Data-efficient Fine-tuning for LLM-based Recommendation

Xinyu Lin[1], Wenjie Wang[1], Yongqi Li[2], Shuo Yang[3], Fuli Feng[4], Yinwei Wei[5], and Tat-Seng Chua[1]
[1]National University of Singapore, [2]The Hong Kong Polytechnic University, [3]University of Technology Sydney,
[4]University of Science and Technology of China, [5]Monash University
{xylin1028,wenjiewang96,liyongqi0}@gmail.com,shuo.yang@student.uts.edu.au
fulifeng93@gmail.com,weiyinwei@hotmail.com,dcscts@nus.edu.sg

arXiv:2401.17197v1 [cs.IR] 30 Jan 2024

ABSTRACT

Leveraging Large Language Models (LLMs) for recommendation has recently garnered considerable attention, where fine-tuning plays a key role in LLMs' adaptation. However, the cost of fine-tuning LLMs on rapidly expanding recommendation data limits their practical application. To address this challenge, few-shot fine-tuning offers a promising approach to quickly adapt LLMs to new recommendation data. We propose the task of data pruning for efficient LLM-based recommendation, aimed at identifying representative samples tailored for LLMs' few-shot fine-tuning. While coreset selection is closely related to the proposed task, existing coreset selection methods often rely on suboptimal heuristic metrics or entail costly optimization on large-scale recommendation data.

To tackle these issues, we introduce two primary objectives for the data pruning task in the context of LLM-based recommendation: 1) high accuracy aims to identify the influential samples that can lead to high overall performance; and 2) high efficiency underlines the low costs of the data pruning process. To pursue the two objectives, we propose a novel data pruning method incorporating two scores, namely influence score and effort score, to efficiently identify the influential samples. Particularly, the influence score is introduced to accurately estimate the influence of removing each sample on the overall performance. To achieve low costs of the data pruning process, we employ a small-sized surrogate model to replace LLMs to obtain the influence score. Considering the potential gap between the surrogate model and LLMs, we further propose an effort score to prioritize some hard samples specifically for LLMs. We instantiate the proposed method on two competitive LLM-based recommender models, and empirical results on three real-world datasets validate the effectiveness of our proposed method. In particular, the proposed method uses only 2% samples to surpass the full data fine-tuning, reducing time costs by 97%.

CCS CONCEPTS

• **Information systems** → **Recommender systems**.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
Conference'17, July 2017, Washington, DC, USA
© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

KEYWORDS

Data Pruning, LLM-based Recommendation, Efficient Fine-tuning

ACM Reference Format:
Xinyu Lin, Wenjie Wang, Yongqi Li, Shuo Yang, Fuli Feng, Yinwei Wei, and Tat-Seng Chua. 2024. Data-efficient Fine-tuning for LLM-based Recommendation. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

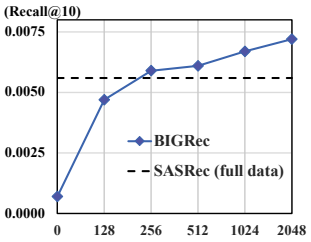
1 INTRODUCTION

Leveraging Large Language Models (LLMs) for recommendation has demonstrated promising efficacy across various tasks, including Click-Through Rate (CTR) prediction [4], sequential recommendation [40], and explainable recommendation [14]. To build LLM-based recommender models, it is crucial to fine-tune LLMs on recommendation data for two primary reasons: 1) there exists a significant gap between previous LLMs' tuning tasks and the recommendation tasks [4], and 2) the rapid and continuous update of recommendation data necessitates frequent fine-tuning of LLMs [41]. For example, there are approximately 160 million new videos and 942 billion interactions emerging on TikTok per day[1]. Thus, frequent fine-tuning is imperative to incorporate up-to-date item information and enhance user behavior comprehension. However, fine-tuning LLMs on large-scale recommendation data demands substantial computational resources and time costs [30], thereby diminishing the practicality of LLM-based recommender models in real-world applications. As such, it is essential to enhance the fine-tuning efficiency of LLM-based recommender models.

Fortunately, the rich world knowledge encoded in LLMs offers a promising solution for efficient fine-tuning: *few-shot fine-tuning*. Previous studies have uncovered that LLMs have the potential to quickly adapt to recommendation tasks by fine-tuning on randomly sampled few-shot data [3, 4, 32] (Figure 1(a)), significantly reducing training time and computational costs. Despite its efficiency, randomly sampled data may lack sufficient representativeness to enable LLMs to effectively comprehend new items and user behaviors. To combat this issue, we introduce the task of *data pruning for efficient LLM-based recommendation*, which aims to identify representative samples tailored for LLMs' few-shot fine-tuning.

A closely related literature to this data pruning task is coreset selection [16]. It tries to select a small but representative subset from the full data, aiming to achieve comparable performance. Existing coreset selection methods generally fall into two categories[2]: 1) Heuristic methods select hard or diverse samples based on predefined metrics [36, 39, 53]. Such heuristic methods do not estimate

[1]https://www.tiktok.com/transparency/.
[2]More detailed related work is discussed and compared in Section 4 and 5.



(a) Few-shot performance on MicroLens-50K.

Model	GPU (GiB)	Time
BIGRec	18.60 × 4GPU	36.87h
SASRec	1.61 × 1GPU	0.45h
% Red.	**97.84%**	**98.78%**

(b) The comparison of training costs between an LLM (BIGRec) and a surrogate model (SASRec). The statistics are based on NVIDIA RTX A5000 on Games.

Figure 1: (a) reveals that BIGRec achieves remarkable performance with only hundreds of samples. (b) shows the low costs of surrogate models.

the impact of selected samples on empirical risk, possibly leading to suboptimal coreset selection. 2) Optimization-based methods mainly optimize the selection of subsets to minimize the empirical risk [5, 55]. However, these methods are inapplicable to large-scale recommendation datasets due to the complex and costly bi-level or discrete optimization problem [20]. Worse still, both heuristic and optimization-based methods rely on the model well-trained by the full data to select the coreset, *e.g.,* calculating pre-defined scores or optimizing the data subset based on the well-trained model (*cf.* Section 2). As such, it is infeasible to directly apply these methods for LLM-based recommendation because of the high training costs of LLMs on the large-scale full recommendation data.

To overcome the above issues, we summarize two principal objectives for data pruning in the context of LLM-based recommendation: 1) high accuracy, which focuses on selecting the samples that can lead to low empirical risk; and 2) high efficiency, which emphasizes the low costs of the data pruning process, *i.e.,* eliminating the dependency of well-trained LLMs on the full data. Nevertheless, pursuing the two objectives faces two challenges:

- To achieve high accuracy, it is essential to measure the influence of removing each training sample on the empirical risk. However, assessing the influence of all samples is costly, as it requires the leaving-one-out retraining for each sample [46].
- To achieve high efficiency, one possible solution is to train a surrogate model for sample selection, *e.g.,* using a small-sized traditional recommender model, which can drastically reduce the GPU memory usage and the training time compared to LLMs (see Figure 1(b)). However, there exists a gap between LLMs and surrogate models, attributable to their divergent capabilities in learning user behaviors (refer to Figure 3). As such, influential samples selected by surrogate models might deviate from the ones on LLMs, potentially hurting the adaptation of LLMs.

To address the challenges, we propose a novel **D**ata pruning method, to **E**fficiently identify the influenti**A**l samples for **L**LM-based **Rec**ommender fine-tuning (shorted as DEALRec). DEALRec leverages two scores, namely influence score and effort score, to identify the influential samples. The ***influence score*** is formulated to estimate the influence of removing each training sample on the empirical risk. It is calculated by extending the influence function [18] via chain rules and second-order optimization techniques [28]. To efficiently calculate the influence score for all samples, DEALRec employs a simple yet effective symmetric property to accelerate the calculation, requiring only the estimation *once* for all samples (*cf.* Section 3.1). Thereafter, DEALRec uses a traditional recommender model as a surrogate model to obtain the influence score and introduces the ***effort score*** to mitigate the gap between the surrogate model and LLMs. The effort score is obtained by calculating the gradient norm of a sample loss *w.r.t.* the parameters of LLMs, intuitively measuring the effort of LLMs to fit a specific sample. By regularizing the influence score with the effort score, DEALRec identifies the influential samples that encompass both the representativeness of the full data and the significance to LLMs. We instantiate DEALRec on two LLM-based recommender models and conduct extensive experiments on three real-world datasets, validating the superiority of DEALRec in terms of both efficiency and accuracy.

In summary, this work offers three major contributions:

- We introduce a data pruning task to identify the influential samples tailored for efficient LLM-based recommender fine-tuning, unlocking the remarkable potential of applying LLM-based recommender models to real-world platforms.
- We propose a novel data pruning method to discover the influential samples for LLM-based recommendation, which effectively and efficiently assesses the influence of removing a sample on empirical risk.
- We conduct extensive experiments on three real-world datasets, demonstrating the effectiveness of DEALRec in achieving both high efficiency and accuracy.

