Building the Next Generation Recommendation Systems

THE 2ND WORKSHOP ON RECOMMENDATION WITH GENERATIVE MODELS, WWW 2024

Jiaqi Zhai, Rui Li on behalf of many amazing colleagues from Meta (MRS, PyTorch, AI Infra, DI, Core Systems, Discovery, IG, …)
May 13, 2024

“recommender systems … is the single largest software engine on the planet”
— Jensen Huang, NVIDIA, 02/22/2024.

Meta AI

Generative Recommenders (GRs) reinterpret main RecSys tasks within a generative framework.

Together with new algorithms like HSTU and M-FALCON, we've improved training & inference efficiency by 10x-1000x vs SotA Transformers and DLRMs.

GRs and HSTU have enabled 12.4%+ topline metric gains, and further demonstrate scaling law in industrial-scale RecSys for the first time, up to LLM compute scale.

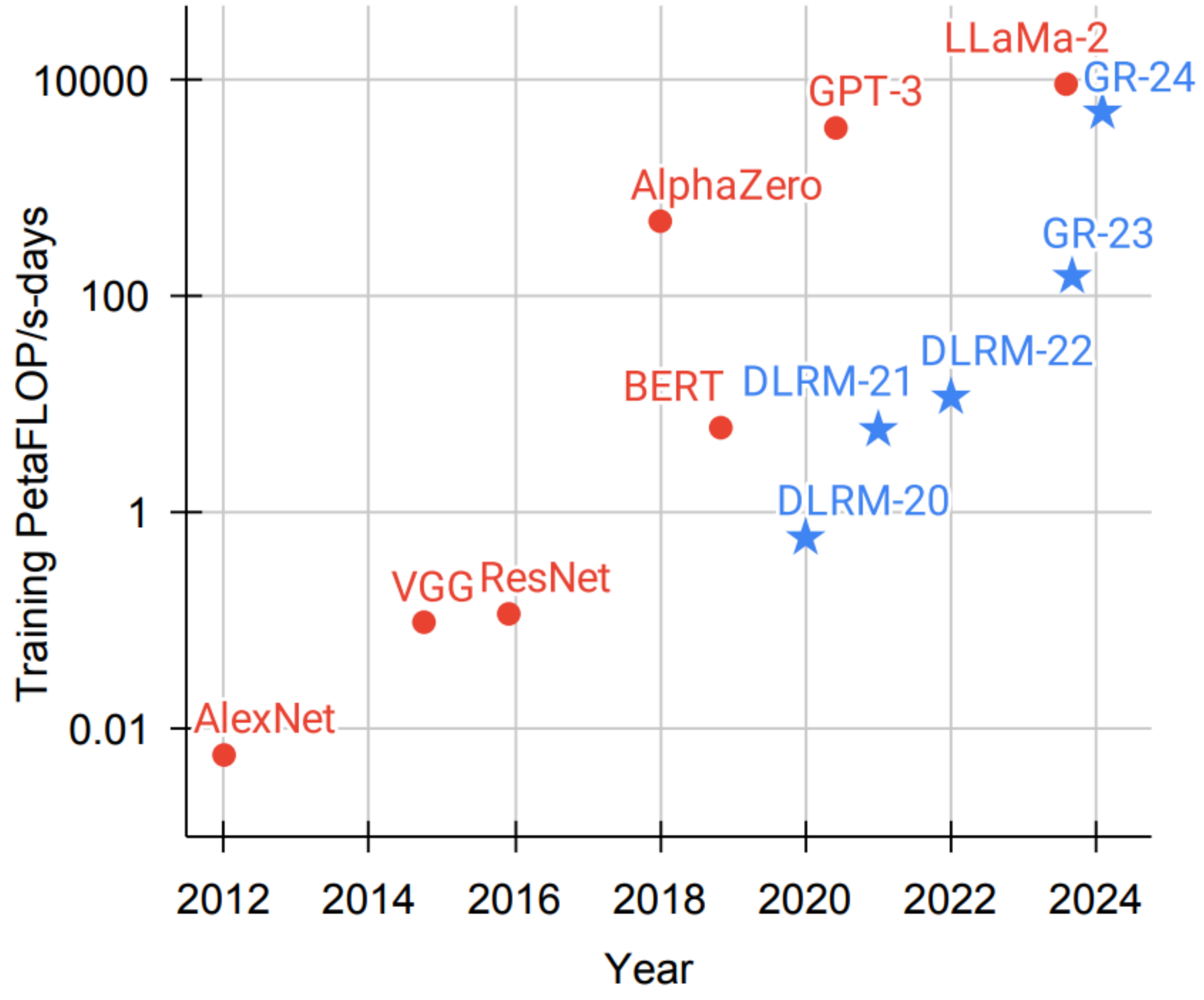


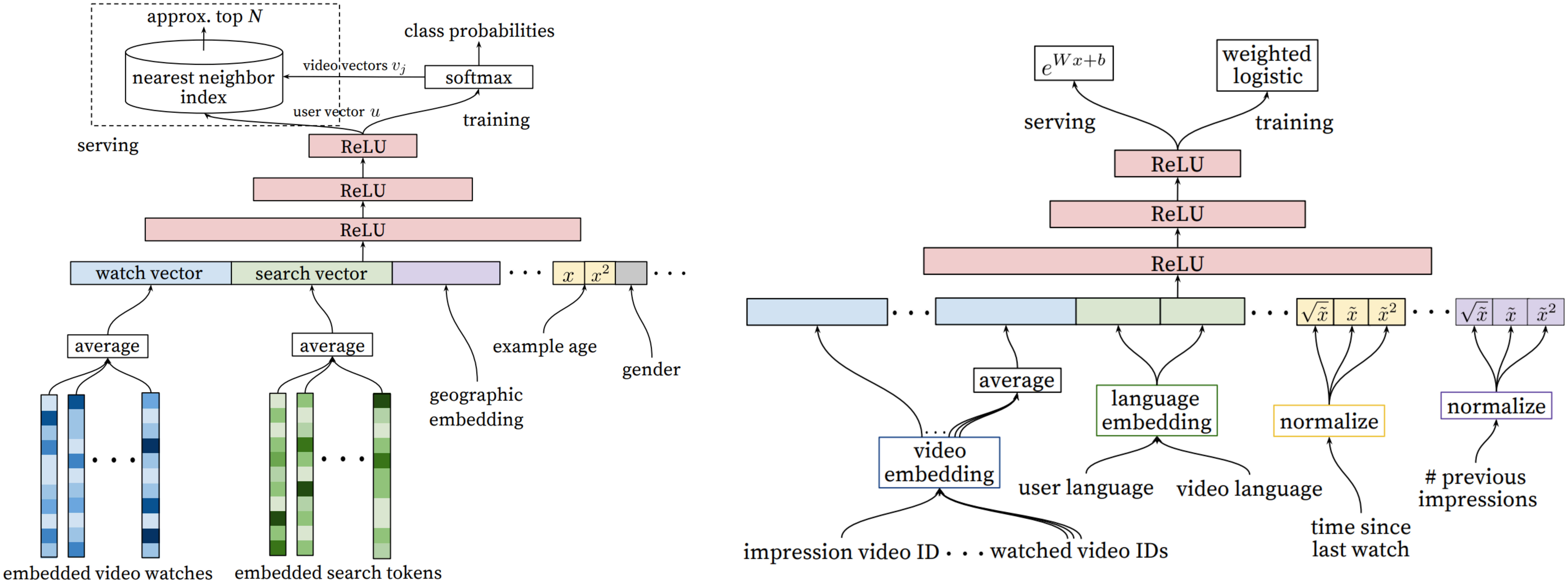
Figure 1. Total compute used to train deep learning models over the years. DLRM results are from (Mudigere et al., 2022); GRs are deployed models from this work. DLRMs/GRs are continuously trained in a streaming setting; we report compute used per year.

Meta AI

I. Background: Deep Learning Recommendation Models (DLRMs) and Generative Models

State of the World: DLRMs & Generative Models

DLRMs: classical IR paradigm (retrieval + ranking) with DNNs



(a) Retrieval. (b) Ranking.

Image credit: Covington et al. Deep Neural Networks for YouTube Recommendations. RecSys'16.

Meta AI

State of the World: DLRMs & Generative Models

Numerous improvements to DLRMs over past decade

- Feature interactions (FMs, DCN, AutoInt, DHEN/Wukong, MaskNet, …)
- Multi-task learning (MMoE, ESMM, PLE, …)
- Sequential (sub-)modules (one-stage DIN, BST, hybrid UBM, SIM, …)
- Debiasing (off-policy correction / REINFORCE, IPW / CLRec, …)
- Beyond two-tower settings (multi-interest / MIND, beam search / “generative retrieval” / TDM, OTM, DR, learned similarities / MoL, …)
- …

State of the World: DLRMs & Generative Models

Generative Models (e.g., LLMs)

- Many explored use cases in RecSys:
  - In-context Learning (e.g., LLMRank, …)
  - Instruction Tuning (e.g., M6-Rec, TALLRec, …)
  - Transfer Learning utilizing World Knowledge (e.g., NoteLLM, …)
  - …

DLRMs + Generative Models: How do we get the best of both worlds?

Classical recommendation models — DLRMs — vs LLMs

- Pros of LLMs
  - Replace feature engineering, to the extent capturable by language;
  - World knowledge benefits cold-start scenarios;
  - Scale with <u>compute</u>.
- Pros of DLRMs
  - Leverage vast number of human-engineered features;
  - Concise representations — efficient and support very long context sizes;
  - Scale with (in-domain recommendation) <u>data</u>.

DLRMs + Generative Models: How do we get the best of both worlds?

Should we build next-gen RecSys on top of current LLMs?

- World knowledge primarily benefits cold-start scenarios…
  - Needs more work to outperform collaborative filtering approaches even on MovieLens-1M.

	Method	ML-1M			
		N@1	N@5	N@10	N@20
full	Pop	0.08	1.20	4.13	5.79
full	BPRMF [49]	0.26	1.69	4.41	6.04
full	SASRec [33]	**3.76**	**9.79**	**10.45**	**10.56**
zero-shot	BM25 [50]	0.26	0.87	2.32	5.28
zero-shot	UniSRec [30]	0.88	3.46	5.30	6.92
zero-shot	VQ-Rec [29]	0.20	1.60	3.29	5.73
zero-shot	Ours	**1.74**	**5.22**	**6.91**	**7.90**

Image Credits: top: Hou et al. Large Language Models are Zero-Shot Rankers for Recommender Systems. ECIR'24.
(Best known ML-1M NDCG@10 as of 05/2024 is 18.9 (paperswithcode), vs LLM zero-shot 6.91)
Bottom: Chang et al. TWIN: TWo-stage Interest Network for Lifelong User Behavior Modeling in CTR Prediction at Kuaishou. KDD'23.

DLRMs + Generative Models: How do we get the best of both worlds?

Should we build next-gen RecSys on top of current LLMs?

- World knowledge primarily benefits cold-start scenarios…
  - Needs more work to outperform collaborative filtering approaches even on MovieLens-1M.
- Tokenization needs to become *orders of magnitude* more efficient…
  - Modern DLRMs often need to handle 10K-100K scale engagement history.

	Method	ML-1M N@1	ML-1M N@5	ML-1M N@10	ML-1M N@20
full	Pop	0.08	1.20	4.13	5.79
full	BPRMF [49]	0.26	1.69	4.41	6.04
full	SASRec [33]	**3.76**	**9.79**	**10.45**	**10.56**
zero-shot	BM25 [50]	0.26	0.87	2.32	5.28
zero-shot	UniSRec [30]	0.88	3.46	5.30	6.92
zero-shot	VQ-Rec [29]	0.20	1.60	3.29	5.73
zero-shot	Ours	**1.74**	**5.22**	**6.91**	**7.90**

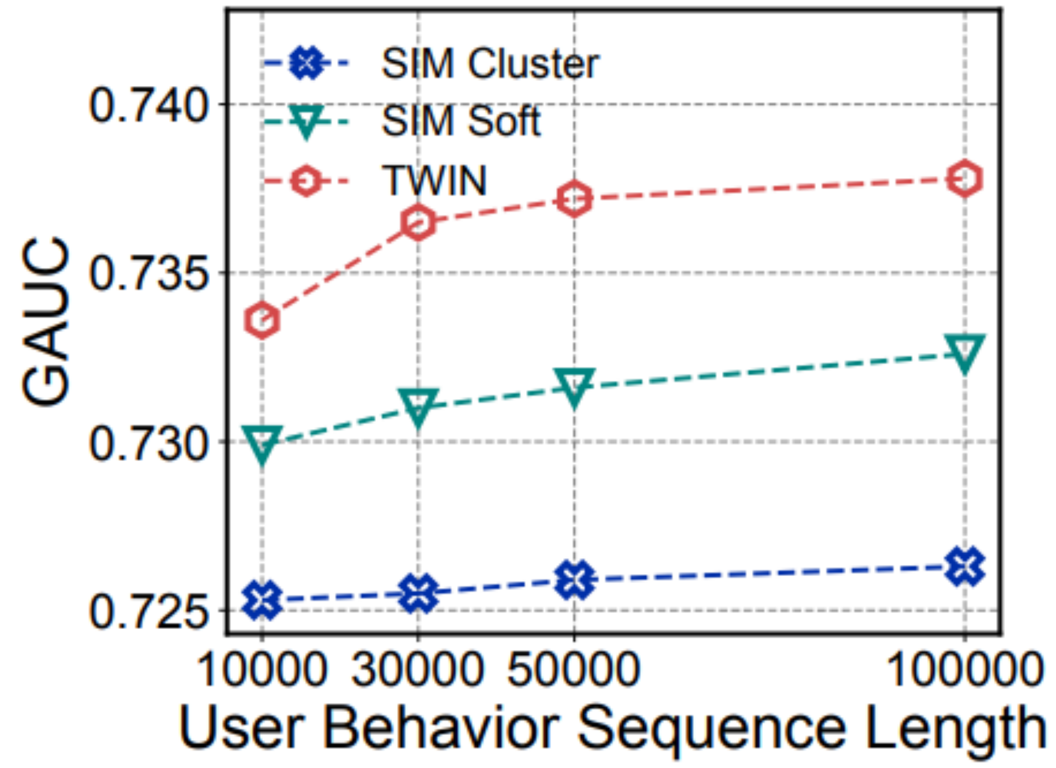


Image Credits: top: Hou et al. Large Language Models are Zero-Shot Rankers for Recommender Systems. ECIR'24.
(Best known ML-1M NDCG@10 as of 05/2024 is 18.9 (paperswithcode), vs LLM zero-shot 6.91)
Bottom: Chang et al. TWIN: TWo-stage Interest Network for Lifelong User Behavior Modeling in CTR Prediction at Kuaishou. KDD'23.

DLRMs + Generative Models: How do we get the best of both worlds?

