LLM and GNN-Enhanced Recommendation Systems for Tackling Cold-Start problem

John Houser
johnhouser@ucla.edu

Jeffrey Ma
jma777@ucla.edu

Parshan Teimouri
parshan@ucla.edu

Jonathan Carlson
jonathancarlson@ucla.edu

Abstract

Research into the cold start problem generally leverages side information or dense recommendation data in order to suggest items for new users. However, these kinds of approaches don't scale well to large amounts of users and items with sparse user-item interactions, becoming less accurate with the same amount of computational resources. We propose an approach to address this sparsity problem by taking a graph-based neural network recommendation system and augmenting its training data with generative data created from models trained on the cold-start data. Our approach uses an Attribute Graph Neural Network (Qian et al. [1]), which models user-to-user and item-to-item interactions via the attribute graph. Furthermore, the data for training the AGNN is augmented via both a GAN-based approach and a LLM-based approach, in order to take advantage of the LLM's emergent properties as well as the GAN's ability to generate precise data given imprecise data. Thus, even with less sparse user-item interactions we can enhance the accuracy of the model. From experiments on the MovieLens100k dataset, we determine that the AGNN augmented with the GAN data outperforms both the LLM and non-augmented approach.

1 Introduction

Recommendation systems are an important part of e-commerce, web applications, and entertainment systems, providing personalized suggestions per user preference. However, such systems often need large amounts of historical user data in order to make predictions. Specifically, the cold-start problem in recommendation systems arises when new users or items have limited or no historical interaction data, hindering the system's ability to make accurate recommendations, leading to sub-optimal user experiences. Addressing the cold-start problem requires innovative approaches that can intelligently leverage auxiliary information or user feedback to bootstrap recommendations for new entities. For example, similar approaches have used GAN to generate more precise data or a Knowledge Graph to give domain knowledge to the recommendation system.

2 Literature Review

Traditional recommendation systems like matrix factorization work well if there are previous user-item interactions. However, if the user-item matrix is sparse because the user has not interacted with many items then matrix factorization performs poorly. Cold users or items are those that do not appear in the training data (Qian et al. [1]). This lack of data may be because the users or items are brand new or not as popular as others.

Recent techniques for recommendation systems include graph neural networks (GNN) where users and items are represented in a bipartite graph. GNN-based approaches rely on these bipartite graphs where the nodes are users or items and interactions between users and items are the edges.

The Attribute-based GNN paper (Qian et al. [1]) makes the distinction between two kinds of cold starts: normal and strict. Normal cold start is where you recommend users or items that do not appear in the training set. Strict cold start means the users or items aren't in the training data and there are no interactions in the test state. An interaction is where the user rates a series of items which are then used to recommend new items.

Existing GNN methods such as STAR-GCN (Zhang et al. [2]) are bounded by the number of interactions and require users to rate items. This limits STAR-GCN to a normal cold start. Another GNN approach is HERS (Hu et al. [3]) which uses user-user and item-item relationships to resolve the strict cold start. These relationships disregard the node's attributes and may only choose popular items for the new user.

The Attribute-based GNN (AGNN) (Qian et al. [1]) attempts to address these limitations by using the attribute graph instead of the user-item interaction graph. For example, an item's attributes are category, description, price and a user's attributes could be their age and location. The idea is to use the attributes of a novel item to make better recommendations. One example is if a movie has the same director as a previous movie the user liked then it's more likely the user will watch the film.

Many of the prior approaches make use of user "side data", like user attributes, which might not always be available due to privacy laws and concerns. This limitation in data can be overcome using a generative model-based approach such as Cold-GAN (Chen et al. [4]). Cold-GAN was able to make use of users with lots of ratings, otherwise known as warm-user data, combined with simulated cold users, called rejuvenated vectors, that are created by dropping ratings of warm-user vectors to learn a generative and discriminative network. This network can learn using an appropriate item loss function. As a result, the generative part of the GAN can be used as an end-to-end system to predict a users' ratings given sparse cold user data.

Large Language Models (LLM) are trained on large amounts of data and their emergent qualities, such as reasoning and world knowledge, could be helpful in many tasks. Recently, the task of prompt engineering has emerged as a serious skill in obtaining meaningful output from LLMs. The right series of words can align the model closer to the intended results. Figuring out this exact combination can be challenging given the number of possible input phrases. Writing instructive, clear, and even encouraging text is a major component of writing high-quality prompts.

