Report for Enhancing Large Language Model based Recommendation Systems

Dustin Yan
Department of Computer Science
San Jose State University
San Jose
dustin.yan@sjsu.edu

***Abstract*—Large Language Models (LLMs) have become overwhelmingly popular. Their use and research outside of Natural Language Processing (NLP) have demonstrated amazing potential on many other applications. However LLMs face an issue in other domains with more than just text-data. In domains where there are more than just text-data, they can perform well, but can definitely improve results if we can deal with the other kinds of data. If we combine other ways to deal with non text-data, we can achieve better results. In this paper, I deal with LLM based Recommendation Systems and the possibility of enhancing a LLM based Recommendation System by dealing with non text-data.**

***Index Terms*—enhance, LLM, personalized, recommendation, Recommendation Systems,**

I. Introduction

I aim to enhance an existing LLM based Recommendation System on e-commerce by dealing with non text-data. Many e-commerce stores have more than just textual data. Often times, products displayed on these stores tend to also have images of the product they are trying to sell. With the LLM dealing with textual data, I plan to implement a Vision Model to deal with the image-data. My goal is to improve an already good Recommendation System by applying a Vision Model on image-data from e-commerce stores. With this approach, I will be using a majority of data from e-commerce stores and show that the use of a Vision Model with a LLM will enhance personalized recommendations. I will be using real world Amazon data to accomplish this goal.

To show if our method is effective in enhancing personalized recommendations, I will be comparing my results to [1] with different hyper parameters. I will perform a baseline with [1] using adjusted hyper parameters and then using the same hyper parameters, I will perform a test on my new method. This will show if our method is effective in enhancing personalized recommendations.

II. Problem Statement

Given all the data on Amazon, such as user descriptions, reviews, images, and item descriptions, where items descriptions could be the price, brand, etc, is it possible to make a more personalized recommendation system? Not just popular items, but items that we will have a high chance of purchasing. With all the data from Amazon, we utilize a LLM to interact with text-data between users and items and a Vision Model to analyze objects, resulting in a generation of items to recommend to the user.

III. Dataset

Before diving deeper, I want to first discuss the dataset used for this paper. I will be using Amazon Review Data (2018), which is a dataset that is sourced from [2]. This dataset has metadata of the item, as well as user reviews of the item.

A. Pre-processing

To use the data, I must make embeddings so our model can utilize the data. All text-data will be embedded and used to train models. Using [1], I will have the necessary written code to perform this task.

For images, I will use a pre-trained Vision model, such as Yolov8 [3]. There will be no need to perform any pre-processing for this step.

Pre-processed text-data was already provided from [1].

IV. Baseline

For the baseline, I will be using a subset of the Amazon dataset. I will be using the Luxury item subset to perform my experiment. This subset was chosen to lighten the amount of data that I have to process.

TABLE I
Dataset

Dataset	Dataset Statistics			
	User	***Item***	***Interaction***	***Content***
AM-Luxury	2,382	1,047	21,911	15,834

Subset of dataset used for the baseline.

A. Baseline Setup

The baseline setup is based on the code provided from [1]. With the existing code base, I changed the hyper parameters to accommodate my environment. To pre-train the LLM, I used 1 epoch. Training on user-item interaction, I used 100 rounds. Fine-tuning, I used 5 epochs. Originally [1] used 10 epochs to pre-train the LLM, 100 rounds on user-item interactions, and 50 epochs on fine-tuning. The experiments were reduced heavily to get results in a short amount of time.

B. Baseline Results

Results using my hyper parameters were not far off the original. I used 2 metrics to evaluate the effectiveness, Recall@20, Recall@40. This indicates items that users are interested in among the 20 or 40 recommended items [1].

Metric	Original	Experiment
Recall@20	.3066	.2611
Recall@40	.3441	.2954

TABLE II
RESULTS OF ORIGINAL PAPER AND MY EXPERIMENT

C. Baseline Discrepancies

From Table II, my results were not far off from the original. The results are different because of the hyper parameters that I changed. The original also ran on Ubuntu 22.04.3 LTS OS, Intel(R) Core(TM) i9-13900KF CPU, with the framework of Python 3.11.5 and PyTorch 2.0.1. I ran on AMD Ryzen 9 5900X 12-Core Processor, with the framework of Python 3.11.5 and PyTorch 2.0.1. There was only a difference of CPUs for the environment. Performing pre-training of the LLM took roughly 6 hours without GPU, and 3 hours with a single NVIDIA RTX 3080. Fine-tuning took approximately 30 minutes per epoch.

With the ideal environment and better computing power, running the original hyper parameters would be possible for me. Using my a personal setup made it almost impossible to run on the original hyper parameters.

V. NEW APPROACH

To enhance personalized recommendations, I propose adding a Vision Model to analyze images from Amazon items. This idea comes from the fact that recommendation systems perform recommendations on one type of data. Recommendation Systems are good right now, but we can definitely improve them. If Recommendation Systems use more than one type of data, this would improve the effectiveness of recommendations.

