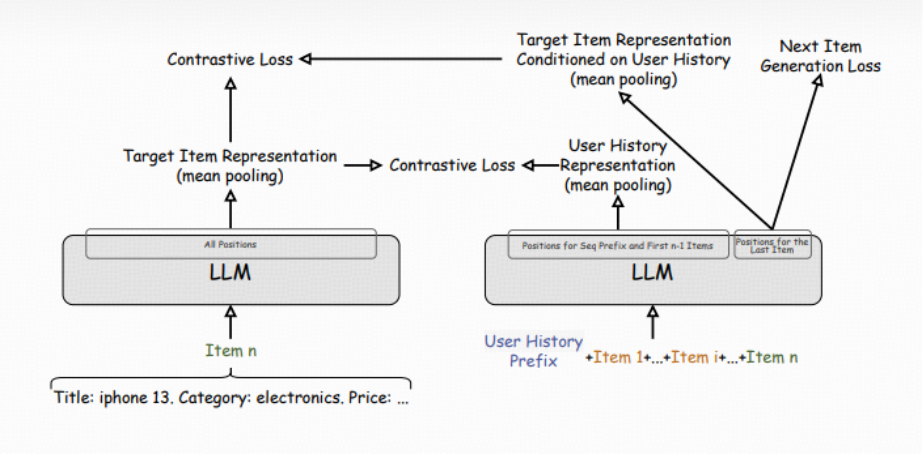
知乎



2024谷歌：CALRec-重塑序列推荐的基于对比技术的LLM推荐

SmartMindAI
专注搜索、广告、推荐、大模型和人工智能最新技术，欢迎关注我

已关注

23 人赞同了该文章

Introduction and Motivation

推荐系统旨在理解用户和商品，并利用学习到的知识为未来的用户交互推荐相关物品。传统的协同过滤简单地将用户和商品映射到共享嵌入空间，但这些不随用户行为而发生变化。近年来，序列推荐采用RNN、CNN和Transformer，它们能捕捉动态行为模式，如Transformer的自注意力机制强调序列中的关联和影响。

在RecSys中，每个输入是用户活动，如购买行为，可以是单一商品ID，也可能包含商品描述的文本信息以丰富理解。处理方式包括将交互历史视为商品ID序列，或结合更复杂特征来提升推荐的准确性。受LLMs广泛应用于各类任务的启发，我们提出利用生成型LLMs改进序列推荐系统。这些模型通过预先在大量文本数据上学习，具有良好的泛化能力。在目标领域进行微调后，性能会有所提升。

在推荐系统中，我们使用纯文本输入和输出，同时融入前瞻性提示设计。借鉴BERT4REC、Recformer和GPT4Rec等模型，我们不仅利用LLMs卓越的语言理解，还进行针对性的微调以适应序列推荐任务。通过在用户行为序列及每个潜在项目前附加特定的文本标签，这种方法不仅能帮助模型理解数据结构和文本格式，还能提升输入的连贯性。我们构建的CALRec框架源于对比学习在推荐系统和NLP中的成功应用，展示了在纯文本环境中进行序列推荐的有效性。

我们的训练目标结合了两个主要目标：一是针对定制的下一个项目生成，这要求模型学习理解和生成；二是辅助的对比性目标，通过对比来优化模型的推荐性能。这种混合策略使得模型既能生成满足用户个性化需求的推荐，又能通过对比学习提升推荐的多样性与相关性。

我们采用了两阶段的LLM微调方法，首先进行多类别联合微调，让模型总体理解各种类型的用户行为和商品特性。然后，在这个基础上，进行类别特定的微调，让模型针对每个类别的推荐任务精细化，以提高预测的精准度和多样性。这样，模型既具备生成个性推荐的能力，又能在对比学习中优化推荐效果。

我们提出了一种新型的近似轮询BM25搜索算法，专为商品检索场景设计。这种方法通过优化BM25索引技术，提供更精确的搜索结果。它通过迭代的查询与文档交互过程，实现对大量商品信息的快速和近似匹配。这种近似轮询策略使得搜索效率得到提升，有助于在海量商品中定位用户可能感兴趣的商品。

