IDentityRec: Revolutionizing Sequential Recommendations by Integrating Collaborative Embeddings with LLM Intelligence

Srijan Chakravorty

Master of Science

Data Science

School of Informatics

University of Edinburgh

2024

Abstract

The thesis investigates the advantages of combining the strengths of sequential pattern learning with robust contextual understanding of large language models (LLMs) to outperform state-of-the-art foundational recommendation baselines. It introduces **IDentityRec**, a novel recommendation architecture designed to integrate the collaborative information from item IDs with the powerful semantic modelling capabilities of an LLM to address pertinent challenges in the space of sequential recommender systems.

Leveraging pre-trained ID embeddings and textual prompts from an instruction tuned large language model, IDentityRec harnesses a unified collaborative embedding space to ensure that the model generates comprehensive, nuanced and personalised ranking lists of recommendations in a single forward iteration. It involves efficiently training a minimal number of modular parameters to fine-tune the model on the downstream recommendation task, without incurring any additional overhead in terms of inference time and storage space. The sophisticated architectural design of IDentityRec ensures that the predictions remain within relevant candidate lists, thus mitigating the issue of out-of-vocabulary recommendations.

Through comprehensive experiments, it is observed the model demonstrates quantifiable improvements in performance to the tune of 35% when assessed on standard evaluation metrics across benchmark datasets. Empirical results highlight that IDentityRec fares exceedingly well in cross-domain generalisation tasks, showing an impressive 43% gain when the knowledge learned from one domain was adapted to another, without retraining it from scratch. The model achieves its objective of generating contextually relevant recommendations in data sparse scenarios, attaining a competitive nDCG score in situations where a limited number of interactions were available. The effectiveness of the architecture allows the robustness of IDentityRec to translate across domains, successfully circumventing the cold start problem in unseen datasets, which is a major challenge for traditional ID based recommendation systems.

The ability to generate optimal recommendations within a fixed vocabulary in a single forward iteration positions IDentityRec as a robust solution for deployment in modern recommender systems where performance and user experience are paramount. The innovative architecture addresses the limitations of traditional LLM-based approaches in terms of efficiency, scalability and practicality, thus paving the way for more personalised and context-aware recommendations on e-commerce platforms.

Research Ethics Approval

This project was planned in accordance with the Informatics Research Ethics policy. It did not involve any aspects that required approval from the Informatics Research Ethics committee.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Srijan Chakravorty*)

Acknowledgements

I express my deep gratitude to my supervisors, Mr. Timos Korres, Dr. Phong Le and Dr. Vladimir Eremichev from Amazon Development Center, Scotland, for their indispensable mentorship, unwavering support and insightful advice throughout the six months since the project was allocated to me. Their expertise, guidance and valuable insights have been essential in helping me develop this research project. My in-person meetings with the aforementioned Applied Scientists at the Amazon office in Edinburgh not only enriched my technical skills but also helped me streamline my thoughts and communicate my developments in a lucid manner.

I also want to express my thanks to the University of Edinburgh, School of Informatics for providing the essential tools and computing setup that facilitated the accomplishment of this research. In particular, I want to thank Dr. Chris Williams for his utmost efforts in ensuring that I could access the best tools at the earliest in order to complete this project. I would also acknowledge the contribution of Dr. Michael Herrman, who was my tutor while drafting the project proposal, for providing me a roadmap that helped me channelise the research in the correct direction.

Last but not the least, I express my heartfelt thanks to my co-researchers, friends and my family who have motivated me along the way by encouraging me at various steps of this journey. This thesis would not have been possible without the collective support of all these individuals. Thank you.

Table of Contents

Chapter 1

Introduction

1.1 Motivation

Sequential recommender systems [46] are software tools that suggest which item a user is likely to interact with next, given a recorded sequence of items the user has interacted with previously, in the form of ratings, clicks, or purchases [46]. They are ubiquitous in modern businesses, allowing companies to discern user preferences and offer tailor-made recommendations, resulting in improved customer experiences and boosting company revenue. Online retail giant, Amazon, directly credited their recommender system accounting for 35% of all sales, driving the company's turnover to touch a billion dollars [22].

Recently, with the emergence of Large Language Models, there has been a marked increase in research aimed at utilizing LLMs for recommendation tasks [27]. Unlike traditional systems that follow the two-tower architecture [10], LLM-based approaches [3, 14, 23, 64] frame the recommendation task as an open-domain natural language generation problem that requires generating a textual description or list of recommended items with respect to an input prompt. However, LLMs, which are optimised for text processing, struggle to extract semantic information from discrete, alphanumeric identifiers like user and item IDs that do not convey linguistic meaning. Without proper ID modelling, LLM-based strategies lack the ability to interpret collaborative signals and fail to harness the relationship between users and items established through ID interactions, leading to a decline in recommendation quality.

Owing to the nature of auto-regressive language models which generate text sequentially, existing recommendation methods that use LLMs typically generate only one recommendation result at a time [6]. Therefore, large-scale recommendation systems

that deal with millions of items and users require multiple generation attempts to obtain optimal recommendation results. This further exacerbates the latency and concurrency issues, making it slow, inefficient and difficult to scale. Such systems fail to efficiently handle large volumes of user requests simultaneously within acceptable time frames. A longer inference time due to multiple recommendation generations is a bottleneck that could potentially translate into higher infrastructure and operational costs for businesses.

Existing approaches [50, 25] predict the next item by evaluating the likelihood over all possible items in the vocabulary, including those which are irrelevant to the user's current context and outside his interest scope. Such a problem definition leads to irrelevant and out-of-range recommendations which do not match the user's demonstrated preferences [23]. This may be detrimental to customer engagement and reduce conversion rates, directly impacting the revenue for businesses relying on recommendation-driven sales. Thus, it is imperative to limit the candidate set based on the user's current interaction history in order to generate relevant and personalised recommendations.

For ID-based collaborative filtering approaches [29], it becomes challenging to accurately predict user preferences when most users interact with only a small subset of items. Presenting newly added items as recommendations when user-item interactions are sparse, allows for increased item visibility to relevant users. Meanwhile, the ability to provide high-quality recommendations for new users with little to no prior interaction history, allows for increased customer purchases and retention [1]. So, generating high quality recommendations in the presence of limited interaction data, while alleviating the issues of data sparsity and cold-start forms a motivation of this thesis. The challenge of enabling models to effectively apply learnt knowledge across unseen domains with different data distributions also presents a compelling avenue for research.

1.2 Contribution

The dissertation delves into the discussion of combining the strengths of sequential pattern learning with robust contextual understanding to outperform state-of-the-art recommendation methods on publicly available datasets. We present an innovative solution, IDentityRec, which bridges the gap between traditional ID-based collaborative filtering methods and the advanced large language model based recommendation architectures. Through the proposed architecture, we pioneer an effective strategy to address the unique challenge of integrating IDs in LLMs for efficient and controllable generation of recommendations.

IDentityRec is a novel recommendation framework which ensures that the model retains both the natural language understanding and collaborative filtering capabilities, addressing the limitations of open-domain generation. It identifies the most relevant items which are aligned with the user's current interaction history and presents them to the user, resulting in highly personalised recommendations. By training a lightweight linear projection layer, the architecture retains the core objective of modelling joint probability distributions over candidate items, enabling the efficient computation of prediction scores across a well defined item vocabulary in a single forward iteration. Filtering and refining the candidate set of items by incorporating additional contextual information via LLMs before generating the final recommendation helps mitigate the issue of out-of-vocabulary recommendations. The robust, domain-agnostic architecture of IDentityRec generalizes well across different types of datasets due to its pre-training on a diverse set of tasks.

The effective design alleviates the issues in traditional methods to ensure that the model is extensible and can handle large-scale recommendation tasks efficiently, meeting the performance requirements for deployment in production-level e-commerce environments. To validate the robustness of the model in sparse data scenarios, empirical evidence is gathered by conducting comprehensive experiments across prominent sequential recommendation datasets. The results demonstrate the superiority and effectiveness of the proposed model in terms of performance and scalability, while an in-depth analysis underscores its efficiency and extensibility in real-world applications.

1.3 Thesis Structure

This dissertation consists of seven chapters. Chapter 2 explains the necessary background literature on traditional and LLM-based recommender systems which are most relevant to this dissertation. Chapter 3 discusses the methodology adopted for the project, the model's architecture and its novelty, while Chapter 4 provides an overview of the experiments undertaken to substantiate the hypotheses. The results from these experiments are presented and analysed in Chapter 5. Chapter 6 discusses these results and argues that they are evidence for the conclusions outlined later in Chapter 7. It also mentions the limitations of the proposed model and potential future work.

Chapter 2

Background

In this chapter, we conduct a literature review of the recent advancements in sequential recommendation systems. We begin by examining their evolution from traditional approaches to attention-based architectures. Then, we proceed to explore the integration of large language models in recommendation systems. Further, the discussion elucidates the prevalent challenge posed by integrating IDs into modern recommendation systems.

2.1 Traditional Recommendation Systems

Traditional recommender systems have been foundational in personalising user experiences, shaping the e-commerce landscape by anticipating user preferences with an almost prescient accuracy. From the days of the Netflix Prize challenge [5], recommendation systems have evolved through groundbreaking innovations. Two primary paradigms form the bedrock of traditional recommender systems. Collaborative filtering [17, 28, 47] recommends items based on the preferences of similar users, whereas content-based filtering [49, 39] utilizes the intrinsic attributes of items to recommend items that are similar to those a user had previously interacted with. Hybrid approaches [40, 4] have been designed in literature to elegantly unify these two paradigms.

2.2 Sequential Recommendation Systems

Sequential recommendation plays a crucial role in personalised recommendation systems, focusing on understanding dynamically evolving user preferences by modelling the latent temporal patterns encoded in their historical activity sequences. Marked a significant departure from traditional recommender systems, they have evolved into

sophisticated engines, deftly guiding users through vast product assortments, while working under the principle that the past actions of a user influence his future behaviour.

Preliminary research to address sequential recommendation leveraged Markov Chain models [16, 44], which capitalise on the assumption that the next item a user interacts with depends primarily on the most recent items they have engaged with. Techniques like FPMC [45] combined matrix factorisation with the Markov Chain framework to adeptly model sequential dynamics and personalised preferences. However, the limitations of such models in capturing complex behavioural patterns and long-term dependencies prompted the adoption of more advanced techniques.

With the advent of deep learning, sequential recommendation models pivoted to using deep neural architectures to generate context-aware recommendations. Seminal models like GRU4Rec [19] and Caser [54] emerged as a notable improvement over their predecessors, attempting to solve the sequential recommendation task by employing Gated Recurrent Units and Convolutional Neural Networks to capture the high-order interactions and user behaviour embedded in latent spaces.

In recent years, SASRec [25] has revolutionised the field by capturing both short-term and long-term dependencies in the sequence more effectively by utilising a self-attention mechanism [59], allowing the model to learn complex patterns and critical relationships from the most relevant parts of the sequence while predicting the next item. BERT4Rec [52] further pushed the boundaries by employing the Cloze [56] objective to predict the masked item, thereby modelling bi-directional dependencies by capturing the contextual information from both past and future items in a sequence.

2.3 LLM Based Recommendation Systems

In the ever-evolving tapestry of recommender systems, Large Language Model (LLM)-based recommender systems, with their unparalleled prowess of natural language understanding, have emerged as the frontrunner of personalization, to forge deeper connections between users and content [12, 61]. Early approaches such as U-BERT [41], which treated LLMs primarily as feature extractors, were constrained by their inability to fully exploit the generative capabilities of LLMs, limiting their potential in dynamic recommendation scenarios. However, with the inception of LLMs like GPT [6], T5 [43], and LLaMA [57], next-generation recommender systems harness the inherent knowledge of pre-trained LLMs [51, 62] to comprehend the dynamically evolving user-item relationships and produce context-aware recommendations.

The core premise in such systems is to reshape sequential recommendation as a language modeling task by converting the behavioral sequence into the textual input prompt to directly generate recommendation results [61]. To align LLMs with the domain of generative recommendation, initial research leveraged prompting [13, 53] and in-context learning [11, 34]. However, foundational models which were trained on a specific recommendation task using dedicated data, continued to outperform these methods. Therefore, fine-tuning was explored to adapt LLMs to recommendation.

P5 [14] utilized a prompt-based learning technique, fine-tuning the FLAN-T5 [43], to address five downstream recommendation tasks by framing them as language generation problem. Based on a similar paradigm, InstructRec [64] adapted the FLAN-T5 model to recommendation tasks by employing instruction tuning with a broader range of texts. TALLRec [3] and GenRec [23] pushed the frontiers of generative recommendations further by instruction tuning on the LLaMA model, creating a holistic and deeply personalized recommendation experience.

2.4 Relevant Challenges

Recommendation systems based on the two-tower architecture [10] have long been dominated by the ID-based paradigm, where users and items are represented by unique, continuous ID embeddings denoting their semantic similarity. However, the reliance on IDs alone introduces certain limitations, such as a lack of generalization to unseen items and an inability to capture rich contextual information about them [2]. Existing approaches [3, 14, 23, 64] have predominantly converted the recommendation task into a natural language generation problem to align it with the inherent capabilities of LLMs. Although impressive progress has been achieved, fundamental dichotomies between these domains remain to be addressed.