2 TASK FORMULATION

In this section, we first introduce LLM-based recommender models and uncover the challenge of real-world applicability. Thereafter, we formulate the task of data pruning for LLM-based recommendation and compare the related work on coreset selection.

- **LLM-based recommender models.** To leverage the competent capabilities of LLMs, LLM-based recommendation typically utilize powerful LLMs directly as the recommender models. Since LLMs are not particularly trained on the recommendation data, fine-tuning is the necessary and key step for LLMs to learn the item knowledge and understand user behavior. Let $\mathcal{U}$ and $\mathcal{I}$ denote the sets of users and items, respectively. We present each training sample, *i.e.,* user sequence, as $s = (x, y)$, where $x = [i_1, i_2, \ldots, i_{|x|}]$ is the user's historical interactions in chronological order, and y is the next interacted item of the user[3], where $\{i_1, \ldots, i_{|x|}, y\} \subset \mathcal{I}$. Formally, given the user sequences of the training set $\mathcal{D} = \{s_u | u \in \mathcal{U}\}$, the target is to fine-tune an LLM for recommendation tasks. The learnable parameters ($\phi \in \Phi$) of an LLM is optimized by minimizing the negative log-likelihood of the next interacted item y conditioned on input x:

$$\min_{\phi \in \Phi} \{\mathcal{L}_{\phi}^{LLM} = -\sum_{t=1}^{|y|} \log P_{\phi}(y_t | y_{<t}, x)\}, \tag{1}$$

where y_t denotes the t-th token of y, and $y_{<t}$ represents the token sequence preceding y_t.

[3]Our main focus lies in sequential recommendation, which holds notable practical significance by intricately considering the temporal aspect in real-world scenarios.

While fine-tuning LLMs has demonstrated effectiveness in recommendation tasks [35], its practical application is hindered by the high resource costs required by LLMs and the continuous influx of new recommendation data [41]. Hence, it is essential to enhance the efficiency of LLM-based recommender fine-tuning.

- **Data pruning for efficient LLM-based recommendation.** To achieve efficient LLM-based recommendation, a promising approach is to reduce the costs by few-shot fine-tuning with randomly selected samples [4]. Nevertheless, the random samples might lose some crucial information for LLMs to acquire the latest information on user behavior or items, *e.g.,* trending items. In this light, we introduce the task of data pruning for efficient LLM-based recommendation, which aims to identify a set of representative samples particularly for LLMs' few-shot fine-tuning. Formally, given all training samples $\mathcal{D} = \{s_u | u \in \mathcal{U}\}$, the target of data pruning is to select a subset $\mathcal{S} \subset \mathcal{D}$, such that the LLMs trained on the subset $\mathcal{S}$ can yield good performance on the testing set. The size of $\mathcal{S}$ is controlled by the given selection ratio r, *i.e.,* $|\mathcal{S}| = r|\mathcal{D}|$.
- **Retrospect of coreset selection.** As the closely related work to this data pruning task, coreset selection methods generally fall into two groups:

1) *Heuristic* methods [8, 12, 47] typically design some heuristic strategies to select samples based on an empirical minimizer:

$$\mathcal{S} = H(\hat{\theta}, \mathcal{D}), \quad \text{s.t.} \quad \hat{\theta} = \arg\min_{\theta \in \Theta} \mathcal{L}(\theta, \mathcal{D}), \tag{2}$$

where $\mathcal{L}(\cdot)$ is the loss function of the task, *e.g.,* image classification [19] or CTR prediction [17], and $H(\cdot)$ denotes the heuristic strategy such as selecting samples with larger prediction entropy [8], or clustering the samples based on the sample representations [6]. However, this group of methods designs the strategy $H(\cdot)$ intuitively and fails to explicitly consider the influence of a sample on the empirical risk. This might lead to suboptimal selection, thereby declining the performance of the model trained by the selected subset.

2) *Optimization-based* methods [5, 26, 27, 52] mainly utilize bi-level optimization techniques to learn the best subset chosen for training:

$$\mathcal{S}^* = \arg\min_{\mathcal{S} \subset \mathcal{D}} \mathcal{L}(\hat{\theta}, \mathcal{D}), \quad \text{s.t.} \quad \hat{\theta} = \arg\min_{\theta \in \Theta} \mathcal{L}(\theta, \mathcal{S}). \tag{3}$$

Besides, there is also some work that employs discrete optimization problems based on the empirical minimizer $\hat{\theta}$ in Eq. (2). Nevertheless, they struggle to be applied to large-scale datasets *e.g.,* recommendation data, due to the complex solving of the optimization problem [20].

Furthermore, as shown in Eq. (2-3), previous coreset selection methods usually require the model to be trained over original training samples $\mathcal{D}$, which however is infeasible for LLM-based recommender models due to the continuous influx of data and the high resource costs of LLMs (*cf.* Section 1).

- Drawing upon the above insights, we consider two objectives for data pruning: 1) ***high accuracy*** emphasizes the low empirical risk of the model trained on the selected samples, and 2) ***high efficiency*** focuses on the low costs of the data pruning process, breaking free from the heavy fine-tuning of LLMs for data pruning.

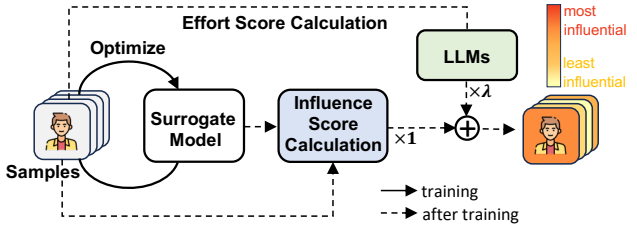


Figure 2: Overview of DEALRec. DEALRec first trains a surrogate model on the full training samples. Subsequently, it calculates the influence score, which is then regularized by the effort score, to identify influential samples.

3 DEALREC

To pursue efficient LLM-based recommendation, we propose a novel data pruning method DEALRec, which involves two key components, *i.e.,* the *influence score* to estimate the influence on empirical risk, and the *effort score* as a regularization to mitigate the gap between surrogate model and LLMs. The overview of our method is presented in Figure 2.

3.1 Influence Score

To achieve good overall performance with the model trained on the pruned dataset $\mathcal{S}$, the key lies in the ability to assess the influence on the empirical risk, *i.e.,* overall performance, caused by removing a sample in training. However, simply assessing the the influence by removing each sample is impractical, because it requires brute force leaving-one-out-retraining for $n = |\mathcal{D}|$ times. To overcome this challenge, we propose an efficient approximation of the influence for all samples by extending influence on parameter change (*i.e.,* a classic result from influence function [28]) via chain rule and second-order optimization techniques. We further utilize the symmetric property to speed up the calculation of the influence score.

- **Influence on parameter change.** To estimate the influence on empirical risk for each sample, we first start with the classic result [33] from research on influence function [9], which gives us the estimation of the parameter change caused by upweighting a sample s for training. Considering a training sample s is upweighted by a small ϵ, the empirical minimizer can be rewritten as:

$$\hat{\theta}_{\epsilon,s} = \arg\min_{\theta \in \Theta} \frac{1}{n} \sum_{s_i \in \mathcal{D}} \mathcal{L}(s_i, \theta) + \epsilon \mathcal{L}(s, \theta). \tag{4}$$

According to [33], the influence of upweighting a sample s on the parameter change is then given as:

$$\mathcal{I}_{\text{param}}(s) = \left. \frac{\mathrm{d}\hat{\theta}_{\epsilon,s}}{\mathrm{d}\epsilon} \right|_{\epsilon=0} = -H_{\hat{\theta}}^{-1} \nabla_\theta \mathcal{L}(s, \hat{\theta}), \tag{5}$$

where $H_{\hat{\theta}} = \frac{1}{n} \sum_{s_i \in \mathcal{D}} \nabla_\theta^2 \mathcal{L}(s_i, \hat{\theta})$ is the Hessian and positive definite by assumption, $\mathcal{I}_{\text{param}}(s) \in \mathbb{R}^m$, and m is the number of parameters. Notably, assigning $-\frac{1}{n}$ to ϵ is equivalent to removing the sample s from training. As such, the parameter change of removing a training sample s can be linearly approximated as:

$$\hat{\theta}_{-s} - \hat{\theta} \approx -\frac{1}{n} \mathcal{I}_{\text{param}}(s) = \frac{1}{n} H_{\hat{\theta}}^{-1} \nabla_\theta \mathcal{L}(s, \hat{\theta}), \tag{6}$$

where $\hat{\theta}_{-s} = \arg\min_{\theta \in \Theta} \sum_{s_i \in \mathcal{D}, s_i \neq s} \mathcal{L}(s_i, \theta)$.

Algorithm 1 Procedure of HVP Estimation

Input: Original training dataset $\mathcal{D}$, parameters of a well-trained model $\hat{\theta}$, iteration number T.

