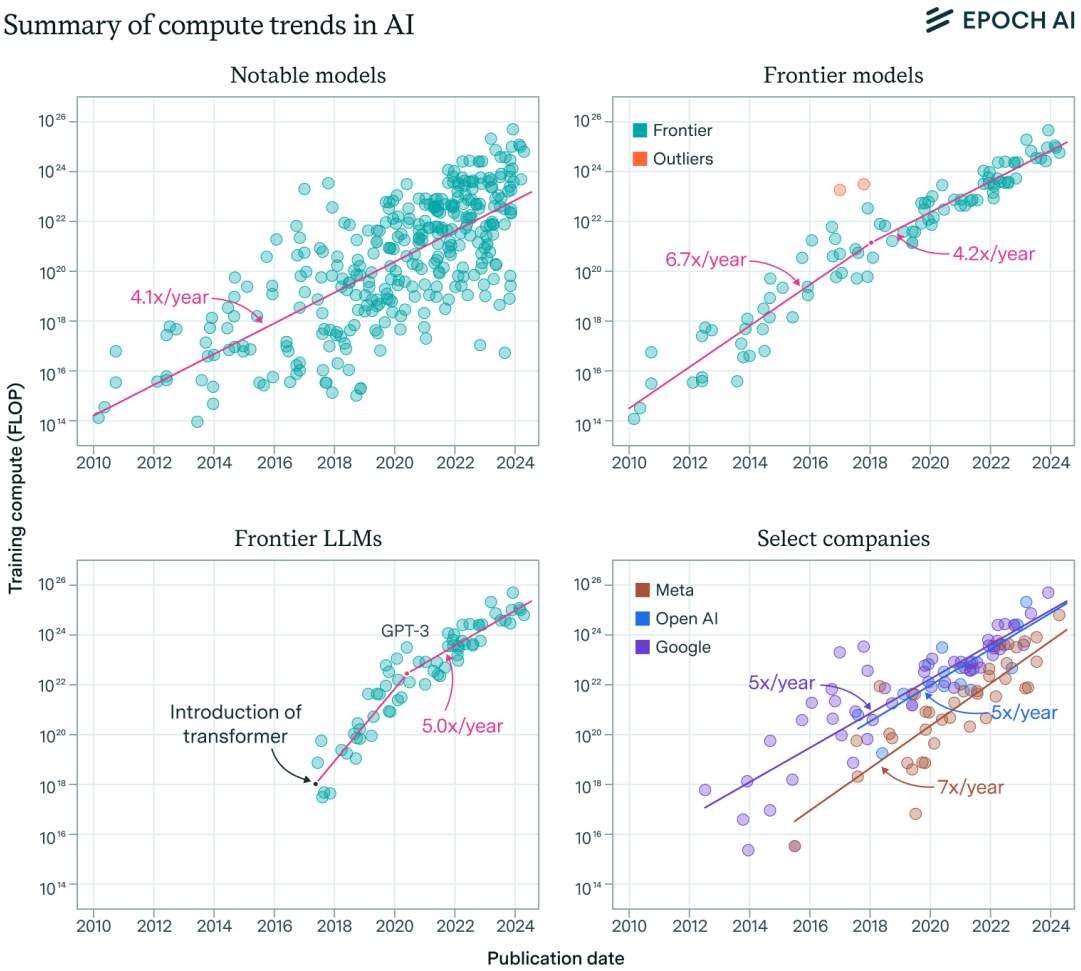
Diffusion Language Models are Super Data Learners

