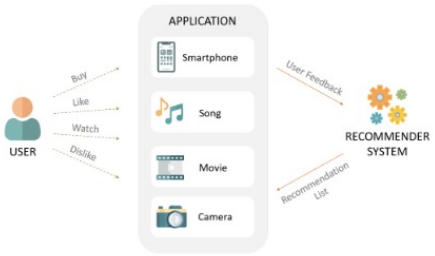

Figure 1. A basic recommender system's flowchart [2]

proposed JSCN model, extracts higher-order connections by carrying out multi-layer spectral convolutions on different user-item bipartite graphs to transfer information across domains by learning a domain-invariant user representation. Finally, it is evaluated on Amazon Review Dataset to achieve state-of-the art performance in terms of precision and MAP scores.

Another paper, proposed by Wang et al. [8] leverages sentiment analysis to bridge the gap between the source and target domains. At first, latent features are extracted in the source domain. Then, the sentiment features are mapped to the target domain using a cross-domain mapping function which is learned using a transfer learning approach. The proposed model is then evaluated against the Amazon Review dataset and achieved low RMSE values.

The work proposed by Li et al. [4] leveraged the use of latent embeddings of features and user preferences across domains instead of explicit information between the source and the target domain to capture hidden complex interactions. It also focuses on dual transfer learning mechanism to enable learning in both source and target domains based on learnt knowledge instead of trivial unidirectional learning. They've shown RMSE, MAE, ho, precision scores using a large-scale anonymized dataset obtained from a European online recommendation service.

Lu et al. [7] proposed a model to capture the consistency of source and target domains in collaborative filtering settings. The smaller the variance of the empirical prediction error produced by their model, the more likely this user is consistent with those from other domains.. Thus, they embed this criterion into a boosting framework to perform selective knowledge transfer.

Zhao et al. [9] proposed a unified multi-task model through the construction of a cross-domain preference matrix. It models the interactions of different domains as a whole. They added a propagation layer to their model Preference Propagation GraphNet which captures how user preferences propagated in the graph. They also defined a joint objective for different domains.

Lie et al. [5] The paper proposes a new method for cross-domain recommendation called BiTGCF, which is based on a Graph Collaborative Filtering network as the underlying model. BiTGCF uses a bi-directional transfer learning approach and achieves improved performance compared to existing state-of-the-art methods on several benchmark datasets.

4 Novelty Proposed

Our work will explore two new avenues. We will use a sentiment-weighted user-item interaction graph to train our model so that the feedback of the user is better captured in the representations learnt. We will also generalize cross-domain representation learning to K domains instead of being limited to just two. We shall propose methods to compare the performance of our multi domain models with differing number of source domains. This will aid in comparing our results with previous literature (predominantly on two domains).

Our research can help find a trend of performance with increasing number of source domains, as well as an optimal number of domains. It will help better identify possible roadblocks for research in multi-domain models for recommender systems.

5 Methodology

5.1 Amazon Review Dataset

The Amazon review dataset [3] is a collection of user reviews and ratings for products sold on Amazon. It contains product ID, user ID, review text, rating, and other metadata. It's commonly used in cross-domain recommendation systems as it provides a diverse set of products from various categories. It's a valuable resource for researchers and data scientists in the field of recommendation systems.

5.2 Data Preprocessing

For conducting the baseline experiments, two domains Digital Music and Cell Phones and Accessories, have been used. The above two domains have been used because of their smaller size and computation limitations. The Digital Music dataset contains 169,781 data samples, and the Cell Phones and Accessories dataset contains 10,063,255 data samples. The samples which don't have any review were removed from the dataset. Some users have rated the same item more than once, and only one of the ratings has been retained after removing the duplicate ratings. The above steps reduced the data samples to 1,120,011 for the Cell phones and accessories domain and 145,146 for the Digital music domain.

Preprocessing steps have been done on the reviews for each of the above domains, which include lowercasing, removal of HTML tags and URLs, punctuation removal, and extra white space removal. All the experiments have been conducted on the preprocessed dataset.

5.3 Solution Proposed

Figure[2] illustrates the methodology that we aim to follow. From the user-item pair in the dataset, a weighted bipartite graph containing two node classes User(u) and Item(i), is constructed by creating an edge u⟶i whenever user u interacts with the item i. The weight W(u->i) will represent the sentiment score S(u->i) of the user u to the item i. Thus incorporating this information in the form of graphs from different domains, we will finally get a shared cross-domain embedding containing three parts: shared user representation,