The performance of Recommender Systems are often challenged by data heterogeneity, which arises when the available side information comes from significantly different domains, and the lack of user and item attributes, usually referred to as data incompleteness. The Large Language Models with Graph Augmentation for Recommendation paper (Wei et al. [5]) leverages LLMs for tackling the cold start problem, by prompting the model to perform these tasks: (i) suggesting items based on user history, (ii) incorporating existing item information in decision making, and (iii) understanding the underlying tendencies of users when it comes to their interaction with items, namely user profiling.

3 Methodology

Our implementation used the MovieLens-100K Dataset (Harper and Konstan [6]) which consists of 100,000 ratings from 943 users for 1682 movies. This means that the user-item matrix is about 94% sparse. This means that of the possible user-item pairs only 6% of them are present in the dataset.

Figure 1 shows the distribution of the reviews in the dataset. We can see that the majority of the reviews are either a 3 or 4. This suggests that users are less likely to give a review on either end of the score range and prefer to give scores closer to the middle of the range. We determined that the MovieLens100k dataset was the easiest to work with given our limited time frame and computational power. Even with just this dataset the size of the GNN graph was in the order of millions of nodes.

3.1 AttributeGNN Architecture

The user-item interaction graph for real-world data is often very sparse. Sparsity makes it difficult to provide good recommendations for users. AttributeGNN attempts to solve the cold start problem

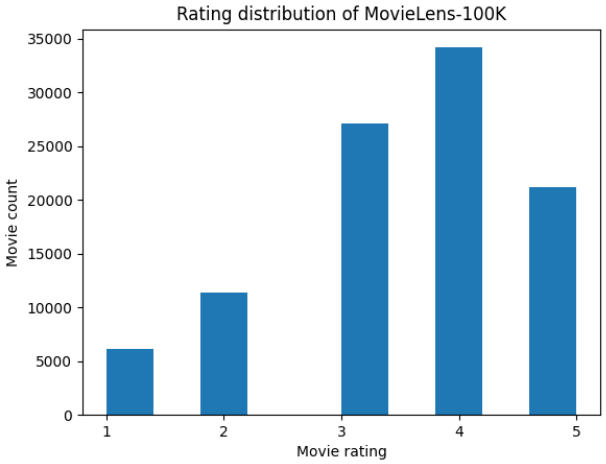

Figure 1: MovieLens-100K Rating Distribution

when there are no user-item interactions for novel users and we attempt to predict items for those cold users. The AttributeGNN model's idea is to use the user-user attribute or item-item attribute graphs to predict cold users or items. For example, a shared user attribute could be a user's location. If two users are in the same city then it's more likely they will have similar interests or should be marketed related products.

The AttributeGNN paper consistently outperforms all baselines in strict cold start scenarios across different datasets, demonstrating its effectiveness in handling cases where there are no pre-existing user-item interactions. This is particularly important in real-world applications where new users or items frequently enter the system without historical interaction data. AGNN's stable performance across various proportions of strict cold start nodes shows its robustness to different levels of sparsity in the user-item interaction data. This is contrasted with some baseline methods, whose performance significantly drops as the proportion of cold start nodes increases.

The overall architecture of AttributeGNN is shown in Figure 2. The extended Variational Autoencoder (eVAE) component plays a pivotal role in AGNN's success, particularly in strict cold start scenarios. By reconstructing preference embeddings from attribute information, eVAE uses attributes in the absence of interaction data. The ablation study demonstrates that removing or simplifying the eVAE structure leads to performance degradation, emphasizing its importance in the AGNN framework.

The comprehensive analysis across various experiments and settings demonstrates AGNN's strong capability in addressing strict cold start problems in recommender systems. Its use of attribute graphs, combined with the gated-GNN structure and the eVAE component, enables effective learning of user and item embeddings from attribute information, offering significant improvements over existing methods.

3.2 ColdGAN Architecture

In contrast to the original paper, we chose to use a Deep Convolutional Neural Network architecture for both the Generator and Discriminator. Figure 3 shows the architecture for both the Generator and Discriminator. The Generator should be arguably "stronger" than the discriminator, as it has a harder job of predicting the ratings themselves instead of if the ratings are fake or real. Thus, the generator is deeper, consisting of 3 Conv1dTranspose layers and 3 Conv1d Transpose layers. Batchnorm and Dropout are utilized for regularization, and ReLU is used as the activation function to introduce nonlinearity. An additional Maxpool layer is added after the Conv1dTranspose layers in order to spatially incorporate information from multiple areas of the rating vector. Meanwhile, the discriminator is slightly more shallow, using 4 Conv1d layers. The Dropout for the layers is significantly higher than the dropout for the generator, as it makes the discriminator slower to train than the generator, giving the generator a chance to "catch up" in strength.