Presented by: **Jinjie Ni**

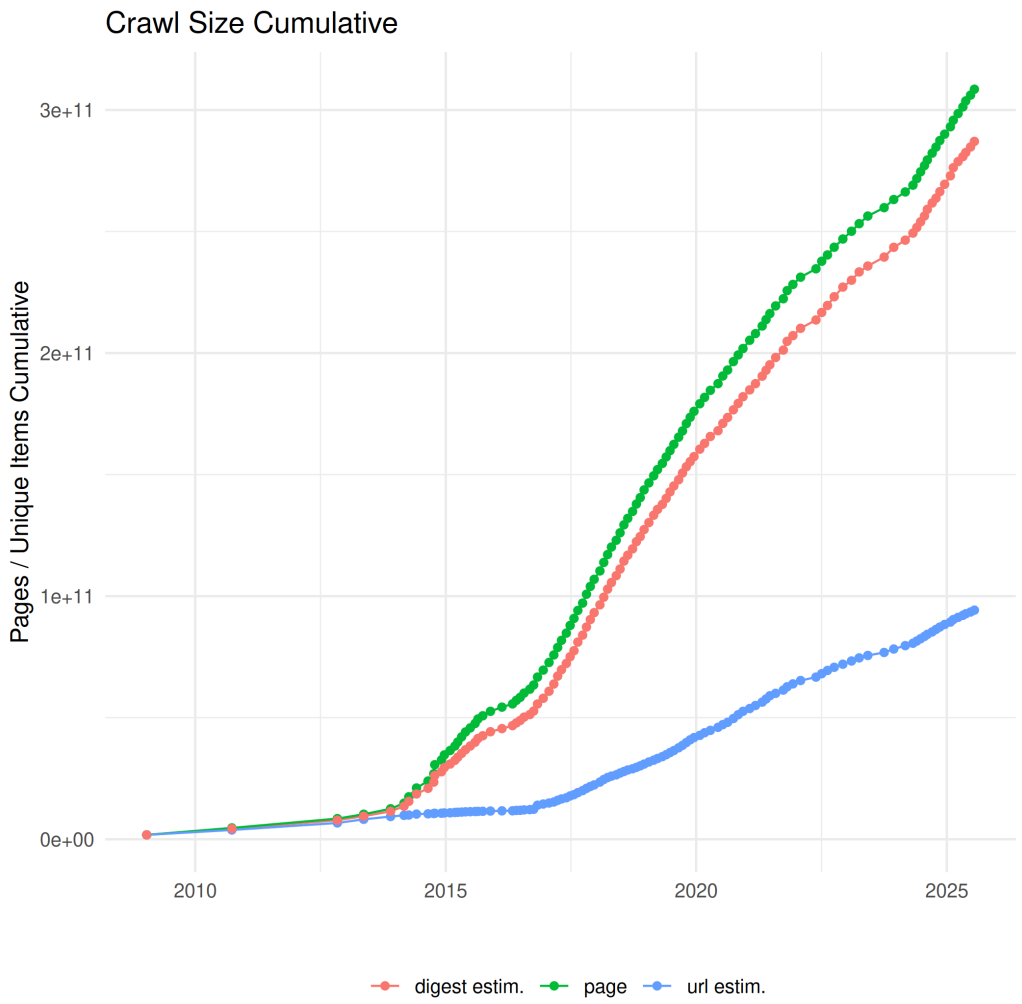
National University of Singapore

Limited Data, Infinite Compute



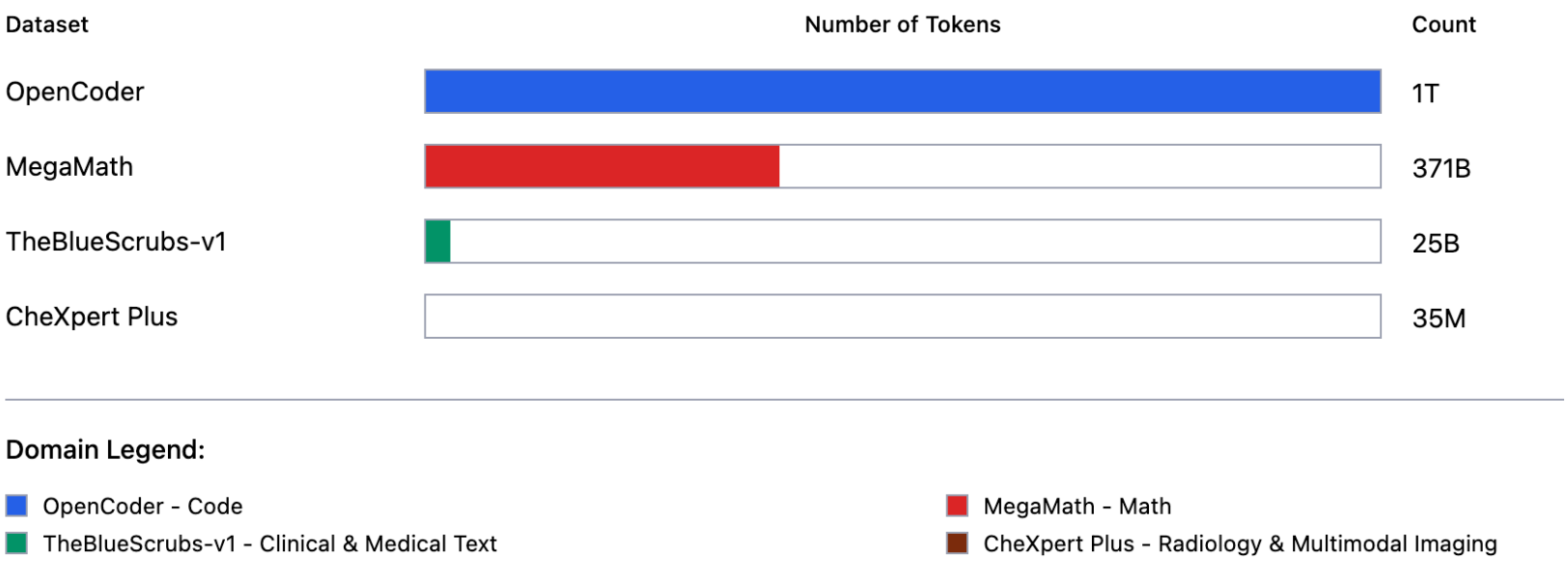
Exponential growth of compute

Limited Data, Infinite Compute



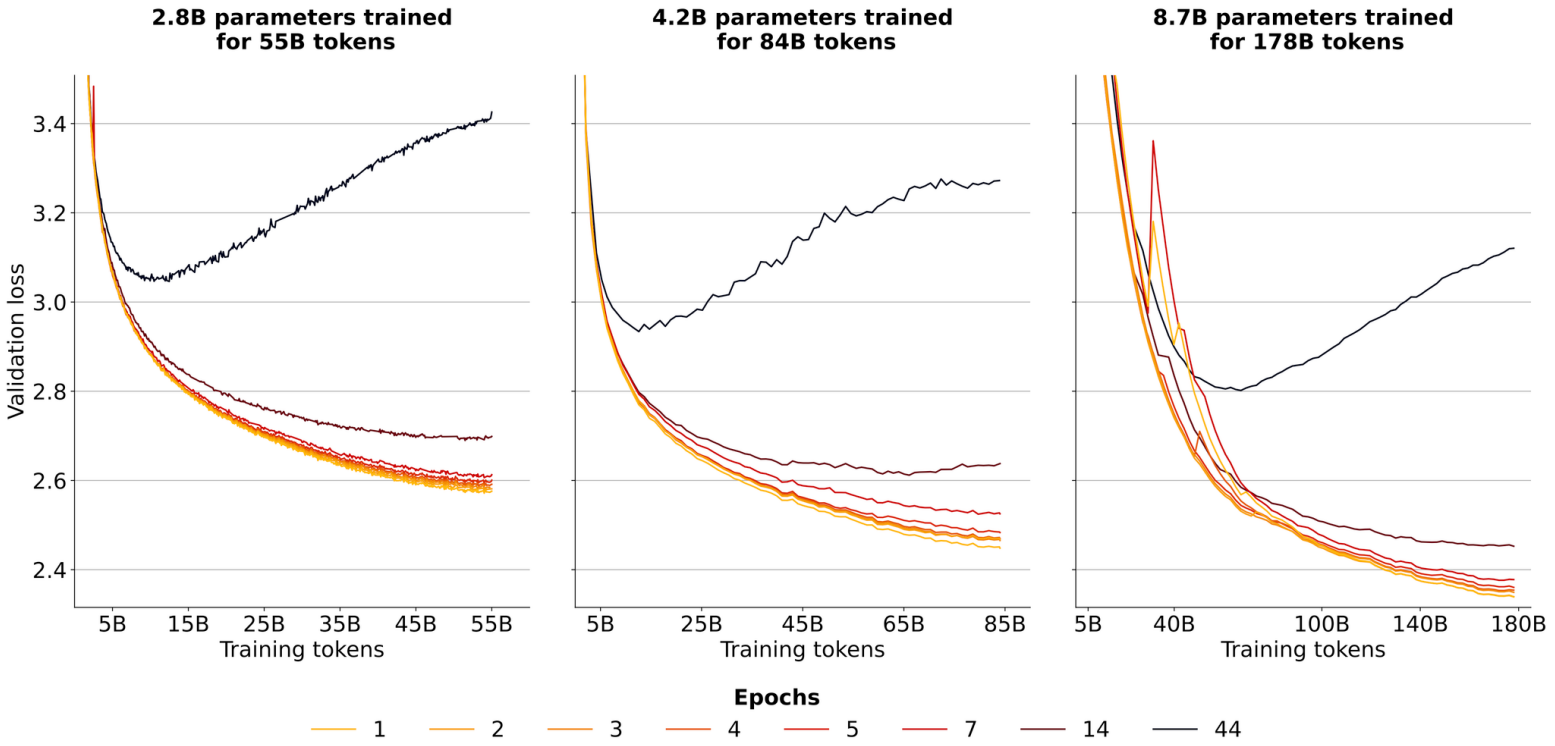
Linear growth of web data

Limited Data, Infinite Compute



Limited domain-specific data

Limited Data, Infinite Compute



Autoregressive LLMs diminish or overfit after a few epochs

Why Autoregressive Receives its Popularity

Despite their quick overfitting, the state-of-the-art auto-regressive models are popular due to:

> Optimal utilization of modern GPU architectures.

> Natural language can be modeled in the left-to-right direction with low loss.

R1: Optimal utilization of modern GPU architectures

> Decoder-only transformer

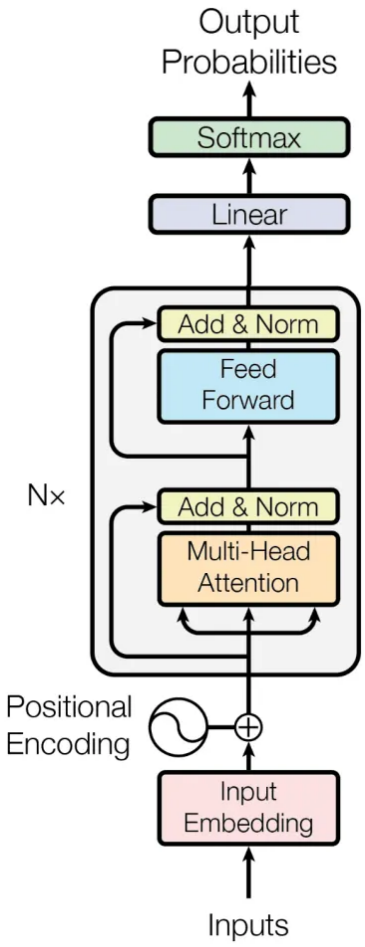
> Causal masking

> Teacher-forcing (training)

> KV cache, Continuous batching, etc. (inference)

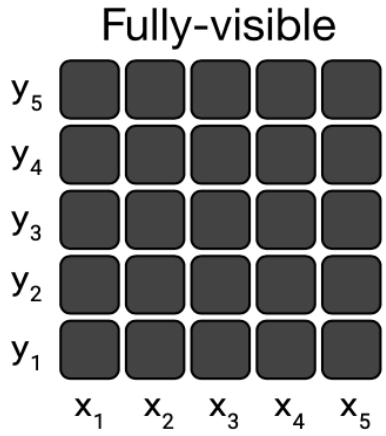
Algorithmically, the above combination maximizes the **training-time signal-to-FLOPs ratio** and **test-time efficiency.**

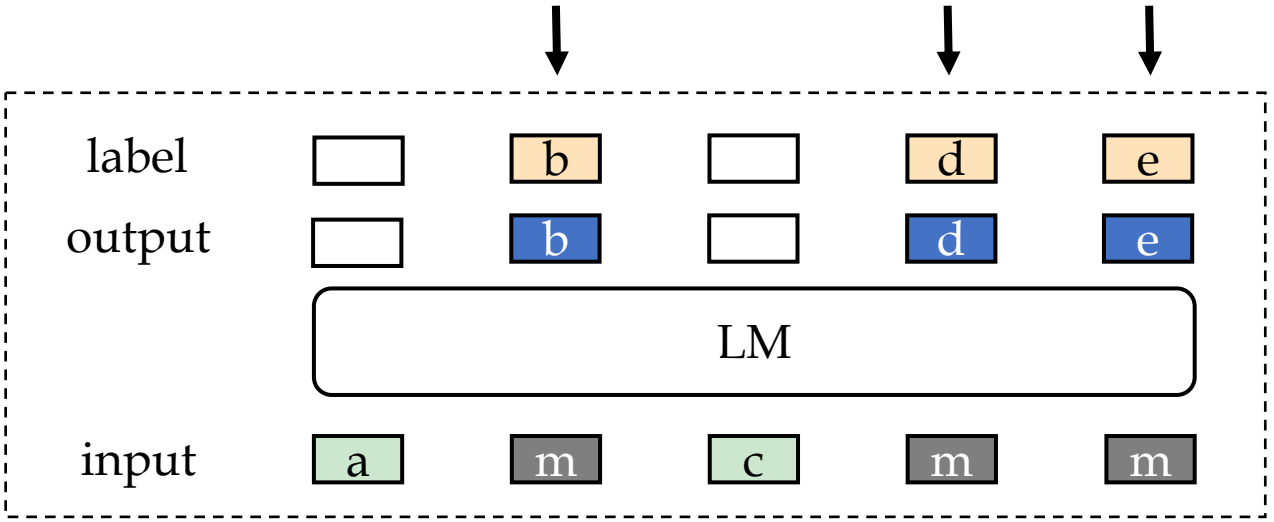
High Training-Time Signal-to-FLOPs Ratio



Masked Modeling

> Only masked positions receive signals.





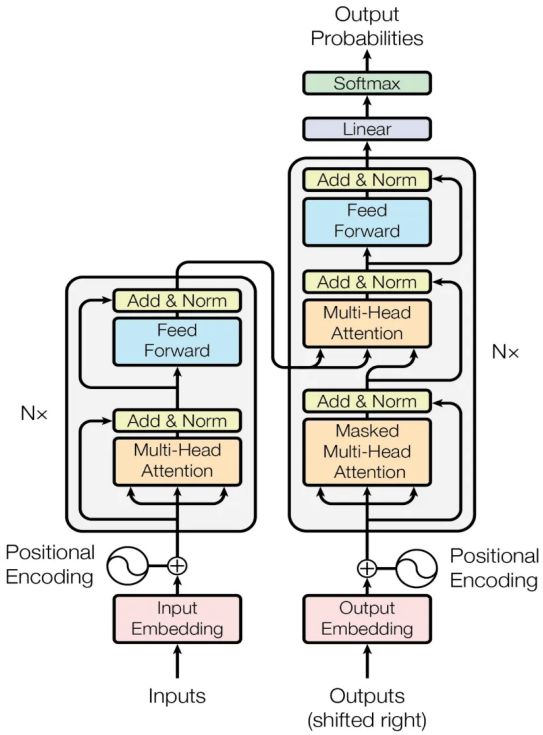
x normal tokens

x model pred

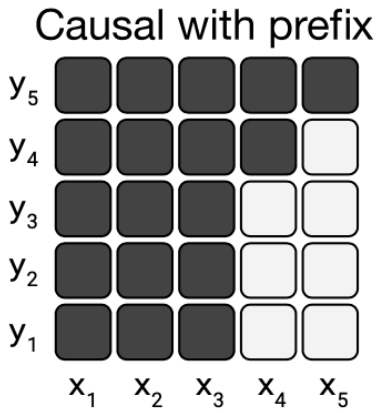
m <mask>

x label

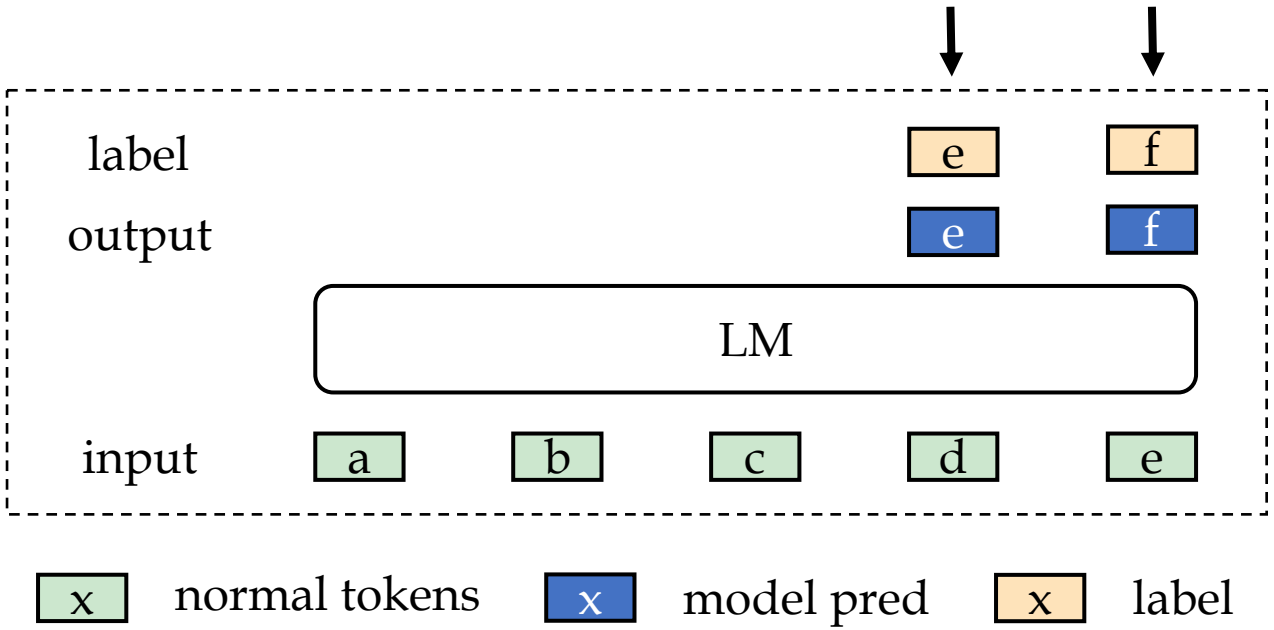
High Training-Time Signal-to-FLOPs Ratio



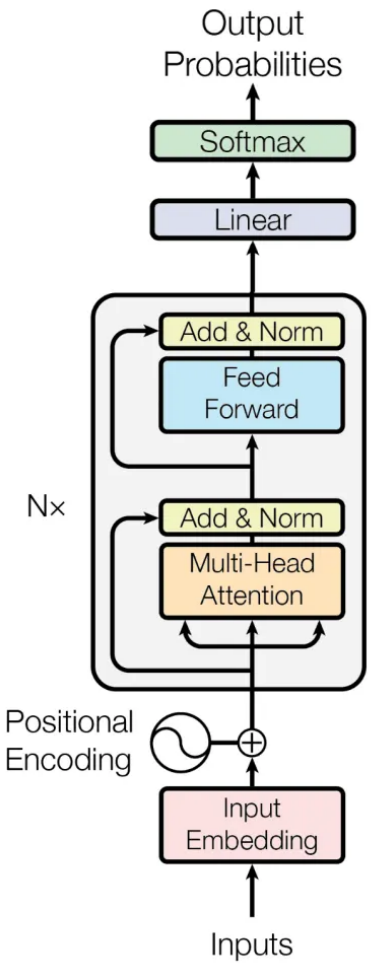
Encoder-decoder



> Prefix does not receive signals.

