An MLP-based Algorithm for Efficient Contrastive Graph Recommendations

Siwei Liu
s.liu.4@research.gla.ac.uk
University of Glasgow
Glasgow, Scotland

Iadh Ounis, Craig Macdonald
first.lastname@glasgow.ac.uk
University of Glasgow
Glasgow, Scotland

ABSTRACT

Graph-based recommender systems (GBRSs) have achieved promising performance by incorporating the user-item bipartite graph using the Graph Neural Network (GNN). Among GBRSs, the information from each user and item's multi-hop neighbours is effectively conveyed between nodes through neighbourhood aggregation and message passing. Although effective, existing neighbourhood information aggregation and passing functions are usually computationally expensive. Motivated by the emerging contrastive learning technique, we design a simple neighbourhood construction method in conjunction with the contrastive objective function to simulate the neighbourhood information processing of GNN. In addition, we propose a simple algorithm based on Multilayer Perceptron (MLP) for learning users and items' representations with extra non-linearity while lowering computational burden compared with multi-layers GNNs. Our extensive empirical experiments on three public datasets demonstrate that our proposed model, i.e. MLP-CGRec, can reduce the GPU memory consumption and training time by up to 24.0% and 33.1%, respectively, without significantly degenerating the recommendation accuracy in comparison with competitive baselines.

ACM Reference Format:
Siwei Liu and Iadh Ounis, Craig Macdonald. 2022. An MLP-based Algorithm for Efficient Contrastive Graph Recommendations. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '22), July 11–15, 2022, Madrid, Spain.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3477495.3531874

1 INTRODUCTION

Recommender systems are designed to efficiently and effectively serve items of interest to users from a large amount of available options. Recently, recommender systems based on graph neural networks have attracted increasing attention in the community, and state-of-the-art recommendation performance has been achieved by such models including LightGCN [8] and UltraGCN [22]. The advantages of applying a GNN, especially multi-layer or heterogeneous GNNs, for the recommendation task lie in two pivotal functions, namely *neighbourhood aggregation* and *message passing*. By applying a neighbourhood aggregation function, the interactive information between users and items can be captured in their representations through message passing over the edges of a user-item bipartite graph. Besides, stacking multiple layers of GNNs can help recommender systems pass message over multi-hop neighbours.

However, the neighbourhood aggregation and message passing of the existing graph-based recommender systems (GBRSs) heavily rely on the pre-computed adjacency matrix $\mathbf{A}$ of the user-item bipartite graph, which is time- and memory-consuming to compute. Although $\mathbf{A}$ is normally sparse, many epochs of iterative matrix multiplications and gradient updates over the concatenation of users and items' embeddings is still less efficient, especially when multiple stacked layers of GNNs are applied [8, 38]. We argue that the inefficiency of the existing GBRSs is mainly caused by the matrix multiplications over unnecessary neighbours. Indeed, some frequently interacted items would be aggregated to most users, causing more time- and memory-consumption with trivial improvements. Given that the performance gained by the GBRSs is attributed to neighbourhood aggregation and message passing [6], the challenge here is to exploit a more efficient manner to perform similar functionalities without using the full neighbourhood graph convolution. In this paper, we propose a Multilayer Perceptron (MLP) based Contrastive Graph Recommender, abbreviated as MLP-CGRec, that uses an MLP-Mixer [31] to encode users and items' representations and conducts efficient contrastive learning. The key idea of our approach is that, instead of aggregating and passing message over all the neighbours, we sample each user and item's neighbours to form each positive pair. To conduct the message passing function efficiently, we propose to sample multi-hop neighbours by raising the adjacency matrix $\mathbf{A}$ to the power of n, which is more efficient than the iterative matrix multiplications. In particular, the used MLP-Mixer in our MLP-CGRec (motivated by of MLP-Mixer in computer vision), is able to model complex user-item interactions and enhance our model's expressive power, ensuring the effectiveness.

Our proposed MLP-CGRec offers multiple positive samples, which will lead to an unbalanced number of positive and negative items. Instead of sampling more random negative items, we use an approximate nearest neighbour search method to efficiently sample an equal amount of contrastive negative items to rebalance learning. To summarise, our contributions are threefold: (1) We incorporate a contrastive loss and a novel graph sampling method to simplify the neighbourhood aggregation and message passing of GBRSs; (2) We employ an efficient MLP-based learning algorithm to enhance the expressive power and recommendation accuracy; (3) We conduct extensive experiments on three public datasets, and show that our proposed MLP-CGRec can achieve high efficiency regarding both the memory and time consumption compared with state-of-the-art GBRSs without a significant loss of effectiveness.

