Proceedings of the 2023 International Conference on Machine Learning and Automation
DOI: 10.54254/2755-2721/44/20230093

Applying self-attention model to learn both Empirical Risk Minimization and Invariant Risk Minimization for multimedia recommendation

Hanyu Zhao[1,5,9,†], Yangqi Huang[2,6,†], Kunqi Zhao[3,7,†], Sizhuo Wang[4,8,†]

[1]Bachelor of Engineering in Computer Science and Technology (Honors), Xiamen University Malaysia, Sepang Selangor Malaysia, 43900, Malaysia
[2]Bachelor of Economics in Finance (Honors), Xiamen University Malaysia, Sepang Selangor Malaysia, 43900, Malaysia
[3]Kunqi Zhao, Sauder School of Business, The University of British Columbia, Vancouver BC V6T 1Z2, Canada
[4]Faculty of Innovation Engineering, Macau University of Science and Technology, Wang Sizhuo, 999078, China

[5]CST2009155@xmu.edu.my
[6]FIN2009310@xmu.edu.my
[7]zhaokunqi@outlook.com
[8]1210000509@student.must.edu.mo
[9]corresponding author
[†]All the authors share the same contribution as the co-first author.

Abstract. Multimedia recommendation systems have many applications in our daily life. However, how accurately capture a customer's preference is an issue that is difficult to deal with. The proposed Invariant Risk Minimization (IRM) and Empirical Risk Minimization (ERM) are ways to learn a customer's preference. Still, both frameworks show some limitations: although ERM performs excellently in a single environment, it fails to generalize well when faced with multiple and new domains. On the other hand, IRM learns invariant features across heterogeneous environments, but it lacks theoretical guarantees and performs less effectively where the invariants are unclear. This paper proposes an **E**RM and **I**RM **O**ptimized **R**ating Framework (EIOR) as our final recommender model with direct rating scores. The EIOR enhances the accuracy and functionality of the multimedia recommendation systems by utilizing self-attention mechanisms to combine IRM and ERM with adjusted attention weights. Specifically, IRM learns invariant parts across different environments, while ERM learns variant parts. With self-attention, we can adaptively allocate attention weights for the two pieces and seek the optimal pair of attention weights based on the loss function. We demonstrate EIOR on a cutting-edge recommender model UltraGCN and use the open multimedia dataset of TikTok to finish all the experiments. The results validate the effectiveness of EIOR by comparing purely operating invariant representations alone with the framework of IRM.

Keywords: Invariant Risk Minimization (IRM), Empirical Risk Minimization (ERM), Self-attention Mechanisms, Invariant Learning

DOI: 10.54254/2755-2721/44/20230093

1. Introduction

The usual assumption of traditional machine learning methods is that the data for model training and testing are independent and identically distributed (IID). Here, the data for training and testing can be said to be In-Distribution (ID). In practical applications, the data obtained after the model is deployed and launched is often not completely controlled. That is to say, the data received by the model may be Out-of-Distribution samples, which can also be called abnormal samples (outlier, weird). For distribution shifts involving confounders, (or) anti-causal variables, and polynomial generative models, the IRM can achieve the desired OOD solution, while ERM can be asymptotically biased [1].

The use of IRM on modern software recommendation systems can effectively solve the deviation caused by Non-Independent Identically Distributed data to model training. The essential idea is to divide the invariant representations for separate learning [2]. It is evident that we should use IRM for OOD samples, but for some IID samples, ERM will show higher effectiveness; how to balance the tradeoff between ERM and IRM is the main focus of our work.

Based on the above illustration, the IRM presents a powerful capability in recognizing invariant features. However, the IRM will excessively focus on the constant part while discarding all the variant parts, where some may contain some helpful information. Under the context of the recommendation, an individual will not only pay attention to internal or invariant factors such as preference and habits but also can be affected by external or variant factors such as comments and product appearance. In this case, we still focus on promoting the accuracy of preference estimation rather than just identifying cause-effects [3]. Moreover, only retaining invariant parts conducted by the IRM will undermine the prediction results given the traditional scenario of the IID. Fortunately, with the ERM's introduction, the recommendation model's performance may be improved; however, since the IRM is categorized as OOD while ERM belongs to In-Distribution Generalization. The properties of the two items determine the incompatibility between the two. In this case, our group's motivation is to trade off the proportion of IRM and ERM self-adaptively applied in the recommendation system to better promote its accuracy and functionality. To realize self-adaptation, our group imports the attention mechanism, which can automatically adjust the weights of IRM and ERM according to the quality of different individuals. We expect the incorporation of both IRM and ERM under the monitoring of the attention model can more accurately extract an individual's proper preference combined with the influence of external factors to recommend his desired results with exactitude and efficiency.

The contributions of our paper are as follows. (1) We compare and analyze the strengths and weaknesses of ERM and IRM, respectively, applied to different environmental conditions. (2) We propose a new multimedia recommender system named EIOR, which considers variant parts and invariant parts to directly compute rating scores reflecting the user’s preference towards the item. (3) We experiment with the proposed balancing mechanism and display the improvements in prediction performance.

2. Preliminaries

2.1. Invariant Learning

Invariant learning in environment partition is to identify the features that do not change across heterogeneous environments. By focusing on features with consistent predictive capability across domains, multimedia recommendations can be more generalizable and adaptive to variations, leading to higher accuracy and efficiency [4, 5].

2.2. ERM & IRM

ERM is a machine learning framework that aims to minimize the risk between the model's predicted and actual output. Due to its vulnerability to changes in the input distribution, ERM is sensitive to noisy data and has difficulty handling the trade-offs between objectives. On the other hand, the machine learning framework of IRM is to learn features invariant to changes across heterogeneous environments, improving the generalization and robustness of the models even with limited training data.

DOI: 10.54254/2755-2721/44/20230093

According to our introduction, ERM under the IID assumption does not always hold in real-world scenarios. However, IRM learns invariant features from the heterogeneity perspective, leading to stable performance under distributional shifts.