Item IDs, being abstract identifiers devoid of semantic content, do not align well with the text-centric operations of LLMs, leading to a loss of contextual richness and limiting the model's ability to generate nuanced recommendations [37]. In cold start scenarios involving new users or items, LLMs cannot effectively predict preferences based on discrete IDs alone [32]. The inefficiency in handling sparse data further limits the system's ability to provide personalized recommendations in situations with limited user-item interactions [7, 32]. To compensate for the lack of semantic content, additional computational overhead may be required, increasing the complexity and resource demands of the system [18, 65].

Chapter 3

Methodology

In this section, we provide a holistic view of IDentityRec's architecture, outlining the novel components involved in the design of our proposed model and how they integrate to mitigate the issues plaguing modern recommendation systems.

3.1 Overview

The proposed model uses a pre-trained sequential recommendation system to obtain high-quality ID embeddings for items and effectively capture sequential patterns and collaborative information. A backbone LLaMA model is instruction-tuned to efficiently adapt the LLM for recommendation related tasks. Through Parameter Efficient Fine Tuning techniques like Low Rank Adaptation, only a small subset of parameters in the LLM are trained, while others remain frozen, thus adapting the recommendation task for a given dataset in a computationally efficient manner. By using a softmax in the output layer, the architecture computes a joint probability distribution over all candidate items in a single forward pass, evaluating and scoring all candidates simultaneously, rather than one at a time. In the inference stage, the model utilises an item linear projection that maps the output embeddings from the LLM into a space where a nearest neighbour search can yield personalised recommendations that are within the candidate set. By leveraging efficient embedding, tuning and projection techniques, the proposed model effectively leverages the collaborative information along with the LLM's contextual knowledge to handle large candidate sets. The sophisticated design allows IDentityRec to generate recommendations quickly and scale effectively to meet the demands of contemporary recommender systems. A concise schematic diagram of IDentityRec's architecture is presented in Figure 1 for enhanced clarity and comprehension.

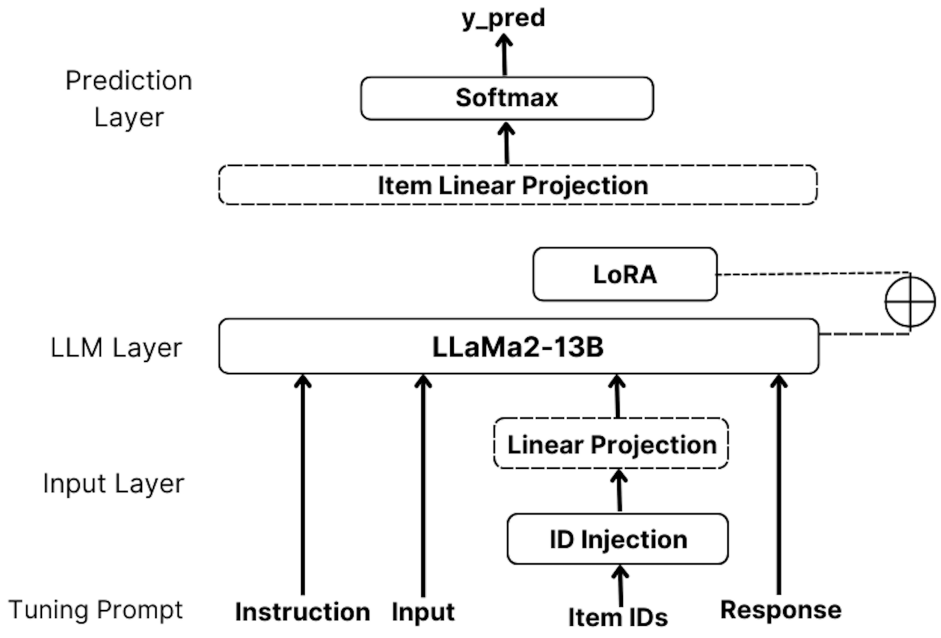

Figure 1: An illustration of the IDentityRec's model training architecture

3.2 Input Processing and Encoding

In a sequential setting, the chronological user-item interaction sequence is represented as $S_u = [i_1^u, i_2^u, \ldots, i_T^u]$, where T is the number of items the user u has interacted with, ordered by the timestamp. During training, at any time step t, the task entails predicting the next item that the user would engage with, depending on the previous $t-1$ items.

We begin by transforming the training sequence into a fixed-length sequences, $S = s_1, s_2, \ldots, s_n$, where $n \in \mathbb{N}$ represents the maximum length of sequences that our model can handle. Since users may have varying numbers of interactions, sequences need to be padded or truncated to a fixed length. Maintaining consistency with [42], the sequences are truncated to include only the 50 most recent interactions. However, if the sequence length is less than the desired length, we repeatedly add a 'PAD' token from the left until its length is n. So the final sequence for each user becomes $S_u = [i_{T-50+1}^u, \ldots, i_T^u]$ for truncated sequences and $S_u = [\text{PAD}, \ldots, \text{PAD}, i_1^u, \ldots, i_T^u]$ for padded sequences.

The processed input sequence is to be fed into a sequential recommendation model for obtaining conceptualised item embeddings. Each item in the sequence, i_j^u, is mapped to a corresponding dense embedding vector $\mathbf{e}_j^u$ of dimensionality d, using an embedding matrix $\mathbf{E} \in \mathbb{R}^{|I| \times d}$, where $|I|$ is the total number of items. A constant zero vector is used as the embedding for the padding item. So, for a sequence $S'_u = \{i_1^u, i_2^u, \ldots, i_T^u\}$

$$\mathbf{E}_u = [\mathbf{e}_{i_1}^u, \mathbf{e}_{i_2}^u, \ldots, \mathbf{e}_{i_L}^u] \quad \text{where} \quad \mathbf{e}_j^u = \mathbf{E}[i_j^u] \quad \text{are the corresponding embeddings} \tag{3.1}$$

3.3 Input Layer

The central premise of this section is to create a unified input representation which ensures that the model benefits from both the collaborative filtering patterns deciphered from IDs and the contextual information present in textual prompts, thus providing a holistic approach to recommendation. To achieve this, a sequential recommendation model like SASRec is pre-trained, the LLM is instruction-tuned and the ID embeddings and textual prompts are projected together as input to the LLM.

3.3.1 Pre-training SASRec for ID Embeddings

As discussed in Section 1.1, Large Language Models are unable to infer the semantic meanings of numeric IDs in the absence of textual features. However, the collaborative information contained in IDs is very effective and crucial in personalised recommender systems. To overcome this limitation, we propose obtaining ID embeddings by directly extracting them from a pre-trained sequential recommender model without modification.

In this thesis, SASRec (Self-Attentive Sequential Recommendation) [25] is chosen as the desired model from which we extract ID embeddings, owing to its superior generalisation ability and effectiveness in sequential recommendation. SASRec, which leverages self-attention mechanisms and transformer blocks to model user-item interaction sequences, is pre-trained on the Amazon Reviews dataset. The item embedding layer in SASRec learns embeddings for each item ID, while the self-attention blocks capture the temporal dependencies and sequential patterns between users and items.

Since the transformer-based SASRec model is order-invariant, positional encodings are added to the item embeddings to capture the sequential order of interactions. Assuming that the positional encoding matrix is $\mathbf{P} \in \mathbb{R}^{L \times d}$ and $\mathbf{p}_j$ denotes the positional encoding for position $\mathbf{j}$, the final input representation to the self-attention layer is the sum of the item embedding and its corresponding positional embedding, given by:

$$\mathbf{X}_u = \mathbf{E}_u + \mathbf{P} \qquad \text{or} \qquad \mathbf{x}_u = \mathbf{e}_j^u + \mathbf{p}_j \tag{3.2}$$

During the training loop, for each batch, the encoded user interaction sequences are fed into the SASRec and the self-attention is computed over each pair of item embeddings in the sequence to capture the dependencies between different items. These scores determine the contribution of each item in the sequence towards the prediction

of the next item. The self-attention score α_{jk} between items i_j^u and i_k^u:

$$\mathrm{A}(Q,K,V) = \mathrm{softmax}\left(\frac{QK^T}{\sqrt{d}}\right)V \quad \text{or} \quad \alpha_{jk} = \mathrm{softmax}\left(\frac{(\mathbf{e}_j^u + \mathbf{p}_j)\mathbf{W}_Q \cdot (\mathbf{e}_k^u + \mathbf{p}_k)\mathbf{W}_K^\top}{\sqrt{d}}\right)V \tag{3.3}$$

where $\mathbf{W}_Q, \mathbf{W}_K, \mathbf{W}_V \in \mathbb{R}^{d\times d}$ are learnable weight matrices for query, key and value transformations respectively and d is the dimensionality of embeddings.

Then, a point-wise feed-forward network (FFN) is applied independently to each position in the sequence. The FFN has two linear transformations with a ReLU activation in between, for adding non-linearity to the model. Mathematically, it is computed as:

$$\mathrm{FFN}(x) = \mathrm{ReLU}(xW_1 + b_1)W_2 + b_2 \tag{3.4}$$

where W_1 and W_2 are weight matrices, b_1 and b_2 are bias vectors and $\mathrm{ReLU}(x)$ is the Rectified Linear Unit activation function applied to the input x.

As we stack self-attention blocks, the network grows deeper. To circumvent the vanishing gradient problem and improve training stability, we employ dropout and layer normalisation after each sub-layer (self-attention and feed-forward). While dropout helps mitigate overfitting, Layer Norm normalises the inputs and improve training convergence. Residual connections are added around the self-attention and feed-forward layers to propagate the low-layer features to higher layers. Self-attention layer g is:

$$g(x) = x + \mathrm{Dropout}(g(\mathrm{LayerNorm}(x))) \quad \mathrm{LayerNorm}(x) = \alpha \cdot \frac{x-\mu}{\sqrt{\sigma^2+\varepsilon}} + \beta \tag{3.5}$$

where $\cdot$ is an element-wise dot product, μ and σ are mean and variance of x and α and β are scaling factor and bias. To ensure that predictions for time step t only depend on previous time steps, a causal mask is used to set the weights to $-\infty$ for future time

$$\mathrm{MaskedAttention}(Q,K,V) = \mathrm{softmax}\left(\frac{QK^T + M}{\sqrt{d_k}}\right)V \tag{3.6}$$

The predicted probability for the next item in the sequence is generated using the output layer using a softmax function over the dot product between the hidden state $\mathbf{h}_t^u$ and the embedding of all possible items:

$$\hat{y}_{t+1}^u = \mathrm{softmax}(\mathbf{h}_t^u \cdot \mathbf{E}^\top) \tag{3.7}$$

where $\mathbf{E} \in \mathbb{R}^{|I|\times d}$ is the embedding matrix containing embeddings for all items in the catalog and $\hat{y}_{t+1}^u$ is a vector of predicted probabilities over all items. The learning

objective is to minimise the cross-entropy loss between the predicted probabilities $\hat{y}^u_{t+1}$ and the true next item i^u_{t+1}. Substituting the probability with the softmax output:

$$\mathcal{L} = -\sum_{u \in \mathcal{U}} \sum_{t=1}^{T_u - 1} \log \hat{y}^u_{t+1}[i^u_{t+1}] \tag{3.8}$$

where $\hat{y}^u_{t+1}[i^u_{t+1}]$ is the predicted probability of the true next item i^u_{t+1}.

During training, the loss $\mathcal{L}$ is backpropagated, while the model parameters are updated until convergence using the Adam optimiser. The hyper-parameters such as learning rate, embedding size and number of attention heads are adjusted based on validation performance. These details are covered in Section 4.3.2. Once the loss on the validation set has stabilised and SASRec has been pre-trained to optimal performance, the attention-weighted sum of values generates the output embeddings for each ID:

$$\mathbf{h}^u_j = \sum_{k=1}^{L} \alpha_{jk} \cdot (\mathbf{e}^u_k + \mathbf{p}_k)\mathbf{W}_V \tag{3.9}$$

The model generates hidden representations (or embeddings) for each item in the sequence after passing through a stack of self-attention layers. For each item i^u_j in the sequence S_u, the final hidden embedding is denoted as $\mathbf{h}^u_j$. After training SASRec, the item embeddings can be extracted from the final hidden state using the item embedding matrix $\mathbf{E}'$ which contains the learned embeddings for all items in the item set, as:

$$\mathbf{E}'[i^u_j] = \mathbf{h}^u_j \tag{3.10}$$

The item embedding matrix $\mathbf{E}' \in \mathbb{R}^{N \times d_s}$ contains the embeddings of items in a high-dimensional space, where N is the number of unique items and $N \times d_s$ represents the number of dimensions in which each item is represented. For a specific item i, its embedding can be found in the i-th row of the matrix $\mathbf{E}'$. These item embeddings which are extracted from the item embedding layer of the trained SASRec capture the sequential dependencies and behavioural patterns learned during training, which are then used in the ID injection explained in Section 3.3.4 to enhance the recommendations.