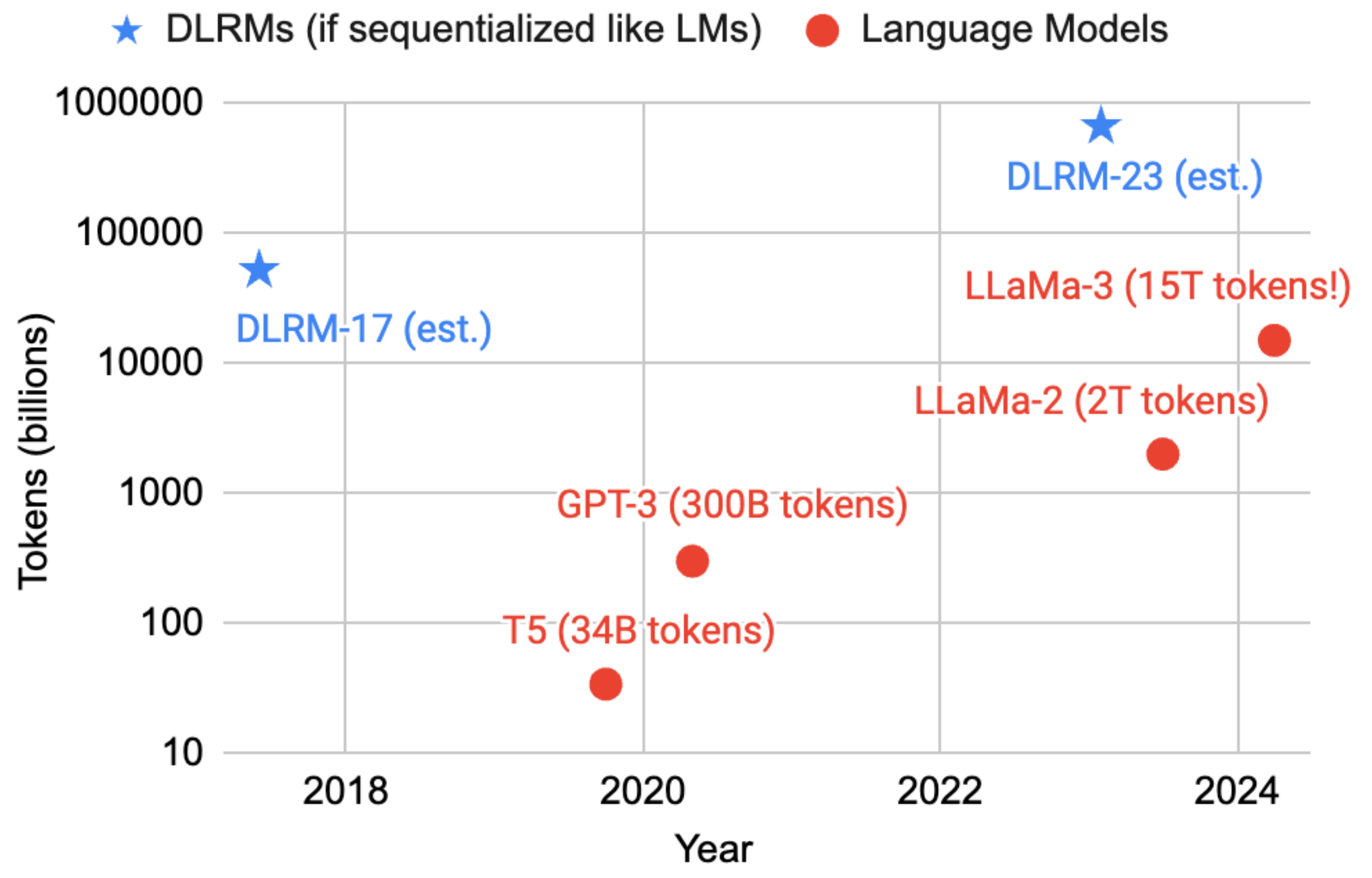
What about a deeper integration… like a “generative” DLRM??

- **Features**: vast number (1K-10K scale); lack explicit structures.
- **Vocabulary**: billion-scale continuously updated in a streaming setting. Invalidates assumptions in generative models and LMs (100K static vocabulary).
- **Cost**: large models utilize huge amount of training data. 300B tokens in GPT-3, 15T in LLaMa-3…

DLRMs + Generative Models: How do we get the best of both worlds?

What about a deeper integration… like a "generative" DLRM??

- **Features**: vast number (1K-10K scale); lack explicit structures.
- **Vocabulary**: billion-scale continuously updated in a streaming setting. Invalidates assumptions in generative models and LMs (100K static vocabulary)
- **Cost**: large models utilize huge amount of training data. 300B tokens in GPT-3, 15T in LLaMa-3…
    - But we generate 100T-1000T tokens *every day* in RecSys!



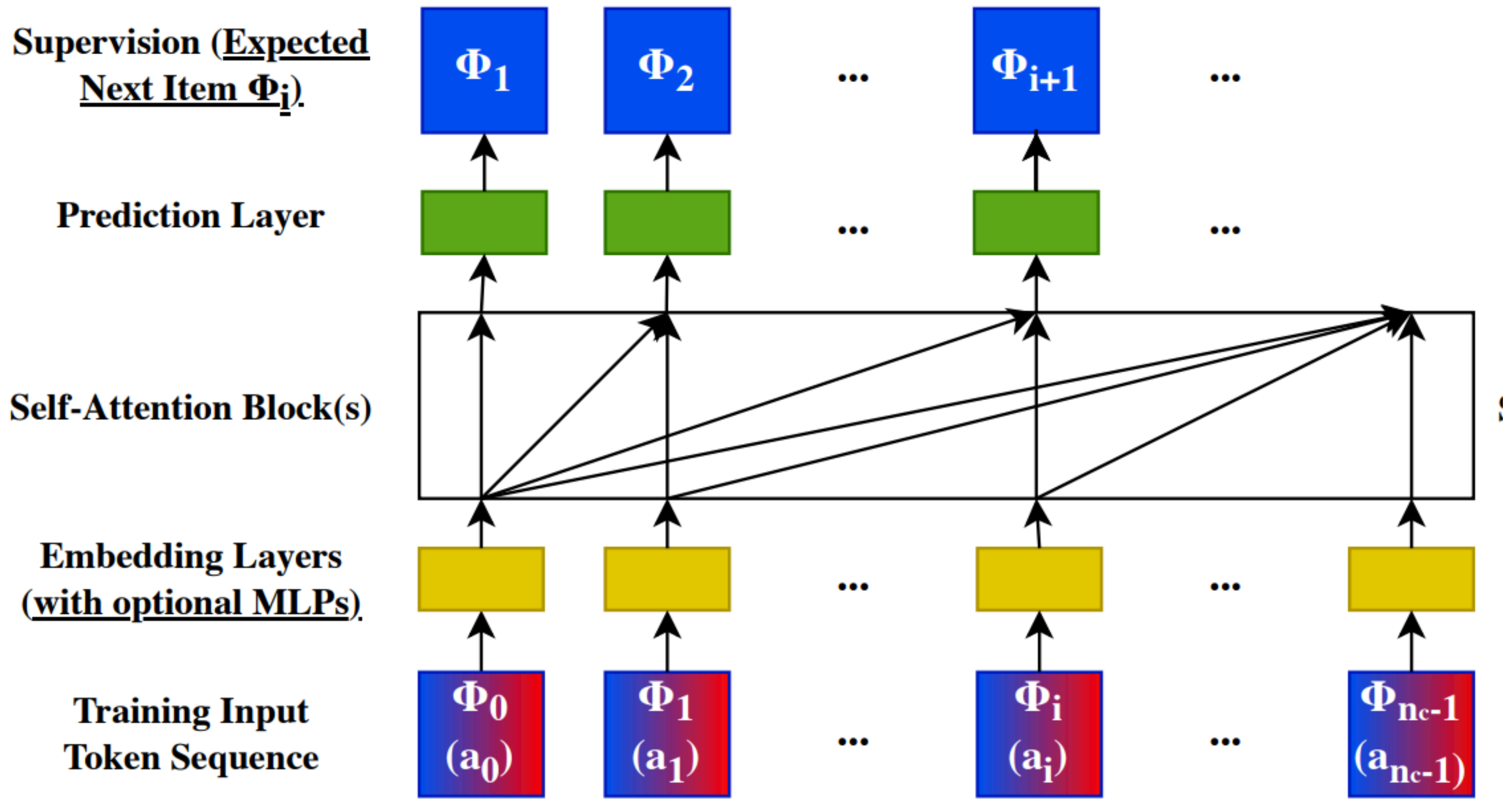
Meta AI

II. Our Solution: DLRMs + Generative Models => Generative Recommenders

Revisiting Formulations: Why Did Prior Sequential Approaches Fail?

How were sequential information utilized previously?

- Academic research - sequential recommenders (e.g., GRU4Rec*, SASRec*, BERT4Rec, …)
  - $(\Phi_0, a_0), \ldots, (\Phi_{i-1}, a_{i-1})$ -> Φ_i
    - => (causal autoregressive*) pointwise retrieval

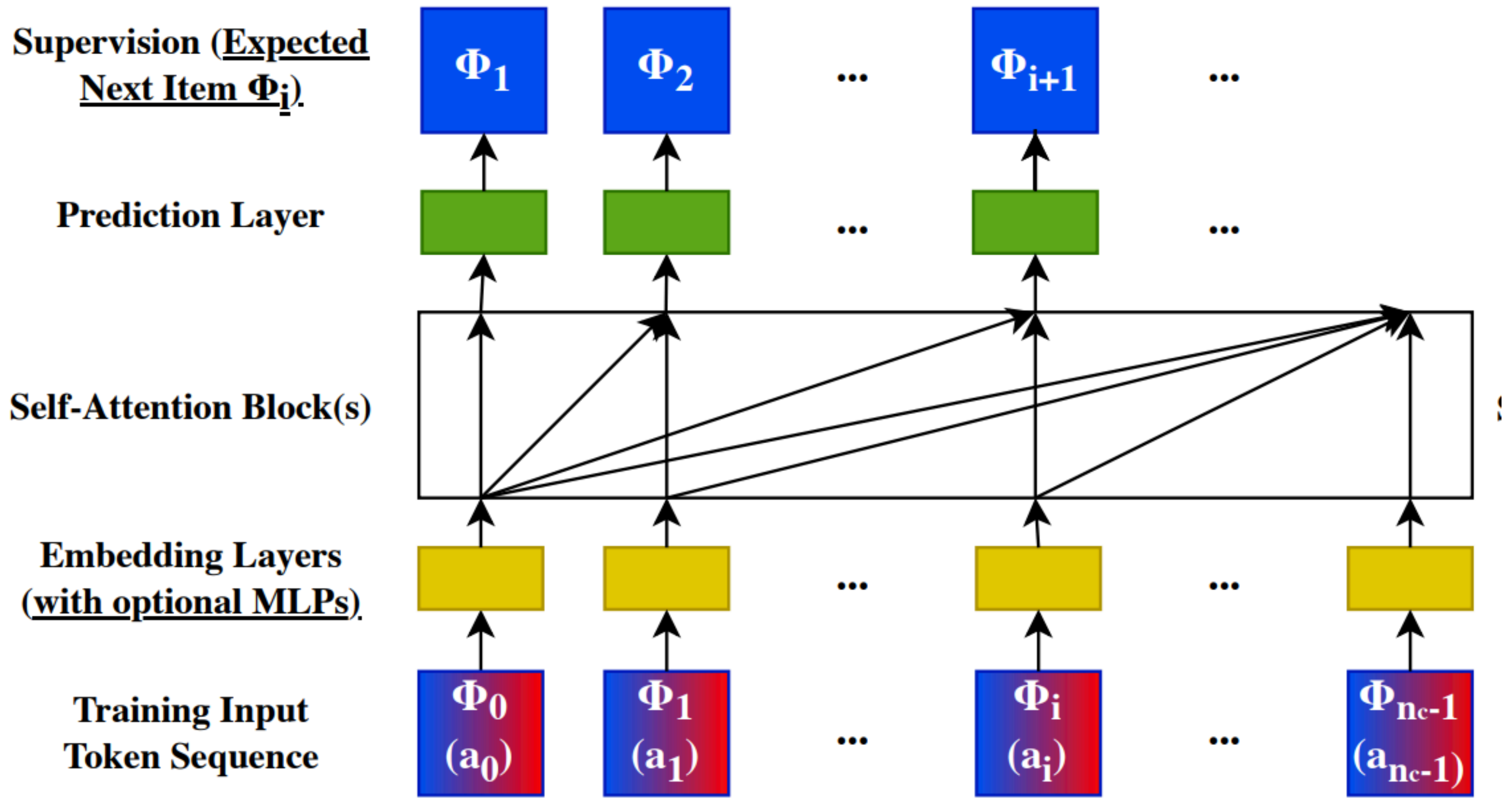


Meta AI

Revisiting Formulations: Why Did Prior Sequential Approaches Fail?

How were sequential information utilized previously?

- Academic research - sequential recommenders (e.g., GRU4Rec*, SASRec*, BERT4Rec, …)
  - $(\Phi_0, a_0), \ldots, (\Phi_{i-1}, a_{i-1})$ -> Φ_i
    - => (causal autoregressive*) pointwise retrieval
- Practical applications - DLRMs with sequential (sub-)modules (DIN, BST, SIM, …)
  - $(\Phi_0, a_0), \ldots, (\Phi_{i-1}, a_{i-1}), \Phi_i$ -> a_i
    - => pointwise ranking



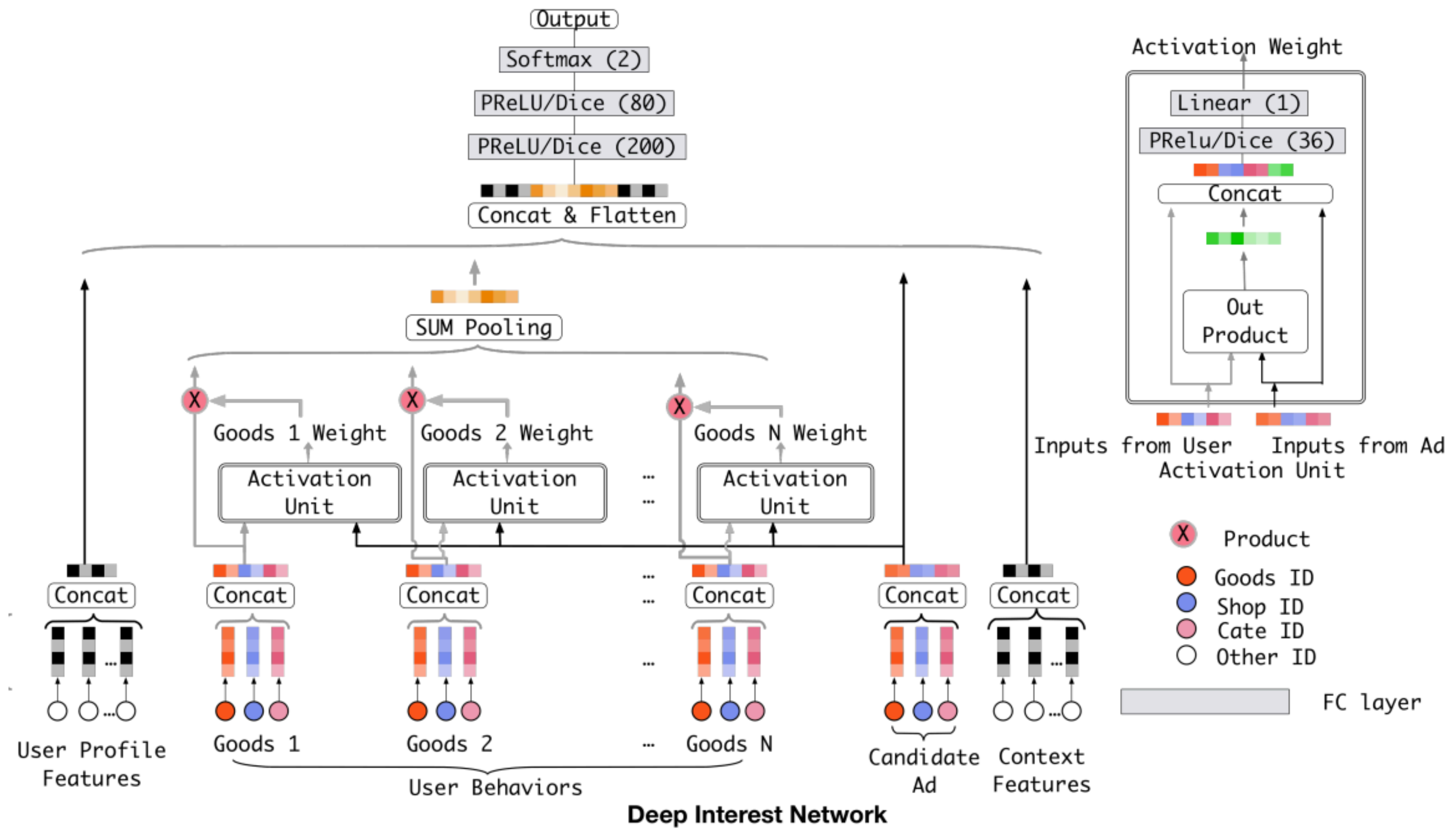


Image credit: (bottom) Zhou et al. Deep Interest Network for Click-Through Rate Prediction. KDD’18

Revisiting Formulations: Why Did Prior Sequential Approaches Fail?