Regarding [6], it evaluates popular IRM methods on deep models with synthetic datasets. The results show InvRat performs more effectively than others. Therefore, we adopt the InvRat method from this paper to build our IRM and ERM model in the methodology section.

2.3. Attention Mechanism

The attention mechanism can be utilized as a resource allocation schema to concentrate on distinctive parts when dealing with overloaded information [7]. Most of the attention mechanisms are focused attention which has been applied to various fields, such as image-based analysis [8, 9], text classification [10, 11], video classification [12], image captioning [13], and recommendation [14, 15, 16].

Specifically, self-attention mechanisms adaptively learn attention weights, facilitating the model to learn between various input elements that would be difficult to capture with fixed attention weights.

According to Xu. et, they. Suggest combining the self-attention model to graph neural networks for session-based recommendation [17]. In the following section, we propose a state-of-the-art approach combining UltraGCN with the self-attention model to adaptively learn the variant and invariant parts regarding the environment for the multimedia recommendation.

2.4. UltraGCN

UltraGCN[18] is an improved Graph Convolutional Networks (GCN) algorithm. It has the following advantages:

1. Adaptive neighbor sampling: UltraGCN[18] can flexibly sample neighbors based on the neighbor status of different nodes, reducing computational and storage costs and improving the scalability and efficiency of the algorithm.
2. Scalability: UltraGCN[18] has good scalability and can easily handle large-scale, dense graph data with significant runtime efficiency and accuracy advantages. This paper uses UltraGCN[18] to implement IRM and ERM.
3. Attention mechanism: UltraGCN[18] uses attention mechanisms to weigh different nodes and features, better exploring the relationships and importance between nodes and improving algorithm accuracy and robustness. In the paper, we apply the attention mechanism to combine IRM with ERM to form a better representation model.

Based on these advantages, UltraGCN[18] has been widely applied in graph neural networks, achieving good performance on various tasks and datasets.

2.5. NDCG

NDCG (Normalized Discounted Cumulative Gain) is a metric used to evaluate the quality of search engines or recommendation systems. It is widely used in the field of information retrieval.

NDCG is based on DCG (Discounted Cumulative Gain) calculations. DCG assigns higher weights to results that rank higher while penalizing the appearance of irrelevant results. Specifically, for a search query or user, the calculation of DCG is as follows:

$$DCG@k = \sum_{i=1}^{k} \frac{2^{rel_i} - 1}{\log_2(i+1)} \tag{1}$$

Here, rel_i is the relevance score of the i-th search result or recommendation result, which is usually a non-negative value. The $\log_2(i+1)$ is a discount factor that doubles the score of higher-ranking results and gradually reduces the scores of later developments.

NDCG eliminates the influence of data size and sorting position by normalizing DCG using Ideal DCG (IDCG). IDCG is calculated by calculating DCG values in the same ranking order when all results are relevant. The formula is:

DOI: 10.54254/2755-2721/44/20230093

$$IDCG@k = \sum_{i=1}^{|\mathrm{REL}|} \frac{2^{rel_i} - 1}{\log_2(i+1)} \tag{2}$$

And NDCG is calculated as follows:

$$NDCG@k = \frac{DCG@K}{IDCG@k} \tag{3}$$

Finally, the value of NDCG ranges between 0 and 1, with 1 indicating that all results are relevant and 0 indicating that no results are relevant.

3. Method

3.1. The process of the method

Here we present a method to improve the accuracy of the representation model by combining the IRM and ERM. We exhibit the workflow of our process in Figure 1. We divide this method into eight essential parts (M1-M8). M1 is a pre-train representation model used to extract the contents from multimedia data, including words, sounds, and pictures. Based on IRM, we create M2 to find the variant part in the content representation model (M1). According to that, we construct M3 to divide the original environment into several subsets. Each subset forms an independent interaction environment; we can get one feature from every subset after experiencing a deep learning process. Then, in M4, we learn an invariant mask to prepare for a uniform representation model. Combining the content representation model (M1) and the result of the consistent cover (M4), we will obtain the invariant representation model (M5), which is also the result of IRM. To the data in variant part (M2), we apply them to construct the ERM representation model (M6) by the training method ERM. After that, we employ an attention mechanism to give different weights for each feature which we obtained from both the ERM representation model (M6) and the invariant representation model (M5). The result of that is attention mechanism representation (M7). After some optimization, we obtain the final model. We will illustrate the M2 and M5 in 3.1.1, M3 in 3.1.2, and M4 in 3.1.3, elaborate on the process of combining IRM and ERM in 3.2 demonstrate M7 and M8 in 3.3, and introduce the backbone in 3.4.

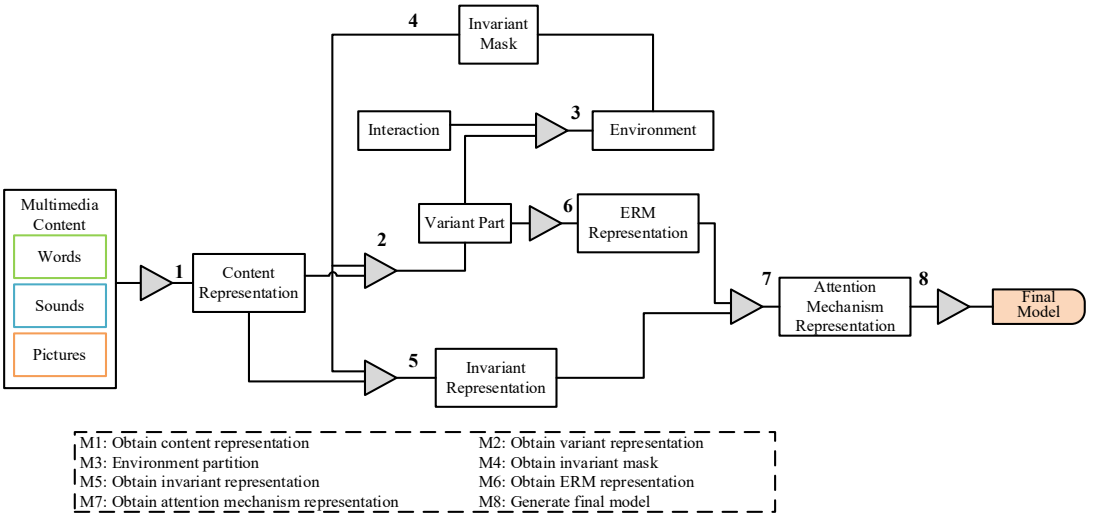

Figure 1. ERM and IRM Optimized Rating Framework (EIOR)