3.3.2 Instruction Tuning the LLM Backbone

The extracted SASRec item ID embeddings encode domain-specific knowledge about user-item interactions that are relevant to recommendation tasks. LLMs, however, are not inherently equipped to handle specific tasks such as recommendation based on structured item embeddings. Instruction tuning [60] is a crucial step in the pipeline

that incorporates the collaborative contextual information from the embeddings into LLM's decision-making process and aligns it with the specific objective of providing contextually relevant recommendations based on historical interactions.

This dissertation follows the Stanford Alpaca template [55] which is designed to provide a structured way to fine-tune a language model using a series of instruction-following examples. These instructions include a clear task description, followed by the relevant input data and the expected output. Using the Alpaca template, instruction-following examples guide the LLM on how to use the provided embeddings. In our context, the instruction part of the template describes the recommendation task, while the input is a sequence of item IDs representing the user's interaction history. The output is a list of recommended items based on user preferences or past behavior. An example of the Alpaca style prompt is illustrated in Table 1.

Instruction Input	
### Instruction:	Given the user's purchase history, predict the next possible item to be purchased.
### Input:	User has interacted with items 32, 198 and 76
Instruction Output	
### Response:	Based on the user's history, item 615 is recommended

Table 1: An instance of the Alpaca template for instruction tuning

The LLaMA [57] has already been trained on a large corpus of text and has a good understanding of language. We implement the Platypus setting [30] by focusing on high-quality, diverse instruction-response pairs. This involves selecting examples that are representative of the types of recommendation tasks the model would be expected to perform, ensuring the instructions are clear and unambiguous and providing responses that are accurate and contextually appropriate. The Platypus setting allows us to combine model ensembles, diverse and augmented data, parameter efficiency, and iterative self-instruction paradigms to fine-tune LLaMA, thus improving its performance on instruction-following tasks while optimizing for resource efficiency and robustness.

We train the LLM iteratively with each example, helping the model improve its ability to understand and generate outputs based on the item ID embeddings. Then we fine-tune it on a dataset of instruction-response pairs to enhance its ability to follow prompts effectively. Supervised fine-tuning minimised the difference between

the model's outputs and the correct responses, refining its ability to predict accurate answers. To further improve the model, a "self-instruct" paradigm was employed, where the model generated additional training data iteratively, enhancing its capacity to handle complex instructions. The fine-tuning leveraged internal attention mechanisms, activation functions like GELU and layer normalisation to ensure stable and efficient training. AdamW [35] was used to optimise the model's parameters. At inference time, the model generates responses to new prompts by associating the provided embeddings with the appropriate textual responses. In this thesis, instruction tuning is conducted for one epoch using the Alpaca template using the auto-regressive objective:

$$\max_{\Phi} \sum_{(x,y)\in\mathcal{Z}} \sum_{t=1}^{|y|} \log\left(\mathbf{P}_{\Phi}(\mathbf{y_t} \mid \mathbf{x}, \mathbf{y}_{<t})\right) \tag{3.11}$$

where x and y represent the textual instruction input and corresponding output responses in the self-instruct data respectively. y_t is the t-th token, $y_{<t}$ represents the tokens preceeding y_t, Φ is the original parameters of the LLM and $\mathcal{Z}$ is the training set.

The instruction-tuned LLM generates responses based on the learned patterns from the fine-tuning process and is ready to be integrated into the recommendation system.

3.3.3 Item Representation

The textual prompts are crucial for leveraging the inherent knowledge within large language models. For an item i and its corresponding textual prompt txt_i, the tokens are of the form $\langle \mathbf{emb}_t^i \rangle = \mathbf{LLM\_Tok}(txt_i)$, where the function LLM_Tok() represents the word embedding layer of an LLM. The pre-trained SASRec ID embeddings for an item i can be expressed as $\mathbf{e_s^i} = \mathbf{SR\_Emb}(txt_i;\Theta_e)$, where $e_s^i \in \mathbb{R}^d$ is the item represented i in d-dimensions and SR_Emb() is the function that extracts the embedding from SASRec.

Unlike the textual prompts that are easily understood by LLMs, the item ID representations may not align well when integrated into prompts. To circumvent this bottleneck, we map SASRec's ID-based item representation, e_i^s, into the language space of LLMs using a trainable projector. This projection results in the creation of a behavioral token $\langle \mathbf{emb}_s^i \rangle = \mathbf{Proj}\ (\ \mathbf{e_s^i}\ ;\Theta_p)$, where Θ_p are the parameters of the projection. The textual and behavioural tokens are concatenated to obtain a holistic definition of an item i:

$$\langle \mathbf{emb_c^i} \rangle = \mathbf{Concat}(\langle \mathbf{emb_t^i} \rangle, \langle \mathbf{emb_s^i} \rangle) \tag{3.12}$$

Therefore, the LLM can comprehend an additional modality encapsulated in item IDs along with textual data from prompts to improve sequential recommendation.

3.3.4 Linear Projection: Creating the ID Injection

The ID embeddings extracted from the SASRec have lower dimensionality ($d_s = 64$) as compared to the word embeddings in LLMs ($d_k = 5120$ in LLaMA2-13B). However, to create a unified input, each low dimensional item ID must be treated similar to a high dimensional word in the LLM. To address this, the IDs are projected using a specialised ID injection component, which is a linear projection represented by a trainable weight matrix. It converts the item IDs into the same dimensionality as the word embeddings by multiplying the pre-trained ID embeddings with a learnable weight matrix $\mathbf{W_{Input}} \in \mathbb{R}^{d_s \times d_k}$. Such a representation preserves the original positions of the ID and word embeddings in the sequence, ensuring the contextual and sequential integrity of the input and maintaining the temporal order.

A standard approach is followed to train the linear projection layer. For each training sample, the input is prepared by concatenating the projected ID embeddings and the textual prompt. The linear projection layer is initialised randomly and the parameters of the LLM are frozen to prevent them from being updated during training. During the forward propagation phase, the concatenated embeddings are passed through the LLM to compute the output predictions using a softmax layer. The optimisation objective is to minimise the cross-entropy loss between the predictions and the ground truth. The loss is backpropagated to update only the weights of the linear projection layer.

This approach of focusing the training efforts on adapting the ID embeddings to match the LLM's input format minimises the risk of disrupting the learned knowledge in both the LLM and the recommendation model. Only the linear projection component is trainable. This helps integrate the collaborative information from the SASRec while simultaneously ensuring that the LLM does not forget previously learned information.

3.4 Large Language Model Layer

The LLM layer takes as input the projected item embeddings and textual representations associated with user-item interactions. It processes the input sequences through its multiple layers of transformers, where each layer applies self-attention mechanisms to capture complex relationships and dependencies between different parts of the input, thus encoding contextual information about items or user preferences. Self-attention mechanisms are used to weigh the importance of different parts of the input sequence, allowing the LLM to focus more on relevant features or words that are more predictive

of user preferences, thereby improving the quality of the output embeddings. The output from the LLM layer is a set of dense, high-dimensional embeddings that encapsulate the semantic meaning and context of the input items. These embeddings are richer and more informative due to the LLM's ability to understand and incorporate complex language patterns and relationships. The item representations are crucial for making accurate recommendations, as they enable the model to understand nuanced preferences and item characteristics. The LLM's ability to generate context-aware embeddings ensures that recommendations are more personalised and aligned with user preferences.

3.4.1 Selecting the Backbone LLM

Currently, there are numerous LLMs available, for instance the GPT series, Gemini, PaLM and LLaMA [6, 9, 57]. However, many of them have restricted the access to their model parameters or APIs, making them difficult to use for research applications. Owing to data security concerns which are of utmost importance in the field of recommendation systems, relying on third-party APIs like ChatGPT to harness LLMs is infeasible. To address real-world recommendation challenges, we aim to simulate the practical use of a public LLM by updating its parameters for recommendation tasks. After thorough evaluation, the LLaMA2-13B model, which is currently among the top-performing open-source LLMs, was selected as the backbone LLM due to its exceptional generalisation capabilities and publicly accessible training data [57].

3.4.2 Parameter Efficient Fine Tuning using LoRA

Large contemporary LLMs like LLaMA are pre-trained on vast amounts of data and full fine-tuning could sometimes lead to the model forgetting some of the valuable pre-trained knowledge. Parameter-Efficient Fine-Tuning (PEFT) [36] approaches enable fine-tuning in a way that allows the model to preserve a majority of its pre-trained parameters while adapting to the task of personalizing recommendations effectively, thus mitigating the risk of catastrophic forgetting.

The core principle behind lightweight tuning postulates that although modern LLMs have an exceedingly large number of parameters, their information is concentrated in a low intrinsic dimension [21]. Therefore, tuning only a minimal subset of its parameters, while preserving the others in their frozen state, may help us achieve performance comparable to that of the full model [36, 31, 33]. This thesis utilises Low Rank Adaptation (LoRA) [21], which optimises efficient fine-tuning by injecting

task-specific knowledge via trainable low-rank decomposition matrices into specific layers of the model architecture, keeping all other parameters of the pre-trained model frozen. Therefore, the learning objective for LoRA modifies equation 3.11 as:

$$\max_{\Theta} \sum_{(x,y)\in Z} \sum_{t=1}^{|y|} \log\left(\mathbf{P}_{\Phi+\Theta}(\mathbf{y_t} \mid \mathbf{x}, \mathbf{y}_{<t})\right) \tag{3.13}$$

where Θ is the LoRA parameters that are updated during the training process [21].

In the proposed model, we want to determine which layers of LLaMA can be effectively fine-tuned to help the model follow the instructions provided in the prompts and adapt to generate relevant recommendations with a significantly high accuracy. In the preliminary investigation, an empirical evaluation was conducted using techniques like Sensitivity Analysis to determine the impact of modifying each layer on the overall performance. We began by applying LoRA to the attention layers only and evaluated the performance. Then, based on the results, we shifted focus to apply LoRA to the weight matrices in the feed forward layers and tested different configurations to find the best balance between performance and efficiency. Although this was time consuming, a valuable insight was derived that applying LoRA on layers closer to the input and output of the transformer is more effective as they contain more task-specific information.

Then, we used gradient-based methods to identify which layers or modules of LLaMA contribute most to the loss during fine-tuning on the recommendation task. A few epochs of fine-tuning were performed across all layers of LLaMA with small learning rates. The gradient norms in each layer were computed, recorded and analysed to identify the layers with higher gradient magnitude. These layers were more sensitive to task-specific fine-tuning, suggesting they should be targeted for LoRA.

During the fine-tuning process, we abide by the settings in Platypus [30] to apply LoRA on the gate_proj, down_proj and up_proj modules and some attention heads of LLaMA (Appendix B). Selectively applying LoRA on these layers while keeping the rest of the model's parameters frozen showed faster convergence and statistically significant improvement in recommendation metrics, whereas, freezing them led to a quantifiable performance degradation. The modified parameters account for less than 1% of the total parameters of the original LLM, thereby improving training efficiency.

3.5 Prediction Layer

The prediction layer of the model plays a crucial role in translating the rich, context-aware output embeddings produced by the LLM into actionable recommendation

scores. The proposed approach redefines the recommendation task as a joint probability distribution modelling problem, generating an N-dimensional vector that contains prediction scores for all candidate items in a forward pass. It represents the likelihood of each item being the next recommendation, optimizing for a one-shot computation of prediction results for a given sequence across all candidate items.

The key component of the prediction layer is an item linear projection, which transforms the LLM's high-dimensional output embeddings into an item space where each dimension corresponds to a candidate item in the recommendation list. This transformation is achieved through a learnable weight matrix $\mathbf{W_{Output}} \in \mathbb{R}^{d_k \times N}$, where d_k is the dimension of the LLM's output embeddings and N is the number of candidate items. The output embeddings are multiplied by $\mathbf{W_{Output}}$, resulting in an N-dimensional prediction vector $\hat{\mathbf{y}} \in \mathbb{R}^N$, where each element $\hat{\mathbf{y}}_\mathbf{i}$ represents the predicted probability score for a corresponding candidate item i. These scores can be used to rank the candidate items, with higher scores indicating higher likelihood of user preference.

If $\mathbf{h} \in \mathbb{R}^{d_k}$ is the hidden state output from the LLM, the prediction computed via the item linear projection can be represented as:

$$\hat{\mathbf{y}} = \mathbf{W}_{Output}^T \mathbf{h} \tag{3.14}$$

The output of the item linear projection is passed through a softmax function to convert the raw scores into a probability distribution. During training, the model learns to adjust the weight matrix $\mathbf{W_{Output}}$ by minimizing the cross-entropy loss, which is the difference between the predicted scores $\hat{\mathbf{y}}$ and the ground-truth of actual user interactions. A full softmax computation ensures proper gradient updates for the model to learn accurate mappings from input sequences to item recommendations by using a probability distribution. The loss function is defined as:

$$\mathcal{L} = -\sum_{i=1}^{N} y_i \log(\hat{y}_i) \tag{3.15}$$

where y_i is the binary ground-truth label indicating if item i is indeed the next item in the sequence and $\hat{y}_i$ represents the predicted probability or score for item i.