SIGIR '22, July 11–15, 2022, Madrid, Spain

2 RELATED WORK

Recently, GBRSs have achieved promising performance in not only the general recommendation [8, 26, 38, 40] but also the sequential [24, 36, 42, 44], social [18, 34, 41, 45] and knowledge-based [7, 35, 37] recommendations. The most crucial functionalities of GBRSs are *neighbourhood aggregation* and *message passing* functions. Specifically, by aggregating information from graph neighbours, recommenders can learn more representative embeddings because neighbouring users are likely to share similar interests, and neighbouring items might share the same properties. Although effective, we notice that GBRSs have higher model complexity than traditional embedding-based recommenders, leading to higher training time and memory consumption. Therefore, building on NGCF [38], LightGCN [8] is proposed as a lightweight variant by dropping the redundant neural operations. Afterwards, GF-CF [30] is proposed by leveraging the low-pass graph filtering theorem to avoid the training process with back-propagation, which requires a large embedding dimension for competitive performance. UltraGCN [22] was proposed to speed up the training by avoiding the message passing function. However, it requires the graph representation learning on dual graphs, i.e. the user-item graph and item-item graph, which even aggravates the memory consumption. Another line of research focuses on graph contrastive learning, where graph augmentation techniques are used to create different views of nodes for the following contrastive objective function [19, 19, 24, 25, 39, 40]. However, graph augmentations based on stochastic perturbations, including random walk and random node/edge dropout [19, 40, 47], are usually performed for every training epoch, which will increase the training time and memory consumption. Different from existing GBRSs, our MLP-CGRec uses a simple neighbourhood construction method and a contrastive objective function to learn information from neighbours. Furthermore, we use a straightforward MLP-Mixer [31] to encapsulate more non-linearity for learnt users and items' embeddings. Therefore, compared with existing GBRSs, our proposed MLP-CGRec is highly efficient and effective.

3 METHODOLOGY

3.1 Preliminaries

We consider a recommender system with a set of users U ($|U| = M$) and a set of items I ($|I| = N$). Let $\mathbf{R} \in \mathbb{R}^{M \times N}$ be the user-item interaction matrix, where the content of the matrix $\mathbf{R}^{M \times N}$ corresponds to implicit feedback [11, 27]. We consider implicit feedback here because it is more abundant, therefore, $\mathbf{R}_{ui} = 1$ if the user u has interacted with the item i, otherwise $\mathbf{R}_{ui} = 0$. Following the BPR [27] framework for training, each triplet (u, i^+, i^-) contains a user, a positive item and a randomly selected negative item.

3.2 Graph Neighbourhood Construction

From the user-item interaction matrix $\mathbf{R}$, we can obtain its corresponding adjacency matrix $\mathbf{A}^{(M+N)\times(M+N)}$. The elements of $\mathbf{A}$ indicate whether the pairs of users and items are adjacent or not in the interaction graph. In existing GBRSs, the multi-hop graph embeddings of users and items are usually computed by:

$$\mathbf{E}^{(l+1)} = (\mathbf{D}^{-\frac{1}{2}}\mathbf{A}\mathbf{D}^{-\frac{1}{2}})\mathbf{E}^{(l)} \tag{1}$$

where $\mathbf{E}^{(l)}$ containing embeddings of all users and items at l-th layer of the GNN, and $\mathbf{D}$ is the diagonal degree matrix of $\mathbf{A}$.

Starting from the initial embeddings $\mathbf{E}^{(0)}$, we follow Equation (1) to compute $l-1$ times to obtain the l-hop embeddings. Although the adjacency matrix and the diagonal degree matrix are both sparse, the users and items' embeddings are however dense and the whole process not only involves the matrix multiplication but also the iterative gradient update for the GNNs at each training epoch. This is why existing GBRSs have a relatively unsatisfactory training efficiency. To avoid this costly operation, we derive how to use the adjacency matrix to explore the l-hop neighbours by raising $\mathbf{A}$ to the power of l below with an induction process:

Lemma 3.1. *The $(i, j)^{th}$ entry $a_{ij}^{(l)}$ of $\mathbf{A}^l$, where $\mathbf{A}$ is the adjacency matrix of $\boldsymbol{R}$, counts the number of walks of length l having start and end nodes i and j respectively.*