3.1.1. Invariant and variant representation. According to the IRM, we divide the content representation model into variant and invariant representations. First, we define some variables: an invariant mask $m \in$

DOI: 10.54254/2755-2721/44/20230093

R, the content representation c_i, the dimension of content representation D, the invariant representation Φ, the variant representation Ψ. Also, we adopt:

$$\Phi = \{\Phi_i \,|i \in I\} \, and \, \Psi = \{\Psi_i \,|i \in I\} \tag{4}$$

To demonstrate the sets of two different representations. And we define the invariant representation Φ_i as:

$$\Phi_i = m \odot c_i \tag{5}$$

when we delete all the data of the invariant representation, the others is the variant part which is defined as:

$$\Psi_i = (1 - m) \odot c_i \tag{6}$$

The most important part of the invariant representation is the generation of consistent mask m and environment partition. The detailed procedure is in the modules M2, M3, and M4; we will discuss these three modules in the following sections: 3.1.2 and 3.1.3.

3.1.2. Environment partition. According to the IRM, to finish the environment, we create a module (M3) to take in the different use-item interactions and output features about these data to form a climate set E. Each domain $e \in E$ reflects a kind of correlation between users and items; some are spurious correlations [2], and some are real correlations. Here is the detailed process. We try to classify the whole environment: some interactions only can form one feature, so we should put them together as a small environment *e*. In order to describe that environment *e*, we learn a predictive model to apply the variant part data:

$$\arg\min_{e} [\mathcal{L}(\Gamma(u, i, \Psi_{\mathrm{i}}|\Theta_{\mathrm{e}}), Re^{tr})] \tag{7}$$

where $\Gamma^{(\mathrm{e})}$ is the predictive model, Θ_e indicates the model parameters. We now have environment *E*, which consists of spurious correlations [2]. To improve the accuracy, we will find some interactions that can recognize a feature with a higher probability. To differentiate the interactions in the environment, we use this formula:

$$\mathrm{e}\,(\mathrm{u}, \mathrm{i}) = \arg\max_{\mathrm{e} \in \mathrm{E}} \left(\Gamma^{(\mathrm{e})}(u, i, \Psi_{\mathrm{i}}|\Theta_{\mathrm{e}}) \right) \tag{8}$$

Finally, we employ a loop to run these two formulas until they converge. Then we get the result of the environment partition

$$\{R(e)|e \in E\} \tag{9}$$

In the next step, we will use this result to find the invariant mask.

3.1.3. Invariant mask. For the invariant representation part, we argue that spurious correlations are unstable in heterogeneous environments, such as cattle on grass and cattle on the beach, where grass and beach have little direct connection to the cattle themselves.

From 3.1.2 we get the result of environment $R(e)$, which consists of variant part Ψ. And the variable of Ψ is invariant mask m. In this part, we will pay attention to this vector:

$$m = (m_1, \ldots, m_D) \tag{10}$$

which is used to generate invariant representation. We'd like to find a vector m that can perform well in both the single-environment and cross-environment predictive models. According to IRM and Heterogeneous Risk Minimization (HRM) [19], we do the following work:

We define

$$\mu = (\mu_1, \ldots, \mu_D) \tag{11}$$

DOI: 10.54254/2755-2721/44/20230093

And

$$\mu_i = \max\{0, \min\{1, m_i + \epsilon\}\}, where\ \epsilon \sim N(0, \sigma_2) \tag{12}$$

After that, we use a predictive model in HRM [19]:

$$\mathcal{L}_{mask} = E_{e \in E}\mathcal{L}^e + \alpha(\|Var_{e \in E}(\nabla_{\Theta mask}\mathcal{L}^e) \odot \mu\|)^2 + \lambda(\|m\|)^2 \tag{13}$$

The first part of this function is the typical recommendation loss, the second part is constraint across $\mathcal{L}_{mask}$ environments, and the third part is a regularization formula $\mathcal{L}_e$ is the average environment loss value; the formula is:

$$\mathcal{L}_e = \mathcal{L}(\Gamma_{mask}(u, i, \mu \odot c_i)|\Theta_{mask}|Re^{tr}) \tag{14}$$

Our purpose is to minimize the $\mathcal{L}_{mask}$, so with the loop continuing, we use the formula:

$$m_i \leftarrow max\{0, min\{1, m_i\}\} \tag{15}$$

clip the mask m. when the $\mathcal{L}_{mask}$ converges, we will get the invariant representation successfully.

3.2. Attention Mechanism

A large factor affecting the prediction accuracy of our model is the ability to filter out invariant representations accurately. Still, the positive impact of changing words on the correct prediction of the model cannot be denied entirely. For example, when a user buys a dress online, there is a high probability and weight that the user likes the dress itself, which is the invariant representation; however, the corresponding changing terms, such as models, lighting, and scenes, can also have a facilitating effect on the user's purchase.

Therefore, we adopt self-attention mechanisms to adaptively learn the variant and invariant parts regarding the environment partition. To adaptively learn these two parts, we use the attention mechanism of adaptive learning to balance their weights dynamically. More effective model fusion is achieved by combining the attention learning process with the UltraGCN prediction process. Based on the self-attention mechanism, it can effectively allocate weights among different environments.

