

## A APPENDIX

### A.1 Time Complexity Analysis

In our analysis of the time complexity of our proposed DiffMM, we consider its three key components. Firstly, the multi-modal graph aggregation module involves a time complexity of  $O((L + 2|\mathcal{M}|) \times |\mathcal{G}| \times d + \sum_m^{\mathcal{M}} |\mathcal{G}^m| \times d)$ . Here,  $L$  represents the number of graph neural layers,  $|\mathcal{G}|$  is the number of edges in the user-item interaction graph,  $|\mathcal{G}^m|$  corresponds to the number of edges in the modality-aware user-item graph of modality  $m$ ,  $\mathcal{M}$  signifies the set of modalities  $m$ , and  $d$  stands for the embedding dimensionality. Secondly, the cross-modal contrastive learning module exhibits a time complexity of  $O(L \times B \times (\mathcal{I} + \mathcal{J}) \times d)$ . Here,  $B$  denotes the number of users/items included in a single batch, while  $\mathcal{I}$  and  $\mathcal{J}$  represent the number of items and users, respectively. Thirdly, the multi-modal graph diffusion module entails a time complexity of  $O(B \times ((\mathcal{I} + d_t) \times d_{diff}) + \mathcal{I} \times d_{diff} + 2\mathcal{I} \times d)$  for training, which can be further simplified to  $O(B \times \mathcal{I} \times d_{diff})$ . Additionally, during inference, the multi-modal graph diffusion module requires  $O(t \times B \times \mathcal{I} \times d_{diff})$  time, where  $t$  denotes the inference time step. It is worth noting that in practical implementation within recommendation systems, the time complexity of the diffusion model is considered acceptable. It is similar to that of the contrastive learning module, ensuring a comparable level of efficiency to other self-supervised recommendation methods.

### A.2 Details of Baselines

This section gives a brief introduction of the compared baseline methods in this work, including 15 baselines from 5 research lines. **Conventional Collaborative Filtering Method. MF-BPR** [19]: This method is a classic factorization-based collaborative recommender, with the incorporation of the pairwise BPR loss function. **GNN-based Collaborative Filtering Methods.**

- **NGCF** [24]: It uses a multi-layer graph convolutional network to propagate information through the user-item interaction graph and learns the latent representations of users and items.
- **LightGCN** [8]: This work simplifies message passing for graph neural network-based recommendation by eliminating redundant transformations and non-linear activation functions.

**Generative Diffusion Recommendation Method. DiffRec** [23]: This work proposes a novel generative recommender model, which reduces the noise scales and inference steps to corrupt users' interactions in the forward process for personalized recommendations. **Self-supervised Recommendation Solutions.**

- **SGL** [32]: This model improves the graph collaborative filtering with the incorporated contrastive learning signals using different data augmentation operators, *e.g.*, randomly node/edge dropout and random walk, to construct contrastive representation views.
- **NCL** [12]: In this approach, positive contrastive pairs are generated by identifying semantic and structural neighboring nodes using EM-based clustering to create contrastive views.
- **HCCF** [33]: A new self-supervised recommendation framework is proposed in this work, which is able to capture both local and global collaborative relations using a hypergraph neural network enhanced by cross-view contrastive learning architecture.

### Multi-Modal Recommendation Systems.

- **VBPR** [7]: This is a representative work to incorporate multi-media features into the matrix decomposition framework.
- **LightGCN-M**: This method is established by utilizing the SOTA GNN-based CF model, LightGCN, as the foundation and incorporating multi-modal item features into the message passing.
- **DiffRec-M**: It is adapted by employing the generative diffusion recommendation method, DiffRec, as the foundation and incorporating multi-modal item features into the diffusion process.
- **MMGCN** [31]: This work utilizes GNNs to propagate the modality-specific embeddings and capture the user preferences related to different modalities for the micro-video recommendation.
- **GRCN** [30]: This multimedia recommender system is a structure-refined GCN, capable of producing refined interactions to identify false-positive feedback and eliminate noise by pruning edges.
- **LATTICE** [37]: This work aims to uncover latent item-item relations via the generated item homogeneous graph, which are established based on similarities of item modal features.
- **CLCRec** [29]: This work tackles the item cold-start problem by enriching item embeddings with multi-modal features through mutual information-based contrastive learning.
- **MMGCL** [35]: This work integrates graph contrastive learning through the application of modality edge dropout and masking.
- **SLMRec** [21]: This method introduces data augmentation for multi-modal content, which consists of two components: noise perturbation over features and multi-modal pattern uncovering.
- **LightGT** [27]: It proposes a new Transformer-based model for multimedia recommendation, with a modal-specific embedding and a layer-wise position encoder for features distillation.
- **BM3** [38]: This work introduces an SSL framework for multi-modal recommendation that eliminates the need for randomly sampled negative samples in modeling user-item interactions.