Proof. **Base case**: When $l = 1$, $\mathbf{A}^l = \mathbf{A}$, and there is walk between node i and j if and only if a_{ij}=1, thus the result holds. **Induction step**: Assume the proposition holds for $l = n$ and consider the case when $l = n + 1$, i.e. the matrix $\mathbf{A}^{n+1} = \mathbf{A}^n\mathbf{A}$. From the induction hypothesis, the value of (i, j) of the matrix $\mathbf{A}^n$ is the count of walks of length n from i to j. Now, the number of walks of length $n + 1$ between node i and node j equals the number of walks of length n from node i to each node v, which is adjacent to node j. Therefore, the number of walks of length $n + 1$ from node i to node j, i.e. the $(i, j)^{th}$ entry $\mathbf{A}^{n+1}$ is the non-zero entries of the column of $\mathbf{A}$, which corresponds exactly to the first neighbours of v. Thus the result holds for $l = n + 1$ as well. **Conclusion**: By the principle of induction, Lemma 3.1 is true for all $l \in \mathbb{Z}^+$. □

From the induction above, we can draw the conclusion that the l-hop neighbours of $\mathbf{R}$ can be obtained by raising $\mathbf{A}$ to the power of l. For example, we can refer to the u-th row of the matrix A^2 to find user u's 2-hop neighbours. Therefore, by pre-defining how many neighbourhoods we aim to incorporate, we can pre-compute the corresponding graph neighbourhood matrices. For the most commonly adopted case in GBRSs, where 3-hop neighbours are included, we can pre-compute $\mathbf{A}^2$ and $\mathbf{A}^3$ of $\mathbf{R}$ for the subsequent sampling. With the graph neighbourhood construction, we obtain additional positive samples but without a balanced number of negative samples, which might lead to the over-repelling of some negative samples from other data points [2]. Therefore, in the next section, we describe how to use the contrastive negative sampling approach to select negative samples to balance the number of positive and negative samples.

3.3 Contrastive Negative Sampling

Given that each training triplet (u, i^+, i^-) is extended to a training instance with multiple positive samples, we need to obtain more negative samples to reach a balance. Similar to how we get i^-, the most straightforward way is to randomly sample more negative items. However, inspired by the recent work of contrastive negative sampling [29, 43], we propose to sample negative items based on a similarity function instead of sampling from random ones.

In the existing work, contrastive negative items are usually defined as those generated negative items obtained from the positive items using the data augmentation method [13, 40]. The motivation

of this sampling method is to obtain hard negative items which can be used to enhance the model's discriminative power. Although effective, generating contrastive negative items can pose challenges regarding the training and memory efficiencies. Since both the augmented and original graphs are created, stored and processed at every epoch, the time consumption will increase and the memory consumption will experience a surge. Besides, there is no theoretical analysis of why those data augmentation techniques, including the random edge/node dropouts, can enhance the overall accuracy [46]. Therefore, we propose to re-sample negative items from existing items, which avoids the redundant data augmentation step.

We use the cosine similarity [3, 29] as the distance metric to search for contrastive negative items from each user's pool of uninteracted items. Specifically, we select those negative items $i_u^-{}'$ that have lower cosine similarity towards the target user u.

$$i_u^-{}' \in f_{\text{top-k}}\big(-\cos(\mathbf{e}_u, \mathbf{E}_{I_u^-})\big) \tag{2}$$

where $\mathbf{e}_u$ is the embedding of the target user and $\mathbf{E}_{I_u^-}$ represents all embeddings of this user's negative item.

To further increase the training efficiency, we adopt the fast neighbour search library, Faiss [14], to reduce the search time. Given the number of users and items in used datasets, we follow the advice in [14] to use the flat indexes and the brute-force search [1] to avoid accuracy loss. In particular, we select those negative items with lower similarity scores to avoid the false-negative items. As the recommendation accuracy increases along the model training, we can become more confident that target users are unlikely to prefer items with lower similarity scores, which will benefit the subsequent contrastive training.

3.4 MLP-Mixer

Although the graph neighbourhood construction and contrastive negative sampling are efficient, the model's expressive power is limited without a nonlinear activation. Recently, some advanced variants of Multilayer Perceptron (MLP) [17, 31, 32] have been proposed for more efficient deep model training. Inspired by their high efficiency and competitive accuracy on the image classification and NLP tasks, we propose to adopt MLP-Mixer to enhance our model's expressive power without a high computational cost. Our MLP module is defined as:

$$\mathbf{E}_{n+1} = \mathbf{E}_n + \mathbf{W}_2\sigma\Big(\mathbf{W}_1\text{LayerNorm}(\mathbf{E}_n)\Big) \tag{3}$$

where $\mathbf{E}_{n+1}$ contains embeddings for all users and items at the $n+1$ epoch; $\sigma(\cdot)$ is the GELU [10] activation; $\mathbf{W}_1$ and $\mathbf{W}_2$ are trainable weight matrices; and LayerNorm$(\cdot)$ denotes the layer normalisation [1], which is used to enhance the training stability.