Up to now, we have obtained stable invariant and changing representations by learning, denoted by Φ_i, Ψ_i respectively. Subsequently, we construct attention mechanisms to learn the learning weights of the invariant and changing representations, that is, to determine the weights of the contributions of the invariant and changing representations to the final prediction results. We piece together the change and invariant representations according to the following equation:

$$\mathrm{c}_j^{re} = \alpha_j^1 \Psi_\mathrm{i} + \alpha_j^2 \Phi_\mathrm{i} \tag{16}$$

where α_j^1 and α_j^2 are the attention weights for variant representation and invariant representation. In other words, they indicate the size of impact factors in two representations' predictions on users' preferences.

3.3. Collaborating filtering

Concerning the collaborative filtering term t_i, *(u, i)* can be written as a user-commodity feature sparse matrix, for which users u_1, u_2 (row vectors) and all commodities i (column vectors) are written as:

$$\begin{bmatrix} R_{1,1} & R_{1,2} & \dots \\ R_{2,1} & R_{2,2} & \dots \end{bmatrix}$$

The similarity of the preferences of user 1 and user 2 can be measured by the cosine similarity:

$$sim(u_1, u_2) = \frac{u_1 * u_2}{||u_1|| \times ||u_2||} \tag{17}$$

The user's preference for an item *i* can be calculated by using the rating formula:

DOI: 10.54254/2755-2721/44/20230093

$$score(u_i, i) = \begin{cases} 1, Hidden\ Feedback \\ x, Based\ on\ user\ ratings \end{cases}$$

Based on the descending order of ratings, we can write the collaborative filtering term based on user u and item i as:

$$sim(u, s) = \sum_{s_i \in S} sim(u, u_i) * score(u_i, i) \tag{18}$$

3.4. Final Prediction Model

Final prediction model. The invariant mask becomes stable by running streams M2-M3-M4 repeatedly in T times until convergence. Therefore, we learn the attention weights w_1 and w_2 of both and the final prediction model based on the invariant and changing representations generated in M5 and M6, respectively.

In order to find the specific w_1 and w_2 values, we obtain the variant mask by taking the inverse of the invariant show to part 3.1.3, and we use these two parts of the show to refine the change representation and the consistent representation, respectively:

$$(1 - invariant\ mask) \times feature$$

We apply an empirical risk minimization model to the change representation part to find the gap between the predicted and empirical environments. We expect and see the loss separately for user-related and user-irrelevant items, where the former term in Eq.19 denotes the loss function for predicting user and user-related articles. The latter term means the loss function for predicting user and user-irrelevant items and takes the square root of the two results to normalize the loss values.

$$\sqrt{\mathcal{L}((\mathrm{u}, \mathrm{iid}, \Phi_{\mathrm{i}})|\ \mathrm{R}_{\mathrm{real}})^2 + \boldsymbol{\mathcal{L}}((\mathrm{u}, \mathrm{iid}, \Psi_i)|\ \mathrm{R}_{\mathrm{real}})^{\ 2}} \tag{19}$$

After obtaining the two sets of environmental losses, we use the attention weights to combine the failure of the changing representation and the failure of the invariant representation into the failure of the overall feature, which is also the loss of our final prediction model. We make the initial attention weights equal and keep the sum of the weights always 1. Thus, we have:

$$Loss\ total = w_1 a_1 + w_2 a_2 \tag{20}$$

Where w_1 is the attention weight of the invariant representation loss, w_2 is the attention weight of the variant representation loss.

The learning is defined below:

$$\arg\min_{\Theta *} L(w_1 * \Gamma_{IRM}(u, i, \Phi_i\ |\Theta *)|R_{tr}\ +\ w_2 *\ \Gamma_{ERM}(u, i, \Psi_i|\Theta *)|R_{tr}\ ) \tag{21}$$

In conclusion, the general training process is introduced in Algorithm One:

Algorithm One: the total training process.

Data: R, R^{-}, R^{tr}

Result: Final Predictive Model $\Gamma * (u, i|\Theta^*, \Phi)$

1 while ($i \leftarrow 1$ to T) do

/* M3 */

2 **do**

3 **while**($e \in$ E) do

4 Optimize Γ(e) via Eq. (7);

5 **continue;**

6 **while** ($e \in$ E) do

DOI: 10.54254/2755-2721/44/20230093

```
7            Compute R_e via Eq. (8);
8          continue;
9       while Converged;
          /* M4 */
10      do
11        Learn m via Eq. (13);
12      while Converged;
13        continue;
          /* M6 */
14      do
15      Using m to extract variant feature
16        ERM learning on variant feature
17          Judge loss of IRM L1 and ERM L2
18      while Converged;
          /* M7 */
19    Optimize equation (21)
```

UltraGCN automatically weighs the learning ratios of ERM and IRM using a loss function to achieve optimal results.

3.5. Backbone: UltraGCN[1, 2]

UltraGCN pushes the representations to encode the user-item graph through the graph-based loss function,

$$\mathcal{L} = \mathcal{L}_O + \gamma_C \mathcal{L}_C + \gamma_I \mathcal{L}_I \tag{22}$$

where γ_C and γ_I are hyper-parameters to balance the importance weights of these loss terms.