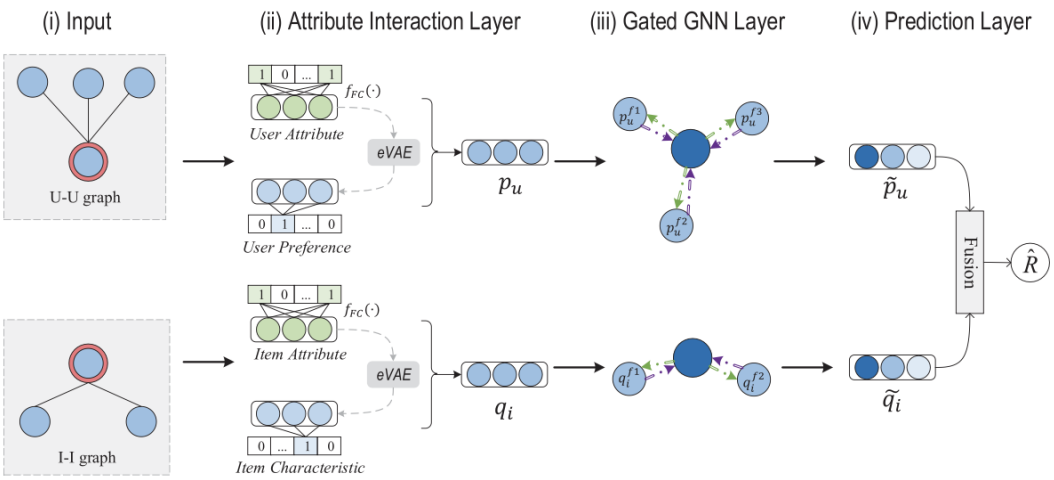

Figure 2: AttributeGNN Architecture

The core of the ColdGAN architecture consists of the rejuvenation function, which generates cold rating vectors from warm rating vectors, as well as the relevant item loss. The rejuvenation function generates cold vectors by probabilistically dropping certain items from the rating vector. This probability is reduced if the item is more popular or if the item was rated earlier in the user's history. The intuition behind this approach is that the items that are more popular and are rated earlier by the user act as "founding items" to determine a user's taste. Thus, one can get a warm, somewhat user specific rating vector given a non-specific cold vector based on what is in the cold vector. The relevant item loss deals with the mode collapse problem of GAN architectures, where the generator might learn the general distribution of ratings instead of a user-specific rating distribution. By averaging the binary cross entropy loss between the sigmoid of the generated vector and the relevance vector (defined as the ratings of the user greater than the average rating of that user), the relevant item loss pushes the generator towards user-specific rating distributions.

3.3 LLMRec Augmentation Strategy

The prompt construction strategy of LLMRec (Wei et al. [5]) can be found in Figure 4. The first strategy, Implicit Feedback, is to use the history of users' interactions to select more relevant items from a pool of candidates. The second one is to ask the model to predict user features, such as age and gender, by feeding the user interaction history to the model. Lastly, item characteristics are requested from the LLM, focusing on their world knowledge found in their training data.

The acquired information is further augmented with the original dataset, resulting in an enhanced feature list. A Graph Neural Network (GNN) is learned on the improved data, enjoying the underlying user-user, user-item, and item-item relations. Figure 5 depicts a summary of the taken approach.

We tested the efficacy of their approach by using the augmented data to train a GNN-based architecture, to measure the accuracy of the generated attributes.

3.4 Our Architecture

Figure 6 shows the system architecture of our proposal. To better compare the LLMRec (Wei et al. [5]) augmentation technique with other architectures, we replaced their GNN model with AGNN. The GNN for the AGNN paper is created from the user data and movie data provided in the dataset. Then, the LLMRec and ColdGAN models are trained independently to generate warm rating vector data from cold vector data. This warm rating vector data is generated for all users, and filtered based on rating value before being fed into the AGNN training data. Finally, the AGNN is evaluated on the test set.

