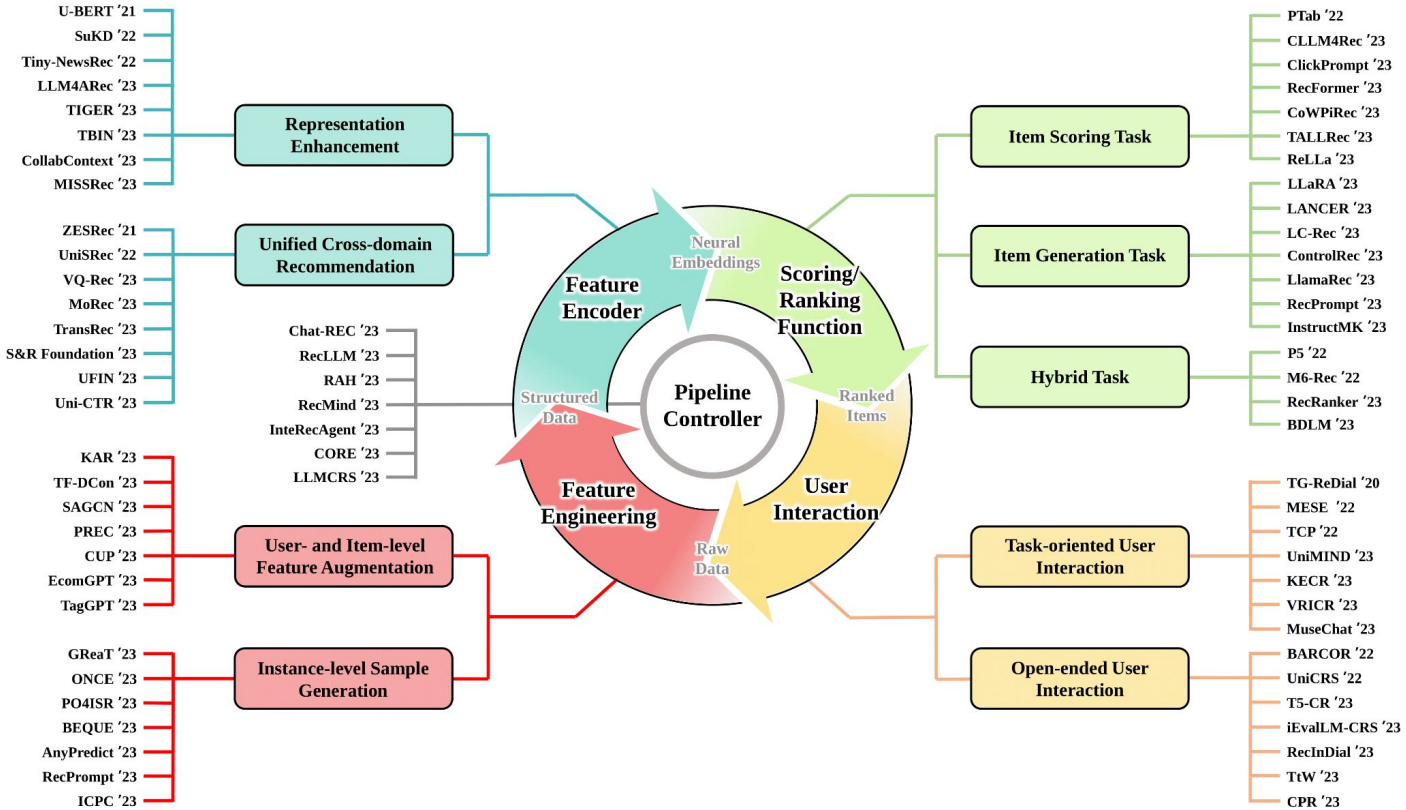
LLMs and Generative Models for RecSys

Hui Yang

Index

- **High Level Overview**
- LLM for Feature Engineering
- LLM for better model Arch and Generative Training

High Level Overview



https://github.com/CHIANGEL/Awesome-LLM-for-RecSys

High Level Paradigms

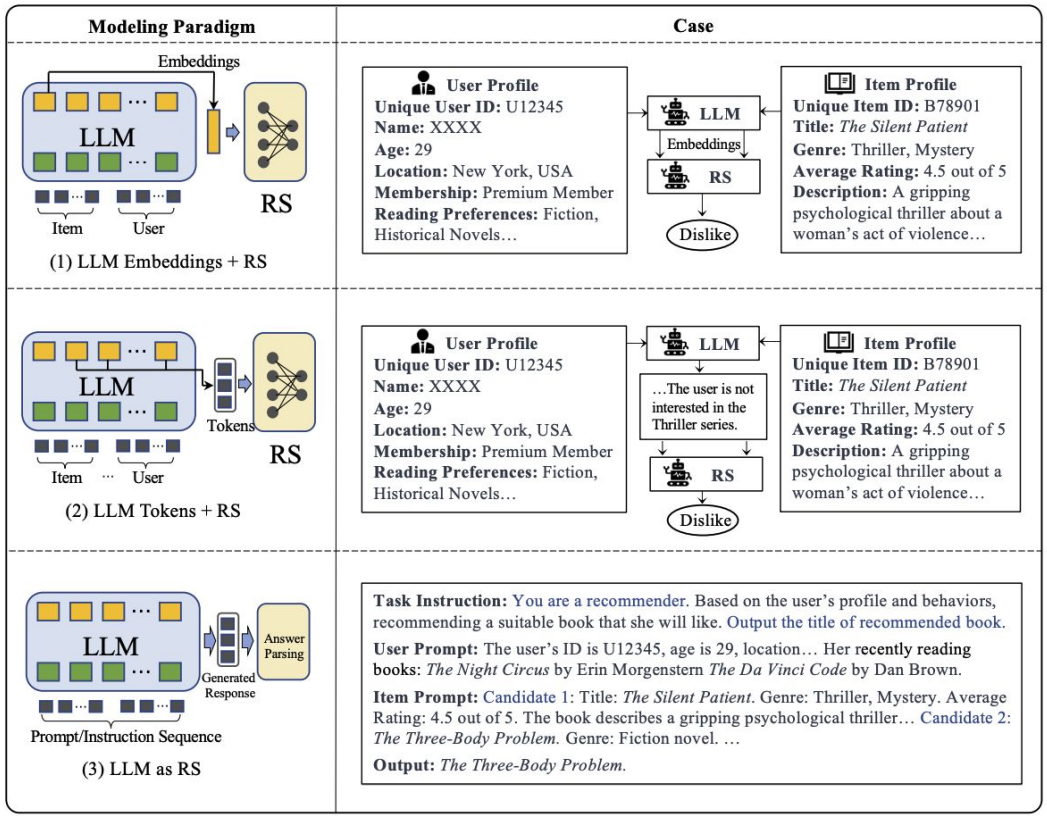


Figure 1: Three modeling paradigms of the research for large language models on recommendation systems.