In Equation (3), we use a self-addition loop i.e. adding the original $\mathbf{E}_n$ to $\mathbf{E}_{n+1}$, which is inspired by the tying parameter technique in MLP-Mixer [31]. The self-addition loop can help the model preserve the information from the previous epoch and avoid overfitting. To justify our choice, we also implement a variant without the self-addition loop, which is identical to the MLP architecture used by the Graph-MLP model. We term this variant as plain MLP and we will compare our model to it in an ablation study.

[1] Using flat indexes and the brute-force search is mathematically the same with the commonly used exact search. However, the overall search process is accelerated by optimising the General Matrix Multiply (GEMM) routines in the cuBLAS library for the GPU acceleration.

Table 1: Statistics of the datasets.

	MovieLens	Yelp	Amazon
Users	6,038	19,539	65,387
Items	3,533	21,266	38,776
Interactions	575,281	450,884	739,380
Density(%)	2.697	0.108	0.029

3.5 Contrastive Training

Contrastive learning aims to encourage similar pairs to stay close to each other while dissimilar ones are far apart in the latent space [4, 5]. In the contrastive recommendation scenario [39, 47], we target learning representations for users and items, where interacted users and items stay closer in the learnt space. Inspired by the performance of InfoNCE loss, we construct our objective function as follows:

$$\mathcal{L} = \sum -\log\frac{f_s(\mathbf{e}_u, \mathbf{e}_{i_u^+}) + \sum_2^n f_s(\mathbf{e}_u, \mathbf{e}_{n^+})}{f_s(\mathbf{e}_u, \mathbf{e}_{i_u^+}) + f_s(\mathbf{e}_u, \mathbf{e}_{i_u^-}) + \sum_2^n f_s(\mathbf{e}_u, \mathbf{e}_{i_{con}^-})} \tag{4}$$

where $f_s(\cdot) = e^{\cos(\cdot)}$; i_u^+, n^+, i_u^- and i_{con}^- denote a positive item, a n-hop neighbour, a random negative item and a contrastive negative item of user u, respectively; n determines how many hops of neighbours to incorporate.

With Equation (4), we aim to encourage users to stay closer with their positive items and multi-hop neighbours while maximising the distance between users with their random and contrastive negative items. To predict each user's preferred items, we use the dot product between the embedding of each user and embeddings of all items, where more preferable items will receive higher scores.

4 DATASETS AND EXPERIMENTAL SETUP

Three public datasets, i.e. MovieLens-1M [2], Yelp [3] and Amazon-electronics [4], are used to evaluate our proposed MLP-CGRec model. In particular, MovieLens-1M is a dataset containing interactions between users and movies; Yelp is a venue check-ins dataset; and Amazon-electronics is a subset of the Amazon review dataset. For the rest of the paper, we respectively denote 'MovieLens' and 'Amazon' as 'MovieLens-1M' and 'Amazon-electronics'. Table 1 provides the statistics of the three used datasets. In the following, we aim to answer the following research questions:

RQ1. Can MLP-CGRec achieve higher efficiency compared with the existing GBRS without significantly degrading its recommendation effectiveness?

RQ2. What is the impact of these additional samples?

RQ3. What is the impact of the MLP-based learning method?

To answer **RQ1**, we compare our MLP-CGRec model with the following baselines: BPRMF [27], NeuMF [9], NGCF [38], LightGCN [8], UltraGCN [22], MixGCF [13] and SGL [40]. Since efficiency and effectiveness are both critical in our study, we compare our MLP-CGRec with all baselines in terms of Normalised Discounted Cumulative Gain@10 (NDCG), Hit Ratio@10 (HR), max memory consumption, average epoch time and total training time, where the max memory consumption is monitored by Tensorboard. Following a common setup, we use a leave-one-out evaluation strategy to split

[2] https://grouplens.org/datasets/movielens/ [3] https://www.yelp.com/dataset
[4] https://jmcauley.ucsd.edu/data/amazon/

Table 2: Experimental results of MLP-CGRec and other baselines on the three used datasets w.r.t. HR@10, NDCG@10, max memory consumption (gigabytes), average epoch time (seconds) and total training time (minutes). The best accuracy is highlighted in bold and the second best result is highlighted with underline. * denotes a significant difference compared to the result of MLP-CGRec using the two one-sided test with p<0.05.