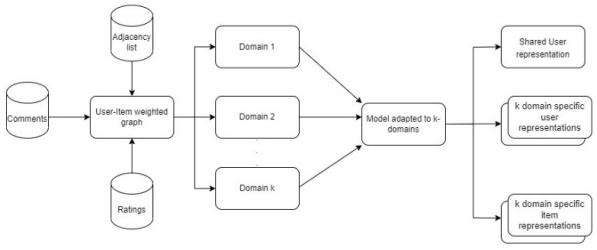

Figure 2. A flowchart of our proposed methodology

	No. of unique users	No. of unique items	No. of rows
Cell phones intersection dataset	335	2817	3517
Digital music intersection dataset	335	2461	4281
Movies intersection dataset	335	6009	9977

Figure 3. Statistics of the generated intersection datasets of each domain

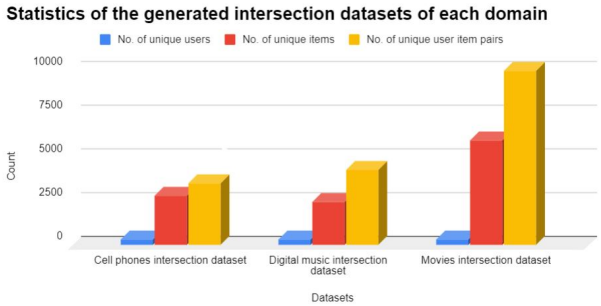

Figure 4. Plot of the statistics of the generated intersection datasets of each domain

domain-specific user representations, and domain-specific item representations.

5.4 Evaluation Metrics

To evaluate our learned representations, we will use HR (hit ratio) and NDCG (Normalized Discounted Cumulative Gain) to evaluate the quality of recommendations.

NDCG Metric

$$NDCG = \frac{DCG}{IDCG}$$

$$DCG = \sum_{i=1}^{R} \frac{Rel_i}{log(r_i + 2)}$$

where, Rel_i means Relevance score, r_i means rank and R means the total number of top K predictions.
And to compute IDCG we simply sort the relevance scores and extract the top K recommendations and use the same DCG metric.

Hit Rate

$$Hit{-}Rate = \begin{cases} 1, & \text{if relevant item lies in top-k recommendation} \\ 0, & \text{otherwise} \end{cases}$$

6 Experiments

6.1 K-Domains

6.1.1 Dataset Formulation.

Initially, we had datasets for three domains, i.e., cell phones, digital music, and movies. All three data sets had columns: UserID, ProductID, and the respective user's rating for that product. In the raw data sets, UserID was in the form of an alphanumeric string, ProductID was in the form of a 10-digit integer, and the rating for each product ranged from 1 to 5. Our aim was first to find the common users who had given ratings for products in each domain and then create datasets for each domain that had the intersection data of the common users and that specific domain, respectively. To achieve our aim, we first removed the tuples of the products with less than ten ratings from each data set. Then we found the common users by finding the common UserIDs that existed in the datasets of all domains. After all the iterations, we found 681 common users from all the domains.

Now to generate the intersection datasets, we, one by one, picked the dataset of each domain and found the intersection of it with the common users' data based on ProductID. This way, we could generate data sets for each domain with the intersection data of the common users and that specific domain, respectively. Each intersection dataset has three columns: [userID, itemID, rating]. All three entities are in the integer format as we converted the old format values to integral values in our code. 'rating' entity values range from 1 to 5. Figure[4] shows the statistics of the generated intersection datasets. The code that we have written can work for K domains. But, we currently have datasets available for three domains only. So, we have worked on them and generated our results from the presently available datasets.

6.1.2 K-Domains Architecture Formulation.

We have k interaction datasets $D^{X_i} = (U, V^{X_i}, E^{X_i})$ where $X_i \in X$. U is the shared user set, V_i^X is the set of items in X_i, and E_i^X is the edge set in that domain. These graphs are represented by binary interaction matrices $A_i^X \in \{0,1\}^{|U| x |V_i^X|}$. The objective is to learn Z_u^S, the shared user representation, the domain specific user representations $Z_u^{X_i}$, and the domain specific item representations $Z_v^{X_i}$, such that the representations are disentangled.

To do so, we make use of three modules.

Embedding Layer The first module is the Embedding layer. It initializes the shared feature matrix for the users, $U^S \in \mathbb{R}^{|U|xF}$, the domain specific feature matrices for the users, $U_i^X \in \mathbb{R}^{|U|xF}$, and the domain specific feature matrices for the items, $V_i^X \in \mathbb{R}^{|V_i^X|xF}$.