https://arxiv.org/pdf/2305.19860.pdf

LLM for RecSys

- Feature Encoder and engineering
  - Directly leverage LLM embeddings - e.x. GPT Embeddings
  - LLM augmented features
- LLM used as a ranker/re-ranker
  - Off the shelf LLM
  - Fine-tuned LLM
- LLM for user-interaction
  - Conversational Recommendation
- LLM inspired architecture
  - MoE
  - Generative modeling

Feature Encoder and engineering

- GPT embedding is already commonly used in retrieval tasks
  - GPT embedding paper
  - Scaling laws on embedding performance vs. model size
- LLM derived features based on eCommerce item textual data
  - Walmart paper for aspects generation based on users potential interests
  - Sony paper to generate movie descriptions purely based on movie names
  - Google paper on LLM to extract User Intent Journeys
- To summarize
  - The world knowledge in LLM is being used to geneate/augment the features
  - Then the features are either
    - Encoded into text embeddings
    - Categorized as sparse features

LLM used as a ranker/re-ranker

- Off the shelf LLM as a recommender with prompt engineering
  - Prompting is needed, and tuning is mainly focusing on the prompts
    - Language Models as Recommender Systems: Evaluations and Limitations
    - Is ChatGPT a Good Recommender? A Preliminary Study
- Fine-tuned LLM as a recommender
  - Supervised Fine Tuning (SFT)
    - GPTRec - a generative sequential recommendation model based GPT-2
    - Genrec: Large language model for generative recommendation.
      - Directly generate the target item to recommend
  - Instruction Tuning (Human Alignment)
    - Can align an LLM with multiple different tasks
- Summary
  - Good LLMs could beat a simple RecSys baseline
  - Nowhere close to the state of the art RecSys in industry settings today

LLM used as a ranker/re-ranker

- LLM as a reranker
  - LLM can be used to aim for specific goals - diverse reranking, relevance, freshness
  - Enhancing Recommendation Diversity by Re-ranking with Large Language Models
  - Large Language Models are Zero-Shot Rankers for Recommender Systems
    - LLMs can
      - 1) struggle to perceive the order of historical interactions, and
      - 2) can be biased by popularity or item positions in the prompts
      - Can be overcome by careful prompt engineering
- LLM as a domain specialist helping recommendation
  - Health, Finance, Medical Care, Law

LLM for user-interaction

- Conversational Recommendation
  - A Large Language Model Enhanced Conversational Recommender System
  - Leveraging Large Language Models in Conversational Recommender Systems

LM inspired architecture

- Generative Sequence Training
- New Transformer Arch: HSTU
- MoE

Index

- High Level Overview
- **LLM for Feature Engineering**
- LLM for better model Arch and Generative Training

LLM for Feature Engineering

- Walmart paper for aspects generation based on users potential interests
  - Constrain to a certain number of user interest aspects
  - Carefully designed Prompts for LLM to generate the tags for each aspect, based on the user reviews/titles of the eCommerce items
  - Treat each tag as a sparse feature into the RecSys
- Sony paper to generate movie descriptions purely based on movie names
  - Directly generate texts from LLM based on the movie names
  - Encode the generated texts as embeddings
- Google paper on LLM to extract User Intent Journeys
  - LLM to capture long term user intents
  - Clusters user interaction histories into Journeys
  - Use LLMs to describe the journeys

eCommerce Item Aspect Generation

- Aspects have better performance than PLM (Pre-training Language Embeddings)
- Uses PaLM LLM

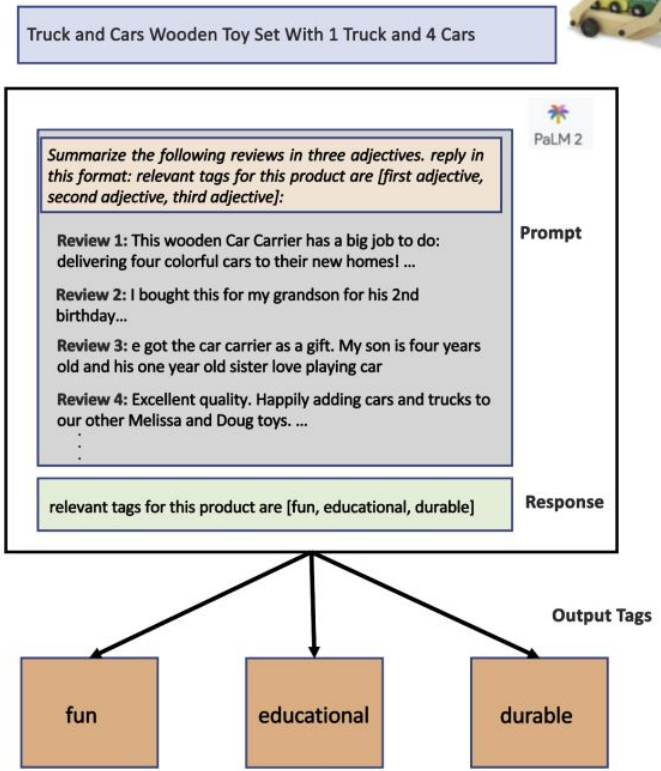
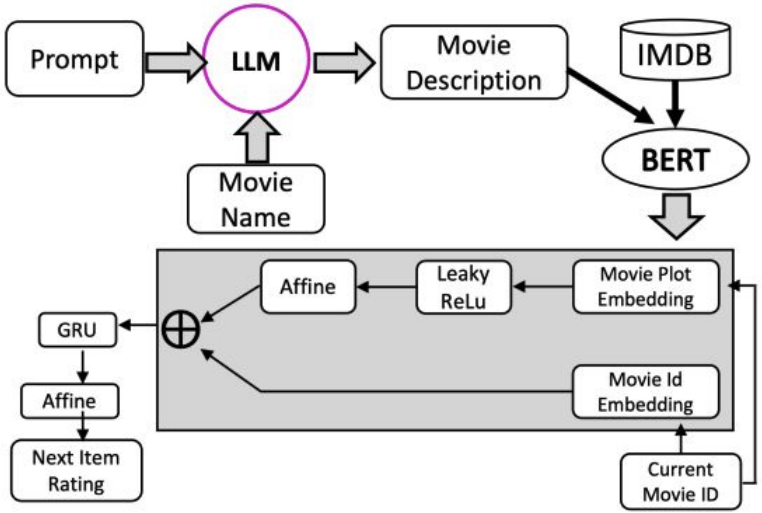


Table 1. MRR and NDCG scores for the benchmark and LLM-based aspect augmented models

Model	NDCG@5	NDCG@10	MRR@5	MRR@10
FFNN-MLE	0.3750	0.4710	0.3229	0.3582
FFNN-PCE	0.3803	0.4775	0.3303	0.3649
FFNN-NDCG	0.3800	0.4768	0.3297	0.3645
PLM-FFNN-MLE	0.3727	0.4747	0.3261	0.3588
PLM-FFNN-PCE	0.3942	0.4918	0.3420	0.3775
PLM-FFNN-NDCG	0.3872	0.4861	0.3351	0.3710
Aug-Concat-MLE	0.4029	0.5044	0.3525	0.3868
Aug-Concat-PCE	**0.4280**	**0.5191**	**0.3700**	**0.4027**
Aug-Concat-NDCG	0.4259	0.5174	0.3690	0.4019
Aug-WD-MLE	0.4005	0.4976	0.3440	0.3790
Aug-WD-PCE	0.4151	0.5086	0.3571	0.3908
Aug-WD-NDCG	0.4169	0.5101	0.3591	0.3927
Aug-2T-MLE	0.4059	0.5017	0.3485	0.3830
Aug-2T-PCE	0.4179	0.5109	0.3598	0.3932
Aug-2T-NDCG	0.4170	0.5102	0.3592	0.3928

Movie Description Generation by Sony



(a) Flowchart of our proposed solution

PROMPT:

"Your are given the name of a Movie as Input. The output consists of three categories: Plot, Cast and Director. Plot consists of the detailed description of the plot of the movie including the key events. Cast consists of the names of cast of the movie. Director Consists of the name of the director of the movie.