Critical expressiveness gap between sequential recommenders & DLRMs

- Features, and … lots of them!
  - Need to engineer and to utilize a very large number of features (often 10K scale, vs ~1 in trad. sequential settings)

Revisiting Formulations: Why Did Prior Sequential Approaches Fail?

Critical expressiveness gap between sequential recommenders & DLRMs

- Features, and … lots of them!
    - Need to engineer and to utilize a very large number of features (often 10K scale, vs ~1 in trad. sequential settings)
    - This is why feature interaction has been the primary research focus in DLRMs (DeepFM, AFM, xDeepFM, DCN, AutoInt, DHEN, MaskNet, …)

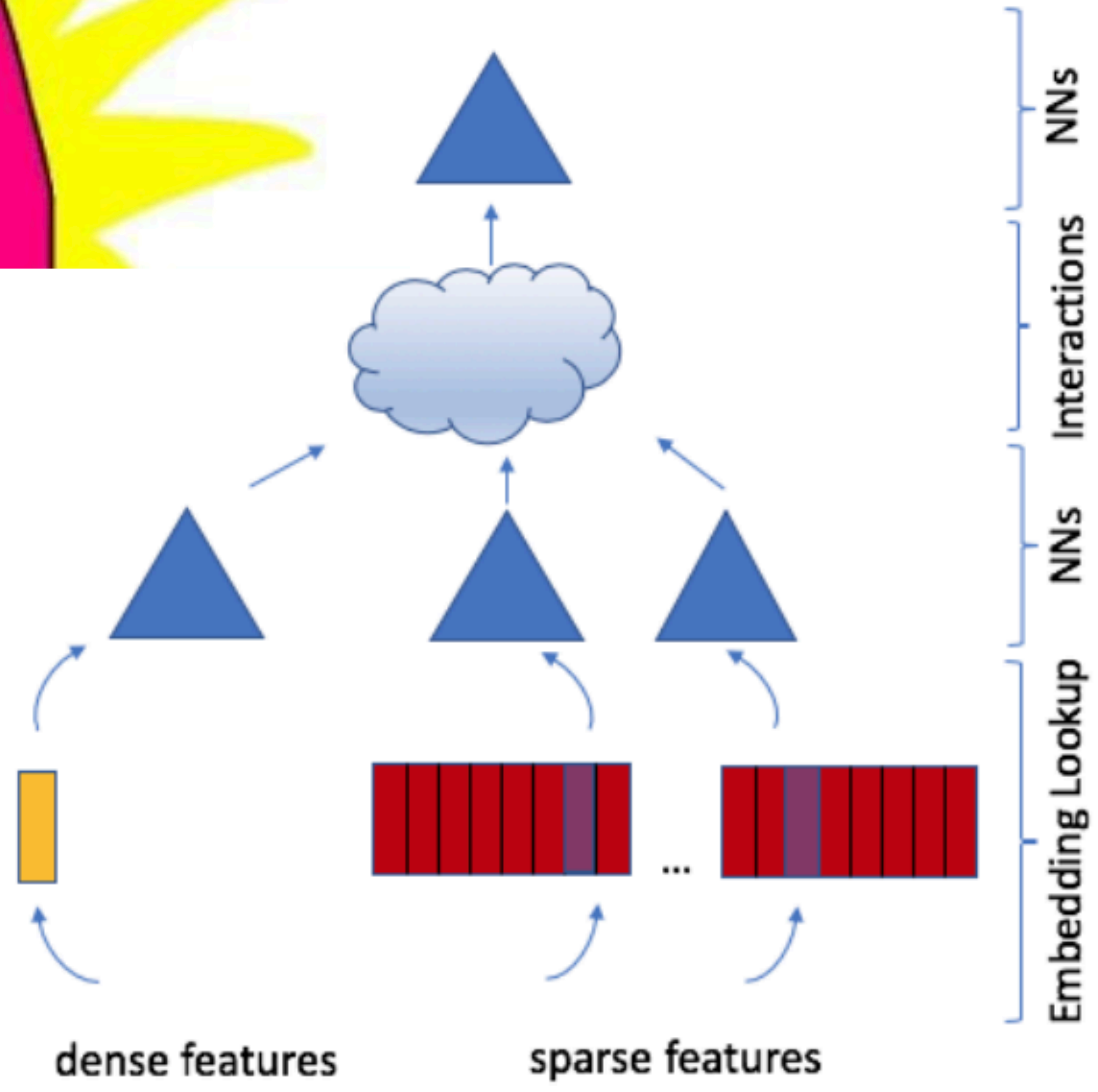


Image credit: Naumov et al. Deep Learning Recommendation Model for Personalization and Recommendation Systems. 2019.

Revisiting Formulations: Why Did Prior Sequential Approaches Fail?

Critical expressiveness gap between sequential recommenders & DLRMs

- Features, and … lots of them!
    - Need to engineer and to utilize a very large number of features (often 10K scale, vs ~1 in trad. sequential settings)
    - This is why feature interaction has been the primary research focus in DLRMs (DeepFM, AFM, xDeepFM, DCN, AutoInt, DHEN, MaskNet, …)
- Examples
    - Good prior for pCTR on a travel video? => user's historical CTR!
    - Am I likely to share a Singapore travel video to my friends? Look at the places I've been to and the items I've shared in the past …

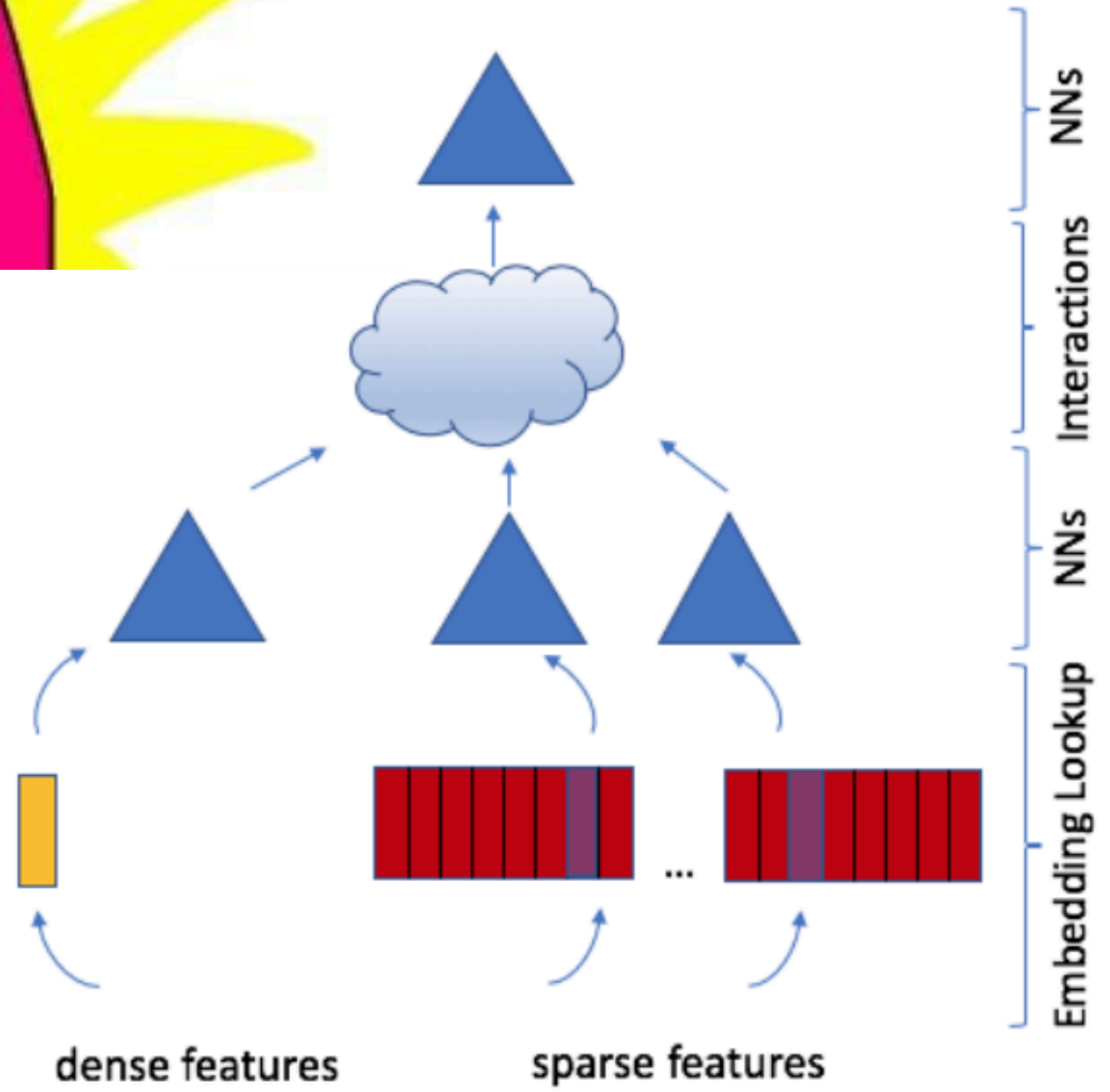


Image credit: Naumov et al. Deep Learning Recommendation Model for Personalization and Recommendation Systems. 2019.

Meta AI

DLRMs + Generative Models => Generative Recommenders (GRs)

How do we close this gap and make sequential methods work?

- We have a related solution: “target-aware attention”, widely used in most industrial DLRMs…
    - Pairwise/cross attention can help with extracting categorical/numerical cross features!

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

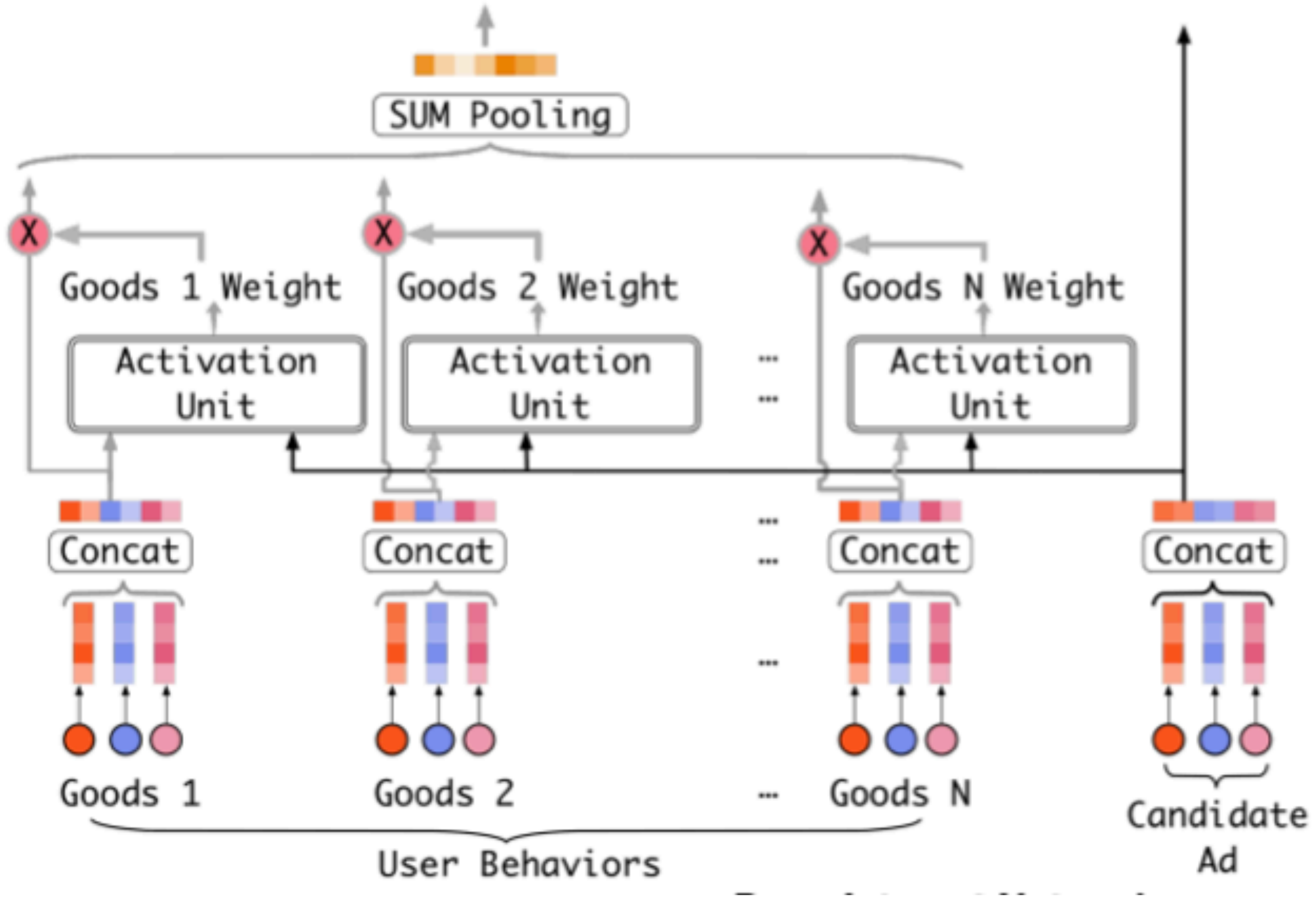


Image credit: (top) Zhai et al. Actions Speak Louder than Words: Trillion-Parameter Sequential Transducers for Generative Recommendations. ICML’24.
(bottom) Zhou et al. Deep Interest Network for Click-Through Rate Prediction. KDD’18.

DLRMs + Generative Models => Generative Recommenders (GRs)

How do we close this gap and make sequential methods work?