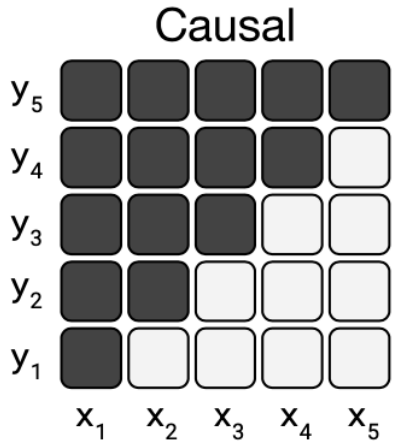


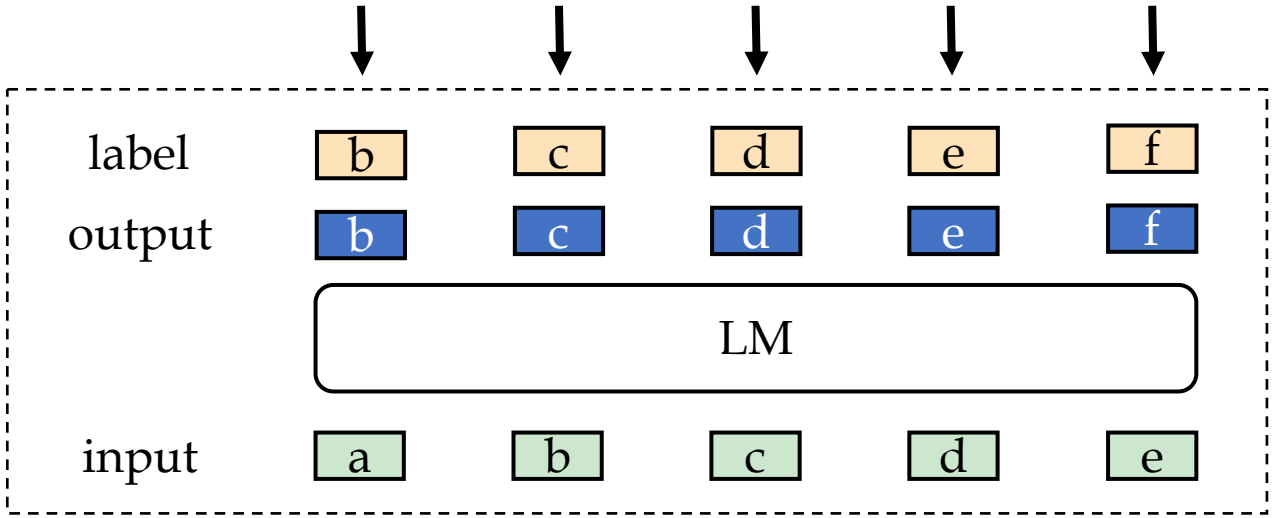
High Training-Time Signal-to-FLOPs Ratio



Decoder

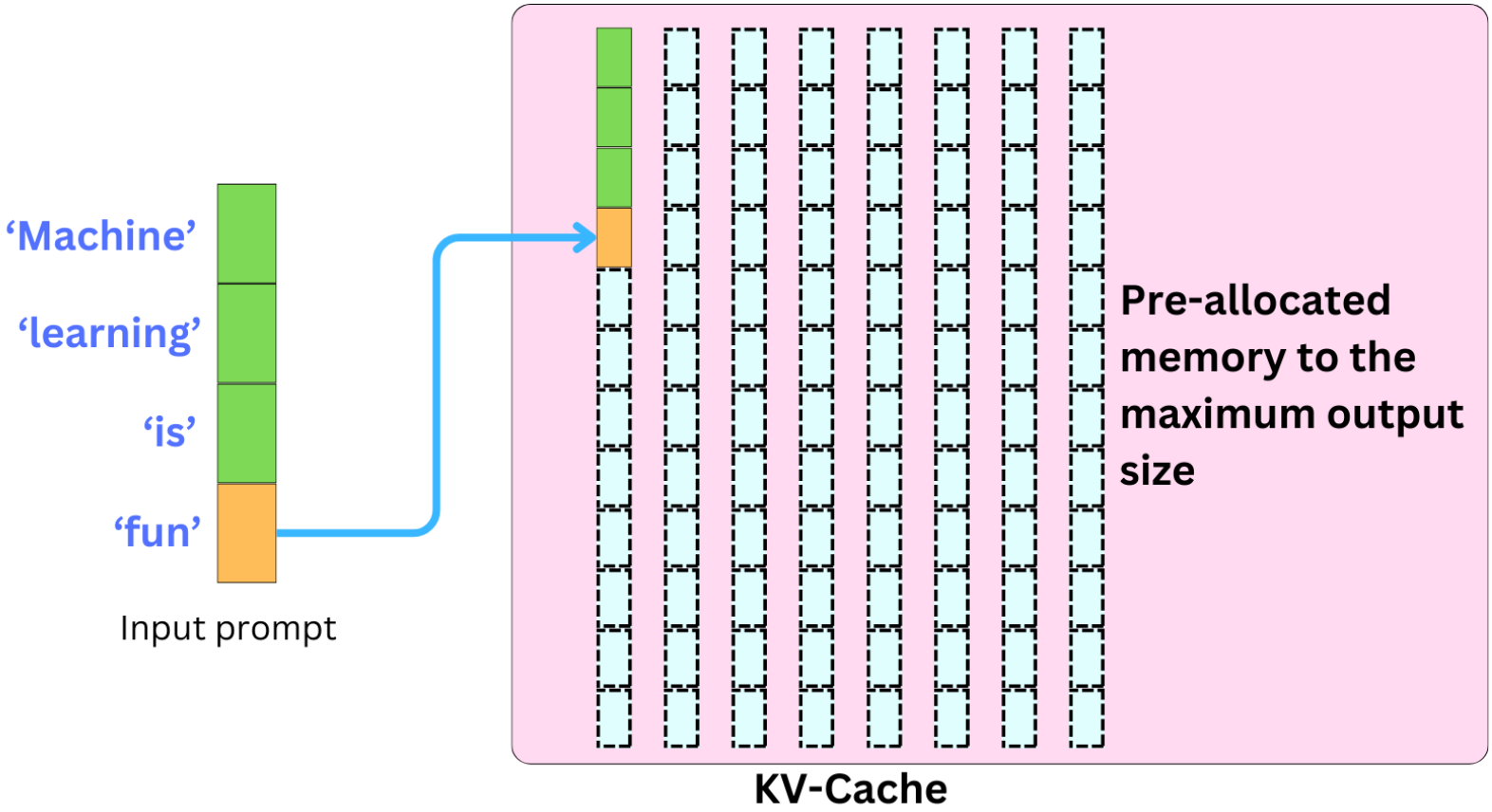
> Full sequence receives signals.





x normal tokens x model pred x label

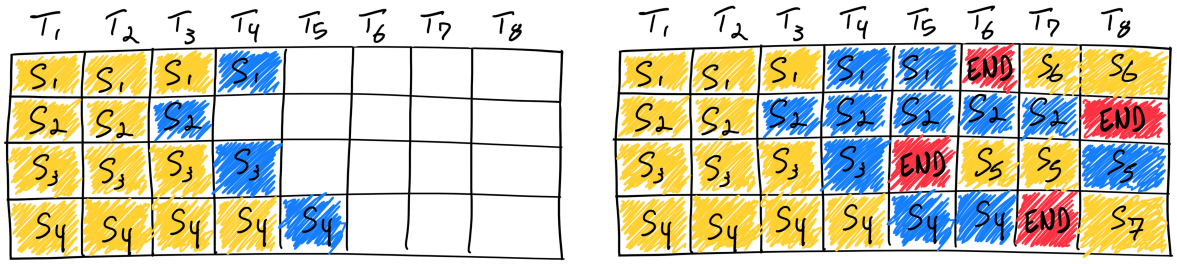
High Inference Efficiency



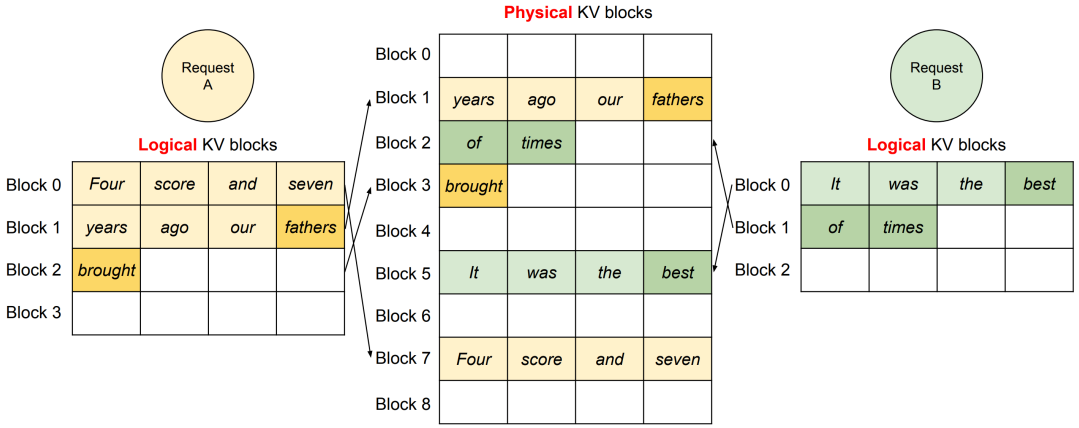
Causality enables KV cache

Inference FLOPs grow linearly!

High Inference Efficiency



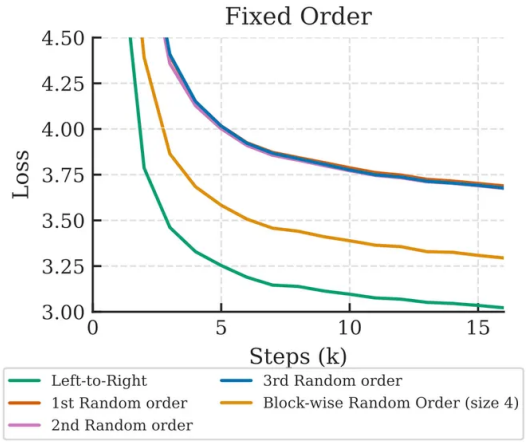
Continuous Batching



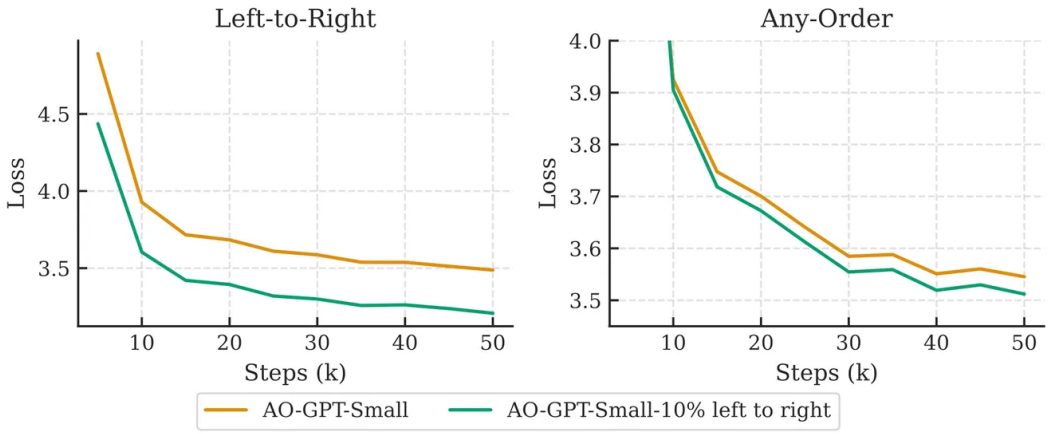
Paged Attention

Token-by-token generation enables fine-grained **batching** and **memory strategies**.