Projecting the LLM's output directly into the item space allows the model to generate a recommendation score for all candidate items in a single forward pass, making it efficient for large-scale recommendation tasks. By using a fixed candidate set represented in the trainable weight matrix $\mathbf{W_{Output}}$, the model ensures that all recommendations are valid items within the candidate set, avoiding out-of-range results.

3.6 Inference Layer

The inference mechanism encompasses the entire process of using the trained model to generate recommendations for users on new, unseen sequential data in real time. It starts with leveraging the pre-trained SASRec model to obtain ID embeddings and projecting and combining them with textual prompts to serve as input into the LLaMA model. The LLM processes this input to generate an output embedding, which is then passed through an item linear projection to get scores for each candidate item. However, unlike the training phase, during inference, the prediction layer replaces the expensive softmax function with a nearest neighbour search in the item embedding space to swiftly compute probabilities and rank the top N items which are most similar to the user's historical interactions and current context. This process, shown in Figure 2, guarantees that the most relevant recommendations are generated in a single forward pass.

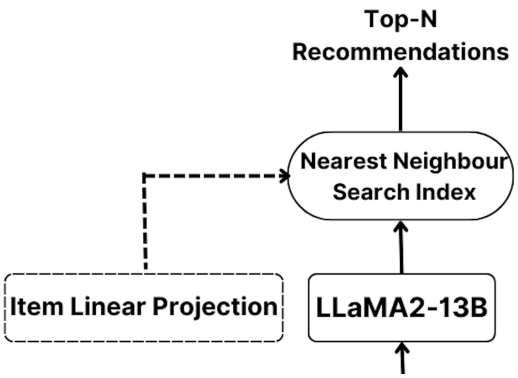

Figure 2: An illustration of the IDentityRec model inference architecture

3.6.1 Nearest Neighbour Search

Softmax normalisation is computationally expensive, especially in practical scenarios where the number of candidate items is very large. Therefore, the architecture focuses on finding the closest matches in a high-dimensional candidate item space without being constrained by the probabilistic interpretation imposed by softmax. An approximate nearest neighbour (ANN) search is employed using efficient KD-Trees data structure and the Annoy algorithm [58] to efficiently generate a ranked list of recommended items. It reduces the computational overhead and speeds up the recommendation process, meeting the low latency criteria of real-world systems and scaling the architecture to handling large candidate item sets. The candidate set is populated by selecting a subset of items that are closest to the user's query embedding in the entire item vector space.

3.6.2 Generating recommendations for unseen sequences

In the proposed inference architecture, the nearest neighbour search calculates the cosine similarity score between the output embedding of the LLM, O, which encapsulates the user's behavioural patterns, and the projected candidate item embedding $\hat{E}_i$. Then, the prediction layer ranks the candidate items based on their similarity to the user's context, with higher scores indicating a higher likelihood for an item to be recommended. This method is scalable as it evaluates all candidate items simultaneously, making it suitable for large-scale recommendation systems with extensive candidate sets.

For each candidate item i in the prediction layer, the cosine similarity score, S_i:

$$S_i = \frac{O \cdot \hat{E}_i}{\|O\| \|\hat{E}_i\|} \tag{3.16}$$

The similarity scores form a joint probability distribution over the candidate set. Ranking them based on the descending order of scores gives the top-N results.

3.7 Model Deployment

The proposed model can be deployed in an extremely efficient and lightweight manner. The backbone large language model serves as a universally shareable foundation for various sequential recommendation tasks, implying that it is sufficient to instruction tune the LLM only once, after which it can be reused across all subsequent tasks.

The use of Low-Rank Adaptation allows the model to be efficient in terms of parameter usage, focusing on adapting key parts of the model with minimal overhead. When dealing with new sequential recommendation datasets, only a minuscule fraction of components need to be trained and stored for deployment. The trainable components include the item ID embeddings $\mathbf{E}$ from a pre-trained sequential recommendation model, the linear projection weights $\mathbf{W}_{\text{input}}$ in the input layer that unify the ID embeddings and textual prompts, the LoRA weights Θ in the LLM layer and the item linear projection weights $\mathbf{W}_{\text{output}}$ in the output layer. These account for roughly 0.07% of the total parameter count, when compared to the 13 billion parameters of LLaMA. Moreover, all the trainable elements are fully modular, allowing the recommender system to quickly adapt to a specific dataset by simply swapping out or updating these interchangeable components, without extensive retraining. This efficiency is critical when deploying models in resource-constrained environments.

The training and deployment pipeline of the proposed recommendation system has been encapsulated step-wise in the algorithm in Appendix A.

Chapter 4

Experiments

In this chapter, we introduce the datasets utilised in this study, followed by a detailed discussion on the pre-processing techniques implemented to ensure the integrity and suitability of the data, particularly addressing the critical issue of data leakage. Then, we proceed to provide an in-depth account of the model's implementation, including relevant configuration details and hyper-parameter tuning strategies for optimizing model performance. This chapter goes on further to outline the metrics selected for performance evaluation and briefly describes the baseline models used for a comparative analysis. The last section mentions how we design and perform a series of experiments to thoroughly assess the model's effectiveness, focusing on challenging scenarios such as data-sparse environments and cold start situations. These experiments are crucial for understanding the model's adaptability in real-world applications.

4.1 Datasets

Reviewing sequential recommender systems literature [25, 8, 52] prompted us to select two popularly used real-world datasets of varying domains, namely Amazon Reviews and Yelp. The **Amazon Reviews** dataset [38, 20] is a comprehensive, multi-domain, e-commerce dataset comprising of a large corpora of reviews, ratings, product metadata and customer information crawled from the Amazon.com website. For our analysis, we concentrate on three specific categories, 'Beauty', 'Sports and Outdoors' and 'Toys and Games' which are notable for their high sparsity and variability.

The **Yelp dataset** [67] is a comprehensive collection of data from the popular online review platform Yelp. It includes detailed information about businesses, their attributes, user profiles and reviews. In our experiments, only the data generated in or after 2019

is used. The Yelp dataset serves the purpose of an unseen dataset on which IDentityRec has not been pre-trained, to test the cross-domain performance of our model. This assessment aids in determining the model's ability to transfer the learned features to a different domain and generalise across varying levels of data sparsity.

Dataset	#Users	#Products	#Interactions	Sparsity
Beauty	22,363	12,101	198,502	99.93%
Sports	25,598	18,357	296,337	99.95%
Toys	19,412	11,924	167,597	99.93%
YELP	30,431	20,033	316,354	99.95%

Table 2: Dataset Statistics

The selected datasets vary not only in their domain, but also in terms of their size and number of user-item interactions. As evident from Table 2, all the datasets exhibit extremely high sparsity, indicating that a vast majority of user-product pairs have no recorded interactions. Since our model deals with IDs, it is specifically trained and tested on such datasets to compare its robustness against traditional collaborative filtering methods which struggle in sparse data scenarios. It also provides a scope to investigate the cold-start problem, which arises when the data is insufficient to recommend new or less popular products. By choosing datasets of varying characteristics and spanning different domains, we hope to increase the confidence that any pattern observed across all of them is indeed a general trend and not a result of specific properties of the data.

4.2 Data Pre-processing

We followed the common practices in literature [45, 52] to prepare each dataset. Explicit ratings were converted into implicit feedback by considering that the presence of a review or rating indicated a user's interaction with the item. The dataset was extensively pre-processed by deleting the items with ratings below 0.0 and removing all transactions which were either duplicates or had corrupted item IDs. Transactions related to items or users with missing attributes were also rejected. To mitigate bias and deal with memory issues while modelling, a smaller, representative subset of the data was taken by randomly sampling the transactions of the original dataset. Henceforth, the terms dataset and data have been used interchangeably in this thesis to refer to this dataset.

For robust evaluation, we applied the 5-core setting and discarded unpopular items having lesser than 5 related actions. Then, the interactions were sorted in an ascending order of their timestamps. We retained the last 50 interactions (most recent) as the historical sequence, padding or truncating sequences with fewer or more than 50 interactions respectively, to ensure a fixed length.

The dataset was split into training, validation and test sets using a leave-one-out strategy to adapt to evolving user preferences. For each user, the most recent item in the sequence history was reserved for testing, the second most recent item was utilised for validation and the remainder was used for training. The 8:1:1 split resulted in a training set containing 80% of the data, with validation and test sets making up 10% each.

4.2.1 Addressing Data Leakage Concerns

In our sequential recommendation task, we aggregated user-wise data to construct their interaction history and then used the strategy discussed above to determine the train, test and validation splits. However, when we pivot to other recommendation tasks like rating prediction, 80% of the dataset is randomly allocated for training and 10% each for validation and testing. As a result, we may encounter certain interactions that appear simultaneously in the training data of one downstream task and in the test set of another recommendation task, leading to data leakage [24]. In multitask scenarios, we reused the splits after removing the overlapping interactions in the training and test sets to ensure that any test data did not leak into the training set.

Using the embeddings generated by pre-training a SASRec model on the same dataset that we would be using for evaluating IDentityRec also raises potential concerns about data leakage, compromising on the model's ability to generalise to truly unseen data. To mitigate this, we enforced a strict temporal split, training SASRec only on the interactions that have occurred before a cut-off time. To procure a consistent test set that would ensure the integrity of our experiments, our validation and test sets were curated to contain future interactions that occurred after this timestamp.

During the IDentityRec training phase, we fine-tuned SASRec's pre-trained embeddings, ensuring that the item and user embeddings were updated and fine-tuned solely based on historical interactions that the model would have observed during the actual training phase of IDentityRec. Once fine-tuning was completed, the embeddings were frozen, thereby disallowing further updates when evaluating on the validation or test set. By effectively forgetting any information that might have been learned from the future

interactions during the initial SASRec pre-training, we avoid embedding leakage.

This maintains the chronological integrity of the user-item interactions and assures a fair comparison with recommendation baselines, allowing us to assess the model's ability to predict future interactions based on historical data.

4.3 Implementation Details

In this section, we delve into the technical details of how we implement the proposed model and briefly discuss the foundational recommendation models against which we want to compare our model's performance.

4.3.1 Baseline Recommendation Models

The baseline methods chosen for comparison are grouped into three categories, which include heuristic-based non-sequential methods like Popularity (POP) and Bayesian Personalised Ranking (Matrix Factorisation), traditional transformer-based sequential recommendation methods like GRU4Rec [19] and SASRec [25] and self-supervised sequential methods like BERT4Rec [52]. However, methods that employ data augmentation [68, 69] are orthogonal to our model and have been excluded from our evaluation to ensure an unbiased comparison. A summary of the underlying working principles of each of these models has been discussed in Appendix C.

All the foundational baselines have been implemented using PyTorch 2.0.1. The maximum sequence length was set to 50 for every model on all the datasets. While the RecBole library [66] was referenced to implement the vanilla BERT4Rec model, other methods including GRU4Rec were implemented using the code from their official public repositories [19]. All the hyper-parameter configurations were the same as that mentioned by their respective authors. We used Xavier initialisation [15] to initialise the model parameters and tuned the hyper-parameters on the validation set by monitoring the performance over a maximum of 20 epochs, post which training terminated. A learning rate of 0.001, batch size of 256 and an embedding dimension of 64 were used with Adam [26] as the optimiser for optimal results.

4.3.2 IDentityRec Training Details

From existing literature [25, 52], it was deciphered that using longer input sequences marginally increases the model performance. So, to suit our computational budget, we

limited ourselves to using moderately long input sequences ($n = 50$ items) which were either truncated or padded to ensure that the sequences were of uniform length.

To obtain item ID embeddings from a pre-trained sequential recommendation model, a vanilla version of the SASRec architecture was trained on the highly sparse subsets of Amazon Reviews data with the maximum sequence length set to 50. The model was implemented in TensorFlow using two self-attention blocks and learned positional embeddings. The learning rate was set to 0.001. We trained the model with a fixed batch size of 128 and a dropout rate of 0.5, using the Adam optimiser [26]. As discussed in Section 3.3.1, we directly extracted the item ID embeddings from this pre-trained SASRec model without any modification.

The LLaMA2-13B model, which serves as the backbone LLM for the IDentityRec, was accessed from the HuggingFace library [48]. We conducted Alpaca based instruction tuning on the LLM using the extracted item IDs for one epoch. To circumvent the memory bottleneck while fine-tuning the 13 billion parameter version of LLaMA, an optimised LLaMA-LoRA architecture was employed. In an attempt to reduce the training time even further, we adopted a data parallel technique, leveraging a GPU cluster for conducting the experiments.

Parameter	**Beauty**	**Sports**	**Toys**
Training Epochs	3	3	2
Warmup Steps	100	200	100
Learning Rate	3e-4	2e-4	2e-4
Learning Rate Scheduler	Cosine Scheduler		
Weight Decay	0.1		
Batch Size	16		
LoRA Modules	[gate_proj, down_proj, up_proj]		
LoRA Rank	16		
LoRA Alpha	16		
LoRA Dropout	0.05		

Table 3: Optimal hyper-parameter configurations for different datasets

The hyper-parameter settings for achieving optimal performance using IDentityRec were obtained by conducting a grid search of training configurations on the validation set. The optimum values of the model parameters listed in Table 3 ensured that the

model generalised well across datasets. To mitigate the impact of randomness, we reported the average outcomes of five runs using different random seeds in Section 5.1.