- We have a related solution: “target-aware attention”, widely used in most industrial DLRMs…
    - Pairwise/cross attention can help with extracting categorical/numerical cross features!
- But this doesn’t quite scale…
    - Common pairwise attention in DLRMs utilizes 1-2 layers — limited model capacity;
    - “target-aware attention” requires the traditional impression (“target”)-level training setting — slows down training by O(N) vs generative training.

$$\phi_2\left(Q(X)K(X)^T + \mathrm{rab}^{p,t}\right)V(X)$$

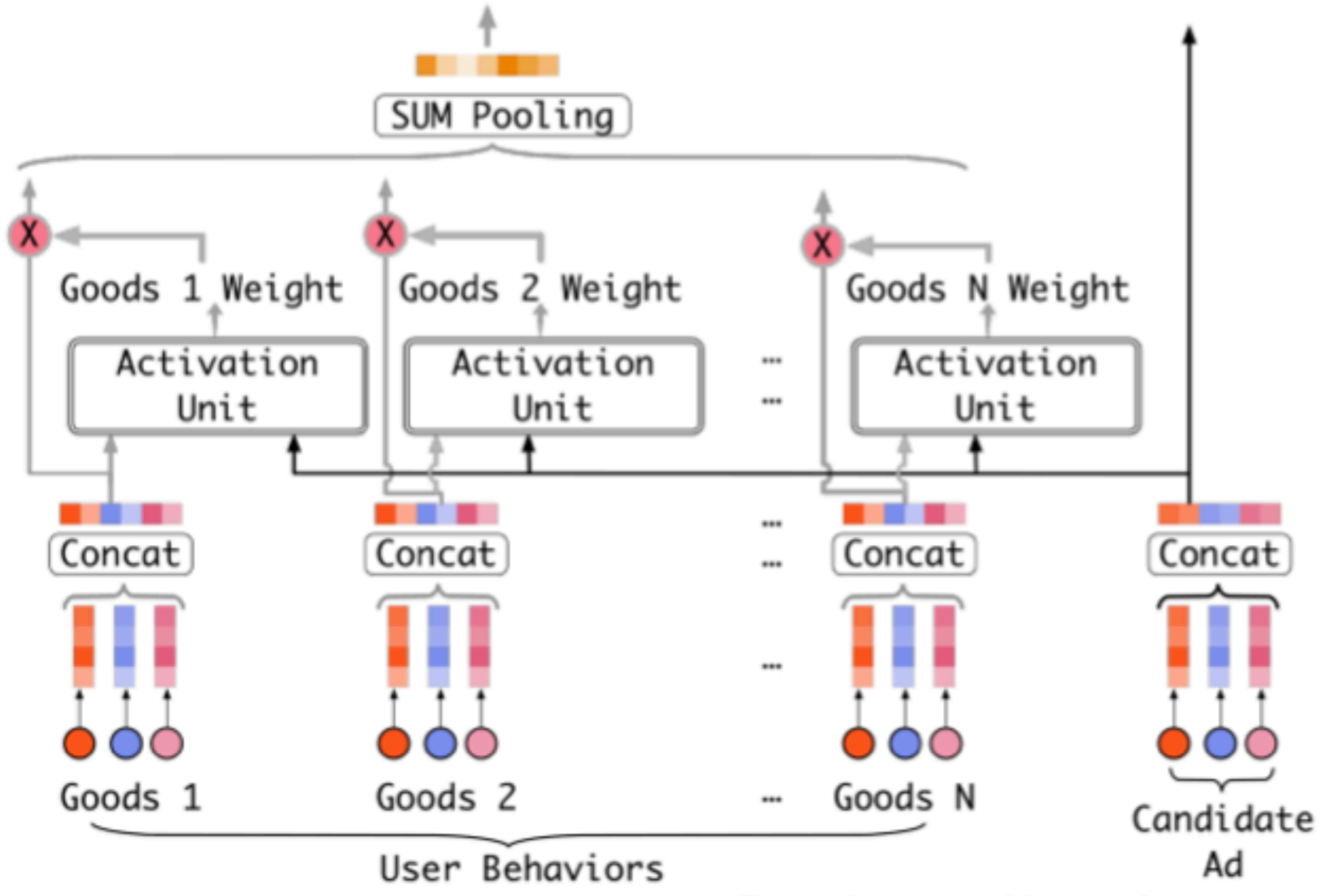
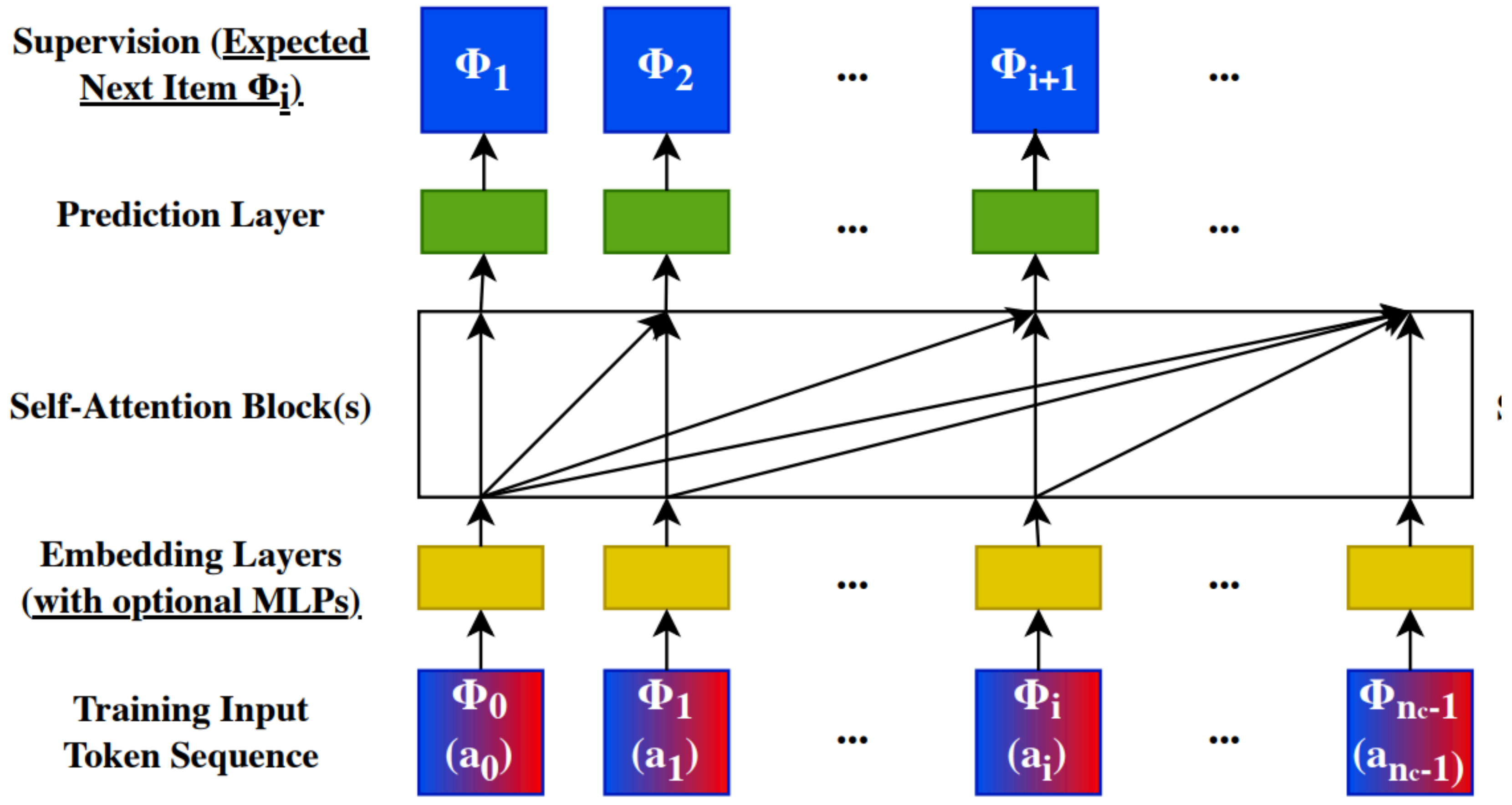


Image credit: (top) Zhai et al. Actions Speak Louder than Words: Trillion-Parameter Sequential Transducers for Generative Recommendations. ICML’24. (bottom) Zhou et al. Deep Interest Network for Click-Through Rate Prediction. KDD’18.

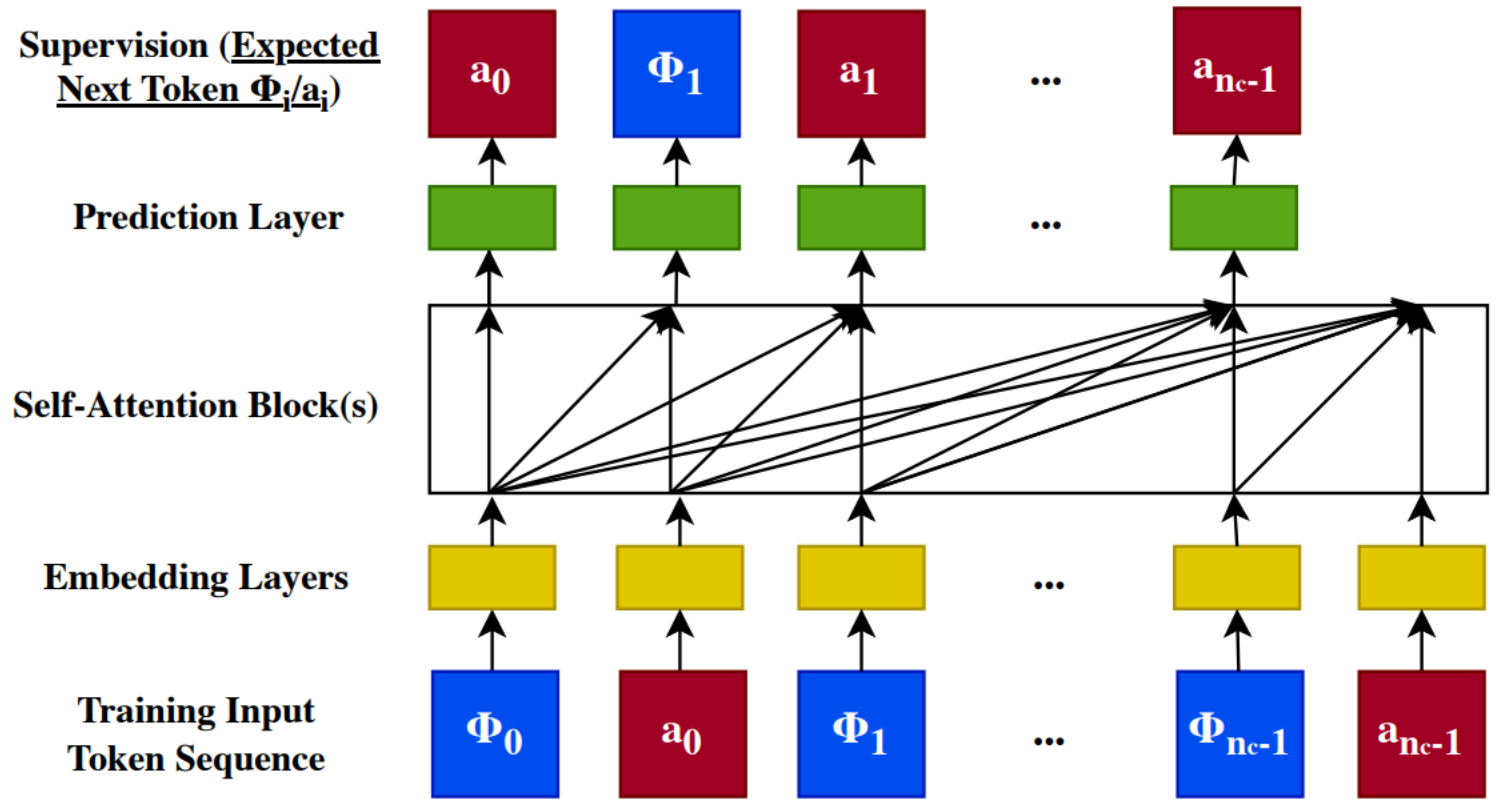
DLRMs + Generative Models => Generative Recommenders (GRs)

Enabling *fully* sequential large-scale models: Generative Recommenders

- "***Actions Speak Louder Than Words***": from **word**(piece)**s** as tokens to (high cardinality, non-stationary) ***actions*** as tokens;
  - user actions as a new *modality* in generative modeling.
- **Addresses expressiveness constraints** w/ traditional sequential recommenders;
  - Interleaves items and actions in a unified time series.
  - Encodes other categorical features as slow-changing time series.
  - Closes quality gaps between academic work and DLRMs.
- **Amortizes compute cost** via interleaving+generative training.



(a) Sequential Recommenders.
Models conditional distribution of $p(\Phi_i|\Phi_0, a_0, \ldots, \Phi_{i-1}, a_{i-1})$

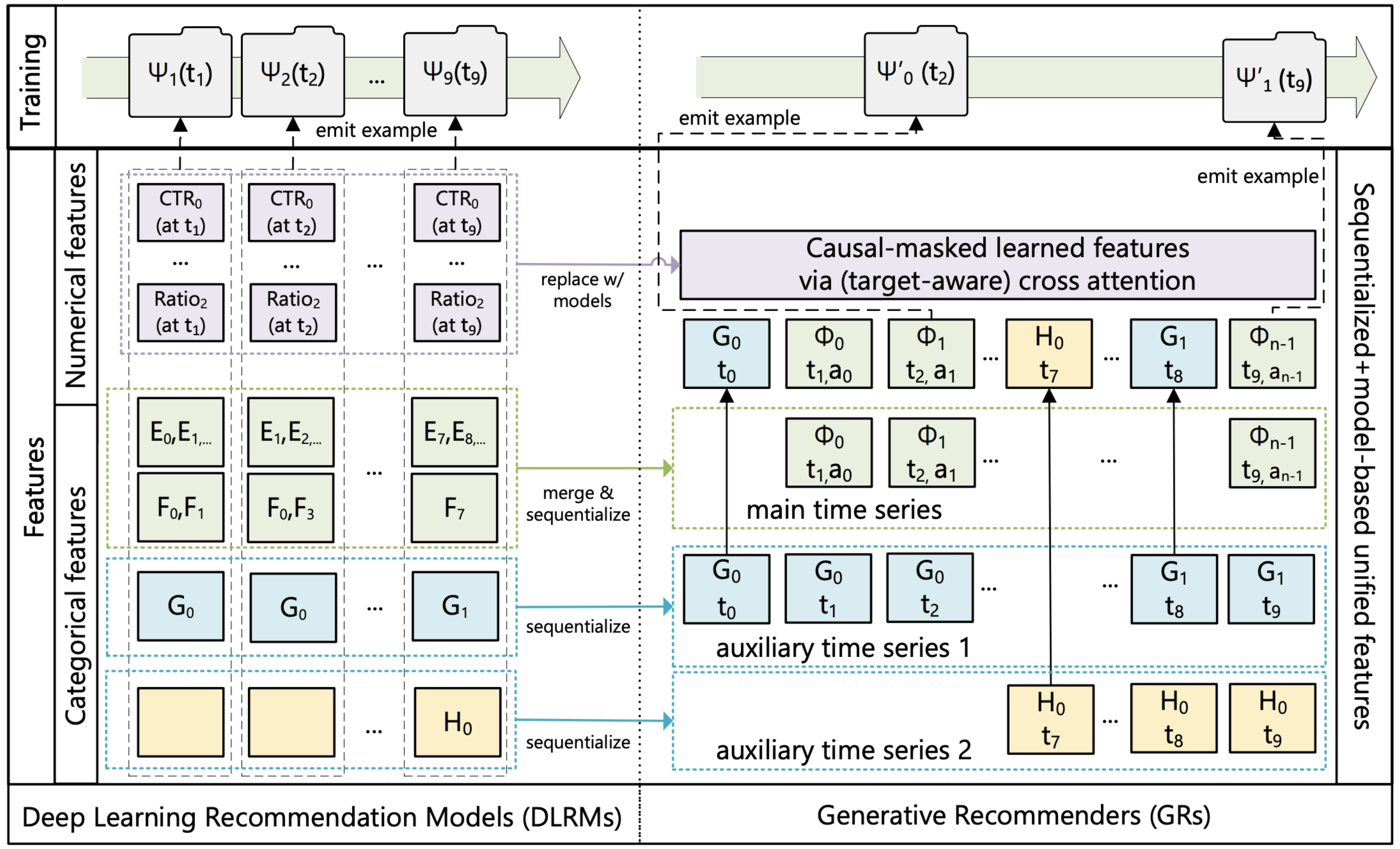


(b) Generative Recommenders.
Models joint distribution of $p(\Phi_0, a_0, \Phi_1, a_1, \ldots, \Phi_{n_c-1}, a_{n_c-1})$