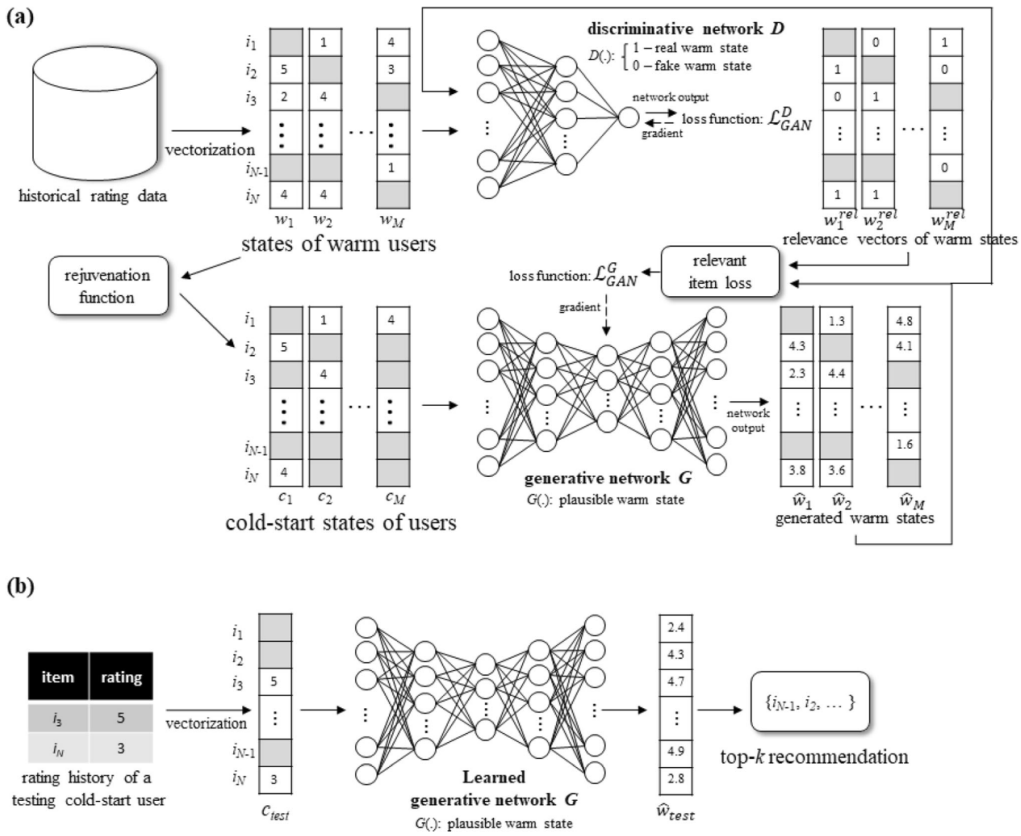

Figure 3: ColdGAN architecture

The paper LLMRec uses OpenAI's GPT-3.5 as its LLM. However, we decided to use the open-source Llama LLM instead. This decision was taken to reduce costs by avoiding paying the API costs of ChatGPT. Instead, we used the 7 billion parameter Llama model which we ran locally on our computer. While it is true that the quality of the outputs from Llama is not on the same level as ChatGPT, this was a sacrifice we decided was necessary.

4 Results and Discussion

4.1 Limitations

The ColdGAN paper (Chen et al. [4]) did not provide the code for their model. Therefore, we had to make estimates about what architecture they used such as how many layers and how many parameters each layer should have. We used the PyTorch DCGAN Tutorial (Inkawhich [7]) as a reference when implementing the ColdGAN model. Since the model's code was not provided then it's possible that the paper's model would have different behavior than our model leading to inconsistent results.

Another limitation was that we lacked the computational resources to use all of the augmented data pairs. So we decided to randomly sample 300,000 pairs from the augmented data. For MovieLens-100K there are approximately 1.6 million user-movies pairs so combining the training data accounts for 25% of the possible pairs. Not having enough computational resources also meant we weren't be able to create larger models which could have performed better on the MovieLens-100K dataset.

A further limitation is that we only evaluated our models on the MovieLens-100K dataset. There are are movie datasets such as MovieLens-1M as well as datasets from different domains such as the Yelp dataset (Yelp [8]) and the Amazon dataset (Ni, Jianmo [9]). We should run our model on these datasets to evaluate how well our models and approach perform in a wider variety of domains.