Variational Bipartite Graph Encoder The user item graph is bipartite in nature. The users, therefore, are always connected by an even number of hops. We take into account the one-hop neighbours of the users while creating our representations in addition to the user-item interactions. First, we create intermediate representations for the users and items, using which we estimate the distributions from which to sample the final representations. The procedures to generate the domain specific user and item representations are analogous to one another, hence we will take the example of the domain specific user representations in X_i. Let $\hat{U}^i$ be the intermediate domain specific user representation for X_i. Then,

$$\hat{U}_i^X = ReLU(Norm((A_i^X)^T)U_i^X W_u^{X_i})$$

, where Norm(.) refers to row normalization. $W_u^{X_i} \in \mathbb{R}^{FxF}$ is a parameter matrix. To get the final embeddings, we estimate the mean and standard deviation for the representations.

$$\mu_u^{X_i} = ReLU((ReLU(Norm(A^{X_i})\hat{U}^{X_i}\hat{W}_{u,\mu}^{X_i}) \oplus U^{X_i})W_{u,\mu}^{X_i})$$

,

$$\sigma_u^{X_i} = SoftPlus((ReLU(Norm(A^{X_i})\hat{U}^{X_i}\hat{W}_{u,\sigma}^{X_i}) \oplus U^{X_i})W_{u,\sigma}^{X_i})$$

The final representation is then sampled from the gaussian distribution as

$$Z_u^{X_i} \sim \mathcal{N}(\mu_u^{X_i}, (diag(\sigma_u^{X_i}))^2)$$

To estimate Z_u^S, we first generate $\bar{\mu}_u^{X_i}$ and $\bar{\sigma}_u^{X_i}$ by passing U^S as input. Then

$$\mu_u^S = \sum_{i=1}^{k} \lambda_u^{X_i} \odot \bar{\mu}_u^{X_i}$$

$$\sigma_u^S = \sum_{i=1}^{k} \lambda_u^{X_i} \odot \bar{\sigma}_u^{X_i}$$

$$\lambda_{u_i}^{X_j} = \frac{N_{u_i}^{X_j}}{\sum_{l=1}^{k} N_{u_i}^{X_l}}$$

$$Z_u^S \sim \mathcal{N}(\mu_u^S, (diag(\sigma_u^S))^2)$$

where $N_{u_i}^{X_j}$ denotes the number of 1-hop neighbours in domain X_j of user u_i. This controls the contribution ratios of different domains.

Generation and Inference The decoders aim to reconstruct observed interactions(user-item map). Given a user-items tuple $(u_i, v_1, v_2 \dots v_k)$, the likelihood of the joint distribution to be maximized is:

$$p_\theta(u, v^{X_1}, \dots, v^{X_k}) = \int \cdots \int p_{\theta^{X_1}}(A^{X_1}|Z_u^S, Z_u^{X_1}, Z_v^{X_1}) \dots p_{\theta^{X_k}}(A^{X_k}|Z_u^S, Z_u^{X_k}, Z_v^{X_k}) p(Z_u^S) p(Z_u^{X_1}) \dots p(Z_u^{X_k}) p(Z_v^{X_1}) \dots p(Z_v^{X_k})\, dZ_u^S dZ_u^{X_1} \dots dZ_u^{X_k} dZ_v^{X_1} \dots dZ_v^{X_k}$$

Here, the prior distributions are assumed to be normal Gaussian distributions, and the remaining conditional probability terms are obtained as decoder outputs.

$$p_{\theta^{X_i}}(A_{j,k}^{X_i}|z_{u_j}^S, z_{u_j}^{X_i}, z_{v_k}^{X_i}) = Sigmoid(< z_{v_k}^{X_i}, z_{u_j}^S + z_{u_j}^{X_i} >)$$

where $< . >$ refers to inner product. Then in the inference step as explained in DisenCDR we try to factorize the approximated posterior distribution as:

$$q_\phi(Z_u^S, Z_u^{X_1}, \dots, Z_u^{X_k}, Z_v^{X_1}, \dots, Z_v^{X_k}|X_1, \dots, X_k) = q_{\phi_u^{X_1}}(Z_u^{X_1}|X_1) \dots q_{\phi_u^{X_k}}(Z_u^{X_k}|X_k) q_{\phi_v^{X_1}}(Z_v^{X_1}|X_1) \dots q_{\phi_v^{X_k}}(Z_v^{X_k}|X_k) q_{\phi_u^S}(Z_v^S|X_1, \dots, X_k)$$

For disentanglement, the last term must encode the domain-shared information while the others encode the domain-specific information.

Objective Function

$$L = \sum_{i=1}^{k}(D_{KL}(q(Z_u^{X_i}|X_i) \parallel p(Z_u^{X_i}))) + D_{KL}(q(Z_u^S|X_1, \dots, X_k) \parallel p(Z_u^S)) + \sum_{i=1}^{k}(D_{KL}(q(Z_v^{X_i}|X_i) \parallel p(Z_v^{X_i}))) - \sum_{i=1}^{k} E_{q(Z_u^{X_i}, Z_v^{X_i}|X_i)q(Z_u^S,|X_1,\dots,X_k)}(logp(A^{X_i}|Z_u^S, Z_u^{X_i}, Z_v^{X_i})) + \beta \sum_{i=1}^{k} D_{KL}(q(Z_u^S|X_1, \dots, X_k) \parallel q(\hat{Z}_u^S|X_i))$$

The first two summations encourage the generated posterior distributions to be close to the prior distributions. The third summation aims to maximize the log likelihood of the decoders. Finally, the last summation encourages Z_u^S to learn the shared information in k domains. $\hat{Z}_u^S|X_i$ are variational distributions attained from k additional VBGEs.