R2: Web Text Can be Modeled Well in Left-to-Right



(a) Convergence speed with different fixed prediction orders: left-to-right, fixed random, and fixed block-wise random.



(b) Impact of adding 10% left-to-right (L2R) data to AO-GPT training on its L2R and any-order loss.

Why?

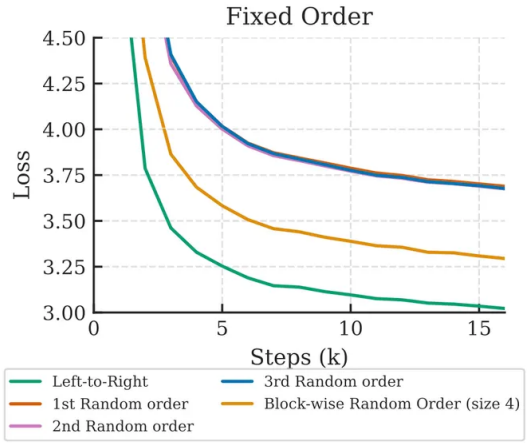
> Most text data are generated by humans, while humans are RNNs.

Left-to-right Modeling is not Optimal

Not all web text are dominated by left-to-right.

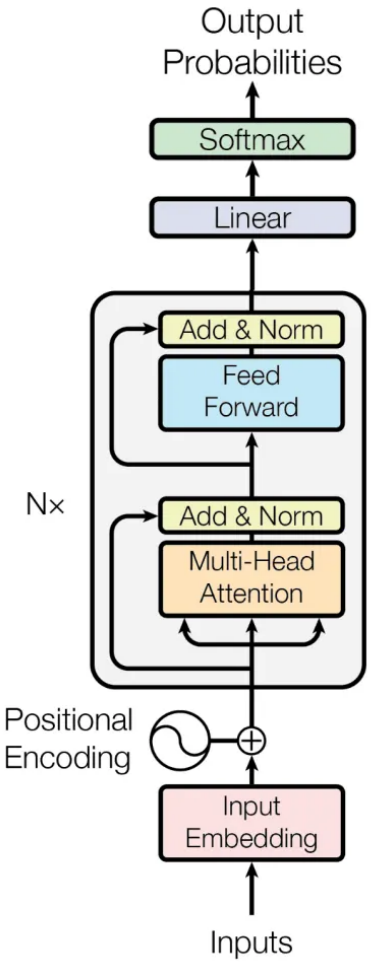
> Code
> Biology data
> Database entries
> Symbolic notations
> etc.

Even general web data can also be modeled in other directions, evidence:

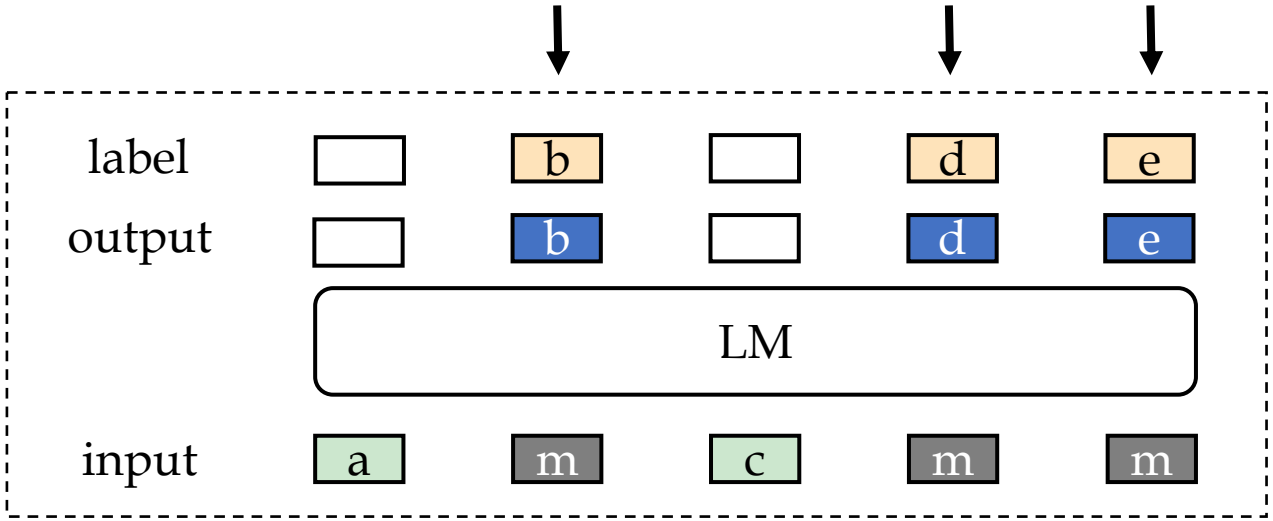


(Masked) Diffusion Language Models

$$\mathcal{L} \triangleq \mathbb{E}_{t,q(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\frac{\alpha_t'}{1-\alpha_t}\sum_{\{i|\boldsymbol{x}_t^i=m\}} -\log p_\theta(\boldsymbol{x}_0^i \mid \boldsymbol{x}_t)\right] \geq -\log p_\theta(\boldsymbol{x}_0),$$



Encoder

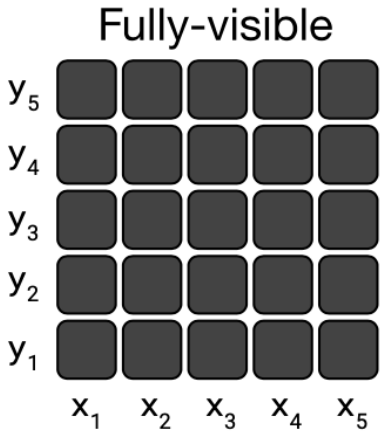


x normal tokens

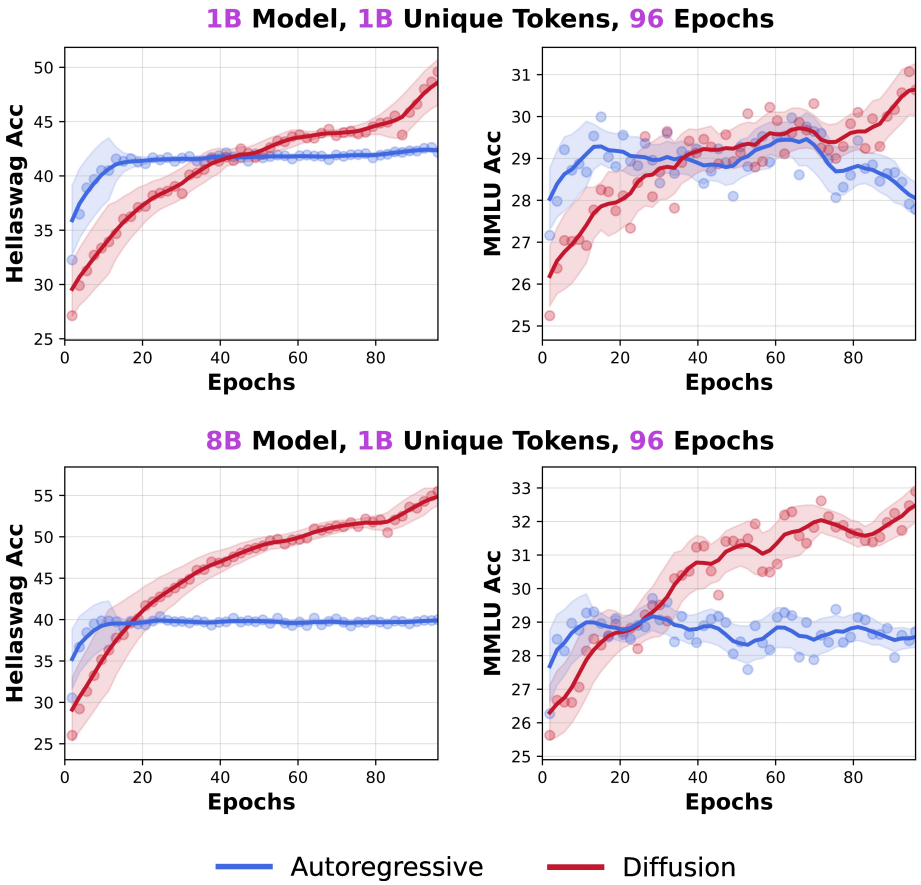
x model pred

m <mask>

x label



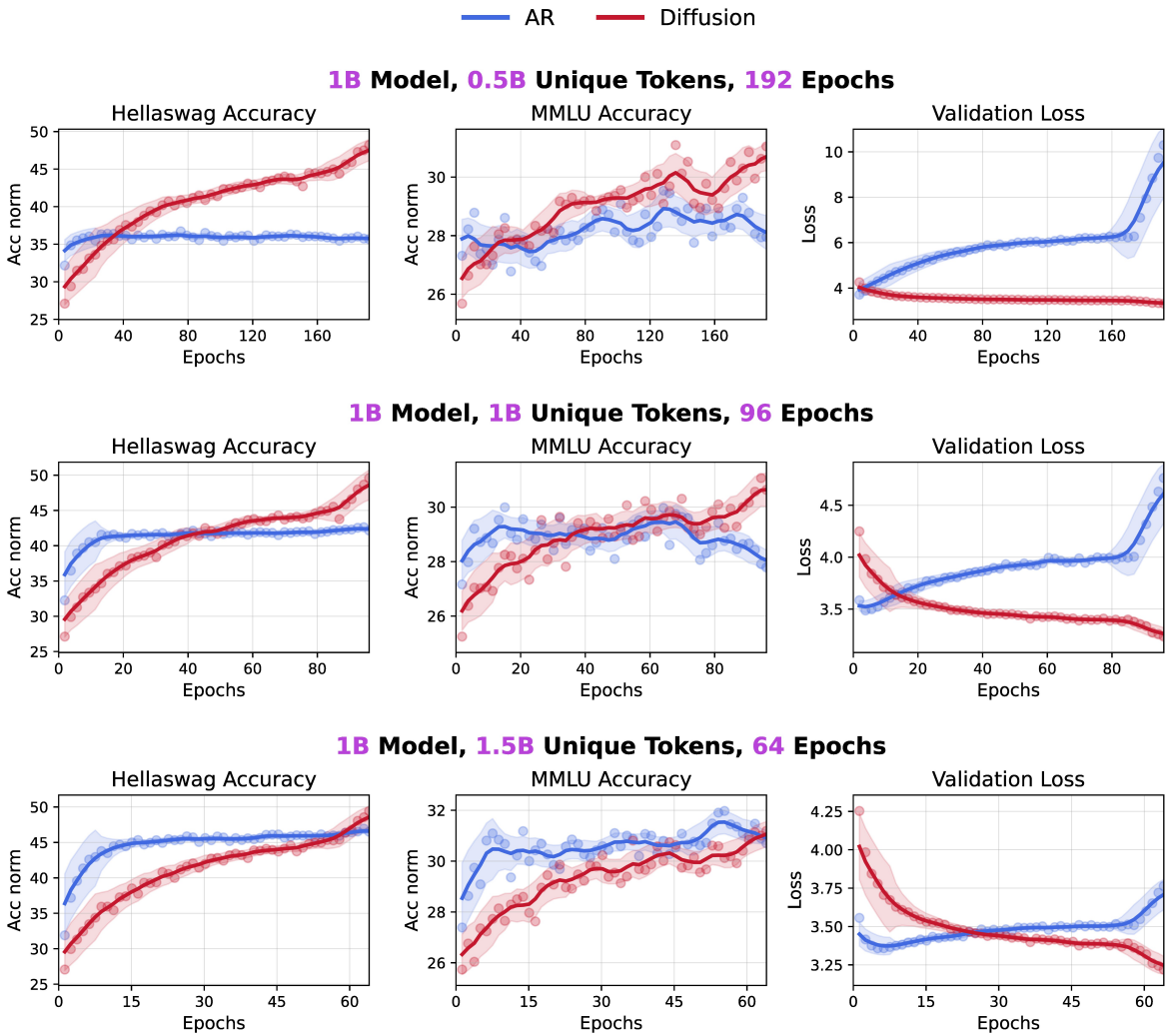
Diffusion Language Models are Super Data Learners