The $\mathcal{L}_O$ indicates the objective loss, and the first and second terms calculate the relevance between multimedia items and targets for positive and negative samples, respectively. The relevance is mapped to a probability value using the logistic function σ. Then, the logarithm of this probability value is taken and negated to represent the matching loss. The objective of the first term is to maximize the relevance of positive samples, allowing the recommendation model to match users with target items better. On the other hand, the objective of the second term is to minimize the relevance of negative samples, enabling the recommendation model to better distinguish users from irrelevant items. The objective loss is,

$$\mathcal{L}_O = -\sum_{(u,i)\in\mathbb{R}} \log\left(\sigma\left(\Gamma(u,i)\right)\right) - \sum_{(u,i)\in\mathbb{R}^-} \log\left(\sigma\left(-\Gamma(u,i)\right)\right) \tag{23}$$

$\mathcal{L}_C$ indicates the user-item constraint loss, which is used to train an adversarial model, such as the discriminator in a generative adversarial network, to enhance the robustness and generalization capability of the invariant representation learning model. In this loss, the first and second terms calculate the relevance between multimedia item u and target items i and j, respectively, based on different weight terms $\beta u,i$ and $\beta u,j$. The relevance values are then mapped to probability values using the logistic function σ. Subsequently, the logarithm of these probability values is taken and multiplied by the corresponding weight terms. Consequently, the objective of the first term is to maximize the relevance of positive samples, while the aim of the second term is to minimize the relevance of negative samples. The user-item constraint loss is,

$$\mathcal{L}_C = -\sum_{(u,i)\in\mathbb{R}} \beta_{u,i} \log\left(\sigma\left(\Gamma(u,i)\right)\right) - \sum_{(u,i)\in\mathbb{R}^-} \beta_{u,i} \log\left(\sigma\left(-\Gamma(u,i)\right)\right), \tag{24}$$

DOI: 10.54254/2755-2721/44/20230093

where the fixed weight coefficients $\beta_{u,i}$ and $\beta_{u,j}$ are derived from the user-item interactive graph R by:

$$\beta_{u,i} = \frac{1}{d_u}\sqrt{\frac{d_u + 1}{d_i + 1}}, \tag{25}$$

Where d_u and d_i denote the degrees of the corresponding nodes. Another constraint relies on an item-item correlation graph $G = R^{tr}$, where R indicates the user-item interactive graph. Thus, $\mathcal{L}_I$ indicates the item-item constraint loss, a regularization term used to encourage the relevance between a multimedia item u and its associated items j in the same temporal sequence i within the recommendation model. The inner summation term measures the relevance between the multimedia item and its associated items using the logistic function transformation and taking the logarithm. The outer summation term aggregates the relevance values of associated items within the same temporal sequence. By minimizing LI, the recommendation model can learn the relevance between the multimedia object and its associated items in the same temporal sequence, thereby better considering the temporal dependencies, The item-item constraint loss is,

$$\mathcal{L}_I = -\sum_{(u,i)\in\mathbb{R}}\sum_{j\in S(i)}\omega_{i,j}\log\left(\sigma\left(\Gamma(u,j)\right)\right), \tag{26}$$

where $S(i)$ indicates the adjacent item set of the item i. The weight coefficient $\omega_{i,j}$ is computed by:

$$\omega_{i,j} = \frac{G_{i,j}}{g_i - G_{i,i}}\sqrt{\frac{g_i}{g_j}}, g_i = \sum_k G_{i,k}, \tag{27}$$

where g_i and g_j denote the degrees of item i and item j in G.

We learn the predictive model by:

$$\arg\min_{e}\left[\mathcal{L}\left(\Gamma\left(u,i,c_j^{re}\right),R^{tr}\right)\right] \tag{28}$$

Here, R^{tr} represents the user's true preference for an item (expressed through ratings), and its loss function can be defined as

$$Loss = |R^{tr} - (\lambda R_{ERM} + (1-\lambda)R_{IRM})| \tag{29}$$

where λ and (1 - λ) can be interpreted as the percentage representation of ERM learning weights and IRM learning weights, calculated as follows:

$$\lambda = \frac{\alpha_j^1}{\alpha_j^1 + \alpha_j^2} \times 100\% \tag{30}$$

UltraGCN automatically weighs the learning ratios of ERM and IRM using a loss function to achieve optimal results.

4. Experiment

4.1. Experiment Settings

4.1.1. Dataset. TikTok platform tracks the viewing data of micro-videos, providing certified written, audiovisual, and auditory representations. To represent the textual content, the initial sentence-based textual representations, encoded as one-hot word vectors, are transformed by summing the word embeddings.

DOI: 10.54254/2755-2721/44/20230093

4.1.2. Evaluation protocols. Building upon prior studies [20, 21], our approach involves assessing the user-item interactions through trained models and subsequently satisfactorily ranking them. Specifically, for each user, we prioritize the top-K items and determine the Precision@K (P@K), Recall@K (R@K), and Normalized Discounted Cumulative Gain (N@K) based on the observed interactions within the testing dataset. To evaluate the efficacy of the trained model, we calculate the average scores across all users.

4.1.3. Baseline. To assess the effectiveness of our model, we adopt the comparative approach outlined in the InvRL article and compare it against state-of-the-art multimedia recommendation methods. Specifically, we consider baselines from three categories as follows:

1. Multimedia CF (M-CF) Category: We include VBPR [22], DUIF [23], and CB2CF [24], which incorporate multimedia content into the original collaborative filtering method (CF).
2. Generic Neural CF (G-NCF) Category: We consider NGCF [25], DisenGCN [26], and MacridVAE [27] as representatives of this category.
3. Multimedia-oriented NCF (M-NCF) Models: Our selection includes MMGCN [28], HUIGN [20], and GRCN [20], which are explicitly designed for multimedia-oriented recommendation tasks.

The performance evaluations of the baselines above are sourced from previous works [20, 21], following the established conventions.