Input:
Movie: 'The lord of the Rings: The Fellowship of the Ring'
Output:
Plot: A young hobbit named Frodo Baggins embark on a perilous journey to destroy a powerful ring that could bring about the destruction of Middle-earth. He is joined by a fellowship of diverse characters, including wizards, elves, dwarves, and humans, as they face numerous challenges and encounter with creatures of the darkness.
Cast:Elijah Wood, Ian McKellen, Viggo Mortensen
Director: Peter Jackson
Input:
Movie:'Toy Story(1995)'
Output:
"

TEXT GENERATED BY LLM:

Plot: In the first full-length animated feature film, Woody a pull-string cowboy toy, is kidnapped by a toy collector and his gang of toys. His friends, Buzz Lightyear and the other toys, embark on a mission to rescue him.
Cast: Tom Hanks, Tim Allen, Don Rickles
Director: John Lasseter

(b) The prompt used and the output generated by the LLM

Movie Description Generation by Sony

- LLM is LLaMa for generation, BERT for encoding
- LLM description alone has comparable performance vs. IMDB plots

Table 2: Hit,NDCG, and MRR for recommendations systems using text description of the movies generated by LLM in different proportions

Metrics	IMDB only	IMDB(70%) +LLM(30%)	LLM only	IMDB|| LLM
HIT@10	0.707	0.706	0.700	0.705
HIT@5	0.592	0.595	0.594	0.591
HIT@1	0.284	0.288	0.278	0.290
NDCG@10	0.484	0.486	0.480	0.485
NDCG@5	0.447	0.450	0.446	0.448
NDCG@1	0.284	0.288	0.278	0.290
MRR	0.426	0.429	0.423	0.428

LLM to Extract User Intent Journeys

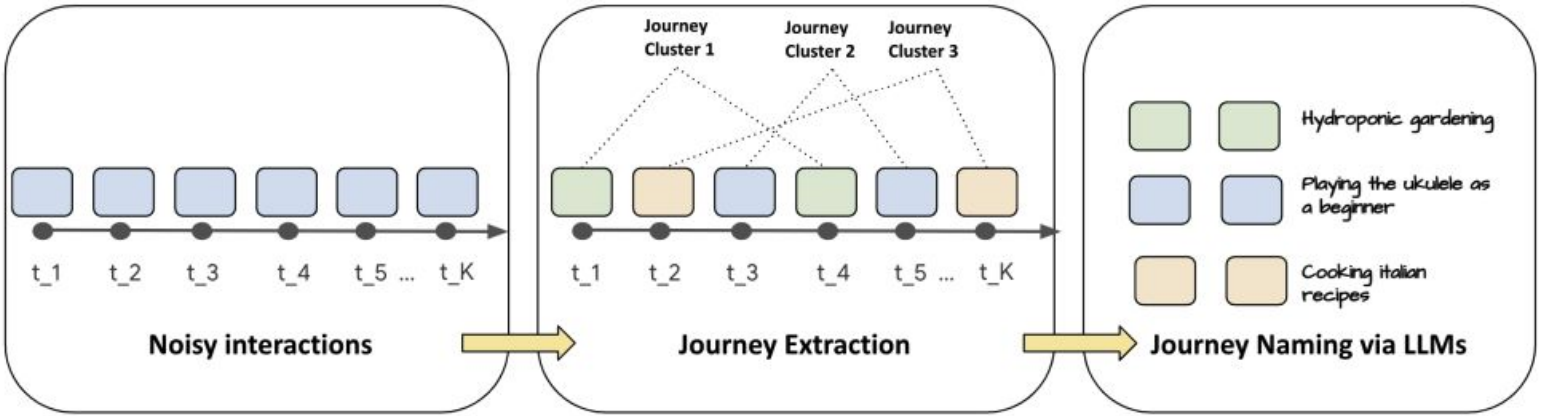


Fig. 1. Our approach uses personalized clustering to uncover coherent user journeys, and names them via prompting LLMs.

LLM to Extract User Intent Journeys

Algorithm 1 Infinite Concept Personalized Clustering (ICPC) on a User

Input: $\mathcal{H}_u = \{i_t, t = 1, \cdots, T\}$ containing list of items the user interacted with; $\epsilon \in [0, 1]$: salient terms similarity threshold; c: Minimum number of items per cluster. Default values: $\epsilon = 0.1, c = 1$.

Initialize: Journey Set $\mathcal{S}_J = \emptyset$

for $t = 1, \cdots, T$ **do**

  $\forall J \in \mathcal{S}_J$, compute ItemJourneySim$(i_t, J)$

  $J^* \leftarrow \arg\max_{J' \in \mathcal{J}}$ ItemJourneySim(i_t, J')

  if *ItemJourneySim*$(J^*, i) \geq \epsilon$ **then**

    $J^* = J^* \cup \{i_t\}$

  else

    Start a new journey $J_{\text{new}} = \{i_t\}$, $\mathcal{S}_J = \mathcal{S}_J \cup \{J_{\text{new}}\}$.

  end

  Update the journey representation based on the added item.

end

Prune journey clusters with less than c items

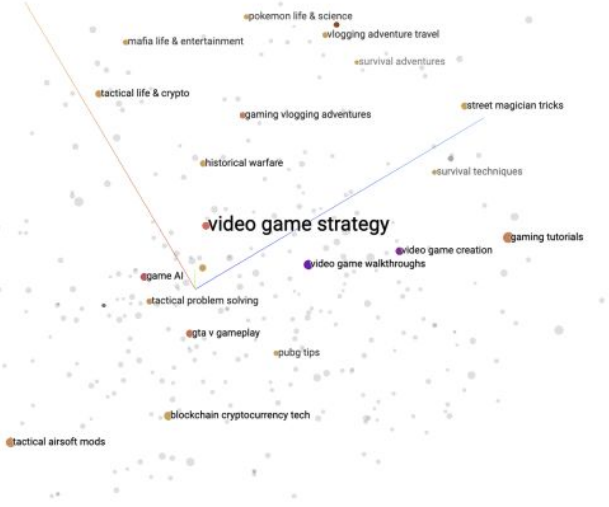


Fig. 3. Visualization of journeys across users, with each point representing an extracted and named journey. We generate embeddings for each journey name through Universal Sentence Encoder [7] and cluster them with UMAP [32]. Highlighted, a *video game strategy* journey and its nearest neighbors in the embedding space, per cosine similarity.