Overall Setup:

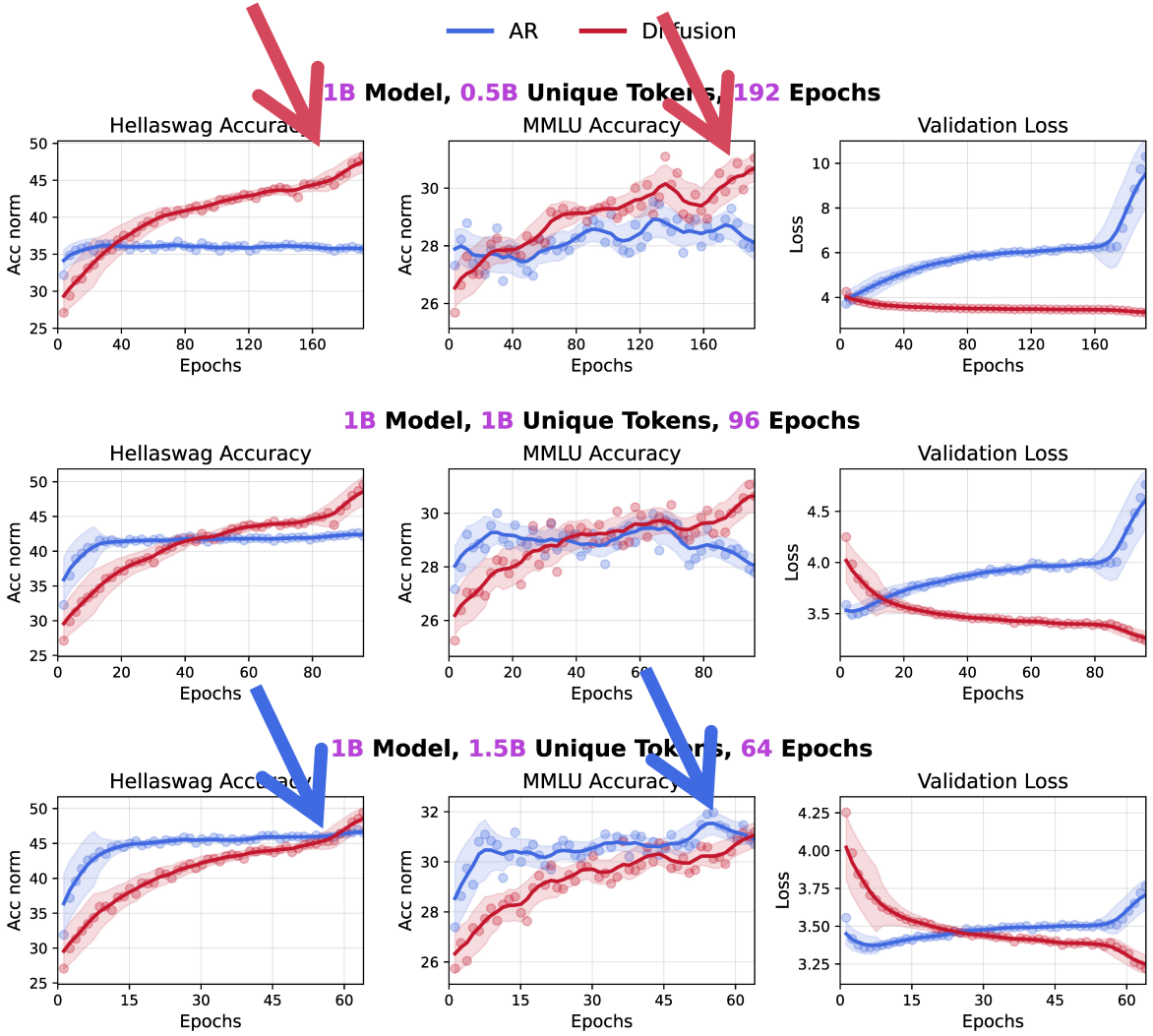
> Dense **1B/8B** models trained on a fixed **96B**-token budget, varying unique tokens from **0.5B** to **96B**.
> A **1B** DLM was also trained for **480** epochs on **1B** unique tokens.

Diffusion Language Models are Super Data Learners



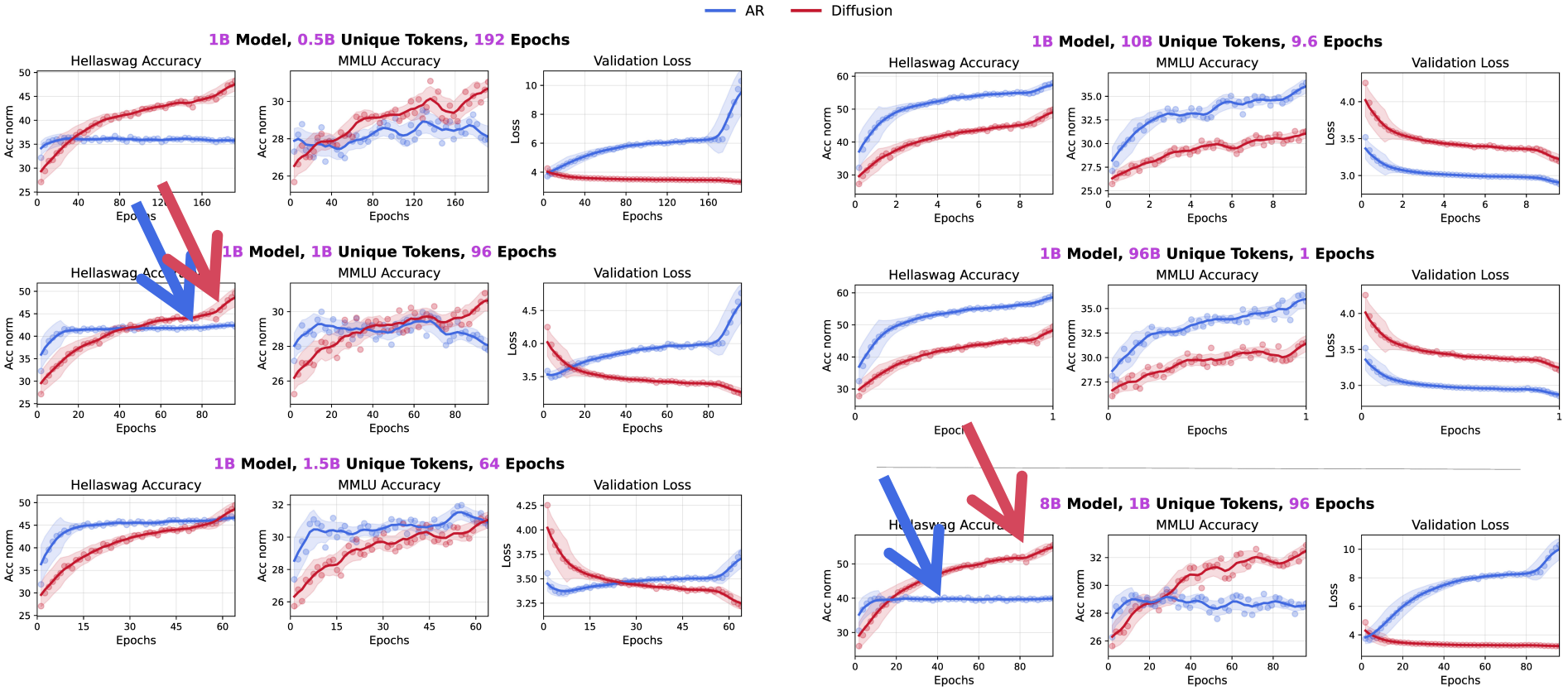
With fixed data budget, DLM clearly **crossover** the AR counterparts at some point by repeating the data.

Diffusion Language Models are Super Data Learners



DLMs exhibit **>3x** data potential compared to autoregressive models.

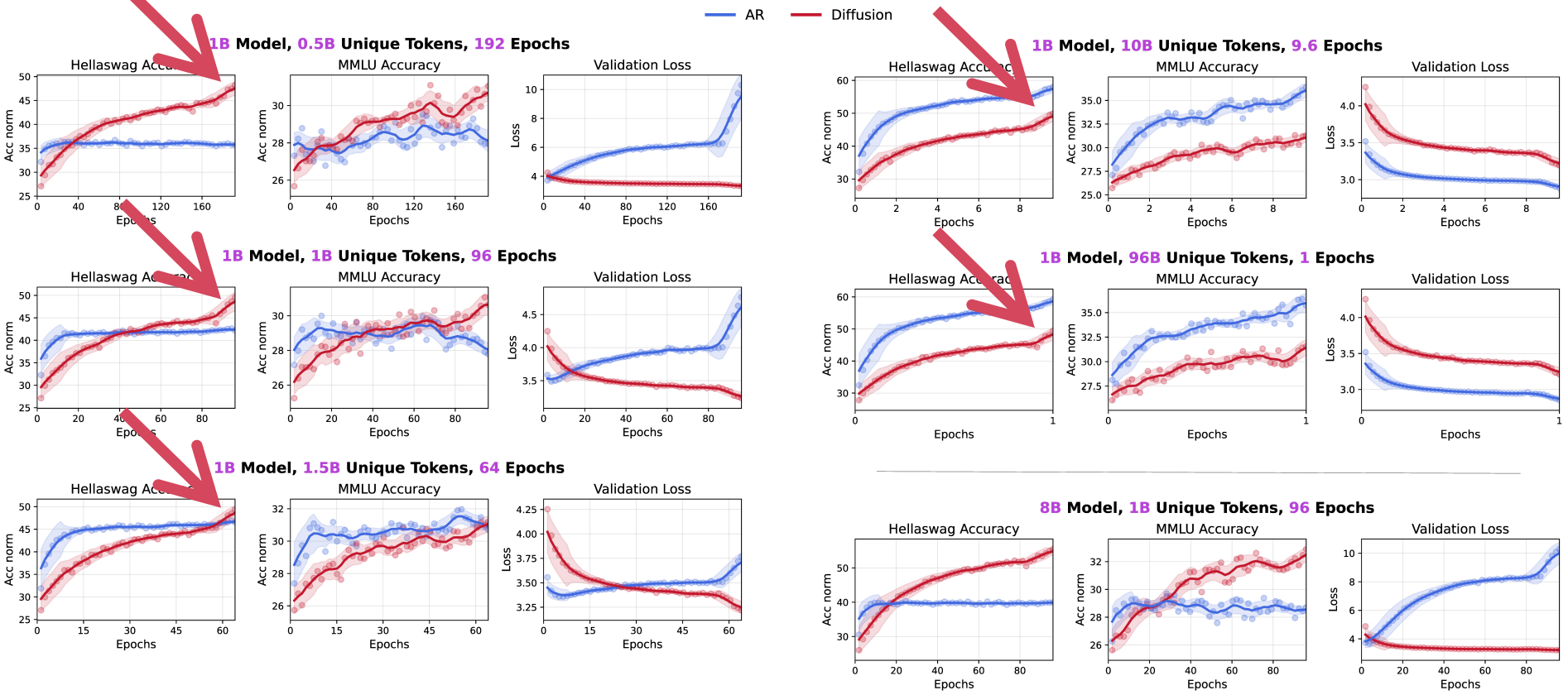
Diffusion Language Models are Super Data Learners



> Increasing the model size from 1B to 8B further unleashes the data potential

> While AR doesn't benefit from a larger model size under data constraint.