Methodology

Preliminaries and Terminology

和物品表示为文本，比如每个(I_i, u)对，然后利用预训练的自回归语言模型，如LLMs，进行序列到序列的生成任务。输入是用户过去的$n-1$个物品描述，目标是生成第n个物品的描述。这样做源于利用实际文本数据中的丰富信息，并充分发挥LLM强大的语言理解和推断能力。

Data Format and Template Design

请给出下一个建议的购买物品。通过这种方式，LLM能够根据已有的上下文信息生成合适的文本响应。

与我们的方法相比，先前基于文本的方法在处理时没有明确指定商品前缀，并且未充分利用一次性用户历史的后缀来提升模型能力，这说明他们在这方面有所欠缺。

Two-Stage Fine-Tuning

相较于之前的研究，我们借鉴了在大量数据上预训练的LLM的优势，提出了一种两阶段微调框架。首先，进行多类别联合微调（阶段1），这是一种非分类的适应性学习，让模型从通用语料中捕捉推荐问题的规律。我们依据Xue等人处理数据不平衡的方法，采用0.3的比例对每个类别进行采样，以减少大类别对微调集的影响。完成阶段1后，进入类别特定微调（阶段2），进一步强化模型针对各类别的表现。整个过程的目标与文中所述一致，都是利用LLM的预训练知识进行序列推荐任务。

Training Objectives

下一个项目生成任务（NIG Task）是LLM微调的核心目标，旨在利用用户过去的文本记录生成未来目标项目的描述。该任务可以通过符号表示用户行为序列$(t_1, t_2, \ldots, t_l)$，其中l表示序列长度✦，首个m个令牌代表除最后一个项目以外的过往项目，后续的$l-m$个令牌是目标项目。微调的目标是使模型能够预测从$(t_1, t_2, \ldots, t_m)$到t_{m+1}的序列，确保生成的文本与实际目标项相一致。这是在适应序列推荐场景下的关键技能。

$$\mathcal{L}_{NIG} = -\mathbb{E}\sum_{j=m+1}^{l} \log P(t_j | t_{1:j-1}; \theta)$$

用户级对比损失和项目级对比损失是我们在模型训练中使用的策略。对于用户级对比，我们首先计算每个用户历史序列$\mathbf{v}^U$与所有其他用户序列的差异，然后使用 InfoNCE（信息增益✦损失）来增强相似性，目标是让模型区分不同用户的独特行为模式。损失函数✦定义为：

$$\mathcal{L}_{user\_contrastive} = -\sum_{u \neq u'} \log \frac{\exp(\text{sim}(\mathbf{v}^U, \mathbf{v}^{U'})/\tau)}{\sum_{i=1}^{|\mathcal{U}|} (\exp(\text{sim}(\mathbf{v}^U, \mathbf{v}_i^U)/\tau) + \exp(\text{sim}(\mathbf{v}^{U'}, \mathbf{v}_i^U)/\tau))}$$

项目级对比损失则针对每个用户的历史项目向量$\mathbf{v}^{T|U}$，与目标项目$\mathbf{v}^T$进行比较，同样是通过 InfoNCE 损失来优化，旨在确保模型能准确预测目标项目的特性：

$$\mathcal{L}_{item\_contrastive} = -\log \frac{\exp(\text{sim}(\mathbf{v}^{T|U}, \mathbf{v}^T)/\tau)}{\sum_{i=1}^{|\mathcal{U}|} \exp(\text{sim}(\mathbf{v}^{T|U}, \mathbf{v}_i^T)/\tau)}$$