In general, the values of learning rate and the number of training epochs were found to vary with respect to the dataset. Training the model for 2-3 epochs on four NVIDIA A100 GPUs using the Adam Optimiser yielded optimal performance. Batch sizes greater or lesser than 16 were prone to failure in convergence. A warm-up strategy initiated with 1/100 of the maximum learning rate was gradually adjusted over the first few hundred steps using a Cosine Learning Rate Scheduler to facilitate stabilisation of the training process towards the end. A uniform weight decay of 0.1 was applied to penalise large weights and prevent overfitting through regularisation.

The rationale behind selecting the specific layers of LLaMA on which the LoRA adaptors were applied has been extensively discussed in section 3.4.2. Through experimentation, it was observed that although the rank and alpha parameters of Low Rank Adaptation did not impact the model much, it was very sensitive to the LoRA dropout value. Changing it caused severe fluctuations in model performance. The alpha value of 16 was used to scale the LoRA layers while a moderately low-rank ($r = 16$) achieved a balance between model efficiency and flexibility. LoRA's dropout rate was set to 0.05 to ensure adequate regularisation and prevent overfitting.

4.4 Evaluation

This section outlines the evaluation metrics that have been used to measure the model's performance, including their mathematical formulation and theoretical description.

4.4.1 Hit Rate

Hit Rate has commonly been used in recommendation literature [25] to measure the proportion of test cases wherein the top N items recommended by the system contain the ground truth item which the user has actually interacted with. The Hit Rate is:

$$\text{HitRate@N} = \frac{1}{|\mathcal{D}|} \sum_{u \in \mathcal{D}} \mathbb{I}(\text{Rank}_u(i_u) \leq N) \tag{4.1}$$

where $\mathcal{D}$ is the set of all user sequences in the test set, $\text{Rank}_u(i_u)$ is the rank of the actual item i_u in the recommendation list for user u and N is the cutoff rank for considering a recommendation a hit. $\mathbb{I}(\cdot)$ represents a binary indicator function (1 if

true, 0 otherwise) and $\text{Rank}_u(i_u) \leq N$ means that the rank of the actual item i_u is within the top N recommended items. Higher the hit rate, better is the model performance.

4.4.2 Normalised Discounted Cumulative Gain

NDCG measures the effectiveness of a recommendation model by comparing the predicted ranking of items against an ideal ranking based on their relevance to the user. The metric penalises relevant items that appear lower in the recommendation list because users are more likely to pay attention to the items at the top. For a ranked list of items $\{i_1, i_2, \ldots, i_N\}$, with corresponding relevance scores $\{r_1, r_2, \ldots, r_N\}$, the Discounted Cumulative Gain (DCG) at rank N and IDCG, which is the DCG of the ideal ranking where items are perfectly ordered by relevance is given by:

$$\text{DCG@N} = \sum_{k=1}^{N} \frac{r_k}{\log_2(k+1)} \quad \text{IDCG@N} = \sum_{k=1}^{N} \frac{r_k^*}{\log_2(k+1)} \quad \text{NDCG} = \frac{\text{DCG}}{\text{IDCG}} \tag{4.2}$$

where r_k^* is the relevance score of the item at position k in the ideal ranking.

Mathematically, NDCG is a value between 0 and 1, where 1 indicates a perfect ranking, which is the case when the predicted ranking matches the ideal one.

4.5 Experimental Design

In this subsection, we discuss several experimental setups which have been designed to assess and evaluate the holistic performance of our model. While the experiments are elucidated in this section, their results, insights and analyses are discussed in Section 5.

4.5.1 Performance Analysis through Negative Sampling

In addition to a regular performance analysis using the aforementioned metrics, we evaluated the ranking ability of our recommendation model using negative sampling [25, 68]. It involved drawing a reliable and representative sample of 99 negative items that the user has not interacted with, sampled according to their popularity, for every positive ground truth item. Popularity sampling makes them harder negatives as popular items are more likely to be relevant or attractive to users, thereby pushing the model to distinguish better. The objective is to maximise the difference in predicted scores between positive (relevant) and negative (irrelevant) samples using a binary cross-entropy loss. Mathematically, the learning objective for a user u is defined as:

$$\mathcal{L}_u = -\sum_{i \in I_u^+} \log(\sigma(\hat{y}_u(i))) - \sum_{j \in I_u^-} \log(1 - \sigma(\hat{y}_u(j))) \tag{4.3}$$

where $\sigma(\cdot)$ is the sigmoid function which maps the predicted scores to probabilities and $\hat{y}_u(i)$ and $\hat{y}_u(j)$ are the predicted scores for the positive item i and the negative item j.

4.5.2 Robustness Analysis and Cold Start

To analyse IDentityRec's effectiveness in addressing data sparsity, the users in each dataset are grouped into three categories based on their number of interactions with items. Having removed all transactions with less than 5 interactions during pre-processing, users with exactly 5 interactions were allocated to the "sparse" group, those with 6-9 interactions were assigned to the "medium" group and users with more than 9 interactions were designated as the "dense" group. The performance of IDentityRec was evaluated in the cold start scenario by comparing the evaluation metrics for each user group against the SASRec baseline. By plotting the nDCG@5 scores of IDentityRec when the number of interactions were gradually increased from 5 to 11, we analysed the IDentityRec's performance as a function of increasing sparsity .

4.5.3 Creating Ablations and Model Variants

An ablation study was designed to evaluate the contribution of each novel component in the architecture. We compared the performance of the IDentityRec against its variants which were created by selectively altering or removing certain elements of the model.

We started with the SASRec + LLaMA variant, which is the IDentityRec itself, that uses the SASRec ID embeddings and fine-tuned LLaMA for generating recommendations. Then, we removed the injected IDs and used only textual prompts as the LLM's input. The next variant, BPR+LLAMA, replaced the pre-trained item ID embeddings from SASRec with the embeddings from a non-sequential, matrix factorisation based model like Bayesian Personalised Ranking (BPR). Another ablation called SASRec + BERT was created by replacing the LLaMA2-13B with a BERT-base-uncased (110M) model in the LLM layer. The final version, SASREC w/o LLM, was an LLM-free variant that utilised only the SASRec for ID injection and linear projections for prediction. For each ablation, we used the same dataset and followed the same training protocol to conduct an unbiased evaluation. The HR@20 score of each variant was analysed to comprehend the effect of removing or altering a component.

Chapter 5

Results and Analysis

This chapter presents a comprehensive analysis of the experimental results, highlighting the performance of the proposed model across various scenarios and benchmarks. It describes how the proposed model compares against baseline methods in terms of performance, drawing key insights to analyse its strengths. This is followed by an elaborate discussion about the results of an ablation study which validates the contribution of the novel components in the model to its overall effectiveness.

The chapter also examines the model's robustness in challenging recommender system scenarios, including data-sparse environments and cold start situations. The results of cross-domain generalisation are studied to observe how the model adapts to different domains. In addition to these evaluations, we conduct a detailed analysis of the model's time and space complexity, providing a holistic view of its computational efficiency and scalability. Through these analyses, this chapter aims to offer a thorough understanding of the model's superior capabilities and practical applicability.

5.1 Performance Analysis

The performance of the proposed IDentityRec model was evaluated on the three Amazon Reviews data subsets, Beauty, Sports and Toys, using Hit@k (H@k) and nDCG@k (N@k) metrics, where k represents the number of recommended items considered for evaluation. By evaluating our model against popular recommendation baselines at different values of k (= 5, 10 and 20), we attempt to gain insights into its performance at different levels of recommendation depth. Too large values of k, might consider a broader range of recommendations and give us a better score, but it dilutes the impact of highly relevant candidate items and hence, it is ignored.

Dataset	Metric	POP	BPR (MF)	GRU4 Rec	SAS Rec	BERT4 Rec	IDentity Rec	Improve
Beauty	**H@5**	0.007	0.012	0.016	0.033	0.019	**0.053**	60.6%
	H@10	0.011	0.036	0.029	0.058	0.041	**0.076**	31.0%
	H@20	0.019	0.059	0.478	0.091	0.059	**0.117**	28.5%
	N@5	0.004	0.006	0.008	0.017	0.018	**0.035**	94.4%
	N@10	0.005	0.012	0.014	0.026	0.025	**0.043**	65.3%
	N@20	0.007	0.018	0.017	0.034	0.036	**0.051**	41.6%
Sports	**H@5**	0.005	0.009	0.014	0.017	0.017	**0.028**	64.7%
	H@10	0.009	0.188	0.027	0.029	0.032	**0.041**	28.1%
	H@20	0.015	0.026	0.044	0.048	0.049	**0.071**	44.8%
	N@5	0.004	0.005	0.009	0.009	0.010	**0.019**	90.0%
	N@10	0.005	0.008	0.014	0.013	0.015	**0.023**	53.3%
	N@20	0.006	0.012	0.017	0.017	0.019	**0.029**	52.6%
Toys	**H@5**	0.006	0.012	0.009	0.044	0.027	**0.056**	27.2%
	H@10	0.008	0.021	0.019	0.069	0.046	**0.079**	14.5%
	H@20	0.011	0.031	0.030	0.099	0.069	**0.118**	19.1%
	N@5	0.003	0.008	0.006	0.023	0.017	**0.040**	73.9%
	N@10	0.006	0.012	0.009	0.032	0.023	**0.048**	50.0%
	N@20	0.006	0.014	0.011	0.039	0.029	**0.055**	41.0%

Table 4: Performance comparison of models on the Amazon Reviews dataset. The best performance is highlighted in bold while the second best performance is underlined.

The evaluation results, presented in Table 4, provides us valuable insights into the performance of the proposed model. IDentityRec outperforms all the popular foundational recommendation baselines on each metric and in every dataset. A general pattern emerges with non-neural methods like POP and BPR performing consistently worse on all the datasets, underscoring the need for models that capture the temporal dynamics and evolving preferences of users. While sequential models like SASRec and BERT4Rec perform comparatively better, the IDentityRec outperforms them significantly, owing to its strength in capturing the users' interaction patterns.

An average improvement of approximately 35% on HR@k and 62% on nDCG@k over the best baseline demonstrates the effectiveness of our model. A substantially high

performance gain under both the metrics highlights the fact that IDentityRec not only recommends relevant items, but does so in a way that prioritises placing the most relevant items in optimal positions within the recommendation list. This propels higher levels of user engagement, especially in e-commerce platforms, where customers are more likely to interact with the top-ranked items in the recommendation list. Another related observation is that the IDentityRec has more pronounced performance improvements on smaller values of k. This implies that even in scenarios where the model is allowed to provide only a few recommendations, IDentityRec effectively identifies the most contextually relevant items based on the history and recommends them to the user.

The most significant performance gain ($\approx 40\%$ in HR@k) is evident in the Beauty dataset. This is likely because it contains a diverse range of products, each with specific attributes that appeal to different sets of users. Users in this category exhibit distinct and strong preferences based on brand loyalty or product types. Our model benefits from learning these interaction patterns from the collaborative and contextual signals and excels in capturing the nuanced preferences of users in such a diverse category.

5.1.1 Evaluating Ranking Performance through Negative Sampling

The use of negative samples help to test the model's ability to distinguish relevant (positive) items from irrelevant (negative) ones, which is critical for high-precision recommendation systems. Metrics like HR@1 and nDCG@5 have been used to measure the model's ability to place the most relevant item at the top of the recommendation list.

	Beauty			Sports			Toys		
Model	H@1	H@5	N@5	H@1	H@5	N@5	H@1	H@5	N@5
POP	0.06	0.21	0.13	0.07	0.23	0.15	0.06	0.19	0.13
BPR	0.04	0.14	0.09	0.04	0.16	0.10	0.02	0.09	0.06
GRU4Rec	0.13	0.31	0.22	0.11	0.30	0.21	0.09	0.27	0.19
SASRec	<u>0.18</u>	<u>0.37</u>	<u>0.28</u>	<u>0.14</u>	<u>0.34</u>	<u>0.25</u>	<u>0.18</u>	<u>0.36</u>	<u>0.28</u>
BERT4Rec	0.15	0.36	0.26	0.12	0.33	0.23	0.12	0.33	0.23
IDentityRec	**0.22**	**0.40**	**0.32**	**0.17**	**0.37**	**0.27**	**0.20**	**0.39**	**0.31**
Improve	21.6%	9.2%	13.1%	19%	7.3%	8.1%	10.5%	6.1%	10.4%

Table 5: Performance comparison of methods with sampled negative items. The best performance is highlighted in bold while the second best performance is underlined.

The experimental design of negative sampling has been outlined in Section 4.5.1. As illustrated in Table 5, our proposed model can still exceed all the baseline models with a significant margin, proving the superiority of IDentityRec in ranking ability. The model shows superior hit rates, particularly in HR@1 and HR@5, demonstrating its effectiveness in ranking relevant items higher. By effectively identifying relevant items and filtering out non-pertinent ones, our model can provide more personalised recommendations. This is crucial in e-commerce for generating the top-5 recommendations.