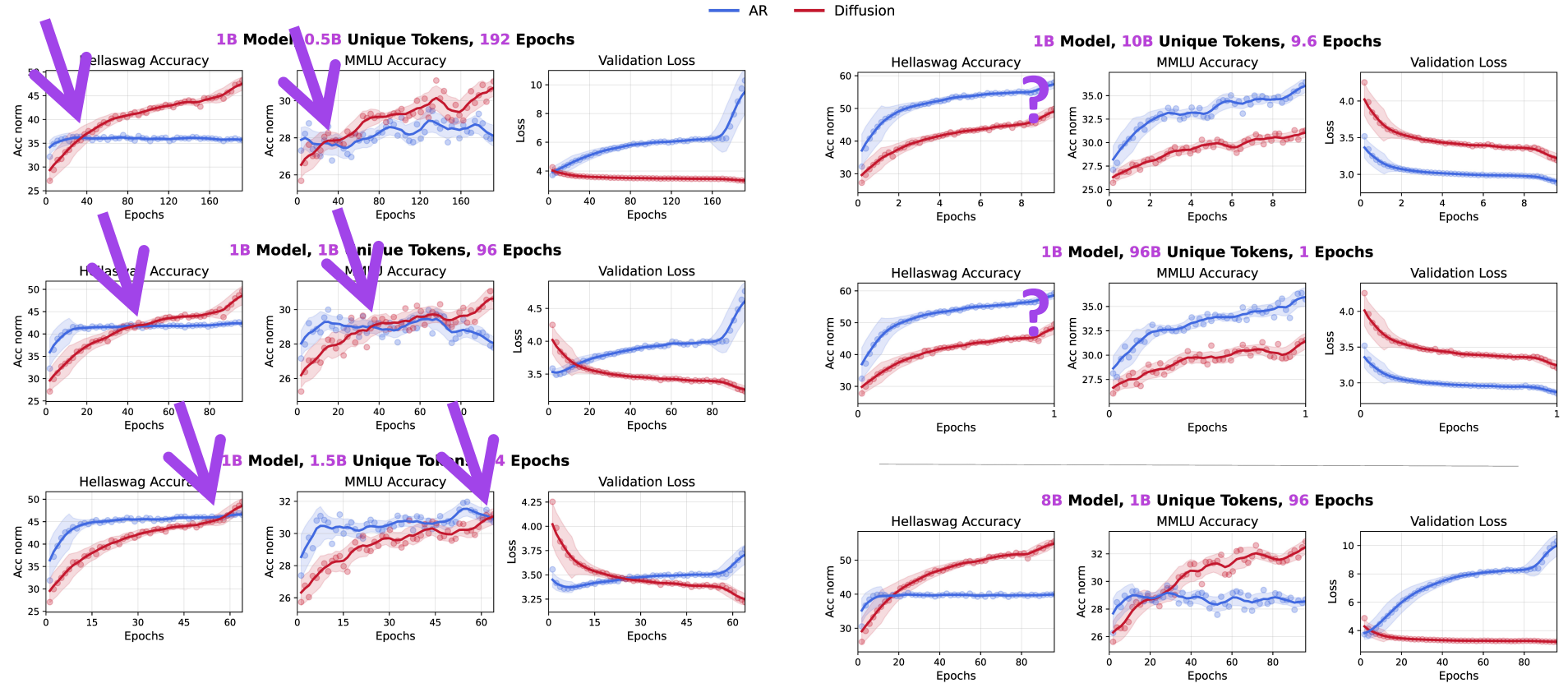
Diffusion Language Models are Super Data Learners



> DLMs show **negligible performance degradation** when drastically reducing unique data from **96B** to **0.5B** tokens.

> Its data potential is higher than we imagine.

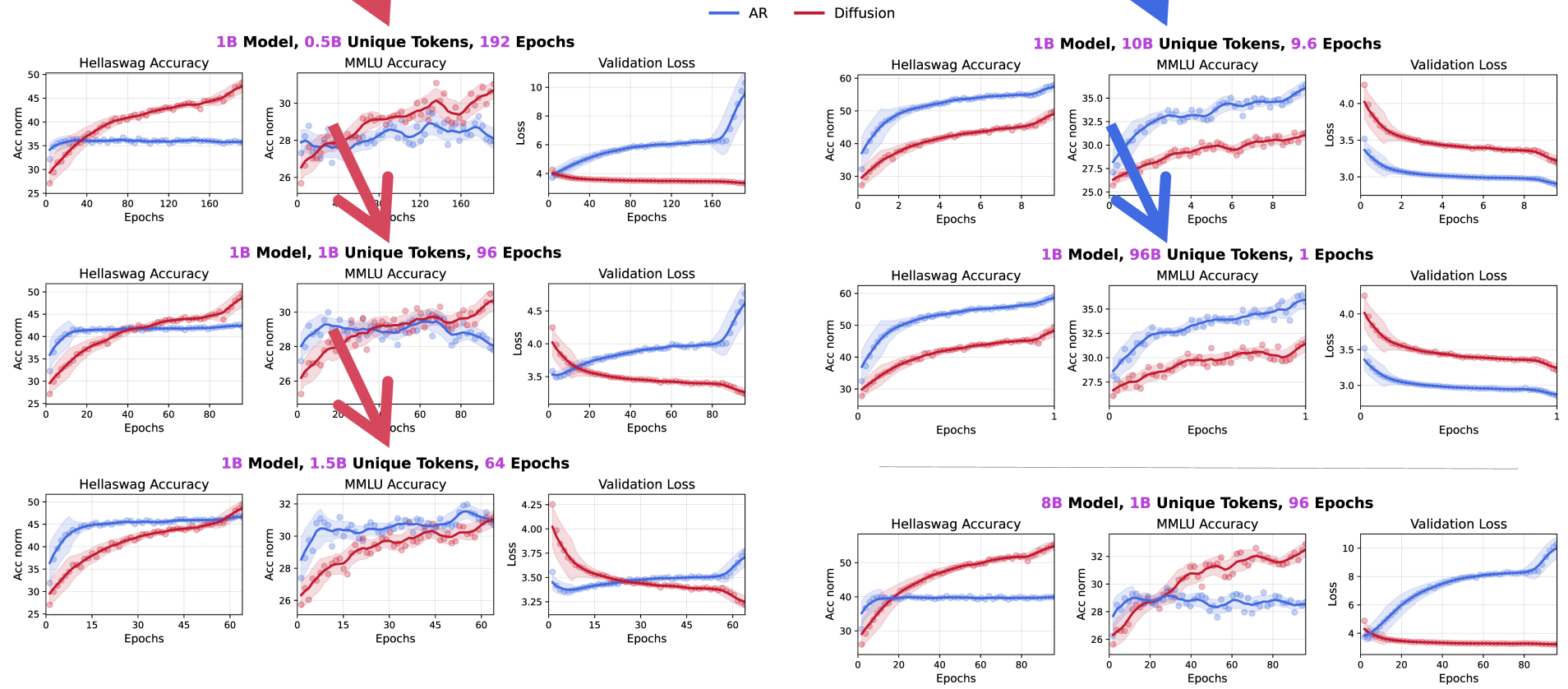
Diffusion Language Models are Super Data Learners



> The crossover point on different evals are similar.

> More unique tokens, later it crossovers.

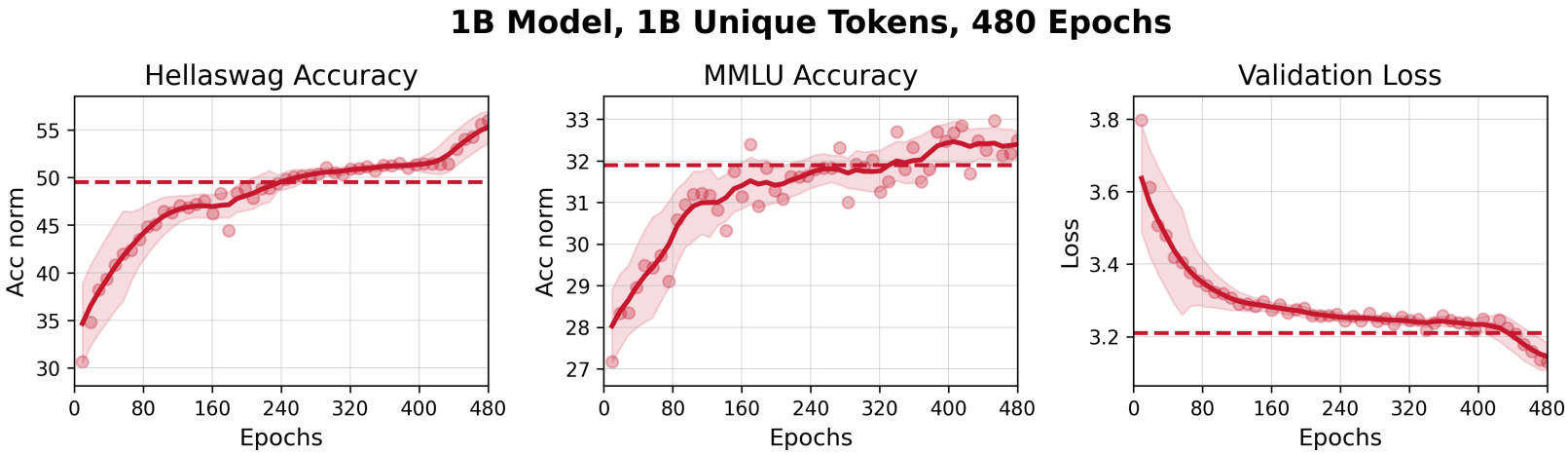
Diffusion Language Models are Super Data Learners



> Under compute-constrained settings, AR fits the data faster;

> Under data-constrained settings, DLM achieves a higher performance.

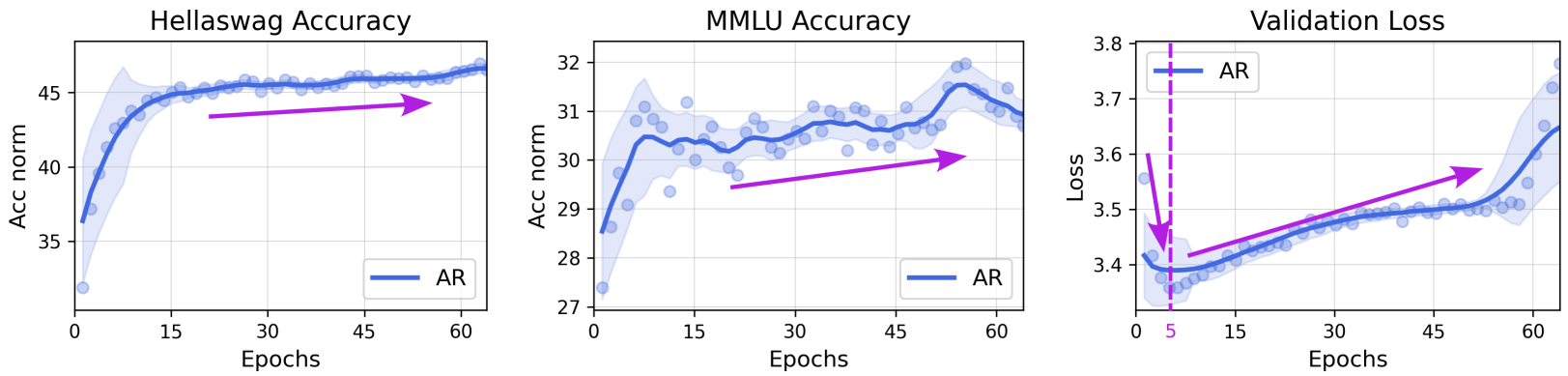
Diffusion Language Models are Super Data Learners