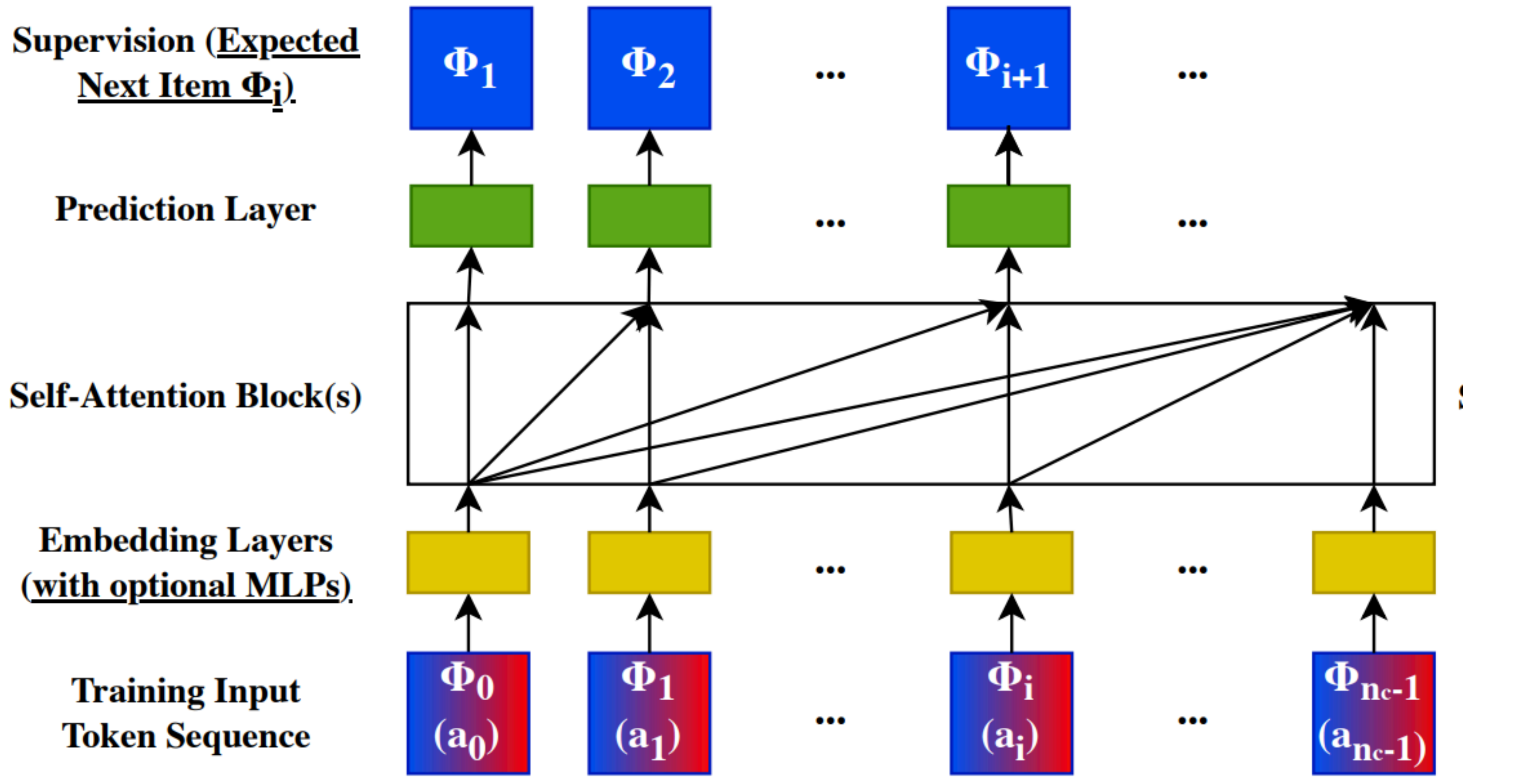
DLRMs + Generative Models => Generative Recommenders (GRs)

Enabling *fully* sequential large-scale models: Generative Recommenders

- Actions Speak Louder Than Words: words as tokens => *actions* as tokens
- Addresses expressiveness constraints w/ traditional sequential recommenders
- Amortizes compute cost via interleaving+generative training

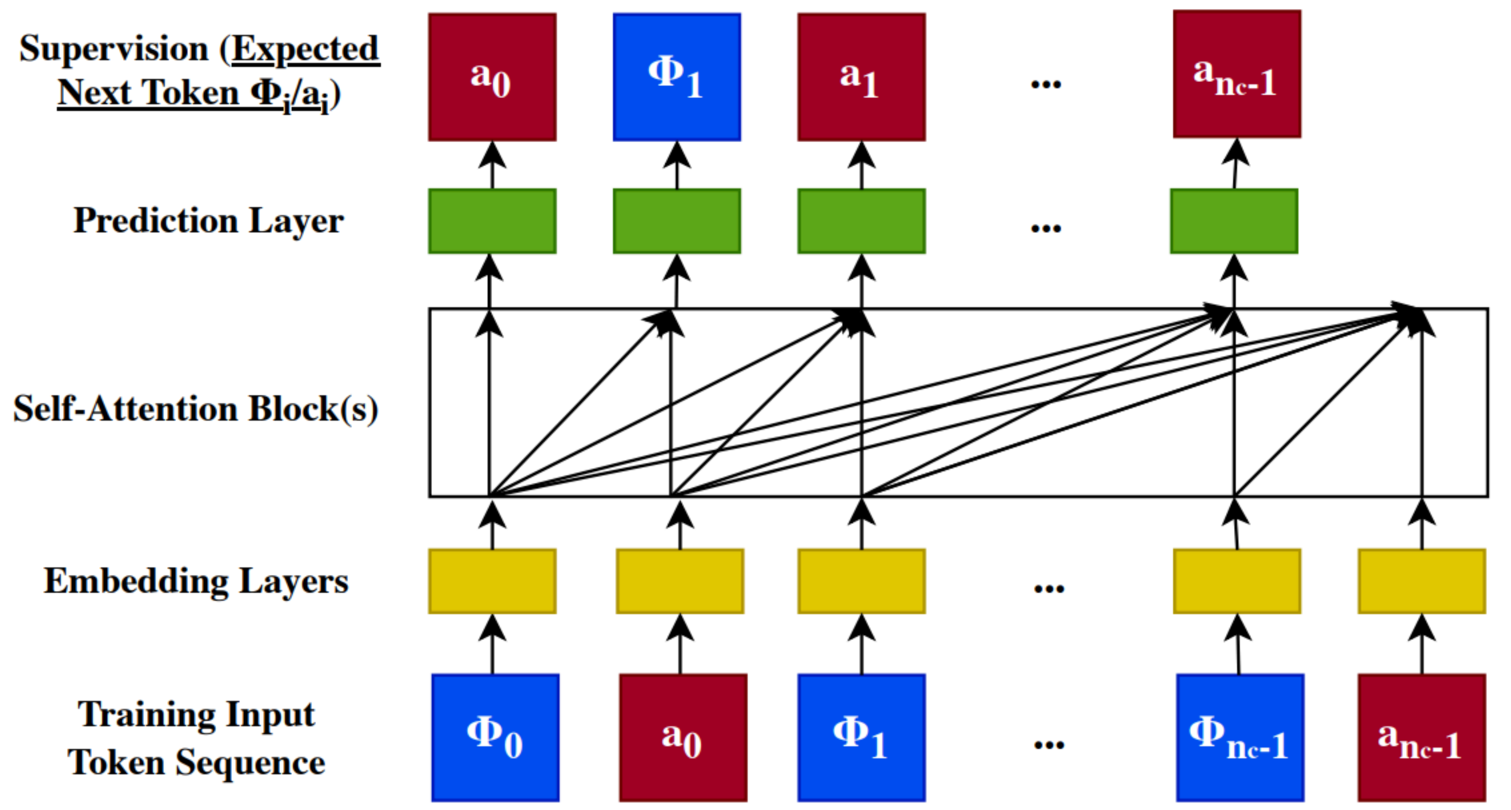


(c) DLRMs vs Generative Recommenders.



(a) Sequential Recommenders.

Models conditional distribution of $p(\Phi_i|\Phi_0, a_0, \ldots, \Phi_{i-1}, a_{i-1})$



(b) Generative Recommenders.

Models joint distribution of $p(\Phi_0, a_0, \Phi_1, a_1, \ldots, \Phi_{n_c-1}, a_{n_c-1})$

Meta AI

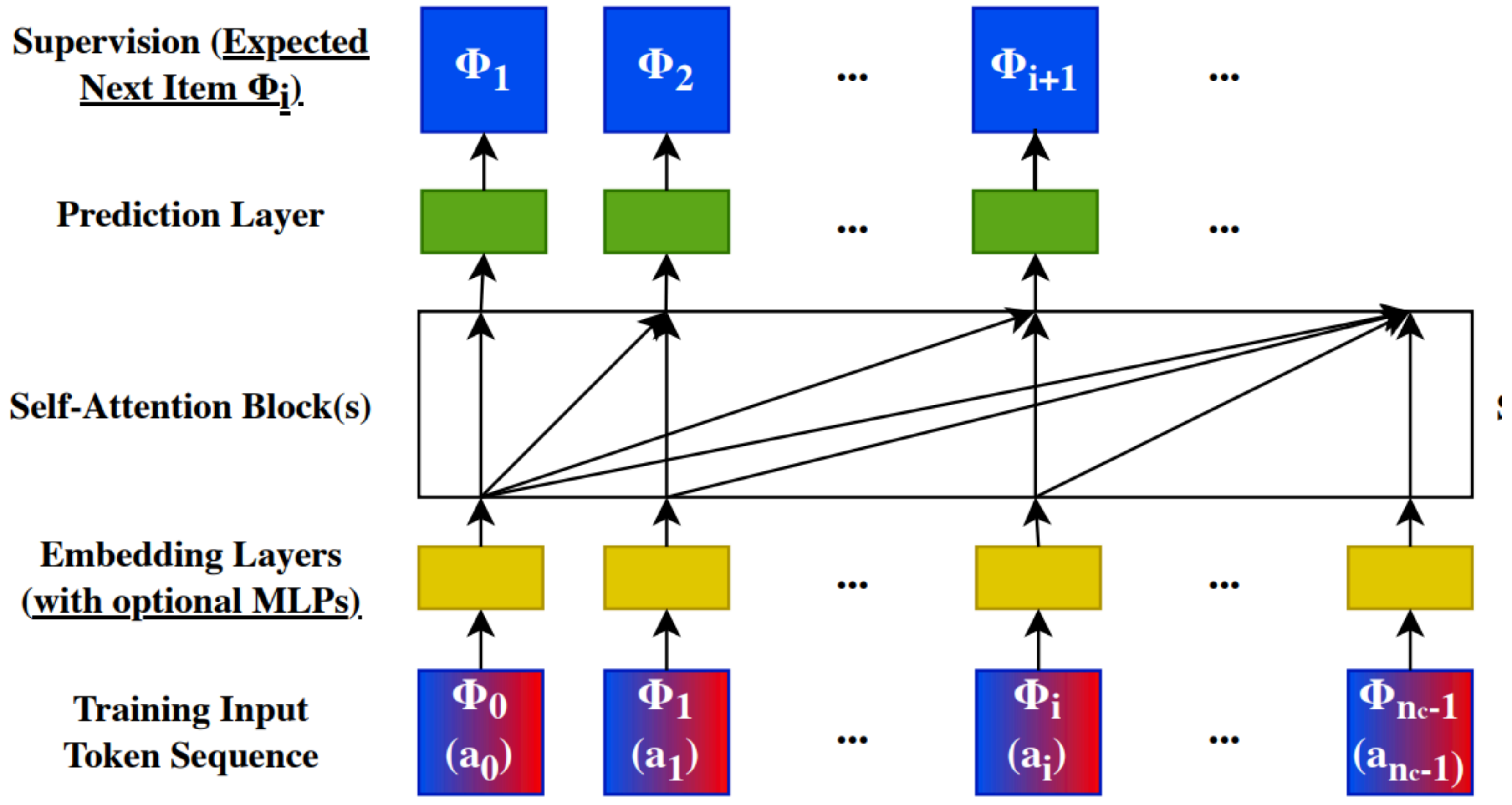
DLRMs + Generative Models => Generative Recommenders (GRs)

Enabling *fully* sequential large-scale models: Generative Recommenders

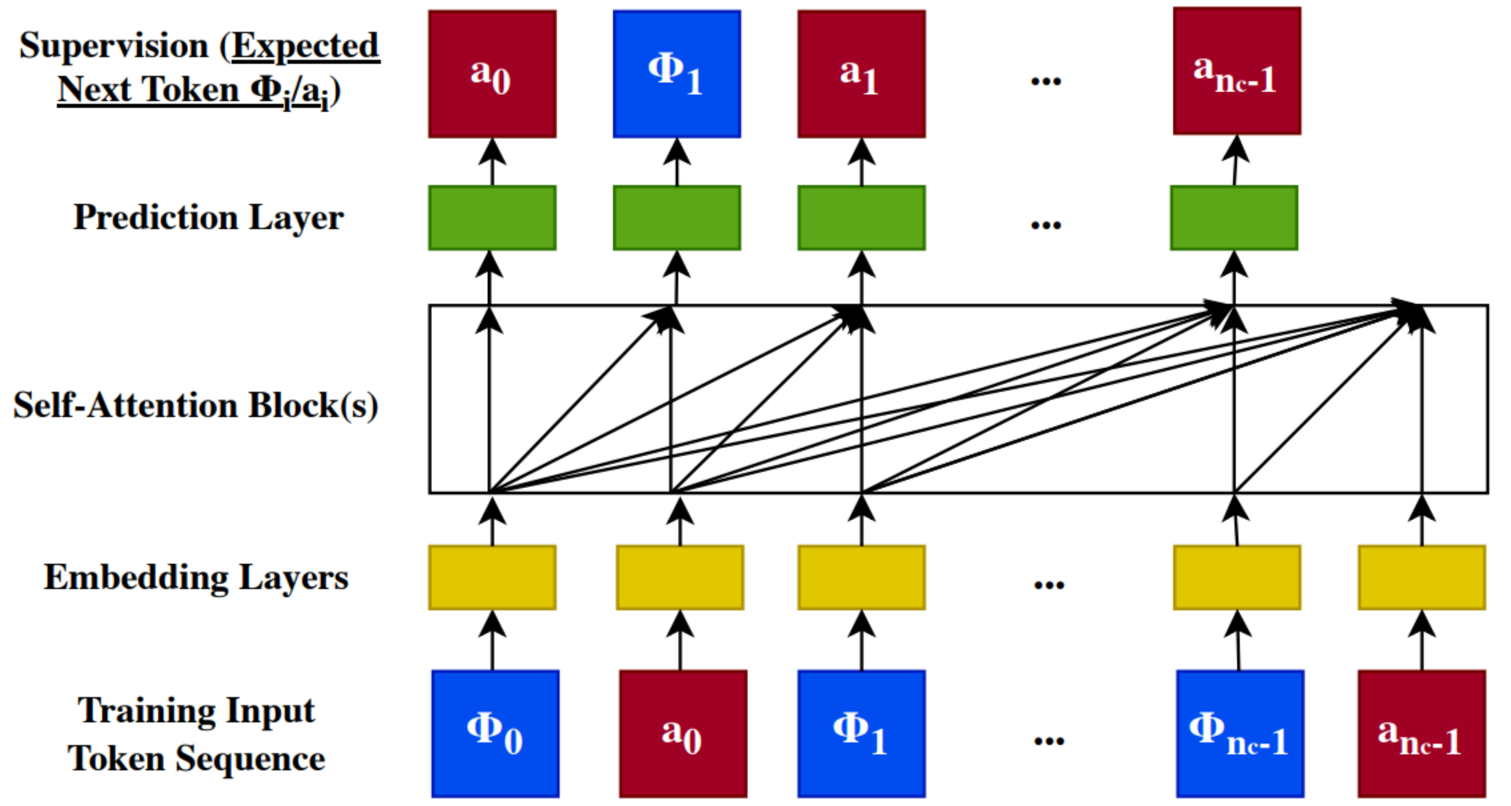
- Actions Speak Louder Than Words: words as tokens => *actions* as tokens
- Addresses expressiveness constraints w/ traditional sequential recommenders
- Amortizes compute cost via interleaving+generative training

	Input for target item i	Expected output for target item i	Architecture	Training Procedure
GRs	$\Phi_0, a_0, \Phi_1, a_1, \ldots, \Phi_i$	a_i **(target-aware)**	Self-attention (HSTU)	**Causal autoregressive (streaming/single-pass)**
GRU4Rec SASRec	$\Phi_0, \Phi_1, \ldots, \Phi_{i-1}$	Φ_i	RNNs (GRUs) Self-attention (Transformers)	**Causal autoregressive** (multi-pass)
BERT4Rec S3Rec	$\Phi_0, \Phi_1, \ldots, \Phi_{i-1}$ (at inference time)	Φ_i	Self-attention (Transformers)	Sequential multi-pass [6]
DIN BST TWIN TransAct	$\Phi_0, \Phi_1, \ldots, \Phi_i$ $(\Phi_0, a_0), \ldots, (\Phi_{i-1}, a_{i-1}), \Phi_i$	a_i **(target aware,** implicitly as part of DLRMs)	Pairwise attention Self-attention (Transformers) Two-stage pairwise attention Self-attention (Transformers)	Pointwise (**generally streaming/single pass**)

(d) Comparisons of DLRMs w/ sequential sub-modules, traditional sequential approaches in academic settings, and Generative Recommenders (GRs).