1: Compute $\sum_i \frac{1}{n}\nabla_\theta \mathcal{L}(s_i,\hat{\theta})$ for $\forall i \in \{1,\dots,n\}$.
2: Initialize $\tilde{H}_0^{-1}\left[\sum_i \frac{1}{n}\nabla_\theta \mathcal{L}(s_i,\hat{\theta})\right] = \sum_i \frac{1}{n}\nabla_\theta \mathcal{L}(s_i,\hat{\theta})$.
3: **for all** $t \in \{1,\dots,T\}$ **do**
4: Randomly sample a training sample $s_t \in \mathcal{D}$;
5: Calculate $\nabla^2_\theta \mathcal{L}(s_t)$ as the unbiased estimator of H;
6: $\tilde{H}_t^{-1}\left[\sum_i \frac{1}{n}\nabla_\theta \mathcal{L}(s_i,\hat{\theta})\right] \leftarrow \sum_i \frac{1}{n}\nabla_\theta \mathcal{L}(s_i,\hat{\theta}) +$
$\left[I - \nabla^2_\theta \mathcal{L}(s_t)\right]\tilde{H}_{t-1}^{-1}\left[\sum_i \frac{1}{n}\nabla_\theta \mathcal{L}(s_i,\hat{\theta})\right]$; ▷ Eq. (10)
7: $\tilde{H}^{-1}\left[\sum_i \frac{1}{n}\nabla_\theta \mathcal{L}(s_i,\hat{\theta})\right] \leftarrow \tilde{H}_T^{-1}\left[\sum_i \frac{1}{n}\nabla_\theta \mathcal{L}(s_i,\hat{\theta})\right]$.

Output: Unbiased estimation $\tilde{H}^{-1}\left[\sum_i \frac{1}{n}\nabla_\theta \mathcal{L}(s_i,\hat{\theta})\right]$.

Based on Eq. (6), an intuitive approach to assess the sample influence for model training is to utilize the L2 norm of a sample's influence on parameter change or an additional discrete optimization problem as proposed in [55]. Nevertheless, large parameter changes do not necessarily lead to performance improvements. Besides, calculating Eq. (6) for all training samples can be computationally costly [20] and is infeasible for recommendation data. To alleviate the issues, we propose an efficient approximation for the influence of removing a sample on the empirical risk.

- **Influence on empirical risk.** Based on the parameter change obtained via the influence function, we can then estimate the influence of upweighting a training sample s by a small ϵ on the loss of an arbitrary sample s':

$$\begin{aligned}\mathcal{I}_{\text{upweight,loss}}(s,s') &\overset{\text{def}}{=} \frac{\mathrm{d}\mathcal{L}(s',\hat{\theta}_{\epsilon,s})}{\mathrm{d}\epsilon}\Big|_{\epsilon=0} \\ &= \nabla_\theta \mathcal{L}(s',\hat{\theta})^{\mathrm{T}} \frac{\mathrm{d}\hat{\theta}_{\epsilon,s}}{\mathrm{d}\epsilon}\Big|_{\epsilon=0} \quad \text{(chain rule)} \\ &= -\nabla_\theta \mathcal{L}(s',\hat{\theta})^{\mathrm{T}} H_{\hat{\theta}}^{-1} \nabla_\theta \mathcal{L}(s,\hat{\theta}).\end{aligned} \tag{7}$$

Similarly, the influence of removing a training sample s on the loss of an arbitrary sample s' can be linearly approximated as:

$$\mathcal{I}_{\text{remove, loss}}(s,s') = \frac{1}{n}\nabla_\theta \mathcal{L}(s',\hat{\theta})^{\mathrm{T}} H_{\hat{\theta}}^{-1} \nabla_\theta \mathcal{L}(s,\hat{\theta}). \tag{8}$$

We can then obtain the influence of removing a sample s on the empirical risk (*i.e.,* influence score) by

$$\begin{aligned}\mathcal{I}_{\text{remove,loss}}(s,\mathcal{D}) &= -\frac{1}{n}\frac{\mathrm{d}\left(\frac{1}{n}\sum_i \mathcal{L}(s',\hat{\theta}_{\epsilon,s_i})\right)}{\mathrm{d}\epsilon}\Big|_{\epsilon=0} \\ &= \underbrace{\frac{1}{n}\sum_i \frac{1}{n}\nabla_\theta \mathcal{L}(s_i,\hat{\theta})^{\mathrm{T}} H_{\hat{\theta}}^{-1} \nabla_\theta \mathcal{L}(s,\hat{\theta})}_{\textbf{(influence score)}}.\end{aligned} \tag{9}$$

However, it is non-trivial to directly obtain $H_{\hat{\theta}}^{-1}$ as forming and inverting $H_{\hat{\theta}} = \frac{1}{n}\sum_{s_i\in\mathcal{D}} \nabla^2_\theta \mathcal{L}(s_i,\hat{\theta})$ requires $O(nm^2 + m^3)$ with n training samples and $\theta \in \mathbb{R}^m$. This results in cumbersome calculation of influence scores for all training samples.

- **Efficient estimation of influence score.** To achieve efficient computation of influence score, we utilize stochastic-based Hessian-Vector Products (HVP) [1] to efficiently approximate $H_{\hat{\theta}}^{-1}\nabla_\theta \mathcal{L}(s,\hat{\theta})$.

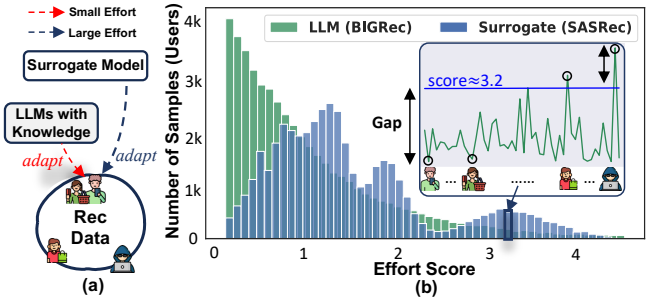


Figure 3: (a) depicts the different learning ability due to the prior knowledge in LLMs. (b) presents the distributions of effort scores of LLM and surrogate model on Games dataset[4].

The idea of stochastic-based HVP estimation is to iteratively obtain an unbiased estimator of $H_{\hat{\theta}}$ and approach the unbiased estimation of HVP, *i.e.,* $H_{\hat{\theta}}^{-1}\nabla_\theta \mathcal{L}(s,\hat{\theta})$. Specifically, we omit the $\hat{\theta}$ subscript for clarity and write the first j terms in Taylor expansion of H^{-1} as $H_j^{-1} \overset{\text{def}}{=} \sum_{i=0}^{j}(I-H)^i$, which can be further rewritten recursively as $H_j^{-1} = I + (I-H)H_{j-1}^{-1}$. From the validity of the Taylor expansion, we have $H_j^{-1} \to H^{-1}$ as $j \to \infty$. Thereafter, denoting $\nabla_\theta \mathcal{L}(s,\hat{\theta})$ as v, the update iteration for the estimated $H_{\hat{\theta}}^{-1}\nabla_\theta \mathcal{L}(s,\hat{\theta})$ at step t can be written as:

$$\tilde{H}_t^{-1} v = v + \left(I - \nabla^2_\theta \mathcal{L}(s_t)\right)\tilde{H}_{t-1}^{-1} v, \tag{10}$$

where s_t is a training sample randomly drawn from $\mathcal{D}$, and $\nabla^2_\theta \mathcal{L}(s_t)$ is an unbiased estimator of the H at step t for fast-to-compute HVP [28]. Despite that stochastic-based HVP can alleviate the computation burdens of the estimation, calculating the influence score for each sample is still costly due to the independent n estimations of $H_{\hat{\theta}}^{-1}\nabla_\theta \mathcal{L}(s,\hat{\theta})$ for each $s \in \mathcal{D}$ (refer to Eq. (9)).

To further enhance the efficiency of acquiring influence scores for all samples, we use symmetric property to rewrite Eq. (9) into:

$$\mathcal{I}_{\text{remove,loss}}(s,\mathcal{D}) = \frac{1}{n}\nabla_\theta \mathcal{L}(s,\hat{\theta})^{\mathrm{T}} \underbrace{H_{\hat{\theta}}^{-1}\left[\sum_i \frac{1}{n}\nabla_\theta \mathcal{L}(s_i,\hat{\theta})\right]}_{\textbf{(constant vector)}}. \tag{11}$$

The reformulation is based on the assumption that $\mathcal{L}(\cdot)$ has continuous second-order derivatives, which is consistent with the assumption for influence function [28], leading to the fact that $H_{\hat{\theta}}^{-1}$ is symmetric. Since $H_{\hat{\theta}}^{-1}\left[\sum_i \frac{1}{n}\nabla_\theta \mathcal{L}(s_i,\hat{\theta})\right] \in \mathbb{R}^m$ is a constant vector for any sample $s \in \mathcal{D}$, we can efficiently obtain influence scores for all samples by only applying HVP estimation **once** for $H_{\hat{\theta}}^{-1}\left[\sum_i \frac{1}{n}\nabla_\theta \mathcal{L}(s_i,\hat{\theta})\right]$. The detailed HVP estimation process is illustrated in Algorithm 1.