4.1.4. Parameter settings. Adam's algorithm can better adapt to the case of sparse gradients by using second-order moment estimates of the slopes (mean of squared angles). This makes it perform better for light data processing in huge matrix tasks. Therefore, we empirically used Adam [29] as an optimizer. This section describes the experimental settings used to evaluate our proposed approach. The details are as follows:

1. Batch Size: We set the batch size to 512, determining the number of samples processed in each training iteration.
2. Embedding Dimension: The dimension of the embeddings was fixed at 64, ensuring consistent representation across the model.
3. Hyperparameter Tuning: We performed individual tuning of the learning threshold and regularization factor for specific embeddings and other parameters. This process involved adjusting the values to optimize the model's performance.
4. Regularization Factors: To control overfitting, we utilized regularization factors with weights of 10-4 for specific ID parameters. We experimented with values of 1, 0.1, 0.01, 0.001, and 0 for other parameters.
5. Learning Rate: We set the learning rate to 10-3 for all parameters to regulate the speed of model convergence during training.
6. Environment Number (ε): The environment number |ε| was varied in the range of {1, 5, 10, 20, 30}, allowing us to explore different environmental contexts for enhanced performance.
7. Parameters α and λ: The parameters α and λ in equation 10 were chosen from the set {1, 0.5, 0.1}, respectively. These values were selected to optimize the trade-off between accuracy and regularization.
8. Learning Rate of Mask Generation: The learning rate of the mask generation module (m) was searched within {0.01, 0.001, 0.0001} to achieve optimal mask generation.
9. Parameters γC and γI: The parameters γC and γI were adjusted in the set {2, 1, 0.1, 0.01, 0} to examine their impact on the model's performance.
10. Iteration Parameter T: The iteration parameter T was initially set to 5, determining the number of iterations for the proposed approach.
11. Training Epochs: The environment segmentation model was trained for 20 epochs, the mask generation model for 40 epochs, and the final prediction model for 500 epochs, ensuring convergence and capturing essential patterns in the data.

DOI: 10.54254/2755-2721/44/20230093

12. Model Selection: The selection of models was based on validation scores, allowing us to identify the most effective models. The corresponding test scores were reported for further analysis.

By adopting these experimental settings, we aimed to thoroughly investigate the performance of our proposed approach and ensure reliable and meaningful results.

5. Result and Discussion

We present the overall performance comparison of different methods in Table 1. The following observations can be made:

Neural collaborative filtering (NCF) approaches generally outperform collaborative filtering (CF) because NCF explicitly considers the interactions between embedding dimensions. This enables a more comprehensive representation of pairwise correlations, enhancing fine-grained information modeling. CNNs are applied to the matrix generated by the outer products, allowing for extracting higher-order correlations and complex patterns within the embedding space. [30]

Moreover, the relatively poorer performance of DUIF highlights the impact of collaborative support. Additionally, M-NCF approaches consistently outperform G-NCF approaches, demonstrating that it is essential in application scenarios such as multimedia recommender systems to correctly analyze multiple data forms and establish interactions between different modalities idiosyncratically. [31, 32]

Notably, by adding a graph regularization term to the standard CNN structure and applying a graph convolution operation to aggregate the information of neighboring nodes, GRCN achieves the best performance among the NCF-based methods, which emphasizes the need for leveraging user behaviors and item contents in an effective multimedia recommendation model. [33]

Our backbone model, UltraGCN, is a generic graph-based CF method. Despite its simple incorporation of multimedia content, UltraGCN significantly outperforms other multimedia recommendation baselines. This impressive performance indicates that UltraGCN can effectively capture collaborative information through constraint losses. The InvRL model uses the same prediction function and training target as UltraGCN, with the only difference being content representation through the learned invariant mask (as described in Section 3.1.3). These significant improvements can be attributed to the constraints imposed by MASK. And the result is, InvRL consistently achieves the best performance across the TikTok datasets, surpassing UltraGCN by 8.71% on Tiktok, respectively [2]. However, the invariant representation obtained from singularity learning through mask masking is limited because it completely abstracts the subject's interaction with the environment, and our model suggests that there is also some connection between the changing and invariant representations.

Compared with InvRL, using the attention mechanism to connect the learning of changing representations with the knowledge of invariant representations in the InvRL model brings more features and means more learnable space. As shown in Table 1, the model using the attention mechanism has slightly improved performance over learning invariant representations using InvRL alone in the Jitterbug dataset, which supports that changing graphics is not useless. This confirms that changing pictures is not meaningless but can uncover information that is useful to us.

Table 1. Evaluation of Performance

Category	Methods	Tiktok		
		P@10	R@10	N@10
M-CF	VBPR	0.0118	0.0628	0.0574
	DUIF	0.0087	0.0483	0.0434
	CB2CF	0.0109	0.0642	0.0613
G-NCF	NGCF	0.0135	0.078	0.0661
	DisenGCN	0.0145	0.076	0.0639
	MacridVAE	0.0152	0.0813	0.0686
M-NCF	MMGCN	0.0144	0.0808	0.0674
	HUIGN [38]	0.0164	0.0884	0.0769
	GRCN [39]	0.0195	0.1048	0.0938

DOI: 10.54254/2755-2721/44/20230093

Table 1. (continued).

Backbone	UltraGCN	0.0182	0.0982	0.0876
InvRL		0.0196	0.1079	0.0951
EIOR		**0.0197**	**0.1094**	**0.0991**
%Impv. over InvRL		0.51%	1.39%	4.21%

The evaluation of performance is presented in the above table. Bold scores indicate the best performance achieved, while underlined scores represent the second-best performance. The abbreviations M-CF, G-NCF, and M-NCF correspond to multimedia CF, generic NCF, and multimedia NCF, respectively.

Through multiple iterations of learning attention mechanism coefficients, it is evident that despite our previous argument that we cannot wholly disregard the feature learning of varying representations, the attention coefficients corresponding to variable expressions are often significantly smaller than those of invariant representations. In other words, their impact is limited.

Furthermore, evidence suggests that a large portion of the data within a set of features represents variations, with only the core regions of the parts being invariant representations. This poses a challenge, as simply distinguishing between varying and invariant representations is insufficient. Taking the example of an image depicting a camel in the desert and a cow in a meadow, the main subjects of the image are the camel and the cow, which occupy only a tiny portion of the picture. However, regarding varying representations, the desert, meadow, and sky hold a significant advantage in terms of feature quantity. This may result in poor performance of our model in learning features related to varying representations in complex background images.