其中sim表示两个向量之间的相似度度量τ是一个温度参数，控制softmax函数的输出分布。通过这两个损失，我们的模型在训练过程中不仅关注用户行为的整体趋势，还重视每个项目与目标项的精确匹配。

$$\mathcal{L}_{TT} = -\frac{1}{N_b}\sum_{i=1}^{N_b} \log \frac{\exp(\cos(\mathbf{v}_i^{T|U}, \mathbf{v}_i^T)/\tau_c)}{\sum_{j=1}^{N_b} \exp(\cos(\mathbf{v}_j^{T|U}, \mathbf{v}_i^T)/\tau_c)},$$

$$\mathcal{L}_{UT} = -\frac{1}{N_b}\sum_{i=1}^{N_b} \log \frac{\exp(\cos(\mathbf{v}_i^{U}, \mathbf{v}_i^T)/\tau_c)}{\sum_{j=1}^{N_b} \exp(\cos(\mathbf{v}_j^{U}, \mathbf{v}_i^T)/\tau_c)},$$

他们结合了用户级对比损失✦ $\mathcal{L}_{user\_contrastive}$ 和项目级对比损失 $\mathcal{L}_{item\_contrastive}$，其中用户级对比是计算用户之间历史序列的余弦相似度✦，项目级对比则是针对目标项目与用户历史下条件化特征的相似度。这些对比对的损失都采用了InfoNCE形式，分别用τ_c作为各自的温度参数。最终的总损失函数是：

知乎

其中λ是一个权重参数，控制对比损失对整体训练目标的贡献程度。通过这种方式，模型在保持个性化推荐的同时，也注重对项目特征的精确理解和匹配。

$$\mathcal{L}_{CALRec} = (1-\alpha-\beta)\mathcal{L}_{NIG} + \alpha\mathcal{L}_{TT} + \beta\mathcal{L}_{UT}.$$

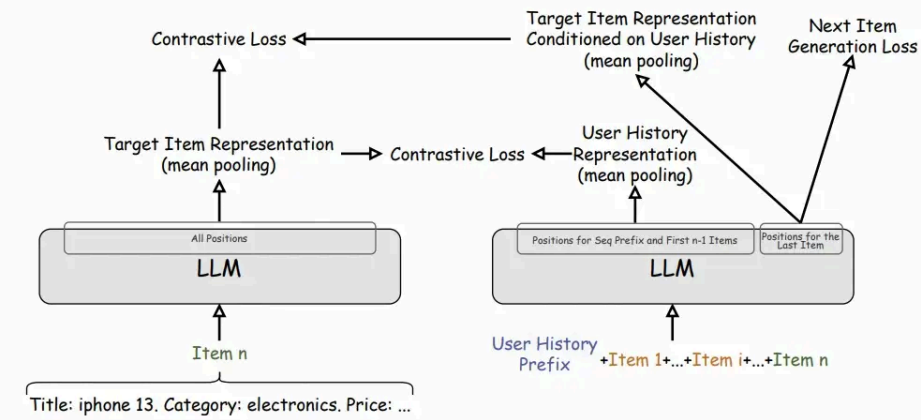


Figure 1: CALRec: Training Framework.

Quasi-Round-Robin BM25 Retrieval

在推理阶段，我们利用用户的历史交互文本信息引导模型。通过应用温度采样（Temperature Sampling），我们生成了N_{gen}个潜在的下一个项目预测。每个预测s都与它对应的序列得分紧密相连，这个得分通常为负实数，数值范围在-10到0之间，这是基于我们对数据中log概率的观察。这样，我们能得到针对所有可能项目的多样性预测。

在推理结束后，我们对生成的文本进行重复消除，仅保留前N_{preds}个唯一预测。接着，我们通过BM25检索算法评估每个预测与N_c个物品的关联度，形成(N_c, N_{preds})矩阵。为了对候选项目进行排名，我们设计了一个策略。首先，对矩阵的每一列进行标准化，使其落入$[0, 1]$区间，便于比较LLM得分与BM25得分。然后，对每个预测的BM25分数，我们引入了一个调整因子$e^{\epsilon s}$，其中s是与文本预测相关的序列得分ϵ是个小正数，使得分数略小于1，以反映LLM得分的权重。最后，通过最大池化操作，得到调整后的BM25匹配分数，以此确定最接近的匹配项目。