6.2 Sentiment

Sentiment Calculation

On the preprocessed dataset, sentiment for each review has been calculated using the VADER sentiment analyzer. VADER sentiment analyzer gives four scores for each text namely negative, positive, neutral, and compound. We have used compound score for our experiments. The compound

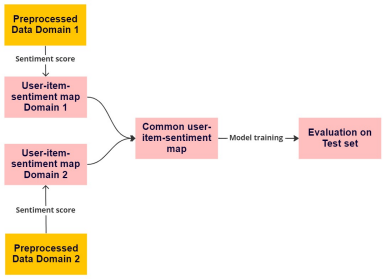

Figure 5. A flowchart of Sentiment-based methodology

score ranges from -1 to +1, we have transformed the compound score to 0 to +1. This results in a user-item map that is then used for model training.

After getting the user-item map along with their scaled sentiment scores, we find the common users-intersection dataset among the 2 domains and divide it into train and test splits for each cross-domain pair(source and target) using the methodology proposed in [5].

Sentiment Architecture Formulation

We have 2 interaction datasets $D^X = (U, V^X, E^X)$ and $D^Y = (U, V^Y, E^Y)$. U is the shared user set between the two domains X and Y, V^X and V^Y is the set of items in the domains X and Y respectively, and E^X and E^Y is the edge set in the domains X and Y respectively. These graphs are represented by sentiment-based interaction matrices $A^X \in [0,1]^{(|U|x|V^X|)}$ and $A^Y \in [0,1]^{(|U|x|V^Y|)}$ for the domains X and Y respectively, where A_{ij} represents the sentiment corresponding to the interaction of user i with the item j. The objective is to learn Z_u^S, the shared user representation, the domain specific user representations Z_u^X and Z_u^Y, and the domain specific item representations Z_v^X and Z_v^Y, such that the representations are disentangled.
To do so, we make use of three modules.

Embedding Layer The first module is the Embedding layer. It initializes the shared feature matrix for the users, $U^S \in \mathbb{R}^{|U|xF}$, the domain specific feature matrices for the users, $U^X \in \mathbb{R}^{|U|xF}$ and $U^Y \in \mathbb{R}^{|U|xF}$, and the domain specific feature matrices for the items, $V^X \in \mathbb{R}^{|V^X|xF}$ and $V^Y \in \mathbb{R}^{|V^Y|xF}$.

Variational Bipartite Graph Encoder The user item graph is bipartite in nature. The users are indirectly connected by an even number of hops. We take into account the two-hop neighbours of the users while creating our representations in addition to the user-item interactions. First, we create intermediate representations for the users and items, using which we estimate the distributions from which to sample the final representations. The procedures to generate the domain specific user and item representations are analogous to one another, hence we will take the example of the domain specific user representations in X. Let $\hat{U}$ be the intermediate domain-specific user representation for X. Then,

$$\hat{U}^X = ReLU(Norm((A^X)^T)U^X W_u^X)$$

, where Norm(.) refers to row normalization. $W_u^X \in \mathbb{R}^{FxF}$ is a parameter matrix. To get the final embeddings, we estimate the mean and standard deviation for the representations.

$$\mu_u^X = ReLU((ReLU(Norm(A^X)\hat{U}^X\hat{W}_{u,\mu}^X) \oplus U^X)W_{u,\mu}^X)$$

,

$$\sigma_u^X = SoftPlus((ReLU(Norm(A^X)\hat{U}^X\hat{W}_{u,\sigma}^X) \oplus U^X)W_{u,\sigma}^X)$$

The final representation is then sampled from the gaussian distribution as

$$Z_u^X \sim \mathcal{N}(\mu_u^X, (diag(\sigma_u^X))^2)$$

To estimate Z_u^S, we first generate $\bar{\mu}_u^X$, $\bar{\sigma}_u^X$ and $\bar{\mu}_u^Y$, $\bar{\sigma}_u^Y$ by passing U^S as input. Then

$$\mu_u^S = \lambda_u \odot \bar{\mu}_u^X + [1 - \lambda_u] \odot \bar{\mu}_u^Y$$

$$\sigma_u^S = \lambda_u \odot \bar{\sigma}_u^X + [1 - \lambda_u] \odot \bar{\sigma}_u^Y$$

$$\lambda_{u_i} = \frac{N_{u_i}^X}{N_{u_i}^X + N_{u_i}^Y}$$

$$Z_u^S \sim \mathcal{N}(\mu_u^S, (diag(\sigma_u^S))^2)$$

where $N_{u_i}^X$ denotes the number of 1-hop neighbours in domain X of user u_i. This controls the contribution ratios of different domains.