> On only 1B unique tokens, it achieves ~**56**% accuracy on HellaSwag and ~**33**% on MMLU, significantly outperforming AR's ~**41**% and ~**29**%, respectively

> Even under such extreme repetition, **performance did not saturate**, suggesting that DLMs can extract substantially more signal from a fixed 1B-token corpus

High Validation Loss ≠ Degraded Intelligence

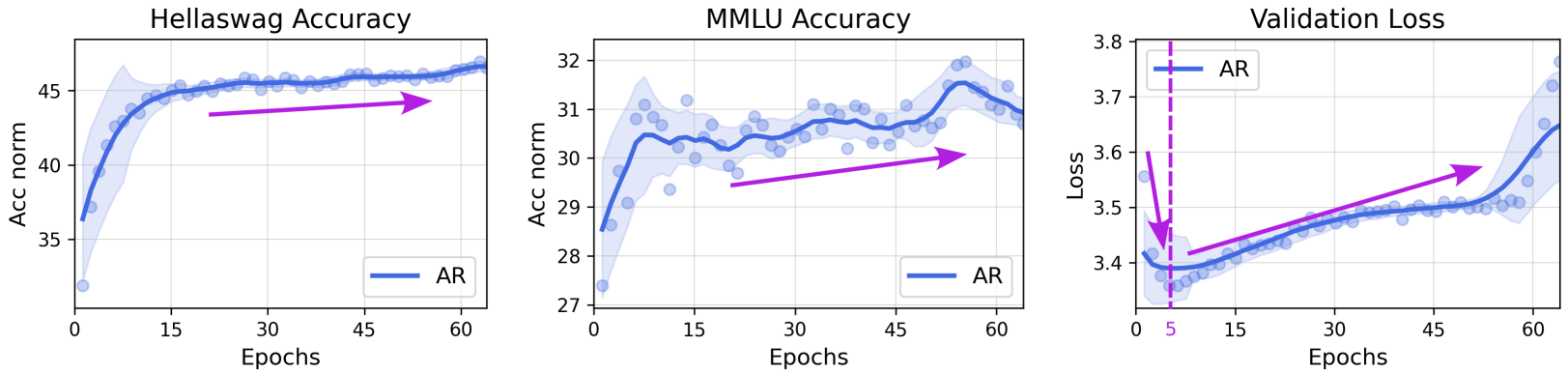


Is val loss a good metric to monitor "overfitting" performance?

> When models get "overfit" on validation subsets, their performance on down-stream evaluations **doesn't necessarily drop**, and may keep improving till the end of training.

> AR is measuring exact negative likelihood, while DLM optimizes an upper bound.

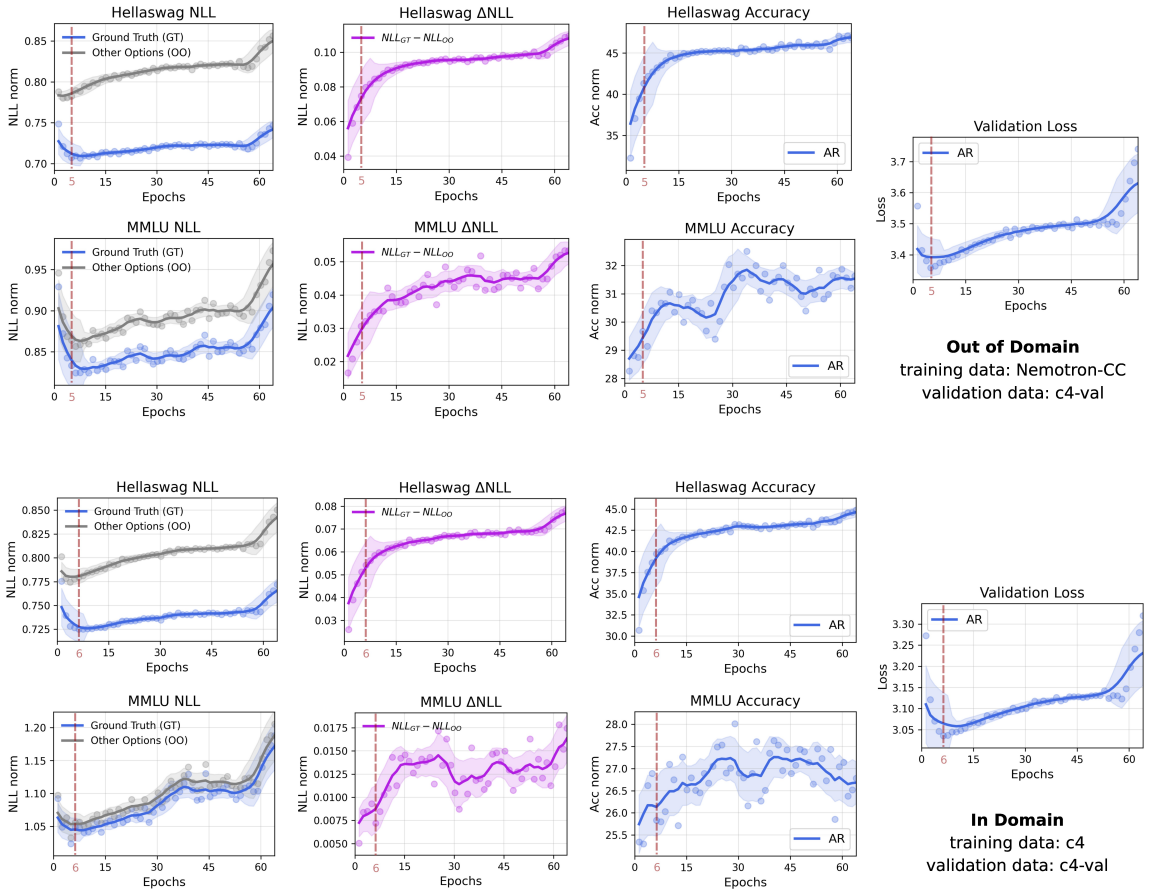
High Validation Loss ≠ Degraded Intelligence



Why?

Validation loss computes an **absolute** cross-entropy loss (NLL); multi-choice evals are decided by **relative** cross-entropy losses (ΔNLL).

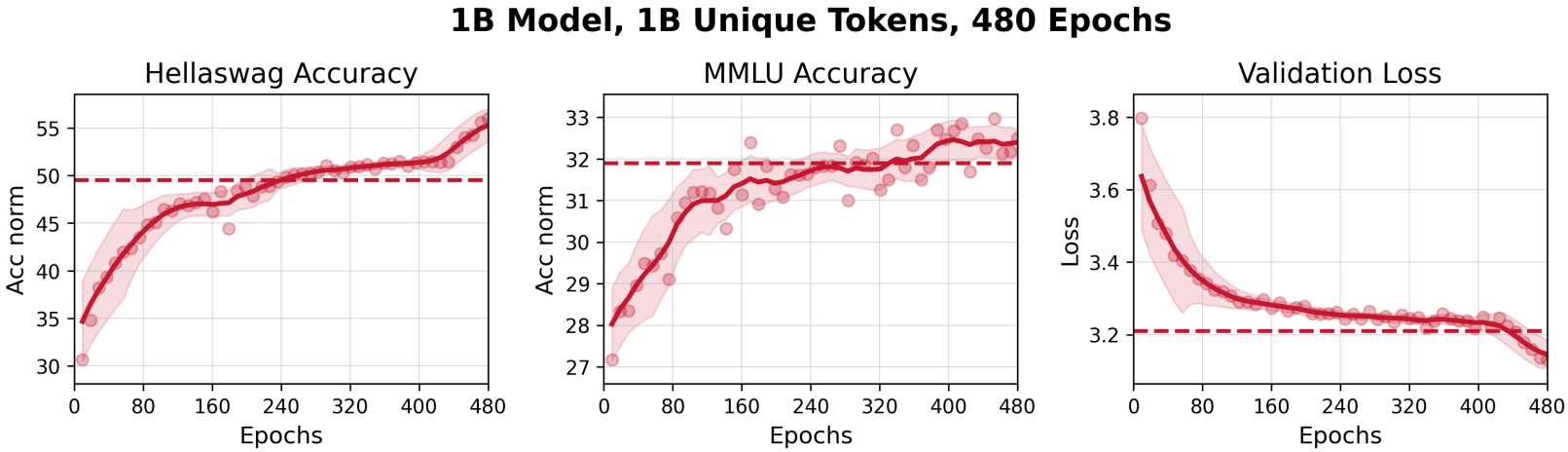
High Validation Loss ≠ Degraded Intelligence



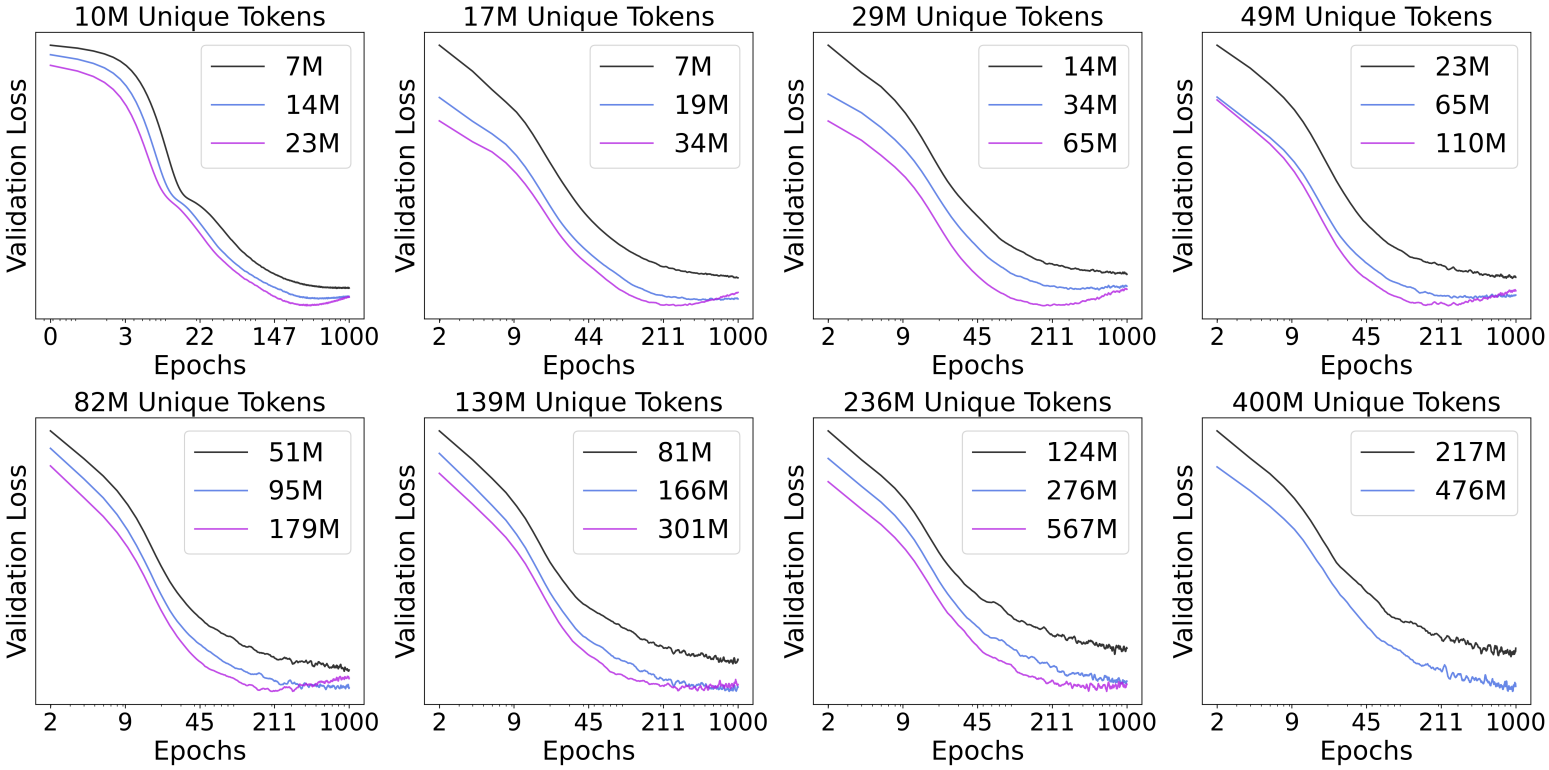
> **NLL**: Negative log-likelihood on the ground-truth and other options of multiple-choice evals (NLLs on other options are averaged).