3.2 Gap Regularization

As shown in Eq. (11), assessing the influence score of a sample requires the optimized parameters $\hat{\theta}$ well-trained over all training samples $\mathcal{D}$. Nevertheless, this poses challenges for LLM-based

[4]We obtain the effort scores for surrogate model by calculating the gradient norm of the parameters of the surrogate model (Eq. (12)).

Algorithm 2 Procedure of DEALRec

Input: Original training dataset $\mathcal{D}$, randomly initialized parameters of surrogate model θ, pre-trained parameters of LLM ϕ.

$\hat{\theta} = \arg\min_{\theta\in\Theta} \frac{1}{n} \sum_{s_i\in\mathcal{D}} \mathcal{L}(s_i, \theta)$.

Obtain estimated $H^{-1}\left[\sum_i \frac{1}{n}\nabla_\theta \mathcal{L}(s_i, \hat{\theta})\right]$ via HVP estimation.

for all $i \in \{1, \ldots, n\}$ **do**

  $I_{s_i} = \frac{1}{n^2}\nabla_\theta \mathcal{L}(s_i, \hat{\theta})^{\mathrm{T}} H_{\hat{\theta}}^{-1}\left[\sum_j \frac{1}{n}\nabla_\theta \mathcal{L}(s_j, \hat{\theta})\right] + \lambda\|\nabla_\phi \mathcal{L}^{LLM}(s_i)\|_2$; ▷ Eq. (13)

$\mathcal{G} = \{G_1, \ldots, G_K\} \leftarrow$ Split training samples $\mathcal{D}$ into K groups according to the final score I_s with even range width.

$\mathcal{S} \leftarrow \emptyset$, $B \leftarrow \lfloor \frac{r|\mathcal{D}|}{K} \rfloor$.

while $\mathcal{G} \neq \emptyset$ **do**

  $k^* = \arg\min_k |G_k|$;

  $\mathcal{S}_{k^*} \leftarrow$ randomly select $\min\{B, |G_{k^*}|\}$ samples from G_{k^*};

  $\mathcal{S} \leftarrow \mathcal{S} \cup \mathcal{S}_{k^*}$; $\mathcal{G} \leftarrow \mathcal{G} \setminus \{G_{k^*}\}$;

  $B \leftarrow \lfloor \frac{r|\mathcal{D}|-|\mathcal{S}|}{|\mathcal{G}|} \rfloor$; ▷ Update sampling budget

Output: Selected samples $\mathcal{S}$ for few-shot fine-tuning.

recommender models due to the continuous influx of large-scale new data in real-world scenarios. In this light, we propose to utilize a surrogate model to replace the LLMs and introduce an ***effort score*** as a gap regularization to complement the learning ability gap between LLMs and the surrogate models.

- **Surrogate model.** To reduce the costs, we propose utilizing a surrogate model, *e.g.,* a small-sized traditional recommender model, to compute the influence scores. Nevertheless, since LLMs acquire rich world knowledge during the pre-training stage, they intricately possess different learning abilities compared to the surrogate model (Figure 3(a)). Therefore, the influential samples on LLMs might deviate from the ones for LLMs.

- **Effort score.** To compensate for the gap, we introduce the effort score, which aims to capture significant samples particularly for LLMs. Specifically, we define the effort score of a sample, *i.e.,* a user sequence, s as:

$$\delta_s = \|\nabla_\phi \mathcal{L}^{LLM}(s)\|_2, \tag{12}$$

where ϕ is the learnable parameters of LLMs[5]. Intuitively, it measures the learning effort of LLMs to fit a specific user sequence, and a larger score indicates a harder sample for LLMs to learn. To elaborate, Eq. (12) measures the change in the model parameters, which can be interpreted as the discrepancy from the current knowledge encoded in LLMs' parameters to the latest item knowledge or user behavior. As such, the effort score can emphasize significant samples particularly for LLMs, supplementing the different learning ability of the surrogate model (Figure 3(b)).

- **Overall score.** By injecting the signals of LLMs' learning ability into the calculation of influence score, we can obtain the final score of each user sequence for LLM-based recommender fine-tuning:

$$I_s = \underbrace{\frac{1}{n^2}\nabla_\theta \mathcal{L}(s, \hat{\theta})^{\mathrm{T}} H_{\hat{\theta}}^{-1}\left[\sum_i \frac{1}{n}\nabla_\theta \mathcal{L}(s_i, \hat{\theta})\right]}_{(\textbf{influence score})} + \underbrace{\lambda\|\nabla_\phi \mathcal{L}^{LLM}(s)\|_2}_{(\textbf{effort score})}, \tag{13}$$

[5]The learnable parameters can be either the whole parameters of LLMs or the learnable parameters from parameter-efficient training, *e.g.,* LoRA [23].

where λ is a hyper-parameter to balance the strength of the gap regularization. Notably, the gap regularization would suppress the easy samples with smaller effort scores while emphasizing the samples that are more difficult to learn, *i.e.,* larger effort scores.

Intuitively, DEALRec identifies the influential samples with two key considerations: 1) the influence score focuses on selecting the representative samples from the full dataset, capturing collaborative filtering information for low empirical risk; and 2) the effort score highlights the non-trivial samples that are significant to the learning of LLMs. The effectiveness of the two scores is empirically validated in Section 4.3.1.

3.3 Few-shot Fine-tuning

Based on the final influential score obtained via Eq. (13), we can select a subset of data $\mathcal{S}$ for LLMs' few-shot fine-tuning, given an expected selection ratio r.

- **Few-shot data coverage.** A straightforward approach is to select the data greedily, *i.e.,* rank the samples based on the overall scores, and then select the top-r percentage of the training data. However, greedily selecting the samples with higher scores might result in very similar samples with low data coverage, which leads to: 1) Inadequacy of samples from other areas, thus hurting the bounded empirical risk [60] and lowering the overall performance (*cf.* Section 4.2). 2) Poor utilization of training samples because of the redundant samples with similar patterns, thereby causing suboptimal selection for few-shot fine-tuning.

- **Coverage-enhanced sample selection.** To address the above issues, we follow [60] to select the users based on the idea of stratified sampling. The core idea is to maintain the budget for the samples in different areas of training distribution, such that the data coverage will be improved to ensure a high-probability bound for the empirical risk (refer to [60] for detailed proof). In detail, we first divide the samples into K groups according to their overall scores. We then iteratively sample n_s user sequences from the group with the fewest samples and discard that group after sampling, where n_s is the average sampling budget for all groups and is initialized with $\lfloor \frac{r|\mathcal{D}|}{K} \rfloor$. If the group size is smaller than the average sampling budget, we select all users from this group and update the average sampling budget for the remaining groups (see Algorithm 2).

Based on the selected few-shot samples $\mathcal{S}$, we optimize the learnable parameters ($\phi \in \Phi$) of LLMs:

$$\hat{\phi} = \arg\min_{\phi\in\Phi} \frac{1}{|\mathcal{S}|} \sum_{s_i\in\mathcal{S}} \mathcal{L}_\phi^{\mathrm{LLM}}(s_i). \tag{14}$$

- **Instantiation.** To instantiate DEALRec on LLM-based recommender models, we first employ a surrogate model to train on original training samples $\mathcal{D}$ and calculate the influence score for all samples via Eq. (11). We then obtain the effort score for LLMs via Eq. (12), where ϕ can be the learnable parameters from any backend LLM-based recommender models. Eventually, we apply the stratified sampling to select the samples for LLMs' few-shot fine-tuning. The detailed data pruning process of DEALRec is demonstrated in Algorithm 2.

Table 1: Statistics of the three datasets.

Datasets	# Users	# Items	# Interactions	Density
Games	49,156	17,332	342,329	0.04%
MicroLens-50K	49,887	19,217	359,048	0.04%
Book	88,263	86,272	5,303,707	0.07%

4 EXPERIMENT

We conduct extensive experiments on three real-world datasets to answer the following research questions:

- **RQ1:** How does our proposed DEALRec perform compared to the coreset selection baselines for LLM-based recommendation and the models trained with full data?
- **RQ2:** How do the different components of DEALRec (*i.e.,* influence score, gap regularization, and stratified sampling) affect the performance, and is DEALRec generalizable to different surrogate models?
- **RQ3:** How does DEALRec perform under different selection ratios and how does DEALRec improve the overall performance?