Generation and Inference The decoders aim to reconstruct observed interactions(user-item map). Given a user-items tuple $(u_i, v_1, v_2 \dots v_k)$, the likelihood of the joint distribution to be maximized is:

$$p_\theta(u, v^X, v^Y) = \int p_{\theta^X}(A^X|Z_u^S, Z_u^X, Z_v^X)p_{\theta^Y}(A^Y|Z_u^S, Z_u^Y, Z_v^Y) p(Z_u^S)p(Z_u^X)p(Z_u^Y)p(Z_v^X)p(Z_v^Y) dZ_u^S dZ_u^X dZ_u^Y dZ_v^X dZ_v^Y$$

Here, the prior distributions are assumed to be normal Gaussian distributions, and the remaining conditional probability terms are obtained as decoder outputs.

$$p_{\theta^X}(A_{i,j}^X|z_{u_i}^S, z_{u_i}^X, z_{v_j}^X) = Sigmoid(< z_{v_j}^X, z_{u_i}^S + z_{u_i}^X >)$$

$$p_{\theta^Y}(A_{j,k}^Y|z_{u_j}^S, z_{u_j}^Y, z_{v_k}^Y) = Sigmoid(< z_{v_k}^Y, z_{u_j}^S + z_{u_j}^Y >)$$

Then in the inference step as explained in DisenCDR we try to factorize the approximated posterior distribution as:

$$q_\phi(Z_u^S, Z_u^X, Z_u^Y, Z_v^X, Z_v^Y|X, Y) = q_{\phi_u^X}(Z_u^X|X)q_{\phi_u^Y}Z_u^Y|Y) q_{\phi_v^X}(Z_v^X|X)q_{\phi_v^X}Z_v^Y|Y)q_{\phi_u^S}(Z_v^S|X, Y)$$

For disentanglement, the last term must encode the domain-shared information while the others encode the domain-specific information.

Objective Function

$$L = D_{KL}(q(Z_u^X|X) \parallel p(Z_u^X)) + D_{KL}(q(Z_v^X|X) \parallel p(Z_v^X))+ D_{KL}(q(Z_u^Y|Y) \parallel p(Z_u^Y)) + D_{KL}(q(Z_v^Y|Y) \parallel p(Z_v^Y))+ D_{KL}(q(Z_u^S|X, Y) \parallel p(Z_u^S)) + \beta D_{KL}(q(Z_u^S|X, Y) \parallel q(\hat{Z}_u^S|X))+ \beta D_{KL}(q(Z_u^S|X, Y) \parallel q(\tilde{Z}_u^S|X)) + BCELoss(p_{\theta^X}, A^X)+ BCELoss(p_{\theta^Y}, A^Y)$$

The first four terms encourage the generated posterior distributions to be close to the prior distributions for both domain specific and shared case. The fifth term,encourages Z_u^S to learn the shared information in the 2 domains domains. The sixth and seventh terms are variational distributions attained from the 2 VBGEs. Finally, the last two terms aim to minimize the BCE loss of the decoders.

Change in the NDCG Metric

DisenCDR [1] has made use of the NDCG metric to test the relevance of the recommendations made by their proposed model. In this metric, they assumed the relevance score to be binary, i.e., 0 or 1, where 0 signifies an irrelevant item and 1 signifies a relevant item. However, it do not take into account the user sentiments while testing their model's performance.

Thus, to incorporate sentiments, we have updated the binary relevance scores to the user-sentiment scores that we get from the VADER sentiment analyzer leading to an updated sentiment-based NDCG metric.

Using the updated NDCG metric, we are able to establish a baseline where we test the DisenCDR model which has not been trained using user sentiments. Thus, we hypothesize that once we incorporate the sentiment scores during the training phase, then we will be able to get better NDCG scores in comparison to the above-explained baseline.

Furthermore, in the testing phase, the original paper developed non-existing 99 user-item edges apart from the original user-item edge. Now since they didn't take into account the user sentiments, so these non-existing edges had a relevance score of 0. But in the sentiment-based NDCG metric, these edge weights can be between 0 and 1. Thus, making it as a hyperparameter for our case.

Training and Testing Phase

Weights	Neg Label Values	Source Hit Rate	Source NDCG	Target Hit Rate	Target NDCG
		DisenCDR	DisenCDR	DisenCDR	DisenCDR
Sentiment - Scaled	0	0.0787	0.0347	0.2158	0.0971
	0.5	0.0820	0.0760	0.1216	0.1141
	Mean	0.0492	0.0482	0.1147	0.1103
Sentiment - 1	0	0.1148	0.0495	0.1298	0.0615
	0.5	0.1541	0.1381	0.1987	0.2111
	Mean	0.1178	0.1178	0.1389	0.1338
Rating Scaled	0	0.1209	0.0550	0.1254	0.0629
	0.5	**0.1634**	**0.1504**	**0.2474**	**0.2274**
	Mean	0.1051	0.1031	0.1076	0.1062

Figure 6. Sentiment-based DisenCDR for different Sentiment Weights and Negative Labels for cellphones as source domain and digital music as target domain

Finally, using the train and test files we trained the DisenCDR model; however, while testing we have changed the NDCG metric to incorporate user sentiments by using the sentiments as relevance scores.