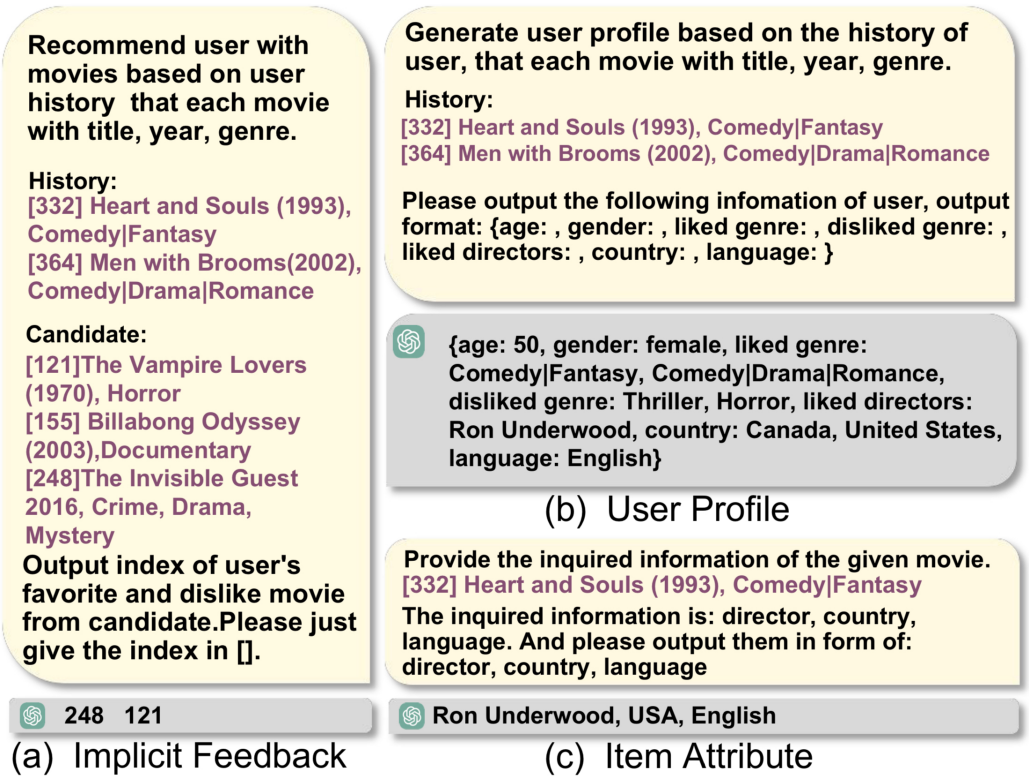

Figure 4: Construction of prompts in LLMRec

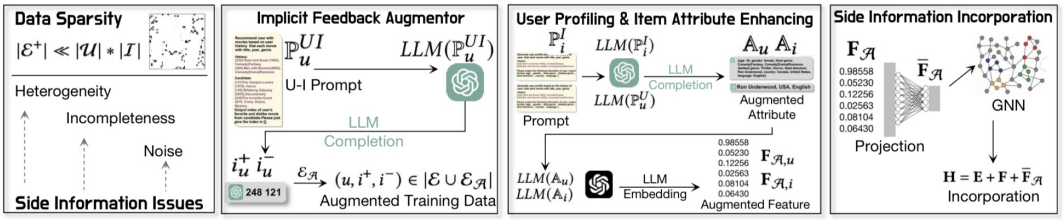

Figure 5: Augmentation Strategy in LLMRec

4.2 Overall performance

The metrics that we use to evaluate the performance of our model are calculated on the test set of the MovieLens-100K dataset which was 20% of the dataset or 20,000 ratings. Furthermore, we evaluated our model in solving the user cold-start problem where there's a new user and we want to predict their rating for a movie they haven't seen.

We evaluate our model using the Root Mean Squared Error (RMSE). For two ratings the RMSE is:

$$RMSE = \sqrt{\frac{1}{N}\sum_{u,i}(\hat{R}_{u,i} - R_{u,i})^2}$$

where N is the number of data points being tested, $\hat{R}_{u,i}$ is the predicted rating and $R_{u,i}$ is the ground truth rating.