Input : *samples*: Output texts sampled from the LLM (size = N_{gen})
Input : *logprobs*: LLM prediction scores for *samples* (size = N_{gen})
Input : C: Item corpus (size = N_c)
Input : N_{preds}: a hyperparam controlling the number of generated text samples to use for BM25 matching.
Input : ϵ: a hyperparam used to scale the LLM log probs score before combining with the BM25 score.
Output: The sorted list of item predictions.

```
// Sort samples by score and remove duplicates
sorted_samples, sorted_logprobs ←
  SortDescending((samples, logprobs), by=logprobs)
sorted_samples, sorted_logprobs ←
  RemoveDuplicates((sorted_samples, sorted_logprobs))
// Keep the top N_preds samples
top_samples ← sorted_samples[: N_preds]
top_logprobs ← sorted_logprobs[: N_preds]
match_scores ← EmptyMatrix(N_c, N_preds)
for i ← 1 to N_preds do
    // Find BM25 match scores of entire corpus for each samples
    bm25_scores ← BM25(top_samples[i], C)
    bm25_scores_scaled ← LinearScale(bm25_scores, axis=0)
    // Modulate by sequence prediction score.
    modulated_bm25_scores ←
      e^(ε·top_logprobs[i]) × bm25_scores_scaled
    match_scores[:,i] ← modulated_bm25_scores
candidate_scores ← MaxPool(match_scores, axis=1)     // size = N_c
return ArgSortDescending(candidate_scores)
```


在处理N_c个物品的向量中，我们整合N_{preds}个预测信息（依据实际目标项在库中的排名来评估或提供前K个推荐）。我们实验表明：

1）最大池化优于其他两种策略（如最小池化和平均池化⁺）； **2**）去除调节会导致性能下降，因为必须结合最大池化与调节以确保LLM的顶级推荐； **3**）对ϵ的选择对结果影响较小，只要它足够小。我们的方法借鉴轮询思想，通过调整BM25分数，类似于非重复预测中的'准轮询'过程，每个最佳匹配项的分数逐渐减少，轮流组合成前N_{preds}推荐，尽管存在并列情况。

Experimental Setup

我们选择PaLM-2 XXS作为任务后端，这是经过多目标训练且表现优秀的大型语言模型⁺，适合多种语言任务。尽管大模型未微调对XXS无明显优势，但考虑到数据集规模小可能引发过拟合⁺，我们选择了这个版本。

		Before Dedup				After Dedup				
Dataset	# of Users	% of duplicate Purchases	# of Items	Total Purchase	Items/User	Purchases/Item	Density	Words/Item	Word Vocab Size	Avg. Word Freq
Scientific	9461	5.5%	5282	66644	7.04	12.62	0.0013	22.82	19178	6.3
Instruments	25577	4.0%	10599	214526	8.39	20.24	0.00079	18.43	22256	8.8
Arts	47197	9.4%	22828	411449	8.72	18.02	0.00038	21.38	40342	12.1
Office	44736 (50%)	6.3%	27482	352151	7.87	12.81	0.00029	21.74	56687	10.5
Games	50940	4.7%	17383	457060	8.97	26.29	0.00051	14.77	17087	16.7
Pet	43135 (20%)	5.9%	37712	380623	8.82	10.09	0.00023	19.37	41332	15.8

Table 1: Statistics of Amazon Reviews Dataset.