Summary: LLM for Feature Engineering

- The simplest thing to try is if text is available, just generate LLM embeddings for them
- If we don’t have enough text data on the entit, LLM can be leveraged for its world knowledge
- If we have a constrained set of aspects like user interests
  - We can use LLM to generate tags
  - Or at least do some summarization/extraction before featurization
- The Google’s User Journey idea could be directly leveraged on our clustering work
  - We can generate naming/categories on each cluster

Index

- High Level Overview
- LLM for Feature Engineering
- **LLM for better model Arch and Generative Training**

Mixture of Experts

- Originally proposed by Geoffrey Hinton
  - *Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer*
- Parameter efficient at serving time
  - In training time we train all the experts' parameters
  - At Serving time each request will only be served by one expert
  - The routing is controlled by a gating network
- Recently one AdKDD paper showed better performance on Ads CTR prediction

Mixture of Experts

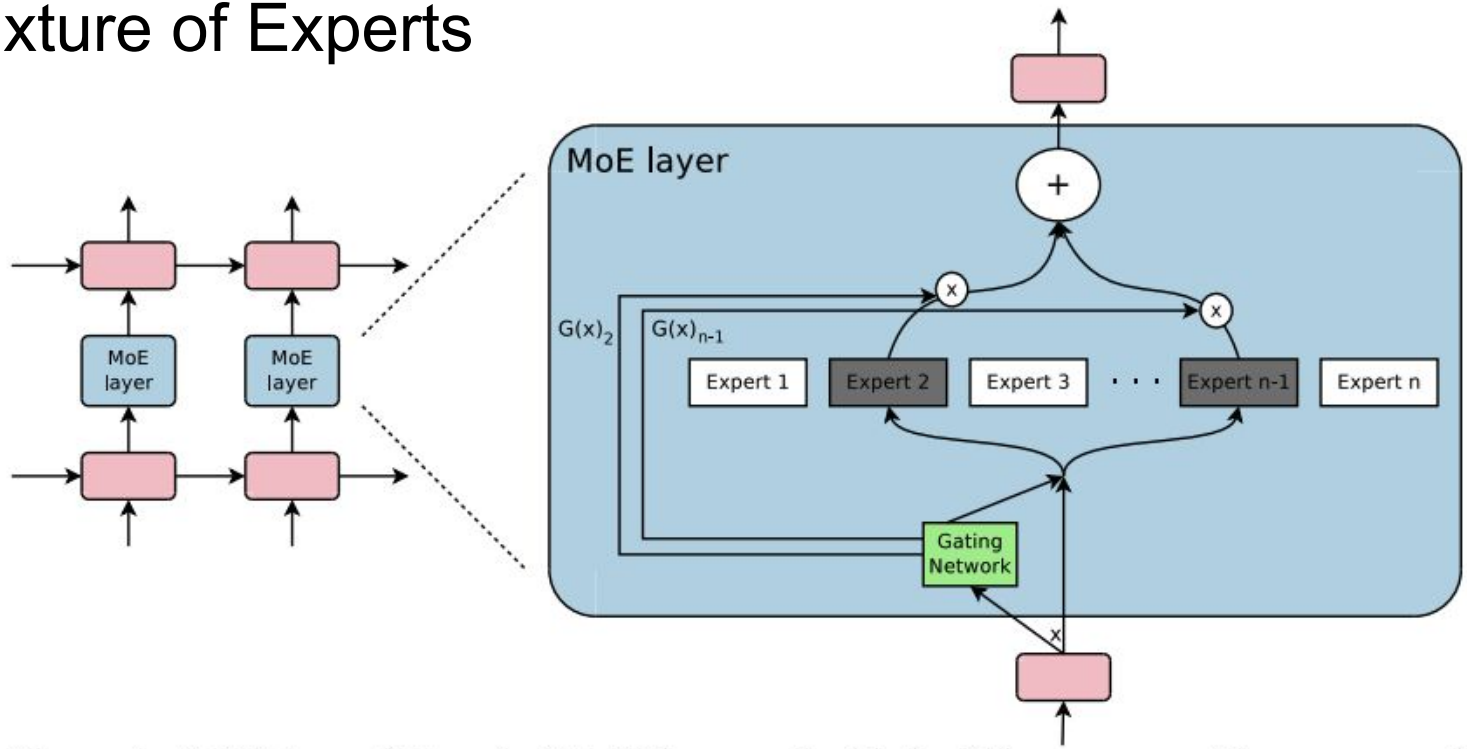


Figure 1: A Mixture of Experts (MoE) layer embedded within a recurrent language model. In this case, the sparse gating function selects two experts to perform computations. Their outputs are modulated by the outputs of the gating network.

Mixture of Experts

- Could be upcycled from a trained model, instead of training everything from scratch

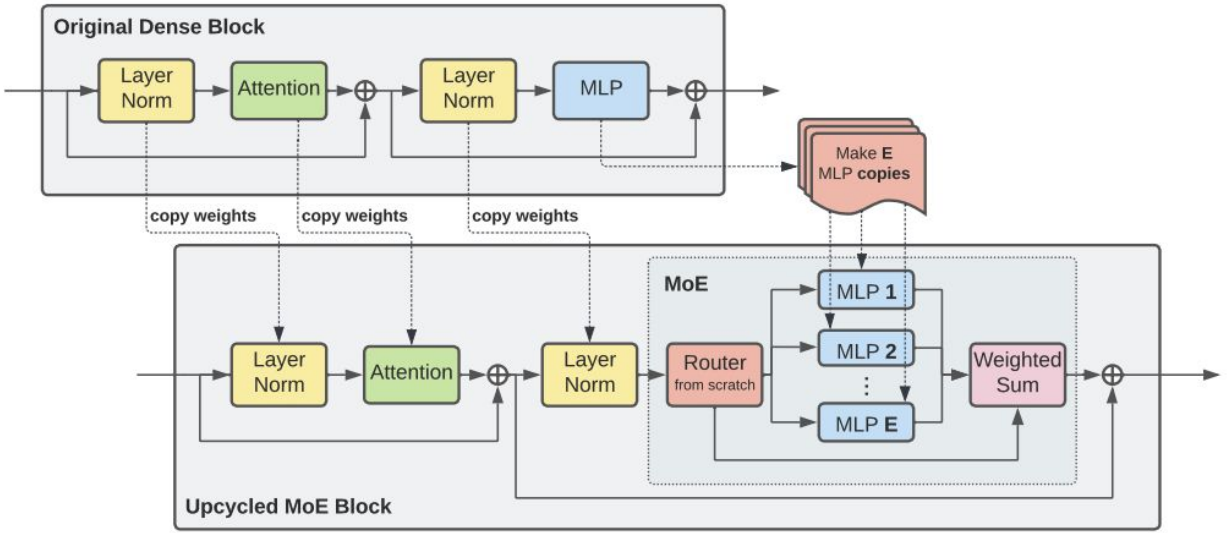


Figure 1: The upcycling initialization process. All parameters, and optionally their optimizer state, are copied from the original checkpoint, except those corresponding to the MoE router, which does not exist in the original architecture. In particular, the experts in the new MoE layer are identical copies of the original MLP layer that is replaced.