Therefore, addressing the insufficiency of differentiating between varying and invariant representations becomes necessary when partitioning varying manifestations.

Besides, to enhance our model's generalization capability, we consider further utilizing multi-head attention [34] to partition the varying representations' environmental aspects. As mentioned, the drawings contain features that contribute significantly to the prediction model and "irrelevant" features. Therefore, we propose assigning higher attention weights to the essential parts of the varying representations while assigning lower weights to the less significant ones. Our future work will focus on dynamic learning of the different models.

To achieve this, we will leverage the multi-head attention mechanism, which has been proven effective in capturing diverse patterns and dependencies within the input data. By incorporating multiple attention heads, each attending to a different aspect of the varying representations, we can better capture the complex relationships and variations in the environment.

Furthermore, we will explore techniques to dynamically adapt the attention weights based on the significance of the varying representations. This can be achieved through adaptive mechanisms such as reinforcement learning or adaptive gating tools, which can iteratively adjust the attention weights during the training process.

By incorporating these enhancements, we expect to improve the model's ability to distinguish between essential and irrelevant features within the varying representations. This, in turn, will lead to enhanced generalization performance and accuracy in handling complex visual data.

In our future work, we will conduct extensive experiments to evaluate the effectiveness of the proposed approach. We will compare the performance of our model with and without the multi-head attention mechanism on various datasets and complex background images. Additionally, we will investigate the impact of different strategies for dynamically learning the attention weights for the varying representations.

DOI: 10.54254/2755-2721/44/20230093

6. Conclusion

This paper introduces the EIOR for multimedia recommendation with explicit rating scores. The model incorporates the learning of the variant part across the environment based on ERM. According to the experiment results, applying self-attention mechanisms with adjusted attention weights for both IRM and ERM illustrates higher rating scores contrasting with the implementation of the IRM framework alone, which indicates that the variant part is not useless under the scenario of the multimedia recommendation. Moreover, the better performance of the EIOR compared with other models shows that the combination and balance between the variant part and the invariant part is more capable of predicting a customer's preference. In the future, we will put more effort into adopting the multi-head mechanism to improve the model's competence further.

Acknowledgment

They contributed equally to this work and should be considered co-first authors.

References

[1] Kartik Ahuja, Jun Wang, Amit Dhurandhar, Karthikeyan Shanmugam Kush R. Varshney, Empirical or Invariant Risk Minimization? A Sample Complexity Perspective

[2] Du, X., Wu, Z., Feng, F., He, X., & Tang, J. (2022). Invariant Representation Learning for multimedia recommendation. Proceedings of the 30th ACM International Conference on Multimedia.

[3] Si, Z., Han, X., Zhang, X., Xu, J., Yin, Y., Song, Y., & Wen, J.-R. (2022). A Model-Agnostic Causal Learning Framework for Recommendation using Search Data. Proceedings of the ACM Web Conference 2022, 224–233. https://doi.org/10.1145/3485447.3511951

[4] Creager, E., Jacobsen, J.-H., & Zemel, R. (2021, July 1). Environment inference for invaria nt learning. PMLR. Retrieved April 1, 2023, from https://proceedings.mlr.press/v139/crea ger21a.html?utm_campaign=The+Batch&utm_source=hs_email&utm_medium=email&_hse nc=p2ANqtz-9GoNXKtgh3kIYhDbN6wuqn6vTgNYaUE_B6t5EpPdQ9phgpRXVhYpkLoF HDJ7S-TWBi8nwc

[5] Du, X., Wu, Z., Feng, F., He, X., & Tang, J. (2022). Invariant Representation Learning for multimedia recommendation. Proceedings of the 30th ACM International Conference on Multimedia. https://doi.org/10.1145/3503161.3548405

[6] Lin, Y., Qing, L., & Zhang, T. (2021). An Empirical Study of Invariant Risk Minimiz ation on Deep Models. Presented at the ICML 2021 Workshop on Uncertainty and Rob ustness in Deep Learn. Retrieved April 2, 2023, from http://www.gatsby.ucl.ac.uk/~balaji /udl2021/accepted-papers/UDL2021-paper-044.pdf.

[7] Niu, Z., Zhong, G., & Yu, H. (2021). A review on the attention mechanism of Deep Learning. Neurocomputing, 452, 48–62. https://doi.org/10.1016/j.neucom.2021.03.091

[8] Song, K., Yao, T., Ling, Q., & Mei, T. (2018). Boosting image sentiment analysis with visual attention. Neurocomputing, 312, 218–228. https://doi.org/10.1016/j.neucom.2018.05.104

[9] Yan, X., Hu, S., Mao, Y., Ye, Y., & Yu, H. (2021). Deep multi-view learning methods: A Review. Neurocomputing, 448, 106–129. https://doi.org/10.1016/j.neucom.2021.03.090

[10] Li, Y., Yang, L., Xu, B., Wang, J., & Lin, H. (2019). Improving user attribute classification with text and social network attention. Cognitive Computation, 11(4), 459–468. https://doi.org/10.1007/s12559-019-9624-y

[11] Liu, G., & Guo, J. (2019). Bidirectional LSTM with attention mechanism and convolutional layer for text classification. Neurocomputing, 337, 325–338. https://doi.org/10.1016/j.neu com.2019.01.078

[12] Long, X., Gan, C., de Melo, G., Wu, J., Liu, X., & Wen, S. (2018). Attention clusters: Purely attention based local feature integration for video classification. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. https://doi.org/10.1109/cvpr.2018.00817

DOI: 10.54254/2755-2721/44/20230093

[13] Huang, L., Wang, W., Chen, J., & Wei, X.-Y. (2019). Attention on attention for image captioning. 2019 IEEE/CVF International Conference on Computer Vision (ICCV). https://doi.org/10.1109/iccv.2019.00473