	MovieLens					Yelp					Amazon				
	NDCG	HR	Mem	Epoch	Total	NDCG	HR	Mem	Epoch	Total	NDCG	HR	Mem	Epoch	Total
BPRMF	0.2031*	0.1274*	2.91	45.8	61.2	0.1221*	0.1520*	5.42	84.5	96.7	0.0745*	0.1138*	5.78	95.1	147.6
NeuMF	0.2114*	0.1398*	3.88	58.7	100.5	0.1330*	0.1598*	6.07	98.9	123.3	0.1093*	0.1678*	6.31	132.2	188.5
NGCF	0.2517*	0.1665*	4.13	65.7	161.0	0.1420*	0.1736*	7.64	112.4	168.6	0.1297*	0.1927*	7.91	168.9	241.4
LightGCN	**0.3303**	**0.2329**	3.96	59.8	118.5	**0.2233**	0.2597	6.43	102.3	136.6	0.1519	0.2177	7.01	149.2	206.0
UltraGCN	0.2646*	0.1862*	3.98	61.3	147.5	0.2111	**0.2619**	6.70	108.9	166.8	0.1302*	0.2021*	7.11	145.2	208.2
MixGCF	0.2737*	0.1989*	4.38	60.3	132.2	0.2003*	0.2378*	6.91	107.8	158.4	0.1403*	0.2107	8.03	143.9	211.4
SGL	0.3116	0.2260	4.08	69.3	135.6	0.1673*	0.2098*	6.61	101.9	145.8	0.1581	**0.2201**	6.58	151.5	184.8
MLP-CGRec	0.3004	0.2176	3.01	48.2	78.5	0.2164	0.2501	5.71	91.3	121.2	**0.1593**	0.2190	5.90	101.3	152.2
Diff (%)	-6.02	-6.57	-24.0	-19.4	-33.7	-3.09	-4.51	-11.2	-10.1	-12.7	+4.87	-0.50	-10.3	-33.1	-17.6

the interactions of each dataset into training, validation and testing sets. However, different from prior works [9, 28] that only use one oracle testing set per dataset with the sampled negative items, we construct 10 different testing sets with different sampled negative items for each dataset using different random seeds, in order to reduce the evaluation bias on some specific testing negatives [15]. Hence, the reported performance of each run is based on the average of the 10 testing sets. For a fair comparison, we conduct all experiments on the same machine with a GeForce RTX 2080Ti GPU. For significant testing, we apply a two one-sided of equivalence test (TOST), with ΔAP=0.05 [16, 20, 21, 23]. The purpose of the TOST test is to examine if MLP-CGRec is significantly equivalent to a baseline with the acceptable range of inequality being ±5%. We define success as outperforming a baseline regarding accuracy and efficiency or outperforming a baseline regarding efficiency but significantly equivalent to it regarding accuracy.

To answer **RQ2**, we compare the following variants of MLP-CGRec: (i) 2-hop neighbours with contrastive negative items and (ii) 3-hop neighbours with contrastive negative items. Hence, we can clearly examine the effect of each type of neighbours. Recall that contrastive negative items are sampled as complements only when multi-hop neighbours are incorporated so we can neglect the condition when only contrastive negative items are sampled. To answer **RQ3**, we use an ablation study to examine the effectiveness of the proposed method when: (i) the MLP module is not applied; (ii) a plain MLP is applied; (iii) MLP-mixer is applied.

The latent dimension and batch size are fixed to 64 and 1000, respectively, for all models. For each dataset, we use 20% of the interactions as a test set; of the remaining, we use 10% as a validation set, and the remainder for training. For the trainable matrices $\mathbf{W}_1$ and $\mathbf{W}_2$ used in MLP-CGRec, we closely follow the implementation details in [31] and set $\mathbf{W}_1$ and $\mathbf{W}_2$ to 32, which is half of the input latent dimension. For the $f_{\text{top-k}}$ function, we empirically set k to 100 according to our early stage experiments. To tune all hyper-parameters, we apply a grid search, where the learning rate is tuned in $\{10^{-2}, 10^{-3}, 10^{-4}\}$; and the L_2 normalisation in $\{10^{-1}, 10^{-2}, ..., 10^{-5}\}$. The node dropout technique is adopted in the NGCF, SGL and LightGCN models, and the ratios vary amongst $\{0.3, 0.4, ..., 0.8\}$ as suggested in [33]. We use 3-hop neighbours in MLP-CGRec following other GBRSs.