7 Results & Analysis

7.1 Sentiment Results & Analysis

As part of introducing sentiments in cross-domain recommendation, we did an extensive set of experiments. We varied the negative labels sentiments scores by choosing among $[0, 0.5, mean - values]$. In this way we varied the non-interactive edges weights in both training and testing by weighting the interactive edge by the original sentiment score for the given user-item interaction. Furthermore, we varied the sentiment labels as scaled versions of 1, Vader-Sentiment labels and the original customer ratings. Finally, for the training purpose of the sentiment-based DisenCDR we have set the number of epochs to 50 and got the following ndcg and hit rate values for both the target and source domains(cellphones and digital music).

We can clearly observe from Fig-6,7 that the training on the original scaled ratings of the user-item pair with negative labels given a score of 0.5 resulted in the highest metric values.

We observe that using user ratings as sentiments are giving better results than sentiment one and VADER sentiment. This is because user ratings depict a true picture of the user experience in a user-item interaction. Using VADER sentiments is not better results than the user-ratings, which may be due to the fact that the amazon dataset contains reviews in foreign languages also like Spanish, French, etc. Fig-8 shows the compound score returned by the VADER sentiment analyzer of the same sentence written in different languages. VADER sentiment analyzer is giving a high

Weights	Neg Label Values	Source Hit Rate	Source NDCG	Target Hit Rate	Target NDCG
		DisenCDR	DisenCDR	DisenCDR	DisenCDR
Sentiment - Scaled	0	0.1815	0.0821	0.1672	0.0732
	0.5	0.1233	0.1167	0.1443	0.1360
	Mean	0.1027	0.0989	0.1148	0.1124
Sentiment - 1	0	0.1280	0.0517	0.1450	0.0665
	0.5	0.2232	0.2024	0.1582	0.1319
	Mean	0.1230	0.1186	0.1631	0.1631
Rating Scaled	0	0.1254	0.0533	0.1046	0.0522
	0.5	**0.1271**	**0.1152**	**0.1601**	**0.1463**
	Mean	0.0829	0.0820	0.1083	0.1065

Figure 7. Sentiment-based DisenCDR for different Sentiment Weights and Negative Labels for digital music as source domain and cellphones as target domain

Sentence	VADER Score
This is the best product that I have ever used in my life.	0.64
Este es el mejor producto que he usado en mi vida. (Spanish)	0
C'est le meilleur produit que j'ai jamais utilisé dans ma vie. (French)	0

Figure 8. Inefficiency of VADER sentiment analyzer to capture sentiment of sentences written in foreign languages

score for the sentence written in English language while it is giving 0 score for the same sentence written in Spanish and French language. Futhermore, using sentiment equal to one to all user-item interactions isn't correct as different users can have different experiences with a product.

Furthermore, we have also varied different hyper-parameters including learning rate, learning decay, disentanglement factor(beta) and dropout as shown in Fig-9,10,11, to try to improve on our previous results of ratings-scaled and negative labels of 0.5. For hyper-parameters tuning we have opted for a greedy approach where we vary one parameter and fix the others.

After performing the hyper-parameters tuning we also compared our results with 2 baselines - base DisenCDR model and Bi-TGCF model by keeping the negative labels as 0.5 in the NDCG metric computation part for both of them and without changing the training part. The comparisions for the same are mentioned in Fig-12,13.

Weights	Learning rate	Source Hit Rate	Source NDCG	Target Hit Rate	Target NDCG
DisenCDR (Rating - Scaled with neg label = 0.5)	0.0001 (decay=0.1)	0.0850	0.0785	0.1082	0.0991
	0.00001 (decay=0.1)	0.0882	0.0816	0.1065	0.0977
	0.0001 (decay=0.05)	**0.0752**	**0.0690**	**0.1203**	**0.1096**
	0.00001 (decay=0.05)	0.0882	0.0816	0.1065	0.0977

Figure 9. Finetuning the model by varying learning rate and decay

Weights	Beta rate	Source Hit Rate	Source NDCG	Target Hit Rate	Target NDCG
DisenCDR (Rating - Scaled with neg label = 0.5)	0.5	0.0852	0.0790	0.1182	0.112
	0.7	0.0852	0.0793	0.1164	0.1098
	1	**0.0787**	**0.0731**	**0.1216**	**0.1143**
	1.5	0.0854	0.0790	0.1182	0.1112