4.1 Experimental Settings

4.1.1 ***Datasets.*** We conduct experiments on three real-world recommendation datasets across different domains: 1) **Games** is from the Amazon review datasets[6], which covers interactions between users and video games with rich textual features such as title and categories. 2) **MicroLens-50K**[7] is a newly released micro-video recommendation dataset [38]. It contains 50k users' interactions with micro-videos and their associated multimodal features, *e.g.,* item title. 3) **Book** is also from Amazon review datasets, containing users' interactions with extensive books. For Games and Book, we follow previous work and discard the interactions with the ratings < 4. For the three datasets, we sort all user-item interactions according to the global timestamps, and then split the interactions into training, validation, and testing sets with the ratio of 8:1:1. Besides, we consider two different fine-tuning settings as follows: 1) *Few-shot fine-tuning* fine-tunes LLM-based recommender models with limited samples at a fixed size, *e.g.,* 1024-shot, obtained via different data pruning methods. 2) *Full fine-tuning* utilizes all samples to fine-tune LLM-based recommender models without data pruning.

4.1.2 ***Baselines.*** We compare DEALRec with the random sampling and several competitive coreset selection methods, including difficulty-based methods and diversity-based methods:

- **Random** obtains the data subset via random sampling, which is a popular and strong baseline in data-efficient training [16].
- **GraNd** [39] is a representative coreset selection method that assumes hard samples are important. It uses the average gradient norm of each sample during training and selects the samples with larger gradient norms.
- **EL2N** [39] proposes to select the samples with larger errors between the labels and the prediction from the model trained by the original dataset. This method is also difficulty-based.
- **CCS** [60] is a competitive method that selects the samples considering both high data coverage and sample importance. We use EL2N as the importance metric for CCS.
- **TF-DCon** [53] is a recently proposed data pruning method for content-based recommendation, which clusters the user sequences based on the user representations obtained from both well-trained recommender models and LLMs for selection.
- **RecRanker** [36] proposes a sampling strategy to select high-quality user sequences. It selects the users with more interactions for better user modeling and utilizes a cluster-based sampling strategy to enhance user diversity.

We do not perform optimization-based methods for comparison because of the inapplicability of complex bi-level or discrete optimization for LLMs on large-scale recommendation data (*cf.* Section 2). We instantiate our proposed DEALRec and all baselines on two competitive backend LLM-based recommender models: 1) *BIGRec* [3] utilizes the item title to present the user sequence for recommendation generation; 2) *TIGER* [40] learns extra tokens from item features to present items, and then converts the user sequence into the sequence of the new item token for next-item generation.

- **Evaluation.** We employ the widely used metrics Recall@K and NDCG@K to evaluate the models [22], with K set to 10 and 20 for Games, and $K = 20$ and 50 for MicroLens-50K and Book.[8] For the two backend LLM-based recommender models, we adopt full ranking protocal [21] for BIGRec. Since TIGER does not support full ranking, we select the top-K items from the generated items according to the probability scores for evaluation.

4.1.3 ***Implementation.*** As for the two backend LLM-based recommender models, we follow the original settings in their paper for implementation. We employ LLaMA-7B for BIGRec and transformer-based encoder-decoder architecture for TIGER as introduced in their paper [40]. All fine-tuning experiments are conducted on four NVIDIA RTX A5000 GPUs. Besides, we adopt the parameter-efficient fine-tuning technique LoRA [23] to fine-tune BIGRec and fully fine-tune the parameters of TIGER. We utilize SASRec [24], a representative sequential recommender model, as the surrogate model in DEALRec. We set the iteration number T for HVP estimation at 5000, and search the regularization strength λ in $\{0.1, 0.3, 0.5, 1.0, 2.0\}$. For cluster-based methods, the number of clusters K is explored in $\{25, 50, 75\}$. As for the coreset selection methods that require the training of LLMs, we consider a feasible implementation [8] by executing them on the same surrogate model as DEALRec.

4.2 Overall Performance (RQ1)

The results of the baselines and DEALRec with two competitive backend LLM-based recommender models on three datasets under few-shot fine-tuning (1024 samples) are presented in Table 2, from which we have the following observations:

- All methods with BIGRec typically yield better performance than those with TIGER, which is attributed to two reasons: 1) BIGRec employs a larger LLM (*i.e.,* LLaMA-7B) compared to TIGER,

[6]https://jmcauley.ucsd.edu/data/amazon/.
[7]https://github.com/westlake-repl/MicroLens/.
[8]We report metrics@20 and @50 because of the challenging modeling of user behavior on book and micro-video recommendations, where the temporal shifts of user interests and the item feature is stronger and thus more difficult to capture [49, 50].

Table 2: Overall performance comparison between the baselines and DEALRec instantiated on two competitive LLM-based recommender models on three datasets. For each backend model, the bold results highlight the best results while the second-best ones are underlined. $*$ implies the improvements over the second-best results are statistically significant (p-value < 0.01) under one-sample t-tests. We run all experiments for 3 times with different random seeds and report the averaged results.

		Games 1024-shot (r=2%)				MicroLens-50K 1024-shot (r=2%)				Book 1024-shot (r=1%)			
	Methods	R@10	R@20	N@10	N@20	R@20	R@50	N@20	N@50	R@20	R@50	N@20	N@50
BIGRec	TF-DCon	0.0102	0.0157	0.0062	0.0078	0.0066	0.0099	0.0027	0.0034	0.0104	0.0144	0.0083	0.0092
	RecRanker	0.0112	0.0166	0.0074	0.0090	0.0024	0.0042	0.0011	0.0014	0.0108	0.0145	0.0090	0.0097
	CCS	0.0164	0.0246	0.0097	0.0122	0.0096	0.0131	0.0041	0.0049	0.0110	0.0145	0.0088	0.0096
	GraNd	0.0158	0.0250	0.0098	0.0125	0.0014	0.0032	0.0006	0.0010	0.0102	0.0136	0.0080	0.0087
	EL2N	0.0154	0.0256	0.0098	0.0128	0.0096	0.0045	0.0041	0.0016	0.0107	0.0149	0.0085	0.0094
	Random	0.0163	0.0241	0.0100	0.0122	0.0108	0.0151	0.0044	0.0054	0.0099	0.0134	0.0083	0.0090
	DEALRec	**0.0181***	**0.0276***	**0.0115***	**0.0142***	**0.0124***	**0.0160***	**0.0055***	**0.0064***	**0.0117***	**0.0155***	**0.0096***	**0.0104***
TIGER	TF-DCon	0.0051	0.0074	0.0033	0.0040	0.0006	0.0057	0.0002	0.0013	0.0028	0.0051	0.0020	0.0027
	RecRanker	0.0028	0.0045	0.0019	0.0024	0.0043	0.0064	0.0011	0.0014	0.0027	0.0052	0.0018	0.0025
	CCS	0.0050	0.0084	0.0031	0.0041	0.0026	0.0061	0.0010	0.0013	0.0026	0.0048	0.0018	0.0024
	GraNd	0.0042	0.0053	0.0027	0.0030	0.0006	0.0014	0.0003	0.0005	0.0008	0.0020	0.0006	0.0010
	EL2N	0.0034	0.0048	0.0024	0.0029	0.0011	0.0016	0.0004	0.0004	0.0005	0.0015	0.0004	0.0007
	Random	0.0062	0.0102	0.0039	0.0051	0.0037	0.0059	0.0011	0.0014	0.0033	0.0066	0.0022	0.0031
	DEALRec	**0.0074***	**0.0114***	**0.0062***	**0.0074***	**0.0058***	**0.0076***	**0.0020***	**0.0020***	**0.0039***	**0.0076***	**0.0026***	**0.0037***

Table 3: Performance comparison between DEALRec under 1024-shot fine-tuning and the full fine-tuning of the BIGRec in terms of both accuracy and time costs. "%Com." denotes the performance achieved by DEALRec compared to the full fine-tuning.

	Games					MicroLens-50K					Book				
	R@10↑	R@20↑	N@10↑	N@20↑	Time↓	R@20↑	R@50↑	N@20↑	N@50↑	Time↓	R@20↑	R@50↑	N@20↑	N@50↑	Time↓
Full	0.0169	0.0233	0.0102	0.0120	36.87h	0.0081	0.0136	0.0038	0.0053	66.64h	0.0076	0.0108	0.0060	0.0068	84.77h
DEALRec	0.0181	0.0276	0.0115	0.0142	1.67h	0.0124	0.0160	0.0055	0.0064	1.23h	0.0117	0.0155	0.0096	0.0104	1.93h
% Com.	**107.10%**	**118.45%**	**112.75%**	**118.33%**	**4.53%**	**153.09%**	**117.65%**	**144.74%**	**120.75%**	**1.85%**	**153.95%**	**143.52%**	**160.00%**	**152.94%**	**2.28%**

thereby benefiting from the stronger generalization ability of large-sized LLMs [32]; and 2) BIGRec leverages item titles to present the user sequence, leading to better utilization of world knowledge in LLMs. In contrast, TIGER learns extra item tokens for LLMs. This might result in cold-start item issues since only limited item tokens are learned while others are maintained randomly initialized under the few-shot fine-tuning setting.