(a) Sequential Recommenders.
Models conditional distribution of $p(\Phi_i|\Phi_0, a_0, \ldots, \Phi_{i-1}, a_{i-1})$



(b) Generative Recommenders.
Models joint distribution of $p(\Phi_0, a_0, \Phi_1, a_1, \ldots, \Phi_{n_c-1}, a_{n_c-1})$

Meta AI

DLRMs + Generative Models => Generative Recommenders (GRs)

Enabling *fully* sequential large-scale models: Generative Recommenders

- *target-aware* autoregressive setting significantly improves performance!

Methods	Offline NEs		Online metrics	
	E-Task	C-Task	E-Task	C-Task
DLRM (pre-GR production model)	.4982	.7842	+0%	+0%
DLRM (DIN+DCN+MMoE)	.5053	.7899	–	–
Trad. sequential recommender setting	.4851	.7903	–	–
Generative Recommender (GR)	**.4845**	**.7645**	**+12.4%**	**+4.4%**

Offline & Online Metric comparisons in ranking setting, with
a) DLRMs (w/ target-aware sequential sub-modules), b) traditional Sequential Recommender settings (e.g., GRU4Rec, SASRec), and c) Generative Recommenders (GRs).
E-task is the main "engagement" task and C-task is the main "consumption" task.

Image credit (slide 13-16): Zhai, Liao, Liu, Wang, Li, et al. Actions Speak Louder than Words: Trillion-Parameter Sequential Transducers for Generative Recommendations. ICML'24.

III. New Algorithms: Accelerating Training & Inference by 10x-1000x for Generative Recommenders

Training - HSTU: Better Quality & 15x Faster vs Transformers

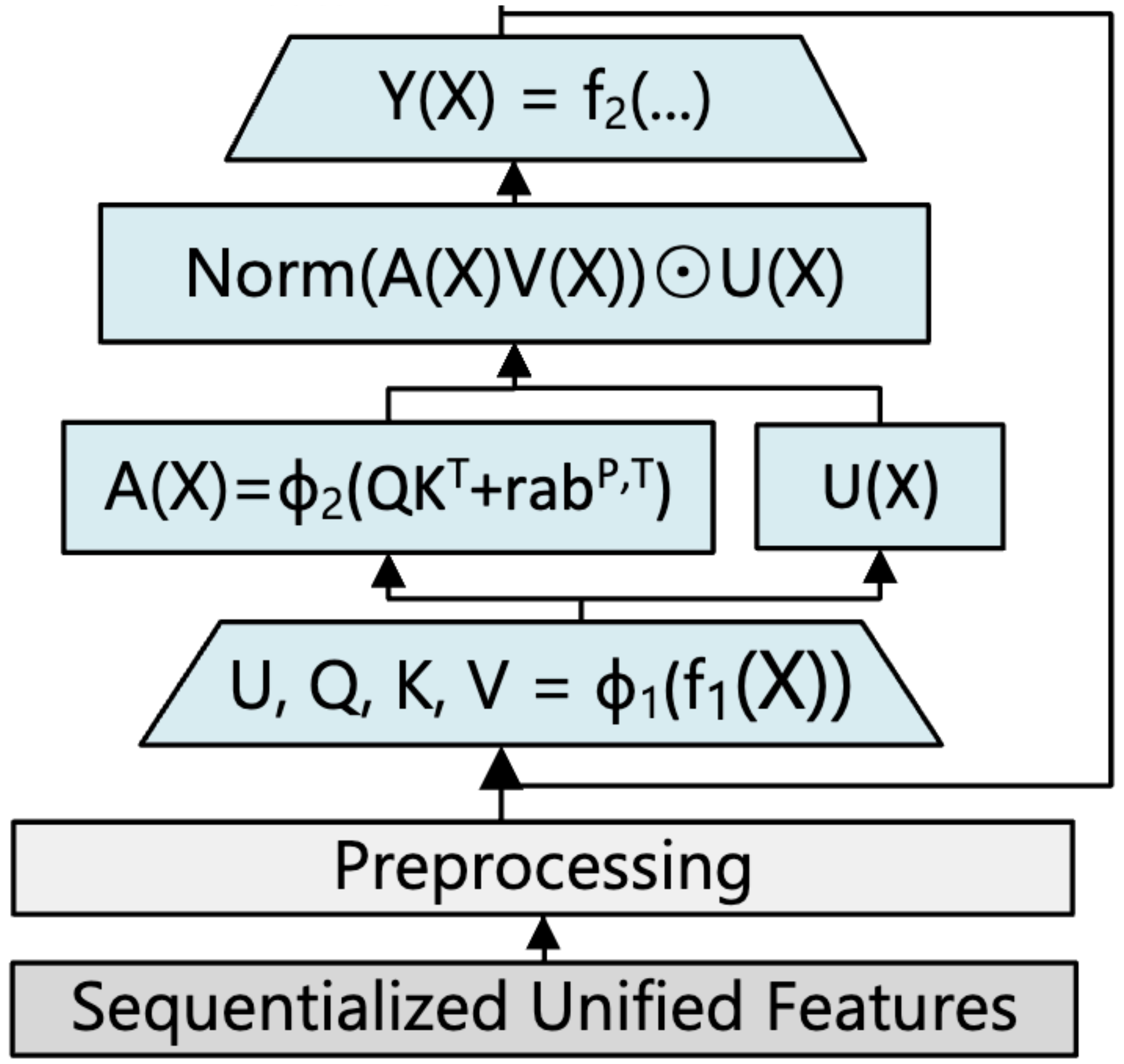
HSTU: Hierarchical Sequential Transduction Units

- Pointwise aggregated (normalized) attention
- Fusing self-attention and MLPs via element-wise gating to reduce compute;
- Grouped GEMM kernel extending memory-efficient attention (Rabe & Staats, 2021) and FA (Dao et al., 2022) to leverage sparsity;
- *Stochastic Length* (SL) further *algorithmically* increases sparsity, and reduce complexity to $O(N^{\alpha}d)$ for $\alpha \in (1,2]$.

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

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

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



Meta AI

Training - HSTU: Better Quality & 15x Faster vs Transformers

HSTU significantly outperforms Transformers in various settings…

- HSTU outperforms Transformers and various sequential baselines on synthetic, public datasets (trad. sequential recommendation settings), and large-scale Generative Recommender settings…

	Method	HR@10	HR@50	HR@200	NDCG@10	NDCG@200
ML-1M	SASRec (2023)	.2853	.5474	.7528	.1603	.2498
	BERT4Rec	.2843 (-0.4%)	–	–	.1537 (-4.1%)	–
	GRU4Rec	.2811 (-1.5%)	–	–	.1648 (+2.8%)	–
	HSTU	.3097 (+8.6%)	.5754 (+5.1%)	.7716 (+2.5%)	.1720 (+7.3%)	.2606 (+4.3%)
	HSTU-large	**.3294 (+15.5%)**	**.5935 (+8.4%)**	**.7839 (+4.1%)**	**.1893 (+18.1%)**	**.2771 (+10.9%)**
ML-20M	SASRec (2023)	.2906	.5499	.7655	.1621	.2521
	BERT4Rec	.2816 (-3.4%)	–	–	.1703 (+5.1%)	–
	GRU4Rec	.2813 (-3.2%)	–	–	.1730 (+6.7%)	–
	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 12. Evaluations of methods on public datasets in traditional sequential recommender settings (multi-pass, full-shuffle).

Architecture	HR@10	HR@50
Transformers	.0442	.2025
HSTU (-rab^{p,t}, Softmax)	.0617	.2496
HSTU (-rab^{p,t})	**.0893**	**.3170**

Table 2. Synthetic data in one-pass streaming settings.

Table 5. 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**

Training - HSTU: Better Quality & 15x Faster vs Transformers

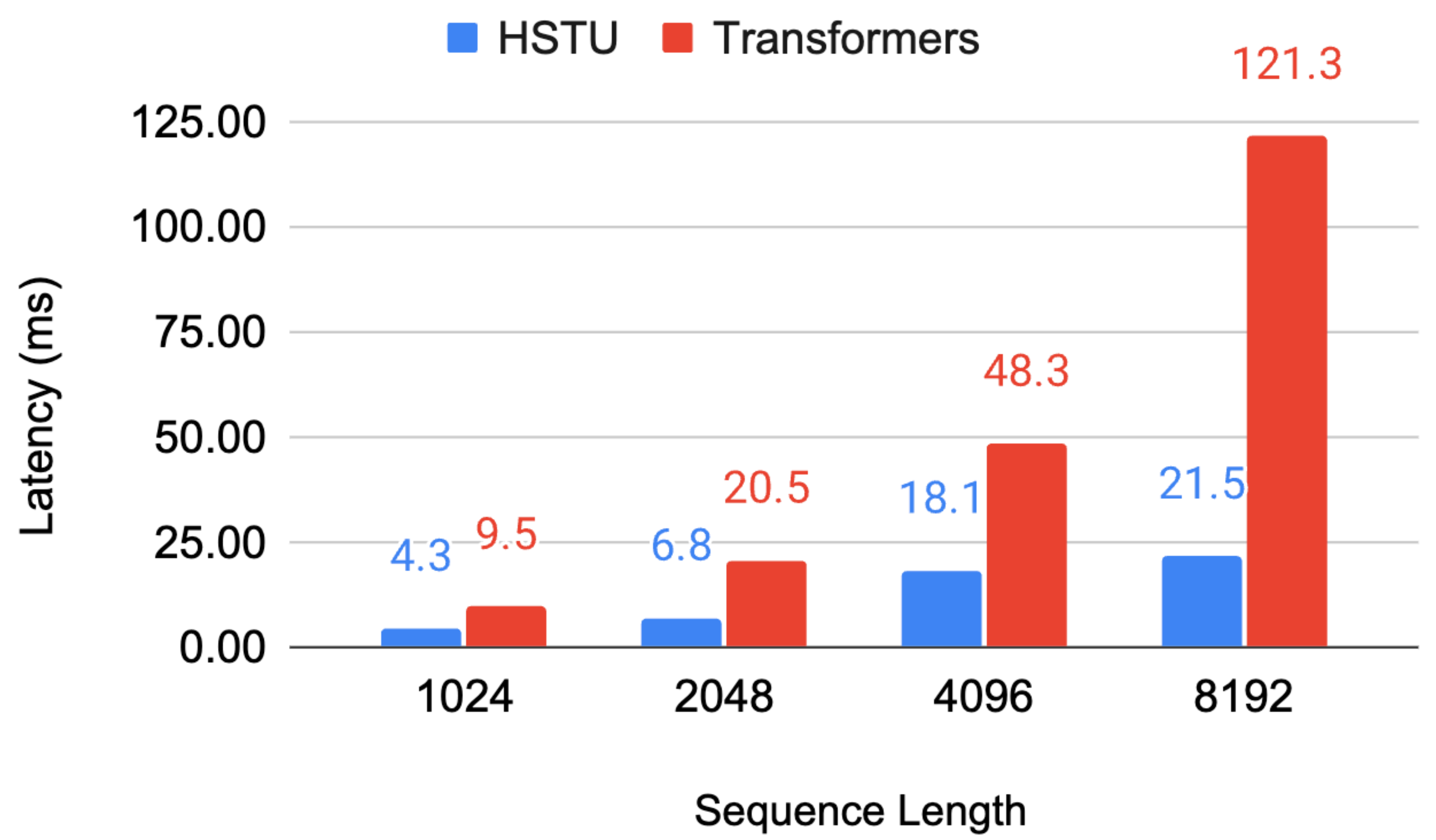
… and achieves 15x Training Speedup on 8K sequences!

- HSTU outperforms Transformers and various sequential baselines on synthetic, public datasets (trad. sequential recommendation settings), and large-scale Generative Recommender settings …
- … while being **15x faster** vs FlashAttention2 (SotA implementation as of 05/2024) on 8k sequences during training, due to HSTU design + SL-induced sparsity.

Alpha (α)	**Max Sequence Lengths**			
	1,024	2,048	4,096	8,192
1.6	71.5%	76.1%	80.5%	84.4%
1.7	56.1%	63.6%	69.8%	75.6%
1.8	40.2%	45.3%	54.1%	66.4%
1.9	17.2%	21.0%	36.3%	64.1%
2.0	3.1%	6.6%	29.1%	64.1%

Table 3. Impact of *Stochastic Length* (SL) on sequence sparsity.

(a) Training Speedup.



(b) Inference Speedup.

Inference - M-FALCON: 900x Speedup vs SotA DLRMs

Microbatched-Fast Attention Leveraging Cachable OperatioNs

EPISODE X: A NEW FRONTIER IN SPEED

IN A PERIOD OF TECHNOLOGICAL REVOLUTION, SCIENTISTS HAVE DISCOVERED A WAY TO ACHIEVE A 1000X INFERENCE SPEEDUP FOR INDUSTRIAL-SCALE RECSYS.