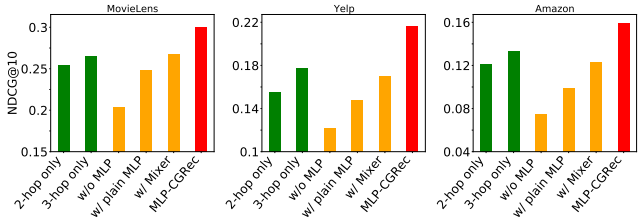

Figure 1: Comparison between different variants of MLP-CGRec, where the green bars are variants with multiple neighbours, orange bars are variants with different MLPs and red bar is the final MLP-CGRec model.

5 RESULTS

Table 2 reports the overall performances of our MLP-CGRec model and all other baselines. The results show that MLP-CGRec achieves competitive performance on all used datasets. On MovieLens, although LightGCN and SGL slightly outperform our MLP-CGRec, both models do not surpass MLP-CGRec by a significant difference according to a TOST test. More importantly, our model generally performs better on larger datasets than on smaller datasets. For example, MLP-CGRec achieves the second-best and the best on the larger Yelp and Amazon datasets regarding NDCG@10, respectively. This observation is consistent with the findings in [12, 31], where both Graph-MLP and MLP-Mixer models only outperform state-of-the-art models on larger datasets for both node and image classification tasks. We notice that SGL and UltraGCN do not always outperform LightGCN as reported in [22] and [40]. This is due to the difference in experimental setup where we additionally use 10 different testing sets to reduce the evaluation bias. Therefore, our results further demonstrate the promising generalisability of a simple MLP-based learning architecture, whose scalability is better than complicated networks. Furthermore, our proposed MLP-CGRec model consistently achieves the best efficiency among all neural recommenders regarding GPU memory consumption and training time. We owe this high efficiency of MLP-CGRec to the simple design of the MLP module and the neighbourhood construction, which can be accomplished quickly using the sparse matrix

multiplication. Therefore, we conclude that our MLP-CGRec can achieve competitive recommendation effectiveness with a higher efficiency compared with existing GBRSs.

Figure 1 plots the NDCG@10 performances of all different variants of MLP-CGRec on all used datasets. In order to answer **RQ2**, we compare the variants of MLP-CGRec without using encoders over different hops of neighbours (green bars). Recall that a user's 1-hop, 2-hop and 3-hop neighbours are items, users, and items, respectively. Here, 1-hop neighbours are neglected because they are already included in the interaction graph neglected. By comparing the performances of variants with 2-hop and 3-hop neighbours, we find that the 3-hop neighbours bring more gains over the 2-hop neighbours, which means that users sharing one interacted item may not necessarily share the same overall interests. This explains why the 3-hop neighbourhood aggregation becomes the most common setup of the GBRSs. In answer to **RQ3**, the variant using Mixer constantly outperforms the one with a plain MLP and the plain MLP surpasses the one with no MLP on all datasets. This observation justifies our choice of incorporating the MLP architecture inspired by the MLP-Mixer as our representation learning module. Lastly, none of the variants can outperform MLP-CGRec, which means the integration of all proposed modules can achieve the best performance.

6 CONCLUSIONS

We propose MLP-CGRec, an MLP-based recommender that uses a neighbourhood construction method and a contrastive objective to replace the classic neighbourhood aggregation and message passing to achieve a competitive recommendation accuracy with an up to 33.7% running time reduction. Our proposed MLP-CGRec model has been demonstrated to achieve the best efficiency and comparable effectiveness compared with state-of-the-art graph-based and contrastive baselines on three public datasets. Furthermore, our ablation study reveals the effects of different proposed modules.

REFERENCES

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. In *Proc. of Deep Learning Symposium*.

[2] Shuo Chen, Gang Niu, Chen Gong, Jun Li, Jian Yang, and Masashi Sugiyama. 2021. Large-margin contrastive learning with distance polarization regularizer. In *Proc. of ICML*.

[3] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *Proc. of ICML*.

[4] Ting Chen, Simon Kornblith, Kevin Swersky, Mohammad Norouzi, and Geoffrey Hinton. 2020. Big self-supervised models are strong semi-supervised learners. In *Proc. of NeurIPS*.