https://arxiv.org/pdf/2212.05055.pdf

Expert Choice Routing to solve the unused experts issue

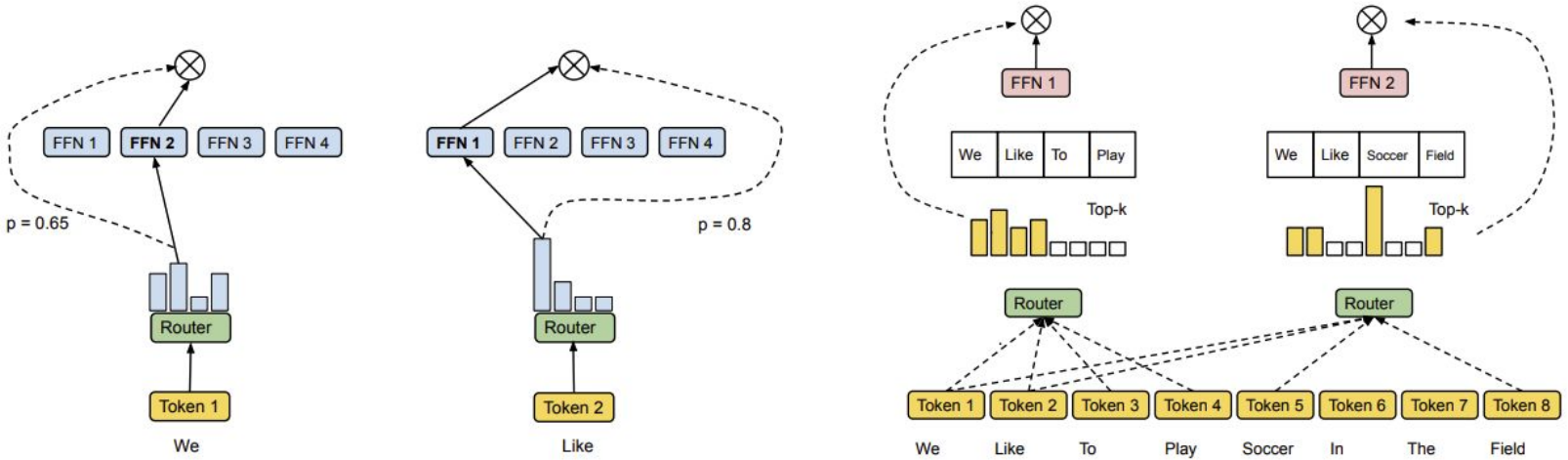


Figure 1: High-level Comparison Between Conventional MoE and expert choice MoE.

https://arxiv.org/pdf/2202.09368.pdf

Mixture of Experts

- One paper shows it's the best arch compared with other popular arch

Table 1: Performance Comparison of Different Algorithms on Criteo, Avazu and iPinYou Dataset.

	Criteo		Avazu		iPinYou	
Model	AUC	LogLoss	AUC	LogLoss	AUC	LogLoss
LR	0.7924	0.4577	0.7533	0.3952	0.7692	0.005605
FM	0.8030	0.4487	0.7652	0.3889	0.7737	0.005576
DNN	0.8051	0.4461	0.7627	0.3895	0.7732	0.005749
Wide&Deep	0.8062	0.4451	0.7637	0.3889	0.7763	0.005589
DeepFM	0.8069	0.4445	0.7665	0.3879	0.7749	0.005609
DeepCrossing	0.8068	0.4456	0.7628	0.3891	0.7706	0.005657
DCN	0.8056	0.4457	0.7661	0.3880	0.7758	0.005682
PNN	0.8083	0.4433	0.7663	0.3882	0.7783	0.005584
xDeepFM	0.8077	0.4439	0.7668	0.3878	0.7772	0.005664
AutoInt	0.8053	0.4462	0.7650	0.3883	0.7732	0.005758
FiBiNET	0.8082	0.4439	0.7652	0.3886	0.7756	0.005679
xDeepInt	0.8111	0.4408	0.7672	0.3876	0.7790	0.005567
DCN V2	0.8086	0.4433	0.7662	0.3882	0.7765	0.005593
AdaEnsemble	**0.8132**	**0.4394**	**0.7687**	**0.3865**	**0.7807**	**0.005550**

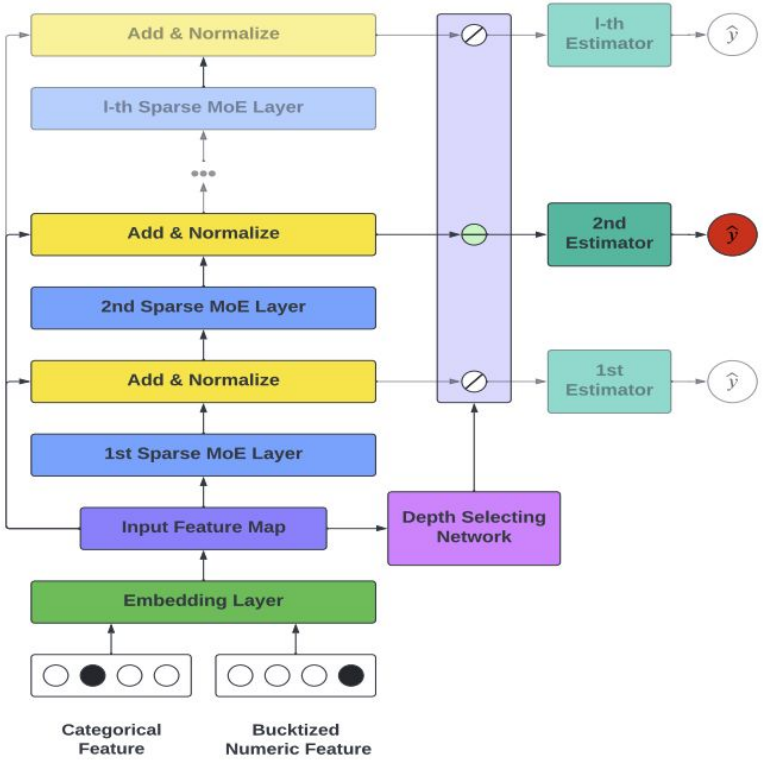


Figure 1: The Architecture of AdaEnsemble

In this example, the depth selecting network selects the 2nd layer to exit and compute the final prediction, therefore the deeper layers was not activated and plotted translucent in the figure.