Metric	AGNN	AGNN-ColdGAN	AGNN-LLMRec	AGNN Paper
RMSE	1.0248	1.0201	1.0269	1.0208

Table 1: RMSE Results for our models

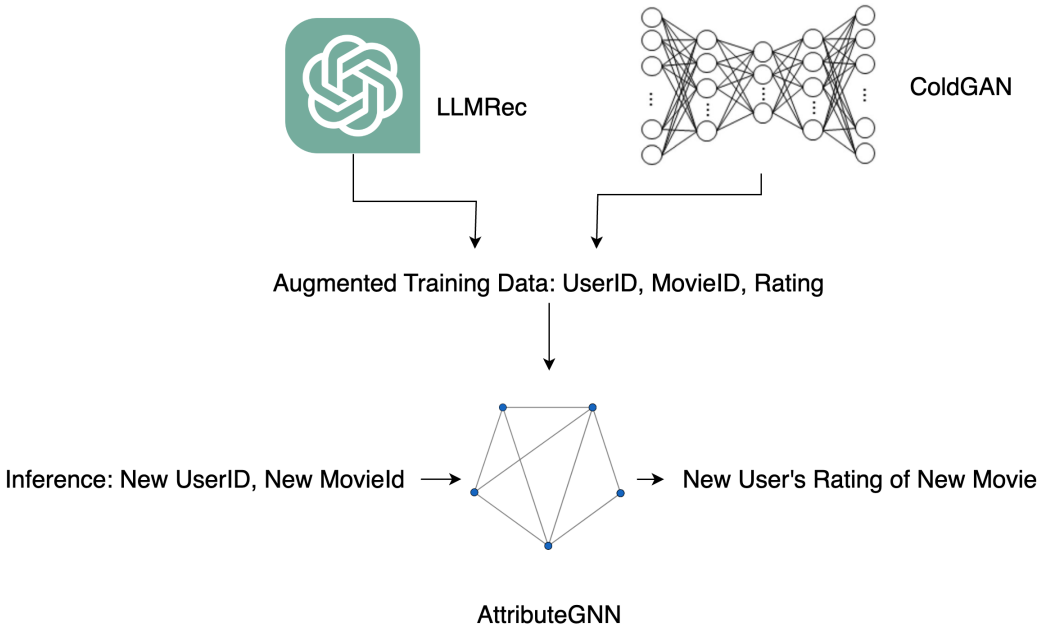

Figure 6: System Architecture of Models

We will treat the AttributeGNN model's RMSE as the baseline when comparing results since the default AttributeGNN model does not use any augmented data for training. From Table 1 we can see that the AttributeGNN with ColdGAN augmented data performed marginally better than the AttributeGNN paper. We can also observe that the AttributeGNN with LLMRec augmented data performed slightly worse than the AttributeGNN paper. Unfortunately, we were not able to reproduce the same RMSE score as the AttributeGNN paper with our AttributeGNN implementation resulting in a marginally higher RMSE score. If we use the AGNN-ColdGAN as a baseline we can see that the AGNN paper has a .068% higher RMSE score, our AGNN implementation has a .46% higher RMSE score, AGNN-LLMRec has a 0.66% higher RMSE score.

4.3 Discussion

While using LLMs has many benefits, it is extremely costly due to the huge inference time and the size of these models. The original implementation bags items and users instead of prompting the LLM one by one for each inference step. Although this approach makes the inference step faster and more efficient, it neglects the weakness of LLMs in outputting long ranked lists and performing multiple logic steps at once. The LLMRec paper (Wei et al. [5]) addresses this issue by pruning the data signals that have a high loss, attributing them to a baseless output made by the model. However, our experiments showed that this issue persists.

As shown in Figure 7 the AttributeGNN with ColdGAN (AGNN-ColdGAN) has the best training loss. This is possibly because we train ColdGAN's GAN on the same training data that we're training AttributeGNN on. This leads to "double-dipping" (even with unseen user-movie pairs as they reflect the data) in the training set resulting in the ColdGAN having low training loss. This also seems to suggest that having the AttributeGNN train on the users' distribution generated from the ColdGAN is helpful to the overall RMSE because it can more easily converge to the data with these "pre-trained" training samples.