[5] Sumit Chopra, Raia Hadsell, and Yann LeCun. 2005. Learning a similarity metric discriminatively, with application to face verification. In *Proc. of CVPR*.

[6] Chen Gao, Yu Zheng, Nian Li, Yinfeng Li, Yingrong Qin, Jinghua Piao, Yuhan Quan, Jianxin Chang, Depeng Jin, Xiangnan He, et al. 2021. Graph neural networks for recommender systems: Challenges, methods, and directions. *ACM Transactions on Information Systems* 1 (2021).

[7] Xuri Ge, Fuhai Chen, Joemon M Jose, Zhilong Ji, Zhongqin Wu, and Xiao Liu. 2021. Structured multi-modal feature embedding and alignment for image-sentence retrieval. In *Proc. of SIGMM*.

[8] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. LightGCN: Simplifying and powering graph convolution network for recommendation. In *Proc. of SIGIR*.

[9] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017. Neural collaborative filtering. In *Proc. of WWW*.

[10] Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415* (2016).

[11] Y. Hu, Y. Koren, and C. Volinsky. 2008. Collaborative filtering for implicit feedback datasets. In *Proc. of ICDM*.

[12] Yang Hu, Haoxuan You, Zhecan Wang, Zhicheng Wang, Erjin Zhou, and Yue Gao. 2021. Graph-MLP: node classification without message passing in graph. *arXiv preprint arXiv:2106.04051* (2021).

[13] Tinglin Huang, Yuxiao Dong, Ming Ding, Zhen Yang, Wenzheng Feng, Xinyu Wang, and Jie Tang. 2021. MixGCF: An improved training method for graph neural network-based recommender systems. In *Proc. of SIGKDD*.

[14] Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data* 7 (2019).

[15] Walid Krichene and Steffen Rendle. 2020. On sampled metrics for item recommendation. In *Proc. of SIGKDD*.

[16] Binsheng Liu, Nick Craswell, Xiaolu Lu, Oren Kurland, and J Shane Culpepper. 2019. A comparative analysis of human and automatic query variants. In *Proc. of SIGIR*.

[17] Hanxiao Liu, Zihang Dai, David R So, and Quoc V Le. 2021. Pay attention to MLPs. In *Proc. of NeurIPS*.

[18] Siwei Liu, Iadh Ounis, Craig Macdonald, and Zaiqiao Meng. 2020. A heterogeneous graph neural model for cold-start recommendation. In *Proc. of SIGIR*.

[19] Zhuang Liu, Yunpu Ma, Yuanxin Ouyang, and Zhang Xiong. 2021. Contrastive learning for recommender system. *arXiv preprint arXiv:2101.01317* (2021).

[20] Sean MacAvaney, Franco Maria Nardini, Raffaele Perego, Nicola Tonellotto, Nazli Goharian, and Ophir Frieder. 2020. Efficient document re-ranking for transformers by precomputing term representations. In *Proc. of SIGIR*.

[21] Joel Mackenzie, J Shane Culpepper, Roi Blanco, Matt Crane, Charles LA Clarke, and Jimmy Lin. 2018. Query driven algorithm selection in early stage retrieval. In *Proc. of WSDM*.

[22] Kelong Mao, Jieming Zhu, Xi Xiao, Biao Lu, Zhaowei Wang, and Xiuqiang He. 2021. UltraGCN: Ultra simplification of graph convolutional networks for recommendation. In *Proc. of CIKM*.

[23] Hafeezul Rahman Mohammad, Keyang Xu, Jamie Callan, and J Shane Culpepper. 2018. Dynamic shard cutoff prediction for selective search. In *Proc. of SIGIR*.

[24] Shameem Puthiya Parambath, Christos Anagnostopoulos, Roderick Murray-Smith, Sean MacAvaney, et al. 2021. Max-Utility based arm selection strategy for sequential query recommendations. In *Proc. of ACML*.

[25] Shameem A Puthiya Parambath and Sanjay Chawla. 2020. Simple and effective neural-free soft-cluster embeddings for item cold-start recommendations. *Data Mining and Knowledge Discovery* 34, 5 (2020).

[26] Shameem A Puthiya Parambath, Nicolas Usunier, and Yves Grandvalet. 2016. A coverage-based approach to recommendation diversity on similarity graph. In *Proc. of RecSys*. 15–22.

[27] Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. 2009. BPR: Bayesian personalized ranking from implicit feedback. In *Proc. of UAI*.