Generative Pre-training : Amazon Paper

- Use Autoregressive Sequence Modeling to training a user representation
- Scaling law observed
- Only clicks events being used
- Different features on the same event are featurized then concatenated
- Combine regular Cross-Entropy loss with contrastive loss when the cardinality is large
- Each feature is predicted and loss calculated

Generative Pre-training : Amazon Paper

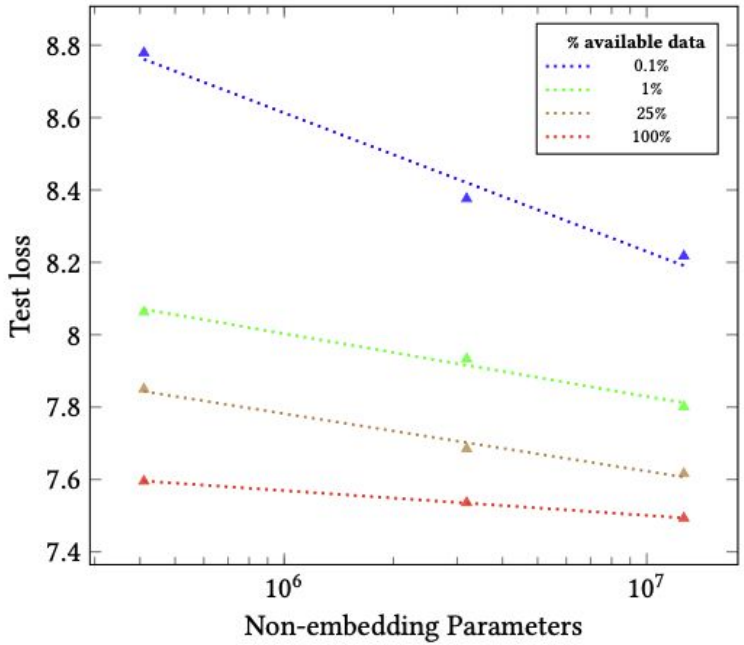


Figure 2: Data Scaling

Table 5: Lift over downstream task performance relative to 54K model

Params	IVR @ fixed FPR	pConversion AUC
3,186,432	+1.63 %	+0.02 %
6,359,552	+3.44 %	+2.51 %
85,136,640	+4.09 %	+3.57 %

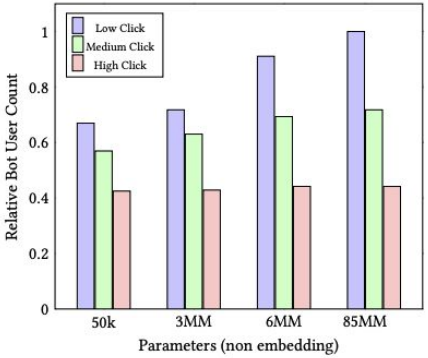


Figure 5: Relative count of bot accounts flagged across click sequence lengths

Meta Paper: Generative Pre-training

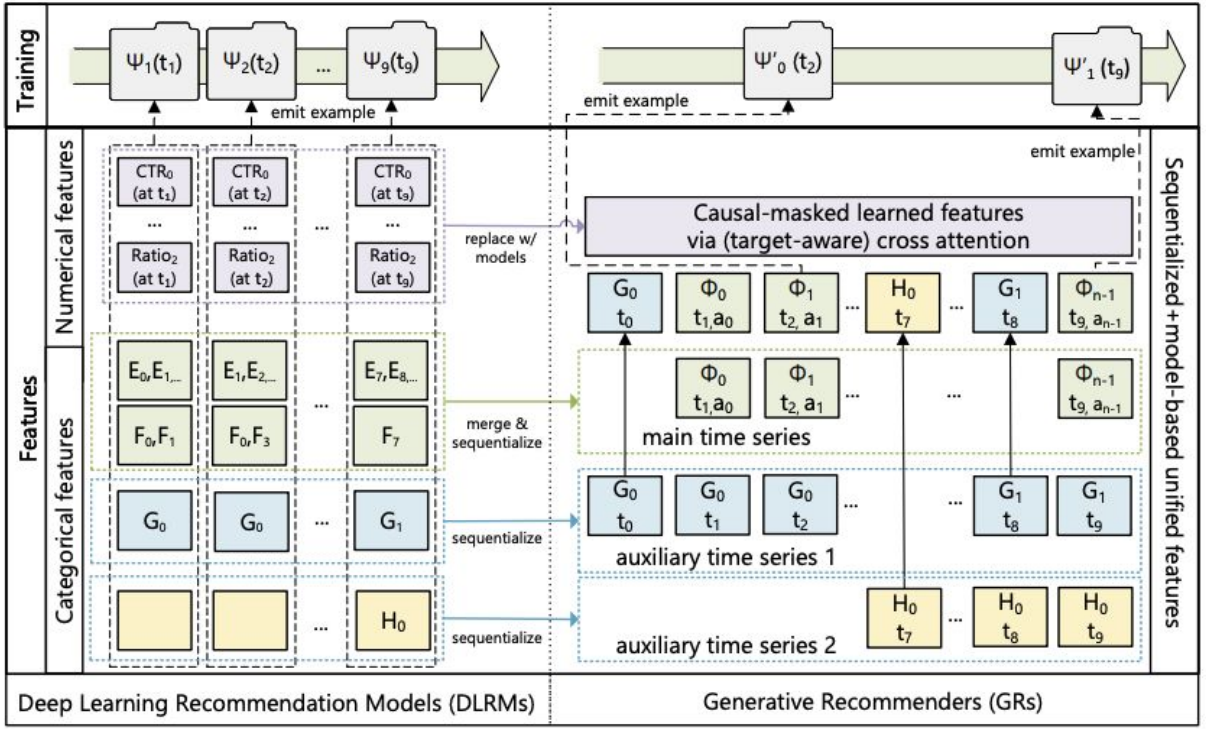


Figure 2. Comparison of features and training procedures: DLRMs vs GRs. E, F, G, H denote categorical features. Φ_i represents the i-th item in the merged main time series. $\Psi_i(t_j)$ denotes training example i emitted at time j.

Given a list of tokens $x_0, x_1, \ldots, x_{n-1}$ ordered chronologically, the time when those tokens are observed $t_0, t_1, \ldots, t_{n-1}$, and other metadata (such as user actions on those tokens a_is, if x_i is a token from the main time series discussed in Section 2.1), a sequential transduction task maps those input sequences to the output tokens $y_0, y_1, \ldots, y_{n-1}$ subject to a mask sequence $m_0, m_1, \ldots, m_{n-1}$ ($m_i \in \{0, 1\}$), where $m_i = 0$ indicates that y_i is undefined. The input tokens come from a dynamic, non-stationary vocabulary $\mathbb{X}$ where the contents (e.g., videos that users engage with) are $\mathbb{X}_c \subseteq \mathbb{X}$. Full notations can be found in Appendix A. We consider causal masked autoregressive settings for these tasks throughout the rest of this section.