5.2 Ablation Study

To gain deeper insights into the contribution of different components in IDentityRec, we conducted a series of ablation studies by systematically eliminating or modifying specific aspects of the architecture and assessing their impact on performance. Throughout this section, we have referred to IDentityRec and SASRec + LLaMA interchangeably, since those are the two most important components in IDentityRec's architecture. Table 6 presents the results of the study described in Section 4.5.3, using the HR@20 metric for evaluation. The analysis has revealed several significant insights.

Model Variant/Ablation	Beauty	Sports	Toys
Full **IDentityRec** Model **(SASRec + LLaMA)**	**0.117**	**0.071**	**0.118**
- Remove Positional Embedding	0.036	0.017	0.059
- Remove Pre-trained Sequential ID Embedding	0.051	0.046	0.072
- Remove Sequential ID Embedding **(BPR + LLaMA)**	0.097	0.062	0.092
- Remove LLM **(SASRec w/o LLM)**	0.022	0.023	0.018
- Replace LLM with BERT **(SASRec + BERT)**	0.084	0.058	0.105

Table 6: Ablations and their corresponding HR@20 values on Amazon Reviews Data

Removing the positional encodings led to a catastrophic drop in performance of about 65.1% on average across all the datasets. This underscored the importance of encoding sequential information in the model as IDentityRec relies on understanding the temporal order of sequences to make accurate recommendations. Taking a step further, we examined the necessity of incorporating ID information in the input layer of the model. Removing the ID embeddings completely and passing only textual prompts as input resulted in a 43% degradation in HR@20, justifying that the collaborative

information contained in the IDs is essential for generating relevant recommendations.

The next ablation probed further into the magnitude of the impact of the pre-trained IDs from SASRec. The performance analysis in Figure 3 revealed a marginal drop in performance of the BPR + LLAMA variant compared to IDentityRec, indicating that SASRec's ID injection mechanism is slightly more effective than that of BPR. This is explained by the effectiveness of SASRec in capturing sequential patterns in user-item interactions compared to a non-sequential, matrix factorisation technique like BPR.

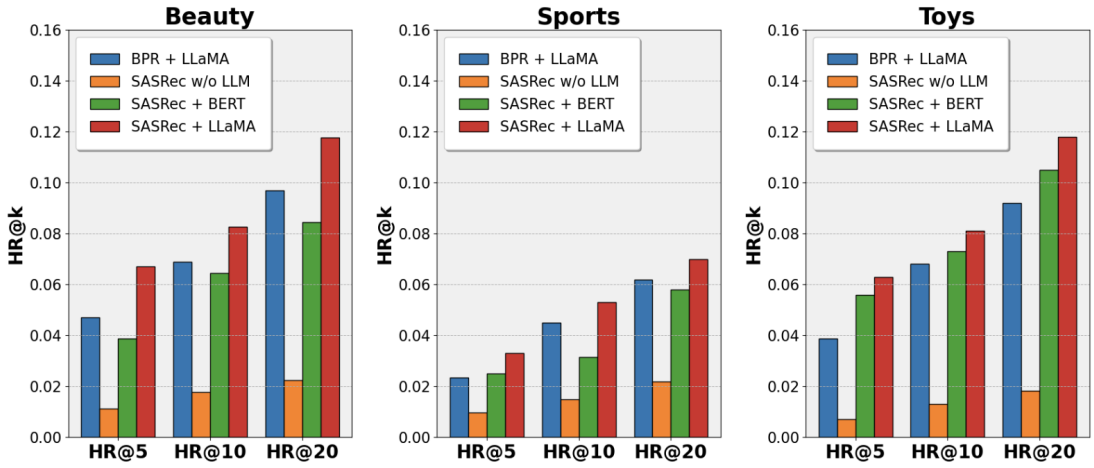

Figure 3: Ablation Studies on Amazon Reviews data across HR@k, k= 5, 10, 20

Now that the importance of using a pre-trained SASRec ID injection has been established, we shift our focus towards studying the need for an LLM in the model's architecture. We observed that the HR@k values plummeted sharply for the SASRec w/o LLM ablation, attaining consistently low scores across all domains, suggesting that the fine-tuned LLM component is crucial for enhancing recommendation performance. We investigate deeper by comparing this LLM-free variant against the LLM-based ablations like IDentityRec and BPR + LLaMA. The performance gain achieved while using LLM-inclusive model variants decisively accentuates the critical role of the LLM in leveraging its inherent knowledge to capture complex patterns and decipher semantic and contextual information to provide more personalised recommendations.

To specifically assess the significance of the LLaMA LLM in the architecture, we contrast the performance of SASRec + BERT with that of the SASRec + LLaMA variant. It is evident from Figure 3 that although SASRec + BERT comfortably outperforms the non-LLM ablation, it falls short of surpassing the SASRec + LLaMA on the HR@k metric, asserting the dominance of a fine-tuned LLaMA over pre-trained language models like BERT. This substantiates our claim that the LLaMA-LoRA architecture is

more adept at grasping the nuances and complexities of the recommendation task.

The IDentityRec model, referred to as the SASRec + LLaMA variant, maintains a clear performance advantage on the HR@k metric in varying domains (Beauty, Sports, Toys) across all tested configurations (H@5, H@10, H@20). This reinforces the fact that SASRec's sequential modelling capabilities and LLaMA's language modelling strengths complement each other effectively, yielding the most accurate recommendations. While the SASRec ID embeddings provide crucial collaborative filtering information, the LLM augments it by capturing complex patterns and contextual details that IDs alone cannot provide, emphasizing the merit of both the components in the architecture.

5.3 Cross-Domain Generalisation

From the results in Table 7, it is evident that IDentityRec, which was originally trained on the Amazon Reviews dataset, is remarkably well-suited for cross-domain generalisation through transfer learning on an unseen target domain like YELP. The key to leveraging our model for cross-domain tasks lies in its ability to adapt to new domains by only requiring minimal updations to its pluggable parameters based on the data from the new domain. Without having to retrain the entire model from scratch, IDentityRec ensures efficient resource utilisation and reduces the computational overhead.

Model	HR@5	HR@10	nDCG@5	nDCG@10	nDCG@20
Popularity	0.56	0.83	0.36	0.43	0.56
GRU4Rec	1.52	2.63	1.04	1.37	1.45
SASRec	1.61	2.65	1.02	1.34	1.79
BERT4Rec	1.86	2.91	1.21	1.71	2.23
IDentityRec	**2.66**	**4.18**	**1.89**	**2.38**	**2.97**
Improvement	43.1%	43.6%	56.1%	39.1%	33.2%

Table 7: Results of cross-domain performance on YELP. The best score is in bold

IDentityRec outperformed the foundational sequential recommender baselines by an impressive margin of 43% on an average, indicating how well the knowledge learned in one domain transfers to another. By generalizing across domains, contextual knowledge and insights gained from user preferences in the source domain (Amazon Reviews) enhanced the recommendation quality in the target domain (Yelp).

The sequential nature of purchase sequences in the Amazon dataset helps the model to capture behavior trends encapsulated in temporal patterns and transfer this learned knowledge to related domains like YELP, where similar sequential dynamics are incorporated in the target data, to make more cohesive recommendations. Moreover, both the source and target datasets contain consumer-centric items, whose reviews encode individual preferences. The high-quality ID embeddings of our model exploits this commonality between the datasets to help the model transfer learned collaborative information between domains. A convergence in performance patterns between the source and target domains further corroborates IDentityRec's robust learning capability and aids in determining which domains can effectively share the model.

By leveraging knowledge and patterns from related domains, the system can make informed recommendations even when direct data is sparse, alleviating data sparsity concerns in cross-domain scenarios. The adaptability of such models that generalise well across diverse e-commerce domains reinforces the model's potential for broad applicability in businesses looking to expand their recommendation systems to new markets or product categories without starting from scratch.

5.4 Robustness Analysis: Data Sparsity and Cold-Start

The graphs in Figure 4 are the result of an experiment conducted in section 4.5.3 to analyse the robustness of our proposed model in sparse data scenarios. IDentityRec consistently outperforms SASRec on the nDCG@5 metric on all datasets across the three user groups (sparse, medium and dense), conforming to the claim that the model is supremely robust and provides accurate recommendations, irrespective of data sparsity.

It is evident that the quantum of performance improvement is most pronounced in the sparsest user group having only 5 interactions, indicating that IDentityRec is better at generating viable and relevant recommendations in cold-start situations. The superiority in performance holds true for the YELP dataset as well, underscoring its robustness to generalise well across domains, even in sparse data scenarios.

As we transition to the user groups with moderate to dense interactions, the general trend suggests that both the models show an improvement in their nDCG score. However, IDentityRec scales better with increasing number of interactions, as evidenced by the sharp elbow in the graphs of the Beauty and Sports domains. Its sophisticated architecture allows the model to leverage additional data to learn user preferences better.

The plateauing performance gain with increased data availability suggests that while

most baselines handle data-rich environments reasonably, IDentityRec has a significant advantage in capturing latent behavioural patterns when the interaction data is sparse.

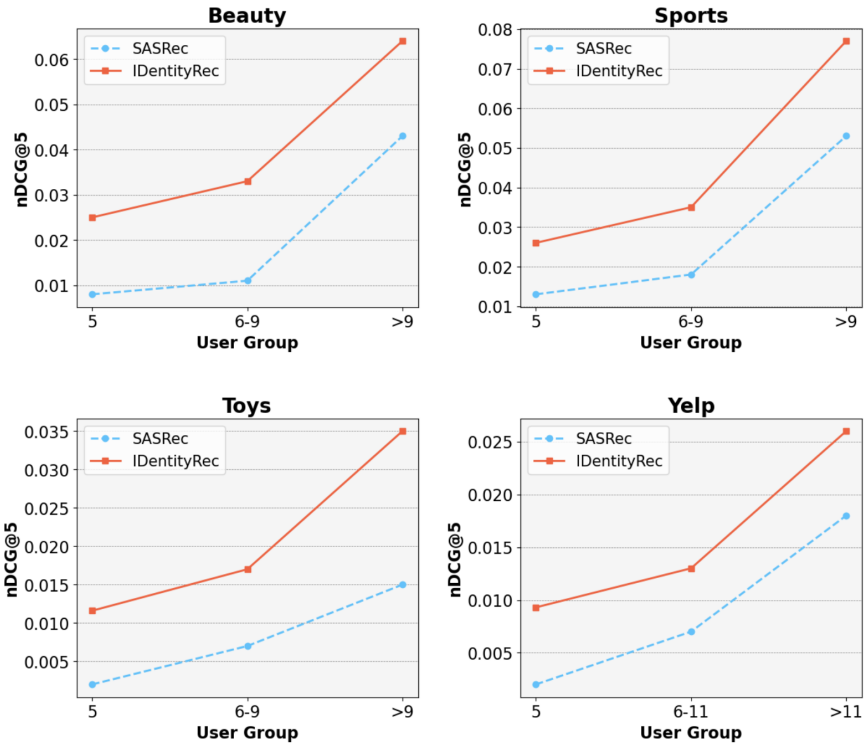

Figure 4: Performance of user groups based on length of interaction sequences

Although the trends within a user group remains consistent, the margin of improvement varies across datasets due to their intrinsic nature and difference in user engagement patterns. Niche categories like Beauty and Toys show the largest relative improvements in sparse data scenarios, suggesting that the model is particularly effective in domains where user preferences are likely to be diverse and less well-defined when the number of interactions are limited. IDentityRec's superiority in capturing fine-grained item similarities through its ID embeddings, makes it adept at modelling long-tail distributions with fewer interactions.

These results underscore the potential of IDentityRec to exhibit a much more stable and robust performance across all interaction levels, suggesting that it is less sensitive to data sparsity and well equipped to handle the cold-start problem. Its superior performance can be attributed to the few-shot learning ability of the instruction-tuned LLM, which extracts more meaningful information from a limited user history and effectively combines it with the injected collaborative signals to provide relevant recommendations in sparse-data settings. This characteristic is especially valuable in e-commerce systems where data can often be sparse, especially for new users or items.

5.5 Efficiency Analysis

Thus far, our analysis has proven how the proposed model is effective in performing sequential recommendation. In this section, we further argue that IDentityRec demonstrates competitive efficiency in terms of time and space constraints as well. To substantiate this claim, in Table 8, we have presented a comparative study of the inference time and storage requirements of the proposed model against the LLaMA2-13B.

As discussed in Section 3.7, IDentityRec postulates plugging in only a minimal set of lightweight, trainable parameters to the backbone LLM during deployment. While LLaMA2-13B is computationally intensive, the use of low rank adaptors helps to mitigate this overhead by allowing efficient fine-tuning, without having to retrain the entire model from scratch. The trainable linear projections in the architecture are simple, lightweight matrix operations that can be computed relatively fast, without acquiring much cost. Additionally, by replacing the computationally expensive softmax operation by an efficient nearest neighbour search during inference, we ensure that the inference time of IDentityRec is approximately identical to that of the LLM backbone.