- Among all coreset selection baselines, difficulty-based (GraNd, EL2N) methods generally perform better than diversity-based methods (TF-DCon, RecRanker). This is reasonable since diversity-based methods merely heuristically encourage selecting users with divergent preference, which lacks the assessments of their contributions to the model training. In contrast, GraNd and EL2N use pre-defined metrics to measure the sample difficulty and select the samples with larger scores, which encourages selecting the samples that are more informative for models' optimization. Besides, CCS improves EL2N in most cases, as it maintains easy samples for selection, thus compensating the knowledge of recommendation data from high-density areas.
- Another interesting observation is that random sampling yields competitive performance or even outperforms other coreset selection methods in some cases, which might attributed to two possible reasons: 1) Uniformly selected user sequences preserve high coverage of the original training distribution compared to other baselines, which ensures a high probability of guaranteed bound for low empirical risk [60]. This observation is also consistent with the findings in [16]. 2) The inferior performance of some coreset selection methods also might be caused by the implementation settings (Section 4.1.3), where they may suffer from the learning ability gap between the surrogate model and LLMs. (*cf.* Section 3.2).
- DEALRec significantly outperforms all coreset selection methods across the three datasets. The consistent performance improvements on both backend models validate the superiority of DEALRec in identifying influential samples for LLMs' adaptation to the recommendation data. The superior performance is attributed to: 1) the accurate and efficient estimation of the influence on empirical risk, *i.e.,* overall performance by removing a sample in training; and 2) the gap regularization based on the effort score to penalize the easy samples for LLMs. By emphasizing the non-trivial samples specifically for LLMs, gap regularization alleviates the learning ability gap between the surrogate model and the LLMs.

- **Comparison with full fine-tuning.** We further compare DEALRec with BIGRec under full training *w.r.t.* accuracy and efficiency, as presented in Table 3. We can find that: 1) DEALRec achieves higher performance compared to the model trained by the full data, indicating the effectiveness of DEALRec for high accuracy. The inferior performance of BIGRec under full training also implies that not all user sequences are informative for model training, or even harmful to the training, *e.g.,* false negative interactions. This has also been observed in CTR prediction [52] and has been discussed in [2] from the view of data redundancy. 2) DEALRec

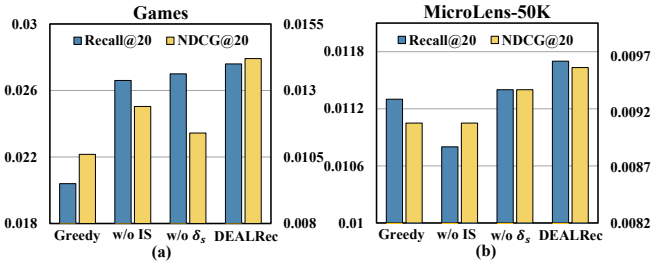


Figure 4: Ablation study of the influence score, effort score, and coverage-enhanced sample selection strategy.

Table 4: Performance comparison between DEALRec with different surrogate models and the BIGRec under full training. "Time" presents the time costs for training the surrogate model on a single NVIDIA RTX A5000.

	R@10↑	R@20↑	N@10↑	N@20↑	Time↓
Full	0.0169	0.0233	0.0102	0.0120	/
BERT4Rec	0.0175	0.0258	0.0103	0.0128	0.76h
SASRec	0.0181	0.0276	0.0115	0.0142	0.45h
DCRec	0.0211	0.0283	0.0117	0.0137	0.61h

significantly reduces the time costs for LLMs' fine-tuning (97.11% reduction of fine-tuning costs on average). With the remarkably declined training costs, DEALRec has the potential to facilitate real-world applications of LLM-based recommender models.

4.3 In-depth Analysis

We carry out further experiments[9] to analyze the effectiveness of each component of DEALRec and the robustness of surrogate model selection. Besides, we investigate how DEALRec performs under different selection ratios and explore how DEALRec performs over different user groups.

4.3.1 ***Ablation Study (RQ2).*** To study the effectiveness of each component of DEALRec, *i.e.,* influence score, effort score, and coverage-enhanced sample selection strategy, we separately remove the Influence Score (IS) and effort score δ_s, referred to as "w/o IS" and "w/o δ_s", respectively. Besides, we replace the coverage-enhanced sample selection strategy by greedily selecting the samples with higher scores, denoted as "Greedy". From the results presented in Figure 4, we can observe that: removing either the influence score or effort score will cause performance drops. This validates the effectiveness of 1) the assessment of overall performance change caused by removing samples from training; 2) additional signals of learning ability captured from LLMs as regularization, alleviating the gap between the surrogate model and the LLMs. Moreover, simply selecting the samples with higher overall scores might weaken the learning of distinct user behaviors and item knowledge (inferior performance of "Greedy"), as discussed in Section 3.3.

4.3.2 ***Robustness on different surrogate model (RQ2).*** To further assess the generalization ability of DEALRec on different surrogate models, we employ three representative sequential recommender models, *i.e.,* BERT4Rec [44], SASRec [24], and a recently proposed DCRec [56] as the surrogate models for data pruning, respectively. The results on Games are presented in Table 4, and we omit the results with similar observations on the other two datasets to save space.

From the table, we can find that: 1) DEALRec with the three surrogate models consistently outperforms BIGRec under full fine-tuning. This demonstrates the strong robustness of DEALRec on different surrogate models. 2) Nonetheless, different surrogate models cause some fluctuations in the accuracy of the LLM-based recommender model. This is reasonable because different model architectures have different expressiveness of user behavior and item knowledge. As such, the selected samples possibly vary across different surrogate models, thus affecting the effectiveness of LLMs' few-shot fine-tuning. 2) It is noted that DCRec surpasses SASRec and BERTRec by a large margin. The possible reason might be that the SOTA DCRec employs contrastive learning to enhance the learning of user representations, thus leading to better user modeling and yielding lower empirical risk. 3) SASRec exhibits the least time costs for training and achieves competitive performance among the three surrogate models. Therefore, based on the empirical results, SASRec could be a good choice of surrogate model for DEALRec to facilitate efficient LLM-based recommender fine-tuning in real-world deployments.

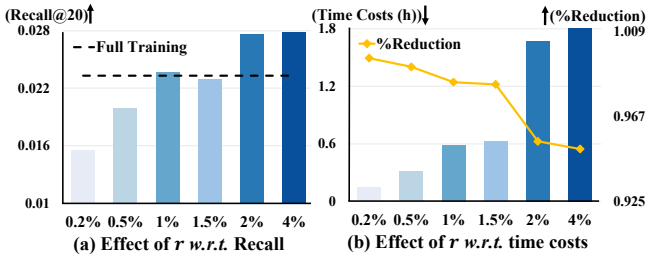


Figure 5: Performance of DEALRec with different selection ratio r *w.r.t.* accuracy and efficiency on Games.

4.3.3 ***Effect of selection ratio r (RQ3).*** To investigate the effect of selection ratio r on DEALRec on both accuracy and efficiency, we vary the ratio r from 0.2% (128-shot) to 4% (4096-shot) and present the results in Figure 5. From the figures, it is observed that: 1) The recommendation accuracy rapidly improves as the number of selected samples increases from 0.2% to 1%, surpassing the full training when $r = 1\%$. Besides, if we continuously increase the selection ratio from 2% to 4%, the benefits from additional samples gradually diminish and only minor improvements in accuracy are observed. We suspect that the gap between and the recommendation data mainly resides in a small subset of the representative user behaviors, which is what DEALRec aims to identify. 2) Meanwhile, although the time costs for fine-tuning LLMs gradually increase because of additional samples, the cost reduction compared to the full training still reaches over 94%. 3) Empirically, setting $r = 1\%$ is recommended to achieve comparable performance to full fine-tuning as well as achieving low costs for real-world deployments.

4.3.4 ***User group evaluation (RQ3).*** To study how DEALRec achieves superior overall performance, we test the DEALRec over user sequences of different difficulties. Specifically, we calculate

[9]We conduct in-depth experiments using BIGRec as the backend model because of its better performance compared to TIGER.

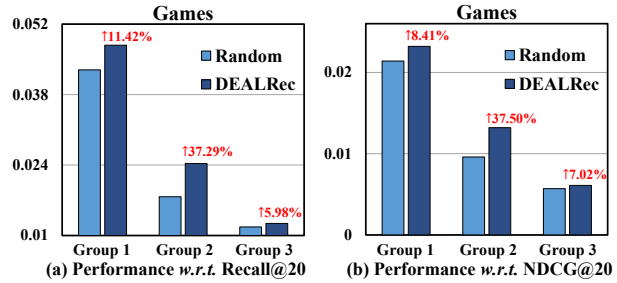


Figure 6: Performance of DEALRec over easy to difficult samples (Group 1 to Group 3).