Retrieval. In GRs, retrieval tasks learn a distribution $p(x_{i+1}|u_i)$ for each user over $x_{i+1} \in \mathbb{X}_c$, where u_i is the user's representation at step i. A typical objective is to select $\arg\max_{x \in \mathbb{X}_c} p(x|u_i)$ to maximize a specific reward. This differs from a standard autoregressive setup in two ways. First, the supervision for x_i, y_i, is not necessarily x_{i+1}, as users could respond negatively to x_{i+1}. Second, y_i is undefined in situations where the next token represents a non-engagement related categorical feature, such as demographics ($x_{i+1} \notin \mathbb{X}_c$). For these cases, we set $m_i = 0$.

Ranking. Ranking tasks in GRs pose unique challenges as modern recommendation systems often require a "target-aware" formulation. In such a formulation, "interaction" of target, x_{i+1}, and historical features up to i needs to occur as early as possible, which is infeasible with a standard autoregressive setup where "interaction" typically happens late (e.g., via softmax after encoder output). We address this problem by *interleaving* items and actions in the main time series. The resulting new time series (before categorical features $x \notin \mathbb{X}_c$) is then $x_0, a_0, x_1, a_1, \ldots, x_{n-1}, a_{n-1}$, where mask m_is are 0s for the action positions. We apply a small neural network to transform predictions at content positions into multi-task predictions in practice. This approach enables us to apply target-aware cross-attention to all n (user, item) engagements in one pass with causal masking.

HSTU: High Performance Self-Attention Encoder

- If you are interested, there's already a PR on using this in oCPM model:
  - https://github.com/pinternal/pinboard/pull/55635

3.1. Pointwise aggregated attention

HSTU adopts a new pointwise aggregated attention mechanism instead of softmax attention in Transformers. This is motivated by two factors. First, in recommendations, the number of prior data points related to target serves as a strong feature indicating the intensity of user preferences, which is hard to capture after the softmax normalization. This is critical as we need to predict both the intensity of engagements, e.g., time spent on a given item, and the relative ordering of the items, e.g., predicting an ordering of candidates to maximize AUC. Second, while softmax activation is robust to noise by construction, it is less suited for non-stationary vocabularies in streaming settings.

$$U(X), V(X), Q(X), K(X) = \text{Split}(\phi_1(f_1(X))) \quad (1)$$

$$A(X)V(X) = \phi_2\left(Q(X)K(X)^T + \text{rab}^{p,t}\right)V(X) \quad (2)$$

$$Y(X) = f_2\left(\text{Norm}\left(A(X)V(X)\right) \odot U(X)\right) \quad (3)$$

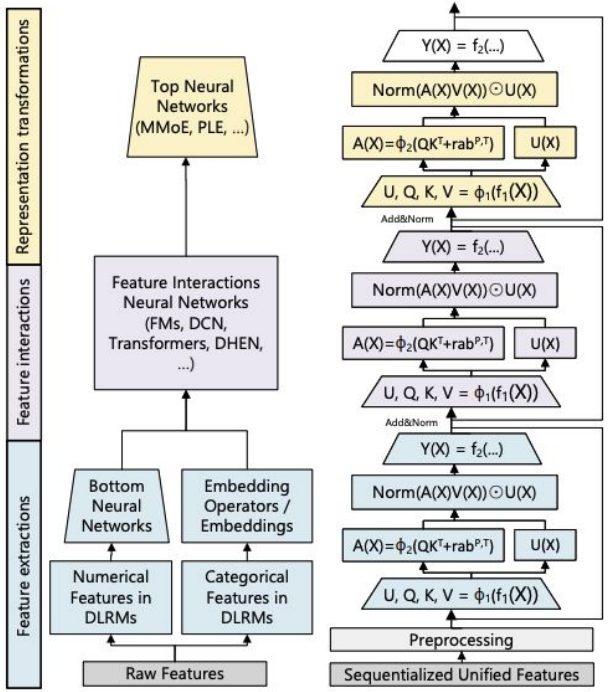


Figure 3. Comparison of key model components: DLRMs vs GRs. The complete DLRM setup (Mudigere et al., 2022) is shown on the left side and a simplified HSTU is shown on the right.

Sequence Sampling

Specifically, let $\Gamma(n, L)$ be a function that selects a subsequence of length L from the original sequence $x_0, \ldots, x_{n-1}$. SL selects input sequences as follows:

$$\begin{cases} x_0, \ldots, x_{n_i-1} \text{ if } n_i \leq N^{\alpha/2} \\ \Gamma(n_i, N^{\alpha/2}) \text{ if } n_i > N^{\alpha/2}, \text{ w/ probability } 1 - N^{\alpha}/n_i^2 \\ x_0, \ldots, x_{n_i-1} \text{ if } n_i > N^{\alpha/2}, \text{ w/ probability } N^{\alpha}/n_i^2 \end{cases} \quad (4)$$

Table 3. Evaluations of methods on public datasets in multi-pass, full-shuffle settings.

	Method	HR@10	HR@50	HR@200	NDCG@10	NDCG@200
ML-1M	SASRec (2023)	.2828	.5508	.7522	.1545	.2441
	HSTU	.3043 (+7.6%)	.5728 (+4.0%)	.7740 (+2.9%)	.1700 (+10.1%)	.2601 (+6.6%)
	HSTU-large	**.3306 (+16.9%)**	**.5897 (+7.1%)**	**.7832 (+4.1%)**	**.1858 (+20.3%)**	**.2730 (+11.9%)**
ML-20M	SASRec (2023)	.2906	.5499	.7655	.1621	.2521
	HSTU	.3252 (+11.9%)	.5885 (+7.0%)	.7943 (+3.8%)	.1878 (+15.9%)	.2774 (+10.0%)
	HSTU-large	**.3567 (+22.8%)**	**.6149 (+11.8%)**	**.8076 (+5.5%)**	**.2106 (+30.0%)**	**.2971 (+17.9%)**
Books	SASRec (2023)	.0292	.0729	.1400	.0156	.0350
	HSTU	.0404 (+38.4%)	.0943 (+29.5%)	.1710 (+22.1%)	.0219 (+40.6%)	.0450 (+28.6%)
	HSTU-large	**.0469 (+60.6%)**	**.1066 (+46.2%)**	**.1876 (+33.9%)**	**.0257 (+65.8%)**	**.0508 (+45.1%)**

Table 4. Evaluation of HSTU, ablated HSTU, and Transformers on industrial-scale datasets in one-pass streaming settings.