[14] Wang, S., Hu, L., Cao, L., Huang, X., Lian, D., & Liu, W. (2018). Attention-based transactional context embedding for next-item recommendation. Proceedings of the AAAI Conference on Artificial Intelligence, 32(1). https://doi.org/10.1609/aaai.v32i1.11851

[15] Celikik, M., Wasilewski, J., Mbarek, S., Celayes, P., Gagliardi, P., Pham, D., Karessli, N., & Ramallo, A. P. (2023). Reusable self-attention-based recommender system for Fashion. Lecture Notes in Electrical Engineering, 45–61. https://doi.org/10.1007/978-3-031-22192-7_3

[16] Ying, H., Zhuang, F., Zhang, F., Liu, Y., Xu, G., Xie, X., Xiong, H., & Wu, J. (2018). Sequential Recommender system based on hierarchical attention networks. Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence. https://doi.org/10.24963/ijcai.2018/546

[17] Xu, C., Zhao, P., Liu, Y., Sheng, V. S., Xu, J., Zhuang, F., Fang, J., & Zhou, X. (2019). Graph contextualized self-attention network for session-based recommendation. Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence. https://doi.org/10.24963/ijcai.2019/547

[18] Kelong Mao, Jieming Zhu, Xi Xiao, Biao Lu, Zhaowei Wang, and Xiuqiang He. 2021. UltraGCN: Ultra Simplification of Graph Convolutional Networks for Recommendation. In International Conference on Information and Knowledge Management, Proceedings. 1253–1262. https://doi.org/10.1145/3459637.3482291 arXiv:2110.15114

[19] Jiashuo Liu, Zheyuan Hu, Peng Cui, Bo Li, and Zheyan Shen. 2021. Kernelized Heterogeneous Risk Minimization. In Advances in Neural Information Processing Systems, Vol. 26. PMLR, 21720–21731. arXiv:2110.12425

[20] Yinwei Wei, Xiang Wang, Xiangnan He, Liqiang Nie, Yong Rui, and Tat Seng Chua. 2022. Hierarchical User Intent Graph Network for Multimedia Recommendation. IEEE Transactions on Multimedia 24 (2022), 2701–2712. TMM.2021.3088307. https://doi.org/10.1109/ arXiv:2110.14925

[21] Yinwei Wei, Xiang Wang, Liqiang Nie, Xiangnan He, and Tat Seng Chua. 2020. Graph-Refined Convolutional Network for Multimedia Recommendation with Implicit Feedback. Proceedings of the 28th ACM International Conference on Multimedia (2020), 3541–3549. https://doi.org/10.1145/3394171.3413556 arXiv:2111.02036

[22] Ruining He and Julian McAuley. 2016. VBPR: Visual Bayesian personalized ranking from implicit feedback. In 30th AAAI Conference on Artificial Intelligence, Vol. 30. 144–150. https://doi.org/10.1609/aaai.v30i1.9973 arXiv:1510.01784

[23] Xue Geng, Hanwang Zhang, Jingwen Bian, and Tat Seng Chua. 2015. Learning image and user features for recommendation in social networks. In Proceedings of the IEEE International Conference on Computer Vision, Vol. 2015 Inter. 4274–4282. https://doi.org/10.1109/ICCV.2015.486

[24] Oren Barkan, Noam Koenigstein, Eylon Yogev, and Ori Katz. 2019. CB2CF: A neural multiview content-to-collaborative filtering model for completely cold item recommendations. In 13th ACM Conference on Recommender Systems. 228–236. https://doi.org/10.1145/3298689.3347038

[25] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat Seng Chua. 2019. Neural graph collaborative filtering. In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval. 165–174. https://doi.org/10.1145/3331184.3331267 arXiv:1905.08108

[26] Jianxin Ma, Peng Cui, Kun Kuang, Xin Wang, and Wenwu Zhu. 2019. Disentangled graph convolutional networks. In 36th International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 2019-June), Kamalika Chaudhuri and Ruslan Salakhutdinov (Eds.). PMLR, 7454–7463.

DOI: 10.54254/2755-2721/44/20230093

[27] Jianxin Ma, Chang Zhou, Peng Cui, Hongxia Yang, and Wenwu Zhu. 2019. Learning disentangled representations for recommendation. Advances in Neural Information Processing Systems 32 (2019). arXiv:1910.14238

[28] Yinwei Wei, Xiangnan He, Xiang Wang, Richang Hong, Liqiang Nie, and Tat Seng Chua. 2019. MMGCN: Multi-modal graph convolution network for personalized recommendation of micro-video. In Proceedings of the 27th ACM International Conference on Multimedia. 1437–1445. https://doi.org/10.1145/3343031.3351034

[29] Diederik P. Kingma and Jimmy Lei Ba. 2015. Adam: A method for stochastic optimization. 3rd International Conference on Learning Representations (2015). arXiv:1412.6980

[30] Du, X., He, X., Yuan, F., Tang, J., Qin, Z., & Chua, T. S. (2019). Modeling embedding dimension correlations via convolutional neural collaborative filtering. ACM Transactions on Information Systems (TOIS), 37(4), 1-22.

[31] Wang, X., He, X., Wang, M., Feng, F., & Chua, T. S. (2019, July). Neural graph collaborative filtering. In Proceedings of the 42nd international ACM SIGIR conference on Research and development in Information Retrieval (pp. 165-174).

[32] Chung, Y. H., & Chen, Y. L. (2021, December). Social Recommendation System with Multimodal Collaborative Filtering. In 2021 IEEE Global Communications Conference (GLOBECOM) (pp. 1-7). IEEE.

[33] Tian, X., Ding, C. H., Chen, S., Luo, B., & Wang, X. (2021). Regularization graph convolutional networks with data augmentation. Neurocomputing, 436, 92-102.

[34] Cordonnier, J.-B., Loukas, A., & Jaggi, M. (2021, May 20). Multi-head attention: Collaborate instead of Concatenate. arXiv.org. https://arxiv.org/abs/2006.16362