实验数据来自公开的亚马逊评论2018数据集，包含至少5次交互的5个子集，主要聚焦于4个商品属性。在实验中，我们运用了亚马逊评论2018数据集的5个核心子集，每个子集由至少5次用户与商品的交互组成。数据集中主要关注4种商品特性。在处理数据时，我们注意到亚马逊评论数据中存在重复现象，大约4%至9.4%的用户购买行为是前一个的精确重复。因此，我们按照严格定义的标准对用户序列进行了去重处理，但对不同时间出现的相同商品、不同评分或评价内容（可能是特殊情况下的行为）则保持了连续记录。模型训练部分，我们使用的是XXS大小的PaLM-2，在TPUv4集群上，通过JAX和PAX这两个开源框架⁺实现。我们以每批512个用户的规模训练，使用1e-4的学习率和1024个令牌的最大输入长度，80个令牌的输出解码长度。在训练中，为增加数据多样性，我们随机抽取用户记录，保留n个物品的最后k个（k可选0-4个），目标始终是最后一个截取项。初始阶段，参数设置为$\alpha = 0.125, \beta = -0.025$在150,000步后停止。通过验证⁺，当$\tau_c = 0.5$时，我们选择性能最优的模型。

在针对亚马逊评论的6个测试类别上，我们展示了CALRec模型在序列推荐任务上的优越性。它在NDCG\@10、Recall\@1和MRR指标上超越了所有8个基线。特别是在Recall\@10，无论在ID-Only还是Text-Only条件下，CALRec都明显领先，但与最强的基线UniSRec的ID+Text变体相比，略逊一筹。