Architecture	Retrieval	Ranking (NE)	
	log pplx.	E-Task	C-Task
Transformers	4.069	NaN	NaN
HSTU (-rab^{p,t}, Softmax)	4.024	.5067	.7931
HSTU (-rab^{p,t})	4.021	.4980	.7860
Transformer++	4.015	.4945	.7822
HSTU (original rab)	4.029	.4941	.7817
HSTU	**3.978**	**.4937**	**.7805**

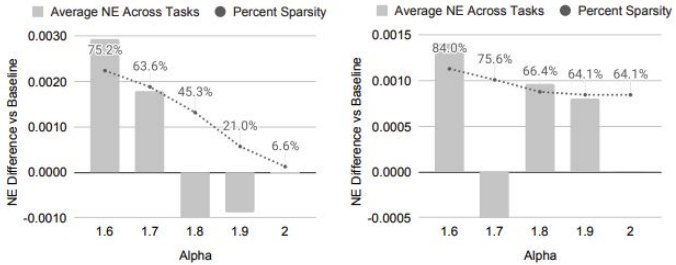
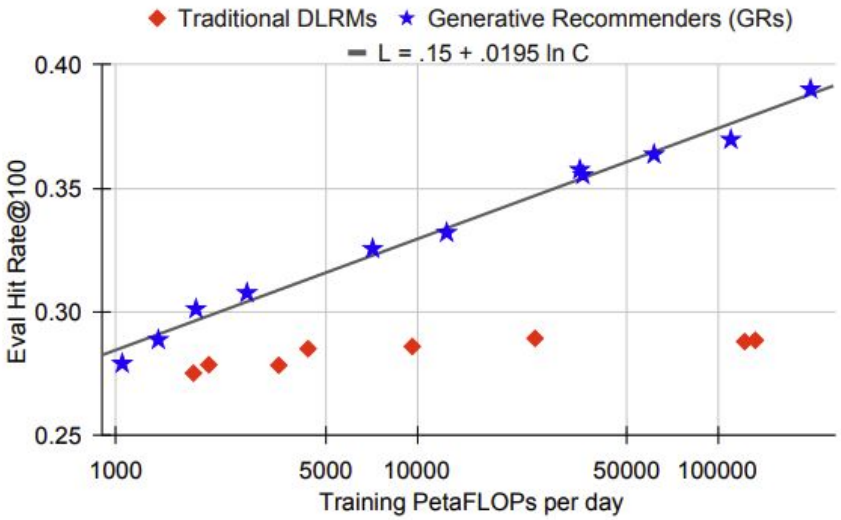


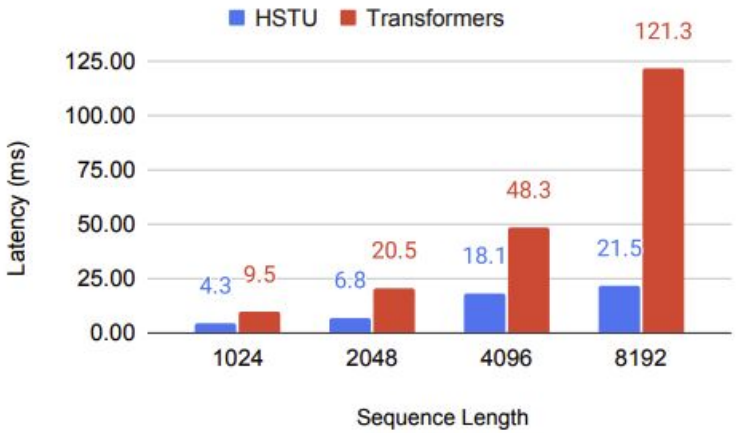
Table 5. Offline/Online Comparison of Retrieval Models.

Methods	Offline HR@K		Online metrics	
	K=100	K=500	E-Task	C-Task
DLRM	29.0%	55.5%	+0%	+0%
DLRM (abl. features)	28.3%	54.3%	–	
GR (content-based)	11.6%	18.8%	–	
GR (interactions only)	35.6%	61.7%	–	
GR (new source)	**36.9%**	**62.4%**	**+6.2%**	**+5.0%**
GR (replace source)			**+5.1%**	**+1.9%**

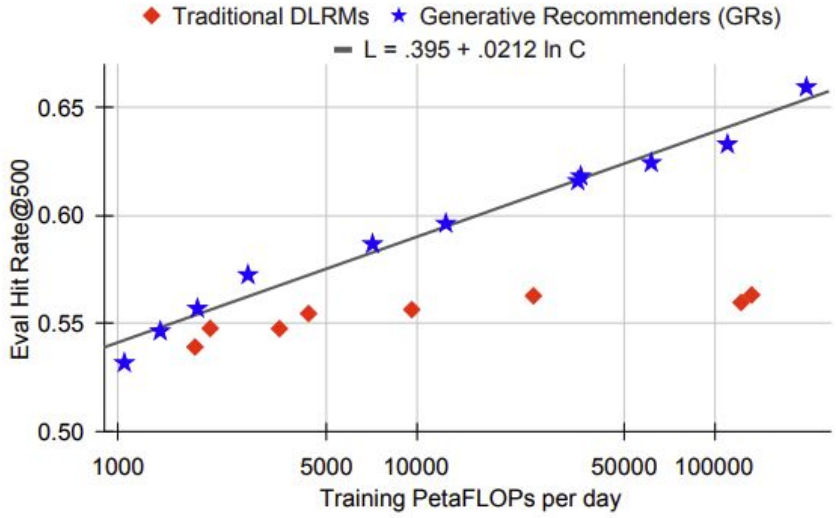
Table 6. Offline/Online Comparison of Ranking Models.

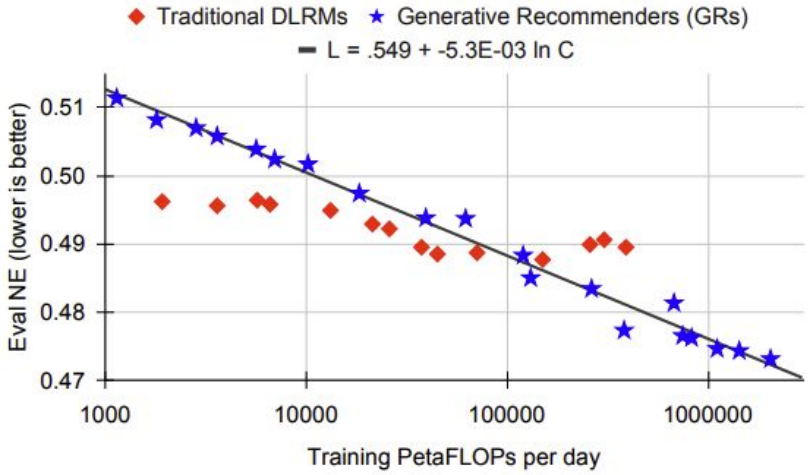
Methods	Offline NEs		Online metrics	
	E-Task	C-Task	E-Task	C-Task
DLRM	.4982	.7842	+0%	+0%
DLRM (abl. features)	.5053	.7925	–	–
GR (interactions only)	.4851	.7903	–	–
GR	**.4845**	**.7645**	**+12.4%**	**+4.4%**





(c) Inference Speedup.





Reference Papers

List of papers:

- Reviews on academic research on Generative RecSys (paper 1 , paper 2, paper 3)
- A good survey and a paper repository:
  - https://github.com/CHIANGEL/Awesome-LLM-for-RecSys
- Model Arch
  - A borrowed idea (Mixture-Of-Experts) and how it's improving Ads Models (paper)
- Generative Training
  - Amazon paperin AdKDD 23' on generative pre-training on Ads sequences (paper)
  - Meta paper of a large-scale generative recommendation model (paper)
- Feature Engineering:
  - Walmart paper to use LLM to generate item features (paper)
  - Sony paper to use LLM for movie description gen (paper)