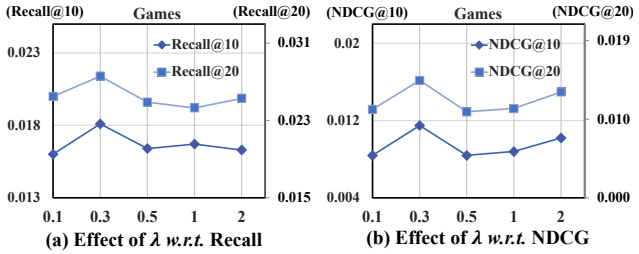


Figure 7: Performance of DEALRec with different λ.

the loss of each user sequence via the model trained by randomly selected few-shot samples; we then divide the users into three groups according to their loss values, from the easier samples with smaller loss (Group 1) to the harder samples with larger loss (Group 3). The results of each group of DEALRec and Random on Games are presented in Figure 6. We can find that 1) the performance of both DEALRec and Random gradually declines from Group 1 to Group 3, because users with larger loss are more difficult to predict. Nevertheless, 2) DEALRec consistently outperforms Random in each group, which validates the effectiveness of DEALRec in considering the influence on overall performance.

4.3.5 ***Effect of regularization strength λ.*** To examine the impact of gap regularization strength, we vary λ from 0.1 to 2 for DEALRec and evaluate the performance as presented in Figure 7. From the figures, we can find that: 1) As we incrementally increase the value of lambda, the overall trend of accuracy has been observed to be generally improved. This makes sense because there exists a gap between the surrogate model and LLMs as discussed in Section 3.2, emphasizing the necessity to regularize the influence score to be aligned with the learning ability of the LLMs. 2) However, blindly pursuing larger lambda is not necessarily beneficial. Therefore, we should carefully choose the λ to balance between the performance-driven influential samples from the surrogate model and the difficult samples for the LLMs.

5 RELATED WORK

5.1 LLM-based Recommendation

Leveraging LLMs for recommendation has gained remarkable attention recently [13, 31, 48, 54], showcasing their potential across various recommendation tasks, including CTR predicton [4], sequential recommendation [32], and cross-domain recommendation [15]. Some early studies explore the recommendation ability of powerful LLMs, *e.g.,* ChatGPT, through the in-context-learning ability [10, 34, 45]. Nevertheless, the performance of LLMs is limited without extra fine-tuning over the domain-specific recommendation data [4, 34]. To fully leverage the potential of LLMs for recommendation, a series of work studies various fine-tuning strategies tailored for recommendation tasks [15, 30, 57].

However, fine-tuning LLMs requires extensive computational resources and time costs, thus hindering the real-world applications of LLM-based recommender models. Therefore, it is crucial to enhance the fine-tuning efficiency of LLM-based recommender models. In this work, we propose the task of data pruning for efficient LLM-based recommendation, which aims to identify representative samples tailored for LLMs' few-shot fine-tuning.

5.2 Coreset Selection

Coreset selection has been widely studied in both traditional machine learning [7, 11, 51] and deep learning [55], benefiting many downstream tasks such as data-efficient learning [47], continual learning [5], neural architecture search [43], and active learning [42]. It aims to select a small but representative subset from the full data that can lead to comparable model performance. Previous work mainly falls into two groups: 1) Heuristic methods [8, 12, 47] typically assume difficult or diverse samples are informative for model training and use pre-defined metrics to compute a score for selection. 2) Optimization-based methods [25, 29, 37, 55] leverages the bi-level or discrete optimization techniques to optimize the data subset that can minimize the empirical risk. However, heuristic methods do not estimate the impact of selected samples on empirical risk, thus might lead to suboptimal coreset selection. And optimization-based methods fail to be applied to LLM-based recommendation due to the cumbersome calculation for complex optimization. Furthermore, previous methods usually rely on the training of the model on full data for selection, which is infeasible for LLM-based recommendation (*cf.* Section 2).

- **Data Condensation** [59] is another potential solution to achieve data-efficient training. However, it is intrinsically different from our proposed task of data pruning. While it aims to synthesize a small but informative dataset [58], our proposed task targets to identify representative samples from the existing samples for LLM's few-shot fine-tuning. Besides, previous work mainly designed for continuous data, which is not applicable to LLM-based recommendation [52]. TF-DCon [53] is recently proposed for content-based recommendation and we compare it in Section 4.2.

6 CONCLUSION

In this work, we proposed the task of data pruning for efficient LLM-based recommendation, which aims to identify representative samples tailored for LLMs' few-shot fine-tuning. Furthermore, we posited two objectives for this data pruning task: 1) high accuracy targets to select the samples that can lead to low empirical risk; and 2) high efficiency strives to consume low costs for the data pruning process. To this end, we proposed a novel data pruning method, namely DEALRec, to efficiently identify the influential samples with two scores. 1) The influence score is formulated to estimate the influence of sample removal on empirical risk, where the calculation is extended from the influence function and is accelerated through

the symmetric property. 2) We introduced a small-sized surrogate model to calculate the influence score efficiently and proposed the effort score to bridge the gap between the surrogate model and LLMs. Empirical results validate the effectiveness of DEALRec in achieving both high efficiency and high accuracy.

This work proposes a data pruning task for LLM fine-tuning, opening up a new research direction for efficient LLM-based recommendation and leaving many promising directions for future work. 1) It is worthwhile to apply DEALRec to more LLM-based recommender models on more cross-domain datasets, improving fine-tuning performance with limited resources. 2) Due to the limited context window length of LLMs, it is promising to select the informative interacted items in users' interaction sequences for LLMs' fine-tuning. 3) Enhancing the inference efficiency of LLM-based recommender models is also a crucial problem for their real-world deployments.

REFERENCES

[1] Naman Agarwal, Brian Bullins, and Elad Hazan. 2016. Second-order stochastic optimization in linear time. *stat* 1050 (2016), 15.

[2] Sharat Agarwal, Himanshu Arora, Saket Anand, and Chetan Arora. 2020. Contextual diversity for active learning. In *ECCV*. Springer, 137–153.

[3] Keqin Bao, Jizhi Zhang, Wenjie Wang, Yang Zhang, Zhengyi Yang, Yancheng Luo, Fuli Feng, Xiangnaan He, and Qi Tian. 2023. A bi-step grounding paradigm for large language models in recommendation systems. *arXiv:2308.08434*.

[4] Keqin Bao, Jizhi Zhang, Yang Zhang, Wenjie Wang, Fuli Feng, and Xiangnan He. 2023. Tallrec: An effective and efficient tuning framework to align large language model with recommendation. In *RecSys*. ACM.

[5] Zalán Borsos, Mojmir Mutny, and Andreas Krause. 2020. Coresets via bilevel optimization for continual learning and streaming. *NeurIPS* 33 (2020), 14879–14890.

[6] Chengliang Chai, Jiayi Wang, Nan Tang, Ye Yuan, Jiabin Liu, Yuhao Deng, and Guoren Wang. 2023. Efficient coreset selection with cluster-based methods. In *KDD*. ACM, 167–178.

[7] Yutian Chen, Max Welling, and Alex Smola. 2012. Super-samples from kernel herding. *arXiv:1203.3472* (2012).

[8] C Coleman, C Yeh, S Mussmann, B Mirzasoleiman, P Bailis, P Liang, J Leskovec, and M Zaharia. 2020. Selection via Proxy: Efficient Data Selection for Deep Learning. In *ICLR*.

[9] R Dennis Cook. 1977. Detection of influential observation in linear regression. *Technometrics* 19, 1 (1977), 15–18.

[10] Sunhao Dai, Ninglu Shao, Haiyuan Zhao, Weijie Yu, Zihua Si, Chen Xu, Zhongxiang Sun, Xiao Zhang, and Jun Xu. 2023. Uncovering ChatGPT's Capabilities in Recommender Systems. *arXiv:2305.02182*.

[11] Dan Feldman, Matthew Faulkner, and Andreas Krause. 2011. Scalable training of mixture models via coresets. *NeurIPS* 24 (2011).

[12] Vitaly Feldman and Chiyuan Zhang. 2020. What neural networks memorize and why: Discovering the long tail via influence estimation. *NeurIPS* 33 (2020), 2881–2891.

[13] Yue Feng, Shuchang Liu, Zhenghai Xue, Qingpeng Cai, Lantao Hu, Peng Jiang, Kun Gai, and Fei Sun. 2023. A Large Language Model Enhanced Conversational Recommender System. *arXiv:2308.06212* (2023).

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

[15] Yuqi Gong, Xichen Ding, Yehui Su, Kaiming Shen, Zhongyi Liu, and Guannan Zhang. 2023. An Unified Search and Recommendation Foundation Model for Cold-Start Scenario. In *CIKM*. 4595–4601.

[16] Chengcheng Guo, Bo Zhao, and Yanbing Bai. 2022. Deepcore: A comprehensive library for coreset selection in deep learning. In *DEXA*. Springer, 181–195.

[17] Huifeng Guo, Ruiming Tang, Yunming Ye, Zhenguo Li, and Xiuqiang He. 2017. DeepFM: a factorization-machine based neural network for CTR prediction. In *IJCAI*. 1725–1731.