Figure 10. Finetuning the model by varying beta factor

Weights	Dropout rate	Source Hit Rate	Source NDCG	Target Hit Rate	Target NDCG
DisenCDR (Rating - Scaled with neg label = 0.5)	0.5	0.1176	0.1072	0.1976	0.1820
	0.2	0.1634	0.1503	0.2595	0.2387
	0.1	0.1601	0.1466	0.2629	0.2418
	0	**0.1961**	**0.1809**	**0.2698**	**0.2490**

Figure 11. Finetuning the model by varying dropout

Weights	Source Hit Rate	Source NDCG	Target Hit Rate	Target NDCG
DisenCDR(Rating - Scaled)	**0.2165**	**0.1990**	**0.1863**	**0.1695**
DisenCDR(Base - Model)	0.2232	0.2024	0.1582	0.1319
Bi-TCGF	0.1221	0.1119	0.1603	0.0758

Figure 12. Comparisions of our best model with base DisenCDR and Bi-TGCF models for digital music-cellphones setup

Weights	Source Hit Rate	Source NDCG	Target Hit Rate	Target NDCG
DisenCDR(Rating - Scaled)	**0.1961**	**0.1809**	**0.2698**	**0.2490**
DisenCDR(Base - Model)	0.1541	0.1381	0.1987	0.2111
Bi-TCGF	0.1353	0.0643	0.2388	0.1543

Figure 13. Comparisions of our best model with base DisenCDR and Bi-TGCF models for cellphones-digital music setup

Source Domain	Target Domain	Target Domain Hit Rate	Target Domain NDCG
Cellphones	Digital Music	0.1289	0.0598
Digital Music	Cellphones	0.1792	0.0778
Digital Music	Movies	0.2577	0.1235
Movies	Digital Music	0.1467	0.0642
Cellphones	Movies	0.2268	0.1093
Movies	Cellphones	0.1321	0.0623
	Mean values	0.1785666667	0.08281666667

Figure 14. Baseline results for 2 domain recommendations using Bi-TGCF (default settings)

Source Domain	Target Domain	Target Domain Hit Rate	Target Domain NDCG
Cellphones	Digital Music	0.1377	0.0584
Digital Music	Cellphones	0.1509	0.096
Digital Music	Movies	0.1391	0.0671
Movies	Digital Music	0.1111	0.0476
Cellphones	Movies	0.0927	0.0415
Movies	Cellphones	0.1415	0.0822
	Mean values	0.1288333333	0.06546666667

Figure 15. Baseline results for 2 domain recommendations using disenCDR (default settings)

7.2 K-Domains Results & Analysis

We have generalized our model to K domains and have currently evaluated our model on 3-domain data. We have considered the "cellphones", "digital music" and "movies" domains for our work. In Fig 16, we have compared the results of our extensive experiments on the model's hyperparameters. Due to a lack of three domain models for this problem, we compare the mean NDCG of the three domains from our model with the mean target NDCG of the results from the two domain models, trained on the six possible permutations of source and target domains from the three available domains. Likewise, we also compare the mean HIT values. The results for the two domain models are shown in Figs 14 and 15. There are many hyperparameters to be tuned, so we do a greedy search for the optimal settings, fixing one hyperparameter value at a time while keeping the others constant. The default values for the parameters are in Fig 17. Unless stated otherwise, the hyperparameters assume their default values.

Experiment Config	Domain 0 NDCG	Domain 0 Hit Rate	Domain 1 NDCG	Domain 1 Hit Rate	Domain 2 NDCG	Domain 2 Hit Rate	Mean NDCG	Mean Hit Rate
default	0.0627	0.1415	0.069	0.1511	0.047	0.1082	0.0595666666	0.1336
dropout 0	0.0758	0.1698	0.0447	0.1066	0.0917	0.1701	0.0707333333	0.1488333333
dropout 0, hdim=fdim=32	0.076	0.1415	0.0558	0.1155	0.0589	0.1134	0.0635666666	0.1234666667
dropout 0, hdim=fdim=64	0.0788	0.1509	0.0621	0.1288	0.061	0.134	0.0673	0.1379
dropout 0, hdim=fdim=256	0.0865	0.1509	0.0806	0.1822	0.1547	0.2938	0.1072666667	0.2089666667
dropout 0, hdim=fdim=512	0.0898	0.1698	0.089	0.1777	0.1287	0.2577	0.1025	0.2017333333
dropout 0, hdim=fdim=256, beta=0.1	0.0717	0.1509	0.0587	0.1377	0.1502	0.268	0.0935333333	0.1855333333
dropout 0, hdim=fdim=256, beta=0.5	0.0827	0.1698	0.06971	0.1644	0.1287	0.2835	0.0937033333	0.2059
dropout 0, hdim=fdim=256, beta=1.5	0.0938	0.1792	0.0811	0.1688	0.1534	0.2989	0.1094333333	0.2156333333
dropout 0, hdim=fdim=256, seed=0	0.0411	0.1132	0.0919	0.1777	0.1686	0.3247	0.1005333333	0.2052
dropout 0, hdim=fdim=256, n_epochs=100	**0.0877**	**0.1792**	**0.0877**	**0.1866**	**0.1695**	**0.2938**	**0.1149666667**	**0.2198666667**