In comparison, the AGNN-LLMRec training loss in Figure 7 performs worse than AGNN-ColdGAN and the base AGNN. Since LLMs are trained on lots of real-world data it's possible they would perform better in a generalized setting (such as different datasets in different domains) compared to ColdGAN. For example, the AGNN-LLMRec might perform better on the Yelp dataset, where the rating distribution is more biased towards 5 and 1 star ratings, since the ColdGAN approach might

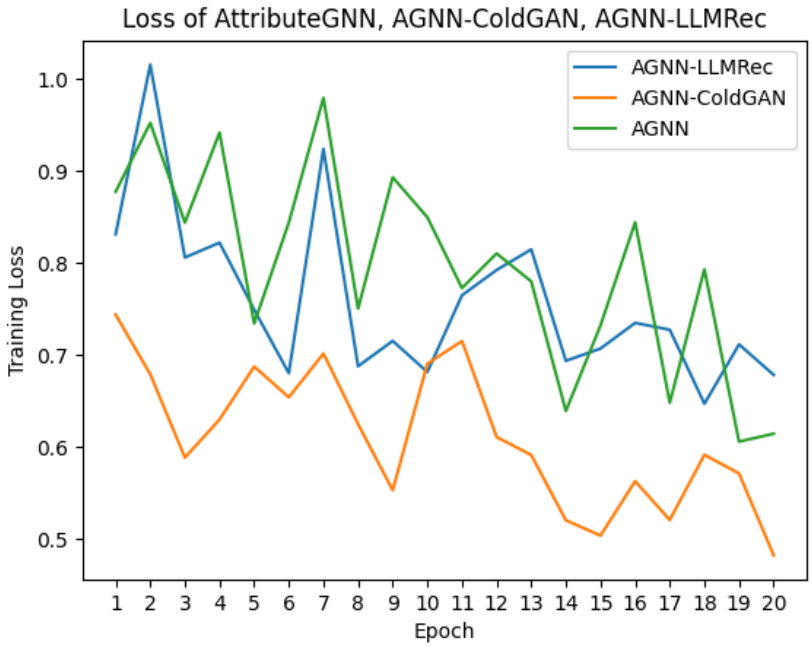

Figure 7: Training loss for our models

not be able to learn the rating distribution given that an item isn't as popular or wasn't rated earlier in comparison to other items.

5 Conclusion

This project focuses on advanced augmentation techniques and their impact on improving the quality of recommendations with cold items and users in a system. Three models are combined and compared with each other. Our experiments show that our combination of AGNN-ColdGAN outperforms AGNN-LLMRec and AGNN-only methods. This suggests that GANs rather than LLMs may be a better tool for generating augmented datasets from sparse data. Additionally, these results indicate that using data augmentation can increase the quality of recommendations in recommender systems when combined with GNNs.

This project can be expanded in the future by testing the aforementioned model combinations on a bigger and wider variety of datasets. This would expose our model to a larger range of side information for different items and users. Additionally, future researchers should consider using bigger models than just the 7 billion parameter Llama model. It is likely that as we increase the size of our LLM, the quality of its outputs increase. The limited size of our model presumably negatively influenced our results.

References

[1] Tieyun Qian, Yile Liang, Qing Li, and Hui Xiong. Attribute graph neural networks for strict cold start recommendation. *IEEE Transactions on Knowledge and Data Engineering*, 34(8): 3597–3610, 2020.

[2] Jiani Zhang, Xingjian Shi, Shenglin Zhao, and Irwin King. Star-gcn: Stacked and reconstructed graph convolutional networks for recommender systems. *arXiv preprint arXiv:1905.13129*, 2019.

[3] Liang Hu, Songlei Jian, Longbing Cao, Zhiping Gu, Qingkui Chen, and Artak Amirbekyan. Hers: Modeling influential contexts with heterogeneous relations for sparse and cold-start

recommendation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3830–3837, 2019.

[4] Chien Chin Chen, Po-Lin Lai, and Chih-Yun Chen. Coldgan: an effective cold-start recommendation system for new users based on generative adversarial networks. *Applied Intelligence*, 53 (7):8302–8317, 2023.

[5] Wei Wei, Xubin Ren, Jiabin Tang, Qinyong Wang, Lixin Su, Suqi Cheng, Junfeng Wang, Dawei Yin, and Chao Huang. Llmrec: Large language models with graph augmentation for recommendation. In *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pages 806–815, 2024.

[6] F Maxwell Harper and Joseph A Konstan. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19, 2015.

[7] Nathan Inkawhich. DCGAN Tutorial - PyTorch Tutorial. `https://pytorch.org/tutorials/beginner/dcgan_faces_tutorial.html`, 2024.

[8] Yelp. Yelp Open Dataset. `https://www.yelp.com/dataset`. Accessed Febuary 12, 2024.

[9] Ni, Jianmo. Amazon Review Data (2018). `https://cseweb.ucsd.edu/~jmcauley/datasets/amazon_v2/`. Accessed Febuary 12, 2024.