AMIDST THE VAST DIGITAL COSMOS, THE POWERFUL M-FALCON ~~STARSHIP~~ ALGORITHM EMERGES AS THE BEACON OF HOPE, PROMISING TO AUGMENT DECISION MAKING PROCESSES ON ONLINE CONTENT AND E-COMMERCE PLATFORMS THROUGH GENERATIVE RECOMMENDERS…

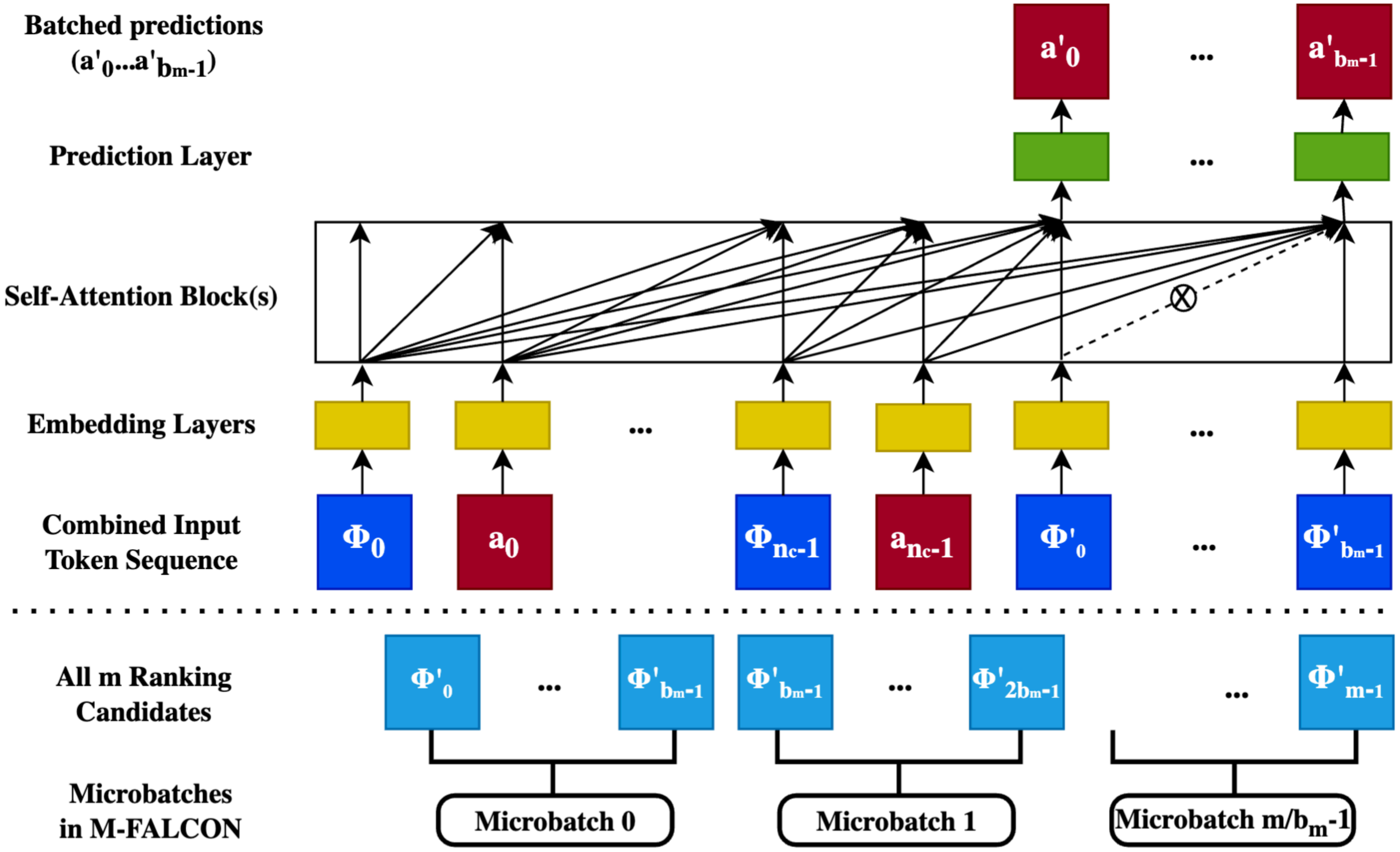
Meta AI

Inference - M-FALCON: 900x Speedup vs SotA DLRMs

Batched Target-Aware Inference + Microbatching + KV Caching

M-FALCON leverages three key insights:

- **Batched inference** enables **compute sharing**, and *can* be efficiently applied to *target-aware* autoregressive settings;
- **Microbatching** scales batched inference to large candidate sets;
- **Encoder-level caching** eliminates redundant ops within & *across* requests.



(b) GR's ranking model inference utilizing the M-FALCON algorithm.

Meta AI

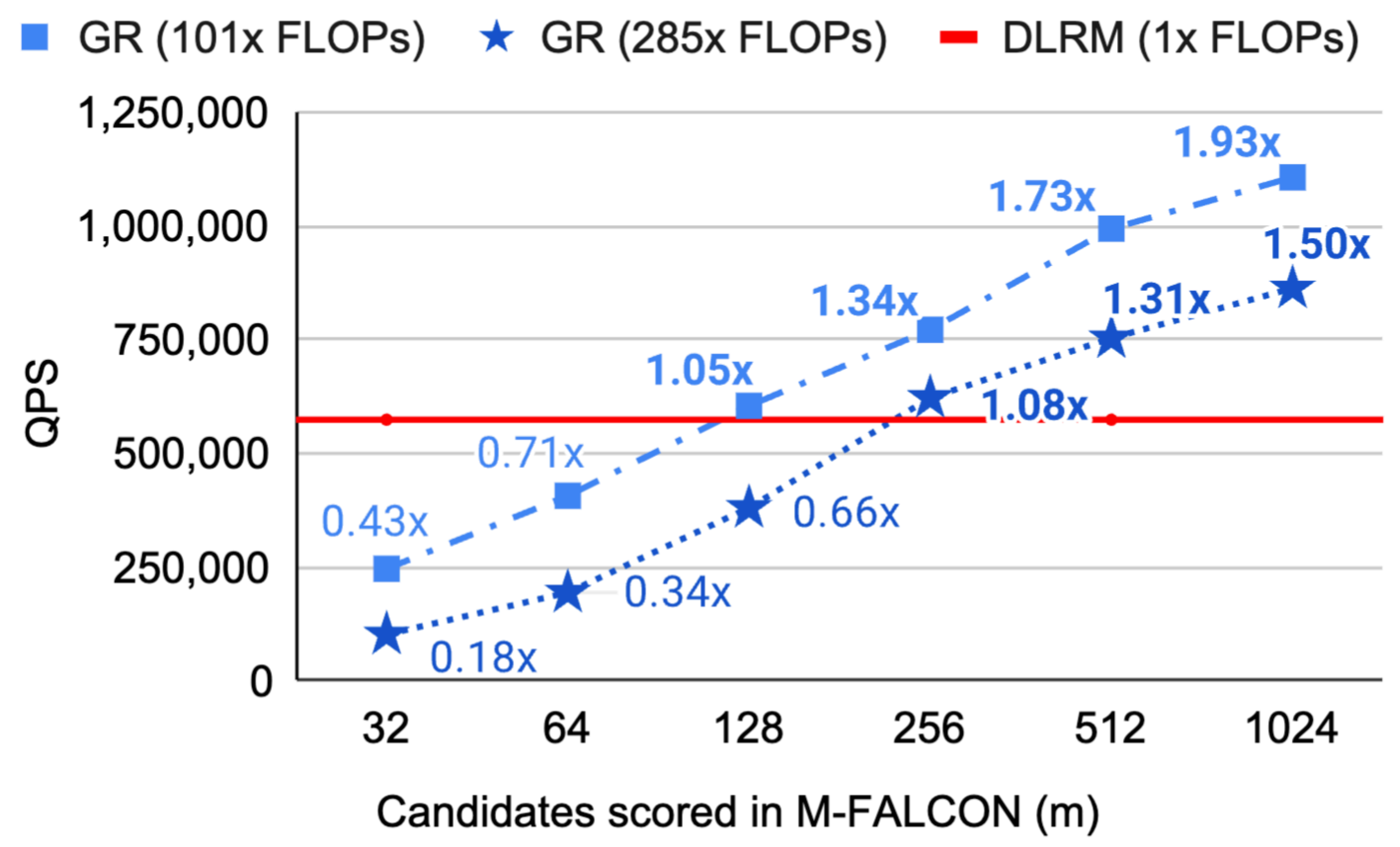
Inference - M-FALCON: 900x Speedup vs SotA DLRMs

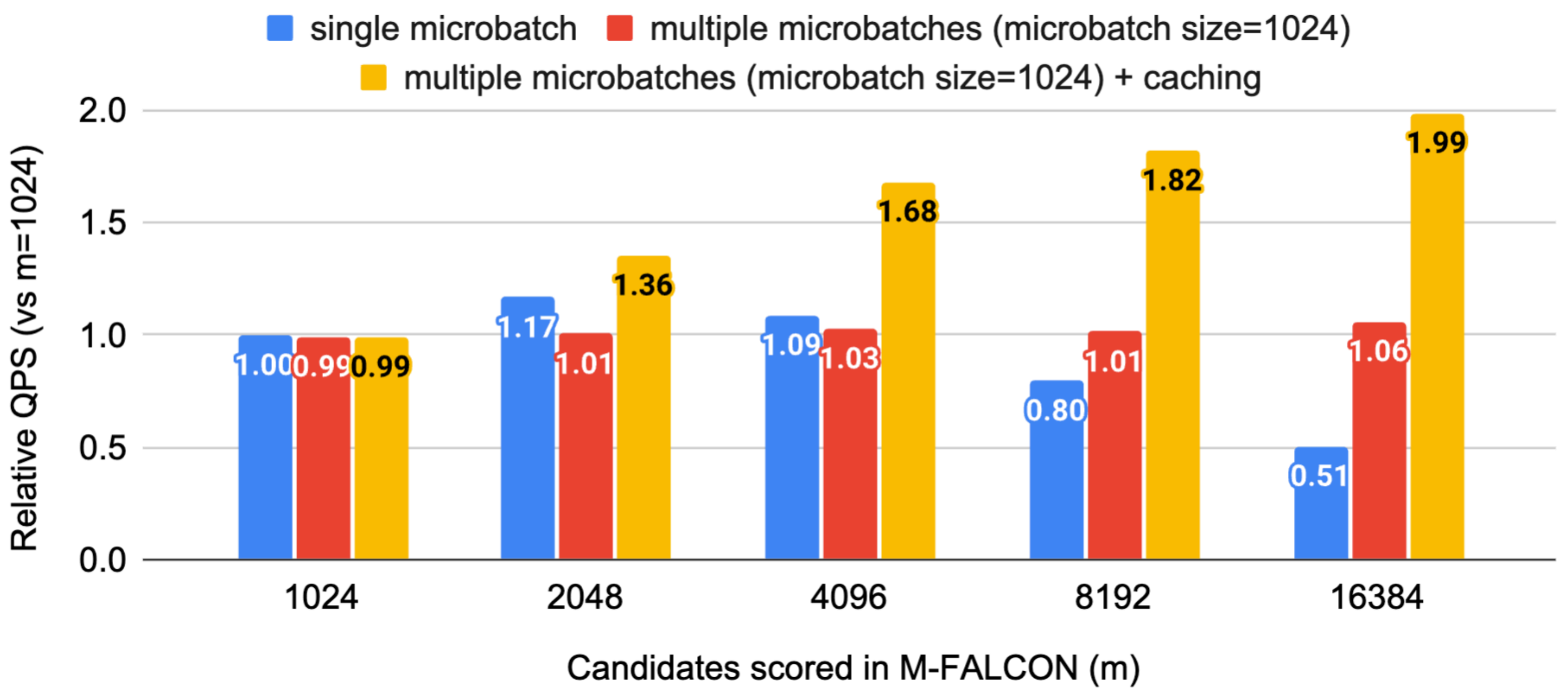
Batched Target-Aware Inference + Microbatching + KV Caching

M-FALCON leverages three key insights:

- **Batched inference** enables **compute sharing**, and *can* be efficiently applied to *target-aware* autoregressive settings;
- **Microbatching** scales batched inference to large candidate sets;
- **Encoder-level caching** eliminates redundant ops within & *across* requests.

These combined enables serving a **285x more complex** GR model at **3x** QPS!





Meta AI

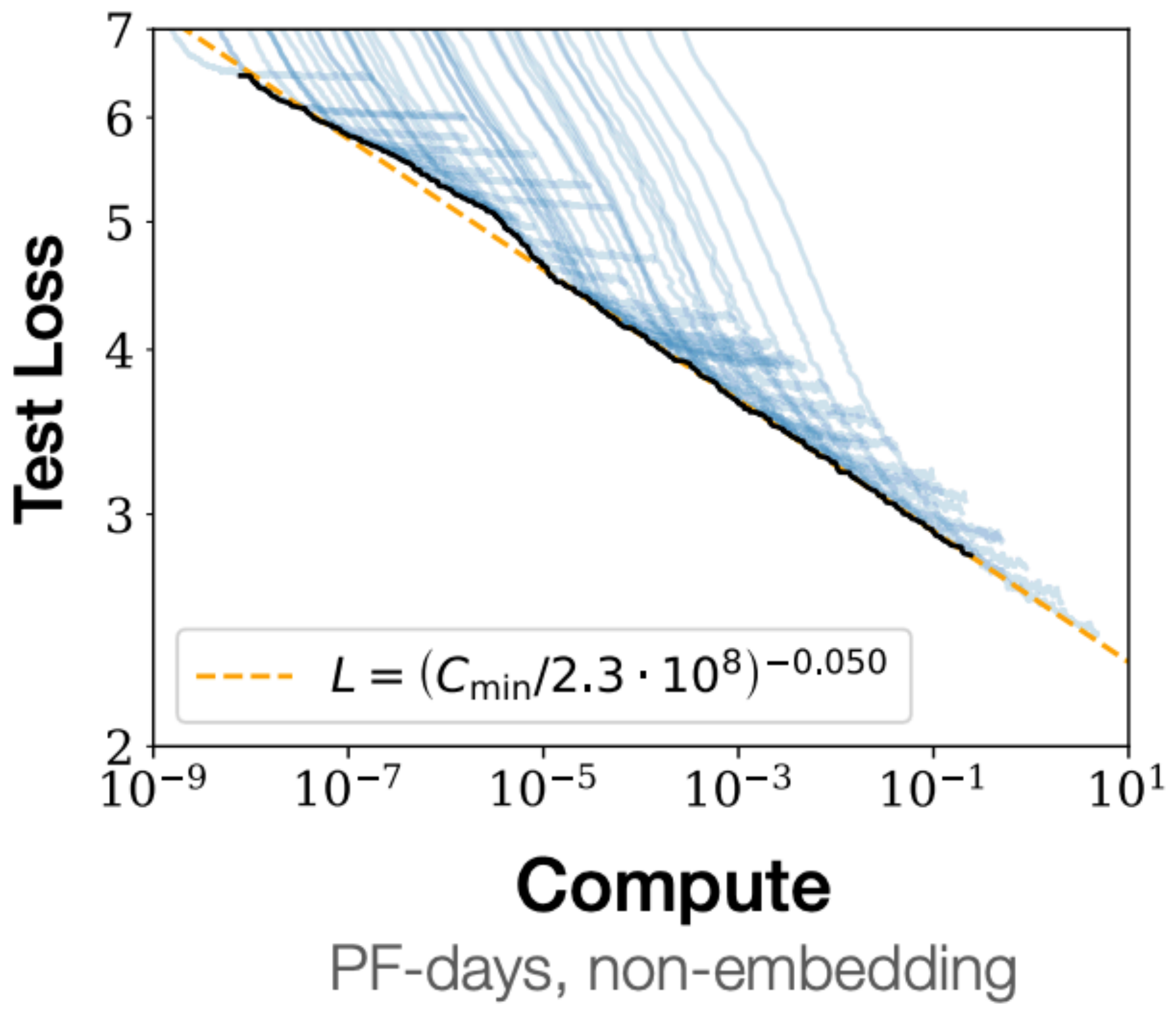
IV. Scaling Law for Recommendation Systems, in Industrial-scale Production Settings