Figure 16. Hyperparameter tuning using greedy search on 3 domain cross recommendation model

Hyperparameter	Default value
Feature dimension	128
Hidden dimension	128
dropout	0.3
learning rate	0.001
beta	0.9
num_epochs	50
seed	2040

Figure 17. Default hyperparameter values

In conclusion, we can clearly say that making use of user sentiments is an essential aspect of a cross-domain recommendation as explained above and also visible in the final comparison tables.

From the experiments, we see that increasing the hidden and feature dimensions to 256 increases the performance across domains. The rest of the hyperparameters work best at their default values. For our baseline models, we consider BiTGCF and DisenCDR. Both are 2 domain models, taking as input data from 2 domains only. Therefore, the results for the three domains cannot be evaluated from only one trained instance of either model. We see that in the 2 domain setting, BiTGCF performs slightly better than disenCDR for our dataset in terms of mean NDCG and HIT. This can be attributed to multiple reasons. The default settings for BiTGCF might be better suited to smaller datasets than DisenCDR. It is also worth noting that while three disenCDR instances had to be trained to get the results for the three domains in their six possible settings, six BiTGCF instances had to be trained for the same purpose since BiTGCF does not treat the source and target domains in an analogous fashion. After sufficiently tuning our 3 domain model, we see the performance improve significantly in all three domains, specifically when dropout is set to 0 and the hidden dimension and feature

dimension are increased. The performance also increases for this setting as the number of epochs is increased. In terms of both mean NDCG and HIT metrics, our model outperforms the results for the three available domains obtained using the existing two domain models. Therefore, on top of providing embeddings for k domains from a single trained model, our formulation extracts relevant information from all domains to improve the predictive performance for each domain.

8 Contribution

- Aabhaas, Jahnvi, Soham - K-domains methodology setup and experiments.
- Eeshaan, Rishi, Udit - Sentiment based CDR methodology setup and experiments.
- Equal contribution in PPT, Video and Report making.

9 Future Work

In this work, we have proposed two mutually exclusive approaches, one using k-domains and the other using the sentiment of the user-item interaction. Future work can combine both approaches into one single approach that takes into account both multiple domains and sentiment.

Furthermore, we would also like to take into account more sophisticated sentiment analysis models that work well for multiple languages.

Also, for the k-domains formulation we would like to try out the model for higher values of k. For this, we would need to acquire and work on larger datasets such that the joint intersection between the domains has sufficient number of data points.

References

[1] Jiangxia Cao, Xixun Lin, Xin Cong, Jing Ya, Tingwen Liu, and Bin Wang. 2022. DisenCDR: Learning Disentangled Representations for Cross-Domain Recommendation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 267–277.

[2] Diana Ferreira, Sofia Silva, António Abelha, and José Machado. 2020. Recommendation system using autoencoders. *Applied Sciences* 10, 16 (2020), 5510.

[3] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *proceedings of the 25th international conference on world wide web*. 507–517.

[4] Pan Li and Alexander Tuzhilin. 2020. Ddtcdr: Deep dual transfer cross domain recommendation. In *Proceedings of the 13th International Conference on Web Search and Data Mining*. 331–339.

[5] Meng Liu, Jianjun Li, Guohui Li, and Peng Pan. 2020. Cross domain recommendation via bi-directional transfer graph collaborative filtering networks. In *Proceedings of the 29th ACM international conference on information & knowledge management*. 885–894.

[6] Zhiwei Liu, Lei Zheng, Jiawei Zhang, Jiayu Han, and S Yu Philip. 2019. JSCN: Joint spectral convolutional network for cross domain recommendation. In *2019 IEEE international conference on big data (big data)*. IEEE, 850–859.

[7] Zhongqi Lu, Erheng Zhong, Lili Zhao, W Xiang, Weike Pan, and Qiang Yang. 2012. Selective transfer learning for cross domain recommendation. Proceedings of the 2013 SIAM International Conference on Data Mining (SDM), pp. 641-649. (2012).

[8] Yongpeng Wang, Hong Yu, Guoyin Wang, and Yongfang Xie. 2020. Cross-domain recommendation based on sentiment analysis and latent feature mapping. *Entropy* 22, 4 (2020), 473.

[9] Cheng Zhao, Chenliang Li, and Cong Fu. 2019. Cross-domain recommendation via preference propagation graphnet. In *Proceedings of the 28th ACM international conference on information and knowledge management*. 2165–2168.