### A.3 Implementation Details

For fair comparison, we present the hyperparameter settings for implementing the proposed DiffMM framework and the baseline methods. Specifically, our DiffMM is implemented with PyTorch, using Adam optimizer and Xavier initializer with default parameters. Training batch size is set as 1024. The dimensionality of embedding vectors is set as 64. The learning rate is set as  $1e - 3$ . The number of GCN layers is set as 1. The decay of  $\mathcal{L}_2$  regularization term (*i.e.*,  $\lambda_2$ ) is searched in the set  $\{1e - 1, 1e - 2, 1e - 3, 1e - 4, 1e - 5, 1e - 6\}$ . The loss weight  $\lambda_0$  is searched in  $\{1, 0.5, 0.1, 0.01, 0.001\}$  and the loss weight  $\lambda_1$  is searched in  $\{1, 0.1, 0.01\}$ . The hyperparameter  $\omega$  is searched in  $\{0.10, 0.25, 0.50, 0.75, 1.00\}$ . The temperature coefficient  $\tau$  is searched in  $\{0.1, 0.5, 1.0\}$ . For the baseline methods, we apply the same Adam optimization algorithm, Xavier parameter initializer, and batch size of 1024 as our DiffMM. The hidden dimensionality for all baselines is also set as 64. Hyperparameters that are shared by baseline methods and our DiffMM, are tuned in the same range as above. Such hyperparameters include the number of GCN layers, the weight for weight-decay regularizer. For self-supervised methods (*e.g.*, SGL, NCL, and HCCF), the temperature

**Table 5: Modality-aware CL with different variants.**

| Dataset                                       | TikTok        |               | Amazon-Baby   |               | Amazon-Sports |               |
|---|---------------|---------------|---------------|---------------|---------------|---------------|
| Variants                                      | Recall        | NDCG          | Recall        | NDCG          | Recall        | NDCG          |
| Modality-aware Contrastive Learning Paradigms |               |               |               |               |               |               |
| Modality View                                 | 0.1064        | 0.0444        | <b>0.0975</b> | <b>0.0411</b> | <b>0.1017</b> | <b>0.0458</b> |
| Main View                                     | <b>0.1129</b> | <b>0.0456</b> | 0.0903        | 0.0390        | 0.0994        | 0.0445        |
| Aligning Methods for Raw Feature Embeddings   |               |               |               |               |               |               |
| Parametric Matrix                             | 0.1065        | 0.04268       | <b>0.0975</b> | <b>0.0411</b> | 0.1012        | 0.0453        |
| Linear  | <b>0.1129</b> | <b>0.0456</b> | 0.0963        | 0.0405        | <b>0.1017</b> | <b>0.0458</b> |

$\tau$  for contrastive learning is searched in  $\{0.1, 0.5, 1.0\}$ , which is the same as our DiffMM. For baseline methods that employ random message dropout (e.g., SGL), the dropout rate is tuned from  $\{0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9\}$ .

#### A.4 Model Analysis for Modality-aware Contrastive Learning

(iv) **Impact of modality-aware CL with different anchors.** In our approach, we have introduced two modality-aware contrastive

learning paradigms that utilize different anchor choices. The results, presented in Table 5, showcase the impact of these paradigms on the model performance across various datasets. In this context, the term "Modality View" refers to using the modality view as the anchor, while "Main View" indicates using the main view as the anchor. The superior performance is highlighted in bold. Based on the evaluation results, employing the modality view as the anchor leads to improved performance on the Amazon-Baby and Amazon-Sports datasets, while the opposite is observed for the TikTok dataset.

(v) **Impact of multi-modal alignment methods.** We introduce the transformation function  $Trans(\cdot)$  to align diverse raw modal feature embeddings  $\hat{f}^m$ . We propose two alignment methods: "Parametric Matrix" using a weight matrix  $W \in \mathbb{R}^{d_m \times d}$ , and "Linear" using a linear conversion process. Evaluation results in Table 5 show that the alignment methods have minimal impact on Amazon-Baby and Amazon-Sports datasets. However, on the TikTok dataset, the "Linear" method outperforms the "Parametric Matrix" approach due to lower dimensionality. This dimension discrepancy can lead to overfitting issues when using the "Parametric Matrix" method.

1161  
1162  
1163  
1164  
1165  
1166  
1167  
1168  
1169  
1170  
1171  
1172  
1173  
1174  
1175  
1176  
1177  
1178  
1179  
1180  
1181  
1182  
1183  
1184  
1185  
1186  
1187  
1188  
1189  
1190  
1191  
1192  
1193  
1194  
1195  
1196  
1197  
1198  
1199  
1200  
1201  
1202  
1203  
1204  
1205  
1206  
1207  
1208  
1209  
1210  
1211  
1212  
1213  
1214  
1215  
1216  
1217  
1218

1219  
1220  
1221  
1222  
1223  
1224  
1225  
1226  
1227  
1228  
1229  
1230  
1231  
1232  
1233  
1234  
1235  
1236  
1237  
1238  
1239  
1240  
1241  
1242  
1243  
1244  
1245  
1246  
1247  
1248  
1249  
1250  
1251  
1252  
1253  
1254  
1255  
1256  
1257  
1258  
1259  
1260  
1261  
1262  
1263  
1264  
1265  
1266  
1267  
1268  
1269  
1270  
1271  
1272  
1273  
1274  
1275  
1276