[18] Frank R Hampel. 1974. The influence curve and its role in robust estimation. *Journal of the american statistical association* 69, 346 (1974), 383–393.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *CVPR*. IEEE, 770–778.

[20] Muyang He, Shuo Yang, Tiejun Huang, and Bo Zhao. 2023. Large-scale Dataset Pruning with Dynamic Uncertainty. *arXiv:2306.05175*.

[21] Xiangnan He and Tat-Seng Chua. 2017. Neural Factorization Machines for Sparse Predictive Analytics. In *SIGIR*. ACM, 355–364.

[22] Xiangnan He, Kuan Deng, Xiang Wang, Yan Li, Yongdong Zhang, and Meng Wang. 2020. Lightgcn: Simplifying and powering graph convolution network for recommendation. In *SIGIR*. 639–648.

[23] Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv:2106.09685*.

[24] Wang-Cheng Kang and Julian McAuley. 2018. Self-attentive sequential recommendation. In *ICDM*. IEEE, 197–206.

[25] Krishnateja Killamsetty, Sivasubramanian Durga, Ganesh Ramakrishnan, Abir De, and Rishabh Iyer. 2021. Grad-match: Gradient matching based data subset selection for efficient deep model training. In *ICML*. PMLR, 5464–5474.

[26] Krishnateja Killamsetty, Durga Sivasubramanian, Ganesh Ramakrishnan, and Rishabh Iyer. 2021. Glister: Generalization based data subset selection for efficient and robust learning. In *AAAI*, Vol. 35. 8110–8118.

[27] Krishnateja Killamsetty, Xujiang Zhao, Feng Chen, and Rishabh Iyer. 2021. Retrieve: Coreset selection for efficient and robust semi-supervised learning. *NeurIPS* 34 (2021), 14488–14501.

[28] Pang Wei Koh and Percy Liang. 2017. Understanding black-box predictions via influence functions. In *ICML*. PMLR, 1885–1894.

[29] Suraj Kothawade, Vishal Kaushal, Ganesh Ramakrishnan, Jeff Bilmes, and Rishabh Iyer. 2022. PRISM: A Unified Framework of Parameterized Submodular Information Measures for Targeted Data Subset Selection and Summarization. In *AAAI*.

[30] Lei Li, Yongfeng Zhang, and Li Chen. 2023. Prompt distillation for efficient llm-based recommendation. In *CIKM*. 1348–1357.

[31] Xinhang Li, Chong Chen, Xiangyu Zhao, Yong Zhang, and Chunxiao Xing. 2023. E4SRec: An Elegant Effective Efficient Extensible Solution of Large Language Models for Sequential Recommendation. *arXiv:2312.02443*.

[32] Xinyu Lin, Wenjie Wang, Yongqi Li, Fuli Feng, See-Kiong Ng, and Tat-Seng Chua. 2023. A multi-facet paradigm to bridge large language model and recommendation. *arXiv:2310.06491*.

[33] Robert F Ling. 1984. Residuals and influence in regression.

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

[35] Qijiong Liu, Nuo Chen, Tetsuya Sakai, and Xiao-Ming Wu. 2024. ONCE: Boosting Content-based Recommendation with Both Open- and Closed-source Large Language Models. In *WSDM*. ACM.

[36] Sichun Luo, Bowei He, Haohan Zhao, Yinya Huang, Aojun Zhou, Zongpeng Li, Yuanzhang Xiao, Mingjie Zhan, and Linqi Song. 2023. RecRanker: Instruction Tuning Large Language Model as Ranker for Top-k Recommendation. *arXiv:2312.16018*.

[37] Baharan Mirzasoleiman, Jeff Bilmes, and Jure Leskovec. 2020. Coresets for data-efficient training of machine learning models. In *ICML*. PMLR, 6950–6960.

[38] Yongxin Ni, Yu Cheng, Xiangyan Liu, Junchen Fu, Youhua Li, Xiangnan He, Yongfeng Zhang, and Fajie Yuan. 2023. A Content-Driven Micro-Video Recommendation Dataset at Scale. *arXiv:2309.15379* (2023).

[39] Mansheej Paul, Surya Ganguli, and Gintare Karolina Dziugaite. 2021. Deep learning on a data diet: Finding important examples early in training. *NeurIPS* 34, 20596–20607.

[40] Shashank Rajput, Nikhil Mehta, Anima Singh, Raghunandan H Keshavan, Trung Vu, Lukasz Heldt, Lichan Hong, Yi Tay, Vinh Q Tran, Jonah Samost, et al. 2023. Recommender Systems with Generative Retrieval. In *NeurIPS*. Curran Associates, Inc.

[41] Noveen Sachdeva, Mehak Dhaliwal, Carole-Jean Wu, and Julian McAuley. 2022. Infinite recommendation networks: a data-centric approach. *NeurIPS* 35, 31292–31305.

[42] Ozan Sener and Silvio Savarese. 2018. Active learning for convolutional neural networks: A core-set approach. (2018).

[43] Jae-hun Shim, Kyeongbo Kong, and Suk-Ju Kang. 2021. Core-set sampling for efficient neural architecture search. *arXiv:2107.06869*.

[44] Fei Sun, Jun Liu, Jian Wu, Changhua Pei, Xiao Lin, Wenwu Ou, and Peng Jiang. 2019. BERT4Rec: Sequential recommendation with bidirectional encoder representations from transformer. In *CIKM*. 1441–1450.

[45] Weiwei Sun, Lingyong Yan, Xinyu Ma, Pengjie Ren, Dawei Yin, and Zhaochun Ren. 2023. Is ChatGPT Good at Search? Investigating Large Language Models as Re-Ranking Agent. In *EMNLP*. ACL, 14918–14937.

[46] Haoru Tan, Sitong Wu, Fei Du, Yukang Chen, Zhibin Wang, Fan Wang, and Xiaojuan Qi. 2023. Data Pruning via Moving-one-Sample-out. *arXiv:2310.14664*.

[47] Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. 2018. An empirical study of example forgetting during deep neural network learning. *arXiv:1812.05159*.

[48] Lei Wang, Songheng Zhang, Yun Wang, Ee-Peng Lim, and Yong Wang. 2023. LLM4Vis: Explainable visualization recommendation using ChatGPT. *arXiv:2310.07652* (2023).

[49] Wenjie Wang, Xinyu Lin, Liuhui Wang, Fuli Feng, Yunshan Ma, and Tat-Seng Chua. 2023. Causal Disentangled Recommendation Against User Preference Shifts. *TOIS* (2023).

[50] Wenjie Wang, Xinyu Lin, Liuhui Wang, Fuli Feng, Yinwei Wei, and Tat-Seng Chua. 2023. Equivariant Learning for Out-of-Distribution Cold-start Recommendation. In *MM*. 903–914.

[51] Kai Wei, Rishabh Iyer, and Jeff Bilmes. 2015. Submodularity in data subset selection and active learning. In *ICML*. PMLR, 1954–1963.

[52] Jiahao Wu, Wenqi Fan, Shengcai Liu, Qijiong Liu, Rui He, Qing Li, and Ke Tang. 2023. Dataset condensation for recommendation. *arXiv:2310.01038*.

[53] Jiahao Wu, Qijiong Liu, Hengchang Hu, Wenqi Fan, Shengcai Liu, Qing Li, Xiao-Ming Wu, and Ke Tang. 2023. Leveraging Large Language Models (LLMs) to Empower Training-Free Dataset Condensation for Content-Based Recommendation. *arXiv:2310.09874*.

[54] Likang Wu, Zhaopeng Qiu, Zhi Zheng, Hengshu Zhu, and Enhong Chen. 2023. Exploring large language model for graph data understanding in online job recommendations. *arXiv:2307.05722*.

[55] Shuo Yang, Zeke Xie, Hanyu Peng, Min Xu, Mingming Sun, and Ping Li. 2023. Dataset pruning: reducing training data by examining generalization influence. (2023).

[56] Yuhao Yang, Chao Huang, Lianghao Xia, Chunzhen Huang, Da Luo, and Kangyi Lin. 2023. Debiased Contrastive Learning for Sequential Recommendation. In *WWW*. 1063–1073.

[57] Junjie Zhang, Ruobing Xie, Yupeng Hou, Wayne Xin Zhao, Leyu Lin, and Ji-Rong Wen. 2023. Recommendation as instruction following: A large language model empowered recommendation approach. *arXiv:2305.07001*.

[58] Bo Zhao and Hakan Bilen. 2023. Dataset condensation with distribution matching. In *WACV*. IEEE, 6514–6523.

[59] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. 2020. Dataset Condensation with Gradient Matching. In *ICLR*.

[60] Haizhong Zheng, Rui Liu, Fan Lai, and Atul Prakash. 2022. Coverage-centric Coreset Selection for High Pruning Rates. In *ICLR*.