Dataset	Metric (opt)/(pes)	ID-Only Methods		ID+Text Methods			Text-Only Methods				Improv. (opt)
		BERT4Rec	SASRec	FDSA	S^3-Rec	UniSRec	UniSRec	RecFormer	LIR	CALRec	
Scientific	NDCG@10	0.0411	0.0555	0.0594	0.0414	0.0644	0.0620/0.0580	0.0639/0.0588	0.0565/0.0486	**0.0788**/0.0740	22.4%
	Recall@1	0.0239	0.0079	0.0357	0.0172	0.0199	0.0242/0.0140	0.0298/0.0168	0.0142/0.0052	**0.0511**/0.0389	43.1%
	Recall@10	0.0636	0.1065	0.0887	0.0727	**0.1181**	0.1101/0.1100	0.1058/0.1057	0.1005/0.0933	0.1124/0.1117	-4.8%
	MRR	0.0389	0.0456	0.0555	0.0380	0.0555	0.0546/0.0492	0.0570/0.0502	0.0461/0.0385	**0.0730**/0.0667	28.1%
Instruments	NDCG@10	0.0680	0.0623	0.0796	0.0606	0.0709	0.0661/0.0655	0.0596/0.0585	0.0344/0.0338	**0.0909**/0.0905	14.2%
	Recall@1	0.0494	0.0158	0.0574	0.0202	0.0237	0.0251/0.0236	0.0266/0.0239	0.0040/0.0031	**0.0718**/0.0706	25.1%
	Recall@10	0.0915	0.1130	0.1092	0.1048	**0.1255**	0.1141/0.1141	0.0940/0.0940	0.0696/0.0692	0.1158/0.1158	-7.7%
	MRR	0.0658	0.0528	0.0765	0.0526	0.0613	0.0576/0.0569	0.0535/0.0521	0.0263/0.0257	**0.0864**/0.0857	12.9%
Arts	NDCG@10	0.0547	0.0619	0.0726	0.0602	0.0729	0.0630/0.0624	0.0662/0.0653	0.0443/0.0431	**0.0864**/0.0853	18.5%
	Recall@1	0.0347	0.0147	0.0460	0.0216	0.0213	0.0220/0.0208	0.0279/0.0254	0.0067/0.0051	**0.0636**/0.0610	38.3%
	Recall@10	0.0799	0.1118	0.1061	0.1052	**0.1320**	0.1103/0.1102	0.1095/0.1095	0.0859/0.0850	0.1140/0.1139	-13.6%
	MRR	0.0513	0.0519	0.0680	0.0520	0.0615	0.0546/0.0540	0.0582/0.0569	0.0344/0.0331	**0.0815**/0.0801	19.9%
Office	NDCG@10	0.0545	0.0602	0.0749	0.0607	0.0713	0.0625/0.0616	0.0687/0.0676	0.0391/0.0379	**0.0976**/0.0966	30.3%
	Recall@1	0.0403	0.0134	0.0547	0.0291	0.0213	0.0250/0.0228	0.0328/0.0302	0.0056/0.0040	**0.0761**/0.0734	39.1%
	Recall@10	0.0709	0.1033	0.0982	0.0941	**0.1220**	0.1023/0.1022	0.1063/0.1063	0.0735/0.0728	0.1213/0.1213	-0.6%
	MRR	0.0520	0.0499	0.0710	0.0535	0.0597	0.0545/0.0533	0.0603/0.0588	0.0309/0.0296	**0.0925**/0.0911	30.3%
Games	NDCG@10	0.0334	0.0451	0.0526	0.0419	0.0501	0.0408/0.0407	0.0377/0.0360	0.0211/0.0197	**0.0595**/0.0585	13.1%
	Recall@1	0.0112	0.0039	0.0210	0.0080	0.0075	0.0111/0.0107	0.0126/0.0091	0.0015/0.0009	**0.0295**/0.0277	40.5%
	Recall@10	0.0646	0.0989	0.0957	0.0884	**0.1074**	0.0820/0.0820	0.0724/0.0723	0.0442/0.0427	0.0986/0.0984	-8.2%
	MRR	0.0313	0.0380	0.0483	0.0367	0.0427	0.0372/0.0370	0.0349/0.0326	0.0171/0.0159	**0.0510**/0.0498	5.6%
Pet	NDCG@10	0.0376	0.0448	0.0537	0.0392	0.0574	0.0539/0.0464	0.0590/0.0471	0.0391/0.0354	**0.0736**/0.0693	24.8%
	Recall@1	0.0276	0.0093	0.0394	0.0162	0.0202	0.0295/0.0129	0.0377/0.0130	0.0094/0.0030	**0.0570**/0.0465	44.7%
	Recall@10	0.0499	0.0773	0.0710	0.0647	**0.0970**	0.0829/0.0826	0.0826/0.0818	0.0680/0.0673	0.0937/0.0934	-3.4%
	MRR	0.0365	0.0382	0.0522	0.0349	0.0503	0.0498/0.0399	0.0549/0.0393	0.0317/0.0272	**0.0696**/0.0639	26.8%
Average (6 Categories)	NDCG@10	0.0482	0.0550	0.0655	0.0507	0.0645	0.0581/0.0558	0.0592/0.0556	0.0391/0.0364	**0.0811**/0.0790	23.8%
	Recall@1	0.0312	0.0108	0.0424	0.0187	0.0190	0.0228/0.0175	0.0279/0.0197	0.0069/0.0036	**0.0582**/0.0530	37.3%
	Recall@10	0.0701	0.1018	0.0948	0.0883	**0.1170**	0.1003/0.1002	0.0951/0.0951	0.0736/0.0717	0.1033/0.1031	-1.5%
	MRR	0.0460	0.0461	0.0619	0.0446	0.0552	0.0514/0.0484	0.0531/0.0483	0.0311/0.0283	**0.0757**/0.0729	22.3%

对比仅基于文本的基线，我们的悲观指标普遍优于它们的乐观指标，这强调了全面评估的重要性。尤其值得注意的是，CALRec在Recall\@1上的优化幅度达到了37.3%，在NDCG\@10和MRR上分别比最先进的SotA方法高出23.8%和22.3%。在各数据类别和评估指标下，CALRec相对于SotA的百分比增益也有所体现，显示了其显著的优势。

Conclusion

我们提出CALRec，一种利用LLMs和对偶学习的两阶段训练框架，专为序列推荐设计。实验基于PaLM-2大型语言模型，结合了针对有限样本的学习模板和创新的BM25检索策略+。在亚马逊评论数据集上，我们的方法展示了显著优势，超越了最先进的序列推荐模型。

原文《CALRec: Contrastive Alignment of Generative LLMs For Sequential Recommendation》

发布于 2024-06-03 10:53 · IP 属地北京

LLM 推荐系统 序列推荐