But there is a problem with this solution. How can we utilize both LLM and Vision Model? This is a problem that is hard to deal with. They both provide different results, so how can we connect them? The original architecture uses text-data embeddings to generate an item for recommendation. A Vision Model, Yolov8 produces an object from an image. What connection can we use between these two models to enhance the system?

From the dataset, we are able to see the relation of users and items on Amazon. With the relationship, I could utilize the Vision model to detect what the object is of the item and its relationship items. This would give me a list of potential items a user buys, visits, likes, or dislikes. With the set of items categorized by buys, visits, etc, distinguish which item has a higher chance for a user to think the item is relevant when given a recommendation. I will also use the Vision model right before recommending an item. At the end of the original architecture, recommendations are added to a target list of recommendations. Within this list, I will run the Vision model on these items to see if the object detected on the item image is in our set of categorized items. If the object falls under a positive annotation such has buys, we can assume that this item will likely be personalized to the user. If the object falls under a negative annotation such as dislikes, we can assume this item won't be personalized. With these added filtering methods, we can will see an increase of personalized recommendations that are relevant to the user.

A. Problems with New Approach

This approach seems too good to be true. Why wouldn't all big companies use this type of Recommendation System to improve recommendations, resulting in more purchases on their e-commerce store. Who wouldn't want to increase profits? Well, training of these big models takes an enormous amount of time and resources. The run time of such a system would also take an immense amount of time. Also implementing such a system is hard. Most Recommendation Systems perform on one data type, and that is usually enough information to create good recommendations. Dealing with two or more data types increases the difficulty of the system. Some problems I see:

- Training and Fine-tuning Vision model to perform on our data
- Runtime of the System - Will need to parse through relationships of users and items. Grab multiple images from the product. Detect object. Get target list from generative LLM. Perform object detection on images from target list.

B. Implementation of the New Approach

Because this system is hard to develop and has many challenges. I was not able to fully implement this system with the allotted amount of time. Instead, I will be discussing the problems I faced when trying to implement this approach. The Vision Model, Yolov8 was easy to get working, but adding it to certain places of the code base was difficult. I couldn't figure out how to completely implement my idea of creating sets of items with positive and negative annotations. After detecting the object, I could save everything into a dictionary and export the dictionary into a JSON file that I can reference at a later time. That was problem 1. The next problem was accessing the target list. The target list was hard for me to adjust. I'm still having trouble accessing the data for the target list. It seems that I would need to access and change existing library code to accomplish this.

VI. INSIGHTS OF NEW APPROACH

Since I was not able to create this new system, I can only suggest possibilities of the new system. Without the results of experiments using the Vision Models, I cannot say for certain if the system will enhance the previous system.

The use of more data should increase related personalized recommendations, but incorporated in the wrong areas, we could see recommendations of the wrong items. In our case,

we will be using the output of the Vision Model to add an extra filtering step. This filtering step should reduce the amount of items added to our target list. This target list will contain a more refined list that are related personalized items.

VII. Future Development

Since I haven't finished developing the code base for this system. Future development for this system will be to complete this system and produce results, so that we can see if the system really enhances personalized recommendations.

There is also a need for future development on multi data type recommendation systems. Its very useful to work on all the data provided from e-commerce stores. There isn't enough systems that utilize Vision Models. Almost every product has an image associated with it. If people can analyze the images, and use it for their recommendation system, we can see better recommendations.

VIII. Conclusion

In recent studies, we have seen generative LLMs show great results. Adding them to a Recommendation System and showing how impressive its results are proves that there is a future with generative LLMs with Recommendation Systems. These systems require more computing power than I have, but produce results that are as good or even better than your modern Recommendation System.

Recommendation Systems are important when we talk about e-commerce. They bring change in many individual shopping lives. They allow customers to explore different, but similar products to what they have bought or viewed in the past. Though the recommendations might not be the best, they lead us to think, maybe I should give this product a chance. Without recommendation systems, customers wouldn't be able to find products that they didn't know they had an interest in.

Unfortunately, I wasn't able to show that adding a Vision Model into a LLM based Recommendation System would enhance personalized recommendations. This report discusses a possible way to enhance a LLM based Recommendation system using Vision Models.There are still more places to explore with LLM based Recommendation Systems.

References

[1] X. Wang, L. Wu, L. Hong, H. Liu, and Y. Fu, LLM-Enhanced User-Item Interactions: Leveraging Edge Information for Optimized Recommendations, https://arxiv.org/pdf/2402.09617.

[2] J. Ni, "Amazon review data," nijianmo.github.io, 2018. https://nijianmo.github.io/amazon/index.html

[3] G. Jocher, A. Chaurasia, and J. Qiu, "YOLOv8 by Ultralytics," GitHub, Jan. 01, 2023. https://github.com/ultralytics/ultralytics