Compute growth of RecSys have lagged behind other fields…

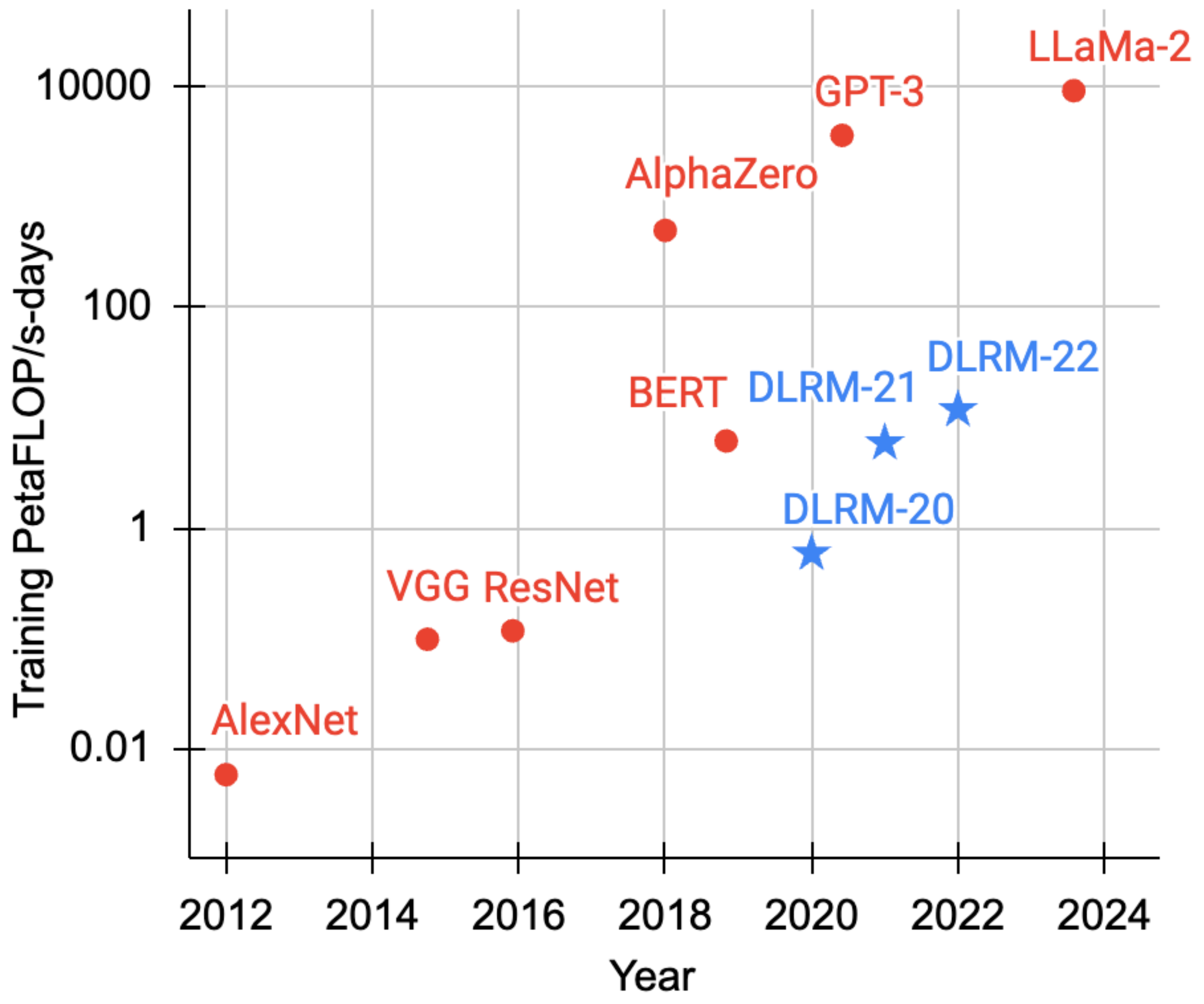
& historically, DLRMs don't scale well with compute

- Many Deep Learning Models, esp. LLMs, benefit from scaling law, where losses etc. scale as a power-law of compute.

- Nevertheless, DLRMs generally scale with data but less well with compute…



Scaling Law for LLMs.
Kaplan et al. Scaling Laws for Neural Language Models. 2020



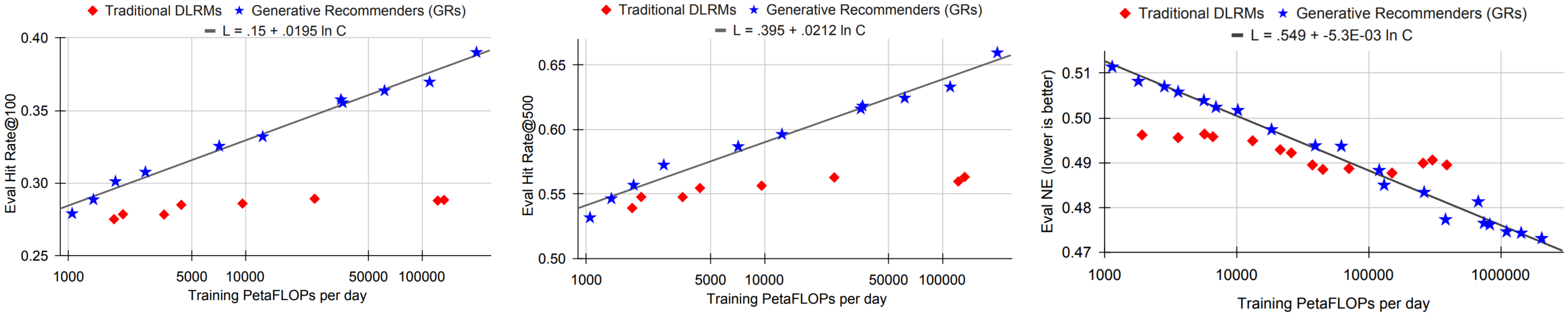
Compute Usage Trends for major Deep Learning Models and representative DLRMs before GRs.

Meta AI

Scaling Law with Generative Recommenders, up to LLM scale

GRs demonstrate scaling law in large-scale RecSys for the first time!

- … for all major metrics, up to GPT-3 175b/LLaMa-2 70b scale



Scalability comparison of DLRMs vs Generative Recommenders (GRs). left: HR@100 (retrieval), middle: HR@500 (retrieval), right: Normalized Entropy (ranking). +0.005 in HR and -0.001 in NE represent significant improvements.

Meta AI

Scaling Law with Generative Recommenders, up to LLM scale

GRs demonstrate scaling law in large-scale RecSys for the first time!

- This enables double-digit topline gains in production settings…
- … while using *less* inference resources, thanks to HSTU+M-FALCON!

Table 6. 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 7. 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 (DIN+DCN)	.5053	.7899	–	–
DLRM (abl. features)	.5053	.7925	–	–
GR (interactions only)	.4851	.7903	–	–
GR	**.4845**	**.7645**	**+12.4%**	**+4.4%**

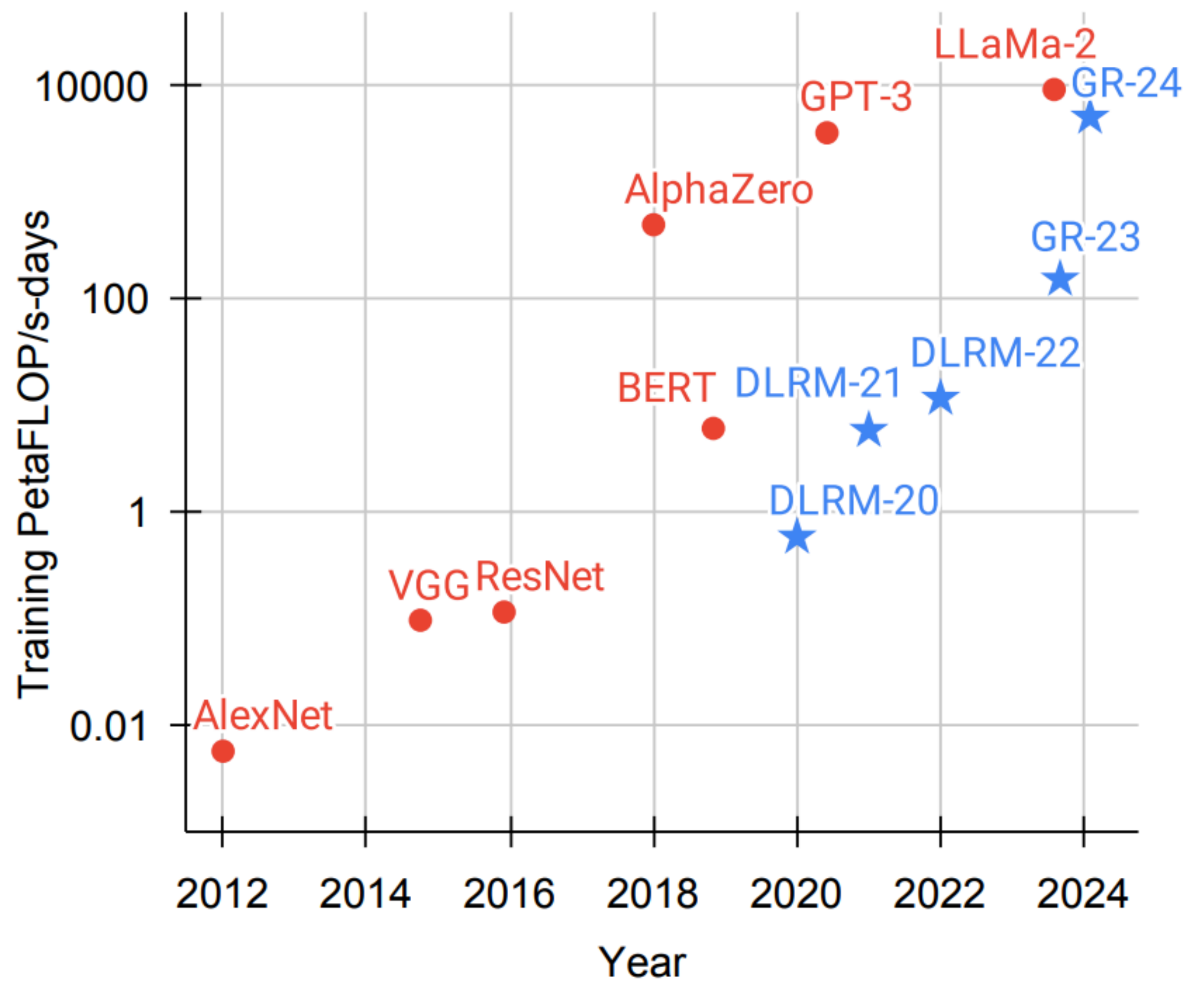


Figure 1. Total compute used to train deep learning models over the years. DLRM results are from (Mudigere et al., 2022); GRs are deployed models from this work. DLRMs/GRs are continuously trained in a streaming setting; we report compute used per year.

Meta AI

Thank You

- **Generative Recommenders** (GRs) reinterpret main RecSys tasks within a generative framework, **unifying heterogeneous feature spaces in DLRMs**, while **addressing expressiveness constraints in traditional sequential recommenders** to significantly enhance performance.
- Our new architecture, **HSTU**, outperforms SASRec **by 65.8% in NDCG**, and offers a **15x training-time speedup vs SotA Transformers (FA2) on 8k length sequences**. Our inference algorithm, **M-FALCON, further enables a 900x speedup at inference time**, through fully amortizing computational costs via microbatching and caching.
- HSTU-based Generative Recommenders, with 1.5 trillion params, improve online metrics by 12.4%+. More importantly, we observe **scaling law in industrial-scale recommendation systems for the first time**, up to GPT-3/LLaMa-2 compute scale, which represents a potential ChatGPT moment for RecSys.

Meta AI

For more info, please check *Actions Speak Louder than Words: Trillion-Parameter Sequential Transducers for Generative Recommendations. ICML'24*. arXiv: 2402.17152 / github. & **We're hiring!** MetaCareers, etc.

References

- DLRMs
  - Covington et al. Deep Neural Networks for YouTube Recommendations. RecSys'16.
  - Guo et al. DeepFM: A Factorization-Machine based Neural Network for CTR Prediction. IJCAI'17. ("DeepFM")
  - Xiao et al. Attentional Factorization Machines: Learning the Weight of Feature Interactions via Attention Networks. IJCAI'17. ("AttentionFM")
  - Wang et al. Deep & Cross Network for Ad Click Predictions. AdKDD'17. ("DCN")
  - Lian et al. xDeepFM: Combining Explicit and Implicit Feature Interactions for Recommender Systems. KDD'18.
  - Zhou et al. Deep Interest Network for Click-Through Rate Prediction. KDD'18. ("DIN")
  - Ma et al. Modeling Task Relationships in Multi-task Learning with Multi-gate Mixture-of-Experts. KDD'18. ("MMoE")
  - Zhu et al. Learning Tree-based Deep Model for Recommender Systems. KDD'18. ("TDM")
  - Ma et al. Entire Space Multi-Task Model: An Effective Approach for Estimating Post-Click Conversion Rate. SIGIR'18. ("ESMM")
  - Chen et al. Top-K Off-Policy Correction for a REINFORCE Recommender System. WSDM'19. ("Top-K REINFORCE")
  - Song et al. AutoInt: Automatic Feature Interaction Learning via Self-Attentive Neural Networks. CIKM'19.
  - Naumov et al. Deep Learning Recommendation Model for Personalization and Recommendation Systems. 2019.
  - Zhou et al. Learning Optimal Tree Models under Beam Search. ICML'20. ("OTM")
  - Tang et al. Progressive Layered Extraction (PLE): A Novel Multi-Task Learning (MTL) Model for Personalized Recommendations. RecSys'20.
  - Pi et al. Search-based User Interest Modeling with Lifelong Sequential Behavior Data for Click-Through Rate Prediction. CIKM'20. ("SIM")
  - Zhou et al. Contrastive Learning for Debiased Candidate Generation in Large-Scale Recommender Systems. KDD'21. ("CLRec")
  - Wang et al. MaskNet: Introducing Feature-Wise Multiplication to CTR Ranking Models by Instance-Guided Mask. DLP-KDD'21.
  - Gao et al. Deep Retrieval: Learning A Retrievable Structure for Large-Scale Recommendations. CIKM'21. ("DR")
  - Zhang et al. DHEN: A Deep and Hierarchical Ensemble Network for Large-Scale Click-Through Rate Prediction. 2022.
  - Chang et al. TWIN: TWo-stage Interest Network for Lifelong User Behavior Modeling in CTR Prediction at Kuaishou. KDD'23.
  - Zhai et al. Revisiting Neural Retrieval on Accelerators. KDD'23. ("MoL")

References (cont'd)

- LLMs
  - Raffel et al. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. 2019. ("T5")
  - Brown et al. Language Models are Few-Shot Learners. 2020. ("GPT-3")
  - Kaplan et al. Scaling Laws for Neural Language Models. 2020.
  - Touvron et al. Llama 2: Open Foundation and Fine-Tuned Chat Models. 2023.
- Sequential Recommenders.
  - Hidasi et al. Session-based Recommendations with Recurrent Neural Networks. ICLR'16. ("GRU4Rec")
  - Kang et al. Self-Attentive Sequential Recommendation. ICDM'18. ("SASRec")
  - Sun et al. BERT4Rec: Sequential Recommendation with Bidirectional Encoder Representations from Transformer. ICDM'19.
- LLM for Recommendations.
  - Cui et al. M6-Rec: Generative Pretrained Language Models are Open-Ended Recommender Systems. 2022.
  - Bao et al. TALLRec: An Effective and Efficient Tuning Framework to Align Large Language Model with Recommendation. RecSys'23.
  - Hou et al. Large Language Models are Zero-Shot Rankers for Recommender Systems. ECIR'24. ("LLMRank")
  - Zhang et al. NoteLLM: A Retrievable Large Language Model for Note Recommendation. WWW'24.
- Efficient Attention.
  - Rabe & Staats. Self-attention Does Not Need $O(n^2)$ Memory. 2022.
  - Dao et al. FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness. NeurIPS'23.