[28] Steffen Rendle, Walid Krichene, Li Zhang, and John Anderson. 2020. Neural collaborative filtering vs. matrix factorization revisited. In *Proc. of RecSys*.

[29] Joshua Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. 2020. Contrastive learning with hard negative samples. In *Proc. of ICLR*.

[30] Yifei Shen, Yongji Wu, Yao Zhang, Caihua Shan, Jun Zhang, B Khaled Letaief, and Dongsheng Li. 2021. How powerful is graph convolution for recommendation?. In *Proc. of CIKM*.

[31] Ilya Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Daniel Keysers, Jakob Uszkoreit, Mario Lucic, et al. 2021. Mlp-mixer: An all-mlp architecture for vision. In *Proc. of NeurIPS*.

[32] Hugo Touvron, Piotr Bojanowski, Mathilde Caron, Matthieu Cord, Alaaeldin El-Nouby, Edouard Grave, Gautier Izacard, Armand Joulin, Gabriel Synnaeve, Jakob Verbeek, et al. 2021. Resmlp: Feedforward networks for image classification with data-efficient training. *arXiv preprint arXiv:2105.03404* (2021).

[33] Rianne van den Berg, Thomas N Kipf, and Max Welling. 2018. Graph convolutional matrix completion. In *Proc. of KDD*.

[34] Janu Verma, Srishti Gupta, Debdoot Mukherjee, and Tanmoy Chakraborty. 2019. Heterogeneous edge embedding for friend recommendation. In *Proc. of ECIR*.

[35] Hongwei Wang, Fuzheng Zhang, Jialin Wang, Miao Zhao, Wenjie Li, Xing Xie, and Minyi Guo. 2018. Ripplenet: Propagating user preferences on the knowledge graph for recommender systems. In *Proc. of CIKM*.

[36] Wen Wang, Wei Zhang, Shukai Liu, Qi Liu, Bo Zhang, Leyu Lin, and Hongyuan Zha. 2020. Beyond clicks: Modeling multi-relational item graph for session-based target behavior prediction. In *Proc. of WWW*.

[37] Xiang Wang, Xiangnan He, Yixin Cao, Meng Liu, and Tat-Seng Chua. 2019. Kgat: Knowledge graph attention network for recommendation. In *Proc. of SIGKDD*.

[38] Xiang Wang, Xiangnan He, Meng Wang, Fuli Feng, and Tat-Seng Chua. 2019. Neural graph collaborative filtering. In *SIGIR*.

[39] Ga Wu, Maksims Volkovs, Chee Loong Soon, Scott Sanner, and Himanshu Rai. 2019. Noise contrastive estimation for one-class collaborative filtering. In *Proc. of SIGIR*.

[40] Jiancan Wu, Xiang Wang, Fuli Feng, Xiangnan He, Liang Chen, Jianxun Lian, and Xing Xie. 2021. Self-supervised graph learning for recommendation. In *Proc. of SIGIR*.

[41] Le Wu, Junwei Li, Peijie Sun, Richang Hong, Yong Ge, and Meng Wang. 2020. Diffnet++: A neural influence and interest diffusion network for social recommendation. *IEEE Transactions on Knowledge and Data Engineering* (2020).
[42] Shu Wu, Yuyuan Tang, Yanqiao Zhu, Liang Wang, Xing Xie, and Tieniu Tan. 2019. Session-based recommendation with graph neural networks. In *Proc. of AAAI*.
[43] Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *Proc. of ICLR*.
[44] Chengfeng Xu, Pengpeng Zhao, Yanchi Liu, Victor S Sheng, Jiajie Xu, Fuzhen Zhuang, Junhua Fang, and Xiaofang Zhou. 2019. Graph contextualized self-attention network for session-based recommendation.. In *Proc. of IJCAI*.
[45] Junliang Yu, Hongzhi Yin, Min Gao, Xin Xia, Xiangliang Zhang, and Nguyen Quoc Viet Hung. 2021. Socially-aware self-supervised tri-training for recommendation. In *Proc. of SIGKDD*.
[46] Junliang Yu, Hongzhi Yin, Xin Xia, Tong Chen, Lizhen Cui, and Nguyen Quoc Viet Hung. 2022. Are graph augmentations necessary? simple graph contrastive learning for recommendation. In *Proc. of SIGIR*.
[47] Chang Zhou, Jianxin Ma, Jianwei Zhang, Jingren Zhou, and Hongxia Yang. 2021. Contrastive learning for debiased candidate generation in large-scale recommender systems. In *Proc. of SIGKDD*.