Model	Inference Time	Storage Space
LLAMA2-13B	0.21 seconds/instance	12 GB
IDentityRec	0.27 seconds/instance	12 GB

Table 8: Time and space requirements of IDentityRec compared to the backbone LLM

ID embeddings do not consume significant storage space as they are typically low dimensional. However, LLaMA2-13B, whose size is in the order of tens of GB, inherently requires substantial storage space for its parameters. Although LoRA adapters introduce some additional parameters ($< 1\%$ of total LLM), they are designed to be storage-efficient. The linear projections add relatively small weight matrices, which when compared to the size of the LLM itself, are considered lightweight in terms of storage. So, the only contributor to storage requirements is the LLaMA model itself. This attests to the high storage efficiency of IDentityRec, relative to the LLM model.

Through our qualitative and quantitative analyses, it is evident that the proposed solution incurs a time and storage cost which is almost identical to that of the vanilla LLM. We extrapolate this observation to conclude that the inference time and storage space requirements of IDentityRec can be attributed solely to the underlying LLaMA model. However, improving the performance of LLMs is beyond the scope of the thesis.

Chapter 6

Discussions

This chapter delivers a thorough analysis and interpretation of the IDentityRec's performance, offering key insights from the experimental outcomes. Initially, we elaborate on the novel elements that amplify the model's performance across various product categories, emphasizing on the strategy of leveraging instruction-tuned language models, adept sequential modelling and effective integration of collaborative and contextual signals. Subsequently, we delve into a brief discussion of how extensible IDentityRec is, underscoring the crucial role of lightweight fine-tuning. The chapter also discusses the shortcomings of our approach, such as the high computational demands and limited use of item metadata. Finally, we explore potential avenues of research that could be undertaken to push the boundaries of generative recommendations.

6.1 Interpreting the Model Performance

The experimental findings reveal that IDentityRec consistently surpasses cutting-edge baselines by 35% on average (HR@k) across various product categories, testifying to the model's effectiveness in sequential recommendation. The amalgamation of user-specific collaborative interaction patterns captured through pre-trained ID embeddings and item-specific semantic information encapsulated in textual prompts of an instruction-tuned LLM, enables the model to provide accurate and contextually relevant recommendations.

IDentityRec is adept at effectively capturing niche user preferences, exhibiting a 40% performance gain in the Beauty dataset where interactions are sparse, diverse and strongly influenced by user specific attributes. This justifies the effectiveness of the LLM, which interprets the nuanced textual information and contextual relationship between products and user preferences in this category. For larger datasets like Sports

which has the maximum number of interactions (Table 2), IDentityRec leverages the additional data and benefits from more consistent patterns of user behavior captured in the ID embeddings, thereby maintaining a significantly high performance ($\approx 46\%$). The superior performance of the model across datasets of different sizes and interaction volumes can be attributed to its scalable architecture.

IDentityRec's strength lies not only in recommending a large number of relevant items (high HR@k), but also in its ability to accurately rank items, as evidenced by its superior performance in nDCG metrics, particularly for smaller values of k. Our experiment of negative sampling aligns the evaluation process with real-world situations, where a recommender system often needs to sift through a large number of irrelevant items to find the most relevant ones.

The use of LoRA for fine-tuning the LLM is a strategic design choice that balances performance and computational efficiency. By targeting specific modules within the LLaMA, the adapters effectively tailor the model to the recommendation task, enhancing its performance. The memory footprint, inference time and the training cost of IDentityRec is significantly low compared to fully fine-tuned models because LoRA updates only a small subset of parameters, specifically within low-rank sub-spaces.

IDentityRec exhibits exceptional cross-domain generalisation, effectively transferring the collaborative information and contextual knowledge learned from the Amazon dataset to apply them to a new, yet related domain like YELP. Such an adaptable model is crucial for practical applications where user data might be fragmented across different contexts or where new users and items are frequently introduced (Appendix D).

The use of pre-trained item IDs serves as a self-supervised learning mechanism, helping the model generalise better, especially in scenarios with limited historical data. IDentityRec's ability to prioritise highly relevant items even in sparsely populated user groups elucidates its robustness in handling the cold-start problem. The correlation between sequence length and model performance suggests that while richer data improves outcomes, there is a point at which the benefits plateau, underscoring the importance of balancing data quality and quantity in recommendation systems.

6.2 Extensibility of IDentityRec

In real-world applications, a recommendation system must be able to efficiently adapt to unseen datasets and perform different downstream tasks on it like ranking and next item retrieval. Even the assumption that the original dataset would remain unchanged is

erroneous because it deviates from industrial scenarios where new items are constantly introduced into recommender systems on a daily basis. This necessitates that the model must be flexible enough to adapt to new datasets and easily accommodate new items in the same dataset without having to retrain the recommendation system each time.

Having analysed the exceptional cross-domain performance of IDentityRec, we can reason that the model effectively generalises to new datasets through transfer learning. By simply training the ID embeddings, LoRA weights and the linear projection matrices of the input and prediction layers, the IDentityRec can be tailored to adapt to specific datasets. These model attributes do not interfere with the core functionality of the LLM, thereby facilitating lightweight model deployment through a universally shareable backbone LLM that requires instruction tuning only once, irrespective of the task.

Given that the pluggable components are ID-agnostic, integrating a new items into the existing dataset only warrants updating the linear projection layer with an additional row. This eliminates the need to retrain the entire model, thereby significantly enhancing the system's extensibility and adaptability to new items, making IDentityRec versatile.

6.3 Limitations and Future Scope

The performance of IDentityRec is heavily dependent on the quality and relevance of the pre-trained SASRec model and the LLM. In the absence of optimal pre-training on the specific domain of the dataset, IDentityRec's generated recommendations may be sub-optimal. Since the nearest neighbour search is constrained by a candidate item space, it might provide less diverse recommendations, favouring popular or frequently interacted items over more niche or diverse options that could interest the user. Incorrect instruction tuning of the LLM might lead to overfitting on specific patterns in the training data. Another evident bottleneck is the large space required to store the model.

A promising direction of research could be exploring strategies like model pruning and quantisation to reduce the computational overhead associated with LLMs. In future, incorporating explainability into IDentityRec through behaviour alignment [63] may help explain the black-box recommendations, allowing for the creation of customisable explanations that can be adapted to individual user preferences. Another avenue for future work could be adapting IDentityRec to handle multi-modal inputs such as visual and audio data to create contextually rich recommendation systems. Exploring federated learning methods to train the model on decentralised data sources while maintaining user privacy could find applicability in privacy-sensitive domains.

Chapter 7

Conclusion

The IDentityRec model represents a significant advancement in the integration of Large Language Models (LLMs) within the realm of sequential recommendation systems. It addresses the key challenges associated with traditional LLM-based recommendation methods, such as inefficiency, limited extensibility and the generation of out-of-range results. IDentityRec offers a robust, lightweight, scalable and efficient solution that is well-suited for practical applications in various domains, particularly in e-commerce. The model's innovative approach of learning collaborative patterns from item IDs and unifying them with rich semantic information from prompts ensures that it can deliver highly personalised recommendations that align with a user's long-term preferences as encoded in the ID embeddings, while also adapting to recent changes in user behaviour as described in a textual input.

The superior performance of the proposed model has been rigorously tested against foundational baseline methods by conducting a comparative analysis of evaluation metrics across four datasets. Extensive experiments and their empirical analysis have reinforced the superiority of our model in data-sparse scenarios and attested to its robust generalisation capabilities when adapting to unseen datasets and new domains. Ablation studies have underscored the effectiveness of incorporating item ID embeddings and justified the need for using an LLM which has been effectively instruction-tuned using domain-specific data to deliver unprecedented performance on the downstream tasks of next-item retrieval and ranking.

However, despite its promising results, IDentityRec is not without limitations. Particularly concerning is its dependency on pre-trained models and the lack of explainability in recommendations. Overall, it lays a strong foundation for future research in this field, offering valuable insights for advancing the state of LLM-based recommender systems.

Bibliography

[1] Gediminas Adomavicius and Alexander Tuzhilin. A survey of collaborative filtering techniques. In *IEEE Transactions on Knowledge and Data Engineering*, page 30–37, 2005.

[2] Trapit Bansal, David Belanger, Jacob Andreas, and Andrew McCallum. Learning to embed categorical features via meta-learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 10571–10581, 2018.

[3] Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. Tallrec: An effective and efficient tuning framework to align large language model with recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems, RecSys 2023, Singapore,*, pages ACM, 1007–1014, 2023.

[4] Justin Basilico and Thomas Hofmann. Unifying collaborative and content-based filtering. In *Proceedings of the twenty-first international conference on Machine learning*, 2004.

[5] James Bennett and Stan Lanning. The netflix prize. In *Proceedings of KDD Cup and Workshop*, 2007.

[6] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*, 2020.

[7] Jun Chen, Liang Zhang, Qi Xu, and Shuang Xie. A survey on recommender systems: Techniques, applications, and challenges. *IEEE Access*, 5:12–54, 2017.

[8] Qiwei Chen, Huan Zhao, Wei Li, Pipei Huang, and Wenwu Ou. Behavior sequence transformer for e-commerce recommendation in alibaba. In *Proceedings of the 1st International Workshop on Deep Learning Practice for High- Dimensional Sparse Data*, page 1–4, 2019.

[9] Aakanksha Chowdhery, Sharan Narang, and Jacob Devlin. Palm: Scaling language modeling with pathways. In *arXiv preprint arXiv:2204.02311*, 2022.

[10] Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In *RecSys '16: Proceedings of the 10th ACM Conference on Recommender Systems*, pages 191–198, 2016.

[11] Sunhao Dai, Ninglu Shao, Haiyuan Zhao, Weijie Yu, Zihua Si, Chen Xu, Zhongxiang Sun, Xiao Zhang, and Jun Xu. Uncovering chatgpt's capabilities in recommender systems. In *Proceedings of the 17th ACM Conference on Recommender Systems, RecSys 2023, Singapore, Singapore, September 18-22,*, page 1126–1132, 2023.

[12] Wenqi Fan, Zihuai Zhao, Jiatong Li, Yunqing Liu, Xiaowei Mei, Yiqi Wang, Jiliang Tang, and Qing Li. Recommender systems in the era of large language models (llms). In *CoRR abs/2307.02046*, 2023.

[13] Yunfan Gao, Tao Sheng, Youlin Xiang, Yun Xiong, Haofen Wang, and Jiawei Zhang. Chat-rec: Towards interactive and explainable llms-augmented recommender system. In *CoRR abs/2303.14524*, 2023.

[14] Shijie Geng, Shuchang Liu, Zuohui Fu, Yingqiang Ge, and Yongfeng Zhang. Recommendation as language processing (rlp): A unified pretrain, personalized prompt predict paradigm (p5). In *RecSys '22: Sixteenth ACM Conference on Recommender Systems, Seattle, WA, USA,*, page 299–315, 2022.

[15] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In Yee Whye Teh and Mike Titterington, editors, *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010. PMLR.

[16] Ruining He and Julian J. McAuley. Fusing similarity models with markov chains for sparse sequential recommendationfusing similarity models with markov chains for sparse sequential recommendation. In *IEEE 16th International Conference on Data Mining, ICDM 2016, December 12-15, Barcelona, Spain.*, page 191–200, 2016.

[17] Xiangnan He, Lizi Liao, Hanwang Zhang, Xia Hu Liqiang Nie, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*, page 173–182, 2017.

[18] Kristof Hidas, Hansi Zeng, and Xiangnan Liu. Transformers for recommendation: A new benchmark and analysis. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 1813–1822. ACM, 2019.

[19] Balázs Hidasi, Alexandros Karatzoglou, Linas Baltrunas, and Domonkos Tikk. Session-based recommendations with recurrent neural networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[20] Yupeng Hou, Jiacheng Li, Zhankui He, An Yan, Xiusi Chen, and Julian McAuley. Bridging language and items for retrieval and recommendation. *arXiv preprint arXiv:2403.03952*, 2024.

[21] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event*, page 299–315, 2022.

[22] Chris Meyer Ian MacKenzie and Steve Noble, 2013.

[23] Jianchao Ji, Zelong Li, Shuyuan Xu, Wenyue Hua, Yingqiang Ge, Juntao Tan, and Yongfeng Zhang. Genrec: Large language model for generative recommendation. In *CoRR abs/2307.00457*, 2023.

[24] Yitong Ji, Aixin Sun, Jie Zhang, and Chenliang Li. A critical study on data leakage in recommender system offline evaluation. *ACM Transactions on Information Systems*, 41(3):1–27, February 2023.

[25] Wang-Cheng Kang and Julian J. McAuley. Self-attentive sequential recommendation. In *IEEE International Conference on Data Mining, ICDM 2018, Singapore,*, page 197–206, 2018.

[26] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, page 197–206, 2015.

[27] Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. Large language models are zero-shot reasoners. In *NeurIPS*, 2022.

[28] Joseph A Konstan, Bradley N Miller, David Maltz, Jonathan L Herlocker, Lee R Gordon, and John Riedl. Grouplens: Applying collaborative filtering to usenet news. In *Commun. ACM 40, 3*, page 77–87, 1997.

[29] Yehuda Koren, Robert M. Bell, , and Chris Volinsky. Matrix factorization techniques for recommender systems. In *Computer 42, 8*, page 30–37, 2009.

[30] Ariel N. Lee, Cole J. Hunter, and Nataniel Ruiz. Platypus: Quick, cheap, and powerful refinement of llms. In *CoRR abs/2308.07317*, page 299–315, 2023.

[31] Brian Lester, Rami Al-Rfou, , and Noah Constant. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP,7-11 November, 2021, Association for Computational Linguistics*, page 3045–3059, 2022.

[32] Chuan Li, Yifan Li, and Xiaodong Liang. Cold-start problem in recommender systems. *Journal of Computer Science and Technology*, 33(5):832–845, 2018.

[33] Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. In *ACL/IJCNLP 2021, August 1-6, 2021. Association for Computational Linguistics,*, page 4582–4597, 2021.

[34] Junling Liu, Chao Liu, Renjie Lv, Kang Zhou, and Yan Zhang. Is chatgpt a good recommender? a preliminary study. In *CoRR abs/2304.10149*, 2023.

[35] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, page 30–37, 2019.

[36] Sourab Mangrulkar, Sylvain Gugger, Lysandre Debut, Younes Belkada, Sayak Paul, and Benjamin Bossan. Peft: State-of-the-art parameter-efficient fine-tuning methods. 2022.

[37] Julian McAuley and Yen Li. Personalized language model for recommendation. In *Proceedings of the 31st ACM International Conference on Information and Knowledge Management (CIKM)*, 2022.

[38] Julian J. McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, Santiago, Chile, August 9-13*, pages ACM, 43–52, 2015.

[39] Robin Van Meteren and Maarten Van Someren. Using content-based filtering for recommendation. In *Proceedings of the machine learning in the new information age: MLnet/ECML2000 workshop, Vol. 30. Barcelona*, page 47–56, 2000.

[40] Michael J Pazzani. A framework for collaborative, content-based and demographic filtering. In *Artificial intelligence review 13 (1999)*, page 393–408, 1999.

[41] Zhaopeng Qiu, Xian Wu, Jingyue Gao, and Wei Fan. U-bert: Pre-training user representations for improved recommendation. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9*, pages AAAI Press, 4320–4327, 2021.

[42] W. Kang R. He and J. McAuley. Translation-based recommendation. 2017.

[43] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. In *J. Mach.Learn. Res. 21 (2020)*, page 140:1–140:67, 2020.

[44] Steffen Rendle. Factorization machines. In *IEEE International Conference on Data Mining (2010)*, page 995–1000, 2010.

[45] Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. Factorizing personalized markov chains for next-basket recommendation. In *Proceedings*

of the 19th International Conference on World Wide Web, WWW 2010, Raleigh, North Carolina, USA, April 26-30, pages ACM, 811–820, 2010.

[46] Francesco Ricci, Lior Rokachand, and Bracha Shapira. Recommender systems handbook. In *Springer*, page 734–749, 2011.

[47] J Ben Schafer, Dan Frankowski, Jon Herlocker, and Shilad Sen. Collaborative filtering recommender systems. In *The adaptive web: methods and strategies of web personalization*, page 291–324, 2007.

[48] Philipp Schmid, Omar Sanseviero, Pedro Cuenca, and Lewis Tunstall, 2023.

[49] Jieun Son and Seoung Bum Kim. Content-based filtering for recommendation systems using multiattribute networks. In *Expert Systems with Applications 89*, page 404–412, 2017.

[50] Fei Sun, Jun Liu, and Jian Wu. Sequential recommendation with transformers. In *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, 2019.

[51] Fei Sun, Jun Liu, Jian Wu, et al. Recommender systems with pre-trained transformers: A transformer-based framework for sequential recommendation. *arXiv preprint arXiv:1904.06690*, 2019.

[52] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. Bert4rec: Sequential recommendation with bidirectional encoder representations from transformer. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China*, page 1441–1450, 2019.

[53] Weiwei Sun, Lingyong Yan, Xinyu Ma, Pengjie Ren, Dawei Yin, and Zhaochun Ren. Is chatgpt good at search? investigating large language models as re-ranking agent. In *CoRR abs/2304.09542*, 2023.

[54] Jiaxi Tang and Ke Wang. Personalized top-n sequential recommendation via convolutional sequence embedding. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, WSDM 2018, Marina Del Rey, CA, USA, February 5-9, 2018*, pages ACM, 565–573, 2018.

[55] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. 2023.

[56] Wilson L. Taylor. "cloze procedure": A new tool for measuring readability. In *Journalism Mass Communication Quarterly 30*, page 415 – 433, 1953.

[57] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurélien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. Llama: Open and efficient foundation language models. In *CoRR abs/2302.13971*, 2023.

[58] Eyal Trabelsi. Comprehensive guide to approximate nearest neighbors algorithms, 2020.

[59] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[60] Jason Wei, Maarten Bosma, Vincent Y. Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M. Dai, and Quoc V. Le. Finetuned language models are zero-shot learners. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29*, 2022.

[61] Likang Wu, Zhi Zheng, Zhaopeng Qiu, Hao Wang, Hongchao Gu, Tingjia Shen, Chuan Qin, Chen Zhu, Hengshu Zhu, Qi Liu, Hui Xiong, and Enhong Chen. Userbert: Pre-training user model with contrastive self-supervision. In *A Survey on Large Language Models for Recommendation*, 2023.

[62] Lu Yu, Shuo Zhang, Wenjie Wang, et al. A survey on context-aware recommender systems based on pre-trained language models. *arXiv preprint arXiv:2109.07328*, 2021.

[63] Jing Yao Xu Huang Defu Lian Xing Xie Yuxuan Lei, Jianxun Lian. Recexplainer: Aligning large language models for explaining recommendation models. 2023.

[64] Junjie Zhang, Ruobing Xie, Yupeng Hou, Wayne Xin Zhao, Leyu Lin, and Ji-Rong Wen. Recommendation as instruction following: A large language model

empowered recommendation approach. In *CoRR abs/2305.07001*, page 224–226, 2023.

[65] Yang Zhang, Zheng Zhang, Xue Yang, and Wenwen Chen. A review of recommender systems based on embeddings. *ACM Computing Surveys*, 53(6):1–35, 2020.

[66] Wayne Xin Zhao, Yupeng Hou, Xingyu Pan, Chen Yang, Zeyu Zhang, Zihan Lin, Jingsen Zhang, Shuqing Bian, Jiakai Tang, Wenqi Sun, Yushuo Chen, Lanling Xu, Gaowei Zhang, Zhen Tian, Changxin Tian, Shanlei Mu, Xinyan Fan, Xu Chen, and Ji-Rong Wen. Recbole 2.0: Towards a more up-to-date recommendation library, 2022.

[67] Kun Zhou, Hui Wang, Wayne Xin Zhao, Yutao Zhu, Sirui Wang, Fuzheng Zhang, Zhongyuan Wang, and Ji-Rong Wen. S3-rec: Self-supervised learning for sequential recommendation with mutual information maximization. In *CIKM*, page 1893–1902, 2020.

[68] Kun Zhou, Hui Yu, Wayne Xin Zhao, and Ji-Rong Wen. Filter-enhanced mlp is all you need for sequential recommendation. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France,*, page 2388–2399, 2022.

[69] Peilin Zhou, Jingqi Gao, Yueqi Xie, Qichen Ye, Yining Hua, Jaeboum Kim, Shoujin Wang, and Sunghun Kim. Equivariant contrastive learning for sequential recommendation. In *Proceedings of the 17th ACM Conference on Recommender Systems, RecSys, Singapore, September 18-22*, pages ACM, 129–140, 2023.

Appendix A

Algorithm of IDentityRec

Algorithm 1 IDentityRec solution

Require: Recommendation dataset $\mathcal{D}$, Instruction dataset $\mathcal{D}_{ins}$, Pre-trained LLM $\mathcal{M}_{LLM}$

Ensure: IDentityRec model $\mathcal{M}$

Train a sequential recommendation model $\mathcal{M}_{seq}$ on $\mathcal{D}$.

Extract item ID embeddings $\mathcal{E}$.

Instruction tune the LLM model $\mathcal{M}_{LLM}$ on $\mathcal{D}_{ins}$.

Prompt tune $\mathcal{M}_{LLM}$ on item ID embeddings and metadata using curriculum learning

// Model Training using pluggable components

Initialise Θ, input linear projection, LoRA weights and item linear projection

for T $\leftarrow 0$ **to** T_{max} iterations **do**

Sample an instance for training.

Obtain the corresponding ID embeddings from $\mathcal{E}$

Project ID embeddings into same dimension as $\mathcal{M}_{LLM}$ using input linear projection

Feed ID embeddings and prompt to $\mathcal{M}_{LLM}$ for output

Predict candidate items using item linear projection.

Compute the cross-entropy loss $\mathcal{L}$ and update Θ using $\mathcal{L}$.

end for

Deploy backbone model $\mathcal{M}_{LLM}$.

Deploy the model for dataset $\mathcal{D}$ with $\mathcal{M} = \mathcal{M}_{LLM} \leftarrow \mathcal{E}, \Theta$.

Appendix B

Description of modified LoRA Layers

Layer	Purpose served in the model	Benefit of Fine-Tuning the layer
gate_proj	Controls flow of information through the network	Adjusts extent to which injected ID embeddings and contextual information influence final representation
down_proj	Reduces the dimensionality of data before processing further	Optimises how the critical information is retained when compressing temporal user-item interactions
up_proj	Increases dimensionality back to the original size to process richer detailed information	Refines final output representations, reconstructing and highlighting important relationships in the data.
Attention Heads	Captures the relationships and dependencies in data	Helps model to focus more on relevant parts of the input, when combining item IDs with word embeddings

Table B: LoRA layers and Fine-Tuning Benefits

Appendix C

Baseline Methods and their Working Principle

Popularity (POP) is a heuristic method that recommends those items which have the highest interaction counts to all users, regardless of individual preferences. Bayesian Personalised Ranking (BPR) utilises matrix factorisation with a pairwise ranking loss to learn user and item embeddings, optimizing for the ranking of relevant items over irrelevant ones. GRU4Rec leverages Gated Recurrent Units to model sequential user behavior, predicting the next item in a sequence based on the user's previous interactions. SASRec uses a self-attention mechanism with multi-head attention to model complex sequential user behavior, focusing on the importance of different items in a user's interaction history for predicting the next item. BERT4Rec adapts the BERT transformer model for recommendation by using bidirectional self-attention with the Cloze objective to predict masked items in a sequence. CL4SRec combines contrastive learning with self-supervised pre-training to enhance sequential recommendation, by maximizing the agreement between positive sequence pairs and distinguishing them from negative pairs. ICLRec utilises a latent intent variable to capture the intent distribution of a user from unlabeled item sequences, optimizing for both item-level and sequence-level representations to improve the quality of transformer-based sequential recommendations through more robust user-item interaction modelling.

Appendix D

Application in E-Commerce

IDentityRec holds significant potential for revolutionizing recommendation systems within the e-commerce sector by providing more accurate, efficient and scalable solutions for personalised recommendations. In e-commerce, where vast amounts of user interaction data are generated daily, IDentityRec's ability to handle large-scale sequential recommendation tasks through efficient ID-based processing facilitates controllable generation of relevant product suggestions, thereby enhancing user experience and driving sales. The model's extensibility allows it to be easily integrated into existing e-commerce platforms without requiring extensive retraining, thus making it a practical choice for businesses looking to leverage cutting-edge recommendation technologies.

In e-commerce, user interactions such as browsing history, cart additions and purchases are often sequential. Extracting item IDs from a pre-trained, transformer-based sequential recommender model helps IDentityRec capture these patterns and generate personalised recommendations based on the user's recent interactions. Instruction-tuning the LLaMA language model on domain-specific data, like a large dataset of product descriptions, reviews and customer interactions specific to the e-commerce domain, renders IDentityRec the ability to understand product attributes and user preferences within the e-commerce context. Using such an instruction-tuned LLM in our model enables it to effectively capture patterns and nuances relevant to this domain.

E-commerce settings require handling massive datasets and generating real-time predictions. Optimizing the model's architecture for scalability by incorporating techniques like Low-Rank Adaptation reduces the computational load without sacrificing accuracy. Moreover, IDentityRec's capability to generate the entire ranking list in a single forward process, coupled with the use of pluggable parameters, ensures that the system can perform real-time inference while maintaining high levels of performance. By guar-

anteeing that the generated recommendations fall within a candidate list, IDentityRec minimises the risk of irrelevant or out-of-range recommendations, thereby improving customer satisfaction and loyalty. This makes the model particularly well-suited for large e-commerce platforms with extensive product catalogues and user-bases.

Recommender systems in e-commerce can impact purchasing decisions, so model interpretability assumes great importance. Providing clear explanations for recommendations to users enhances user trust and satisfaction. By carefully adapting the model to fit the e-commerce context, considering scalability and user-centric features, we enhance IDentityRec's effectiveness in driving personalised, relevant and trustworthy recommendations that meet the needs of online shoppers.