> **ΔNLL**: The differences between the NLLs on ground-truth and other options, which keeps growing.

When Does Diffusion Language Models Saturate?



Diffusion Language Models also Overfit the Data



> Overfitting eventually emerges after prolonged training.

> **Larger unique data size delay** overfitting, while **larger models accelerate** its onset.

What is the Real Advantage of Diffusion Language Models?

Three angles to interpret:

> Reduced Inductive Bias via **Bidirectional Modeling**.

> Super-Density: **more training and test time FLOPs** per task.

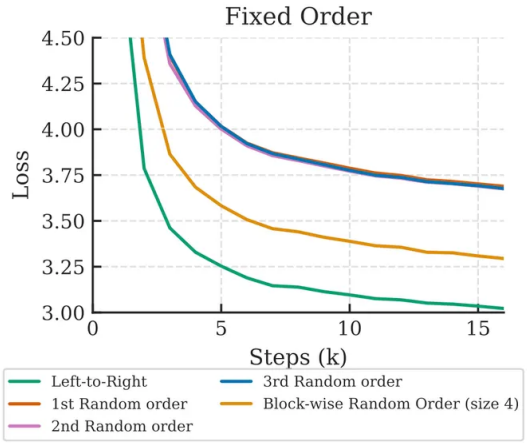
> **Data augmentation** via injecting noise.

Reduced Inductive Bias via Any-Order Modeling

Not all web text are dominated by left-to-right.

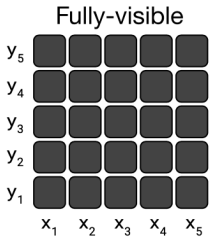
> Code
> Biology data
> Database entries
> Symbolic notations
> etc.

Even general web data can also be modeled in other directions, evidence:



$$\mathcal{L} \triangleq \mathbb{E}_{t,q(\boldsymbol{x}_t|\boldsymbol{x}_0)}\left[\frac{\alpha_t'}{1-\alpha_t}\sum_{\{i|\boldsymbol{x}_t^i=m\}} -\log p_\theta(\boldsymbol{x}_0^i \mid \boldsymbol{x}_t)\right] \geq -\log p_\theta(\boldsymbol{x}_0),$$

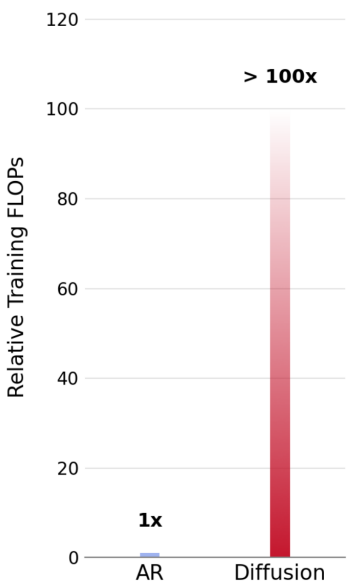
+

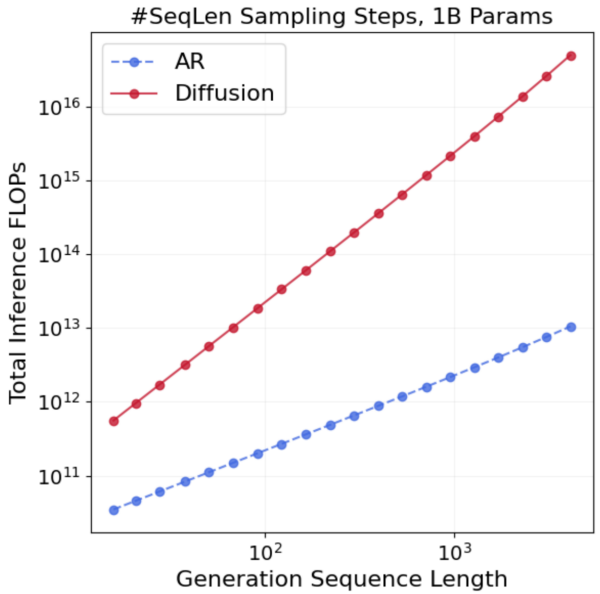


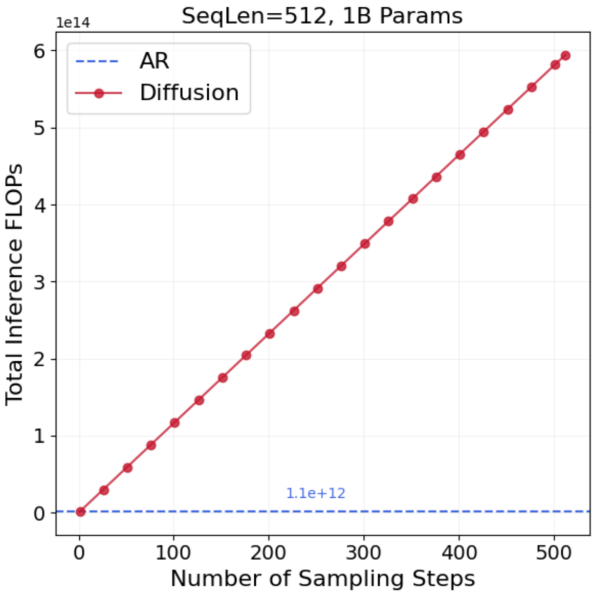
=

Any-Order Modeling

Super-Density: More Training & Test Time FLOPs Per Task







> **Two orders of magnitude (>100×)** more training FLOPs than AR to achieve full data potential.

> Sequence length 16 -> 4096, inference FLOPs **16× -> 4700× of AR**.

> Equals to AR's inference FLOPs when diffusion **sampling steps = 1**.

Data Augmentation via Injecting Noise

> When we do multi-epoch training, we are augmenting the data by injecting different noise into each data point.

> A training sequence of length N can be corrupted into 2^N data samples at most.

> A 1B dataset of 2048 sequence lengths will be expanded into **$488281.25 \times 2^{2048} \sim= 1.6\times10^{2119}$ TB** tokens at most, which is more than enough to fully fit the data.

Insights Behind the Data Augmentation

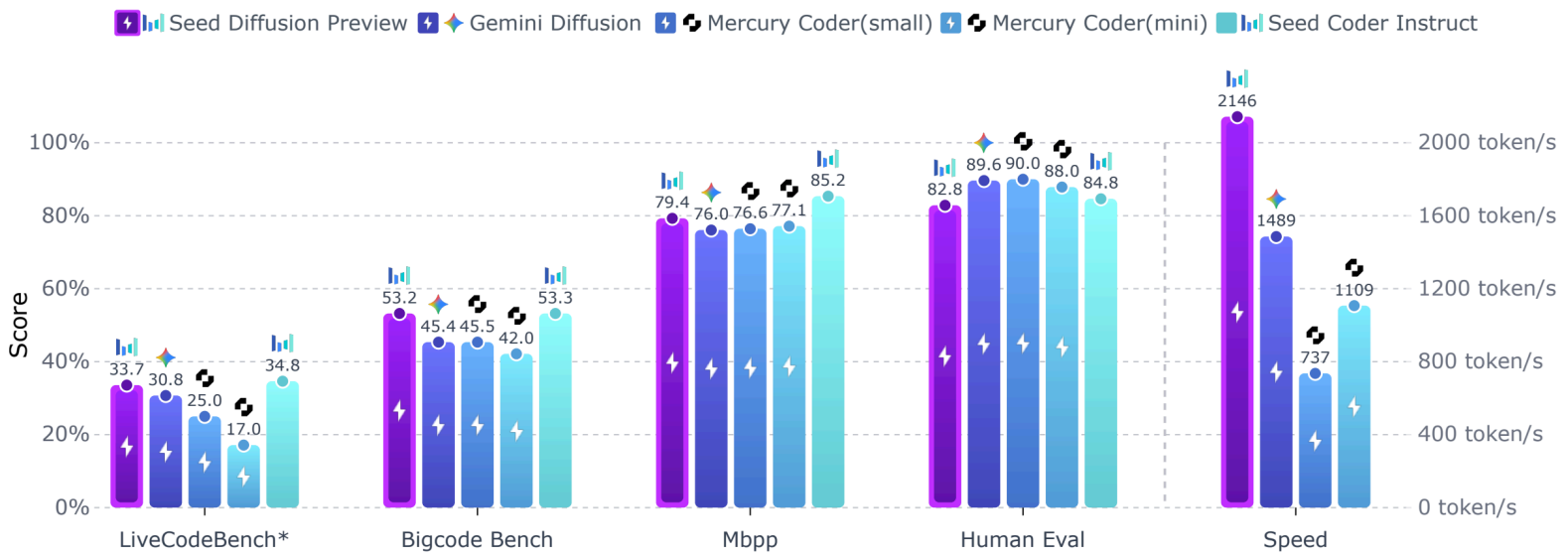
$$\mathcal{L} \triangleq \mathbb{E}_{t,q(\boldsymbol{x}_t|\boldsymbol{x}_0)} \left[\frac{\alpha_t'}{1-\alpha_t} \sum_{\{i|\boldsymbol{x}_t^i=m\}} -\log p_\theta(\boldsymbol{x}_0^i \mid \boldsymbol{x}_t) \right]$$

> The objective function explicitly requires each data point in the pre-training dataset to be corrupted at **multiple masking ratios and combinations** for more effective training.

> The more Monte Carlo sampling, the more precise the expectation.

> Therefore, ”data augmentation” is just approaching the true expectation. The true higher limit arises from the **bi-directional modeling** and the **super density**.

Efficient Serving of DLMs (batch-size = 1)



Efficient Serving of DLMs (batch-size = 1)

First of all, under this condition, it will be tricky for AR to use tensor core (requiring bsz >= 16), which makes it much slower.

However, we can suppose it can use the tensor core to simplify the problem.

Suppose we are doing tensor production of shape [4096, 4096] with [4096, N]:

To fully utilize the 132 SMs of an H100 GPU without getting bounded by throughput, we got **N ~= 128 at most** to not drastically increasing the latency.

In this condition, we assume latency(N=1) ~= latency(N=128), roughly.

Efficient Serving of DLMs (batch-size = 1)

Suppose we are generating 512 tokens, each with 4096 dimension:

AR requires **512 steps**, each step forwarding **1** tensor;
DLM requires **≤ 512 steps**, each step forwarding **512 / n_step** tensors.

To achieve comparable / lower latency than AR, we can either:

> Forwarding one block at a time, **blk_size < 128**, sampling 512 times (block diffusion with kv cache);

> Forwarding 512 tokens per step (consume 4x time), **predicting > 4 tokens per step** (multi-token prediction);

> It could be much faster as we are saving more time on moving the data between HBM and SMs;

> Combining them.

Efficient Serving of DLMs (batch-size > 1)

It's now throughput bound, i.e., throughput scales linearly with input batch size.

Suppose we are generating 512 tokens, **to achieve comparable / higher throughput (toks/s) than AR, we can either:**

> Use block diffusion with kv cache, **block size = 1**.

> Generate **512** tokens in one step.

> Use block diffusion and multi-token prediction, generate **512** tokens in **512/blk_size** steps.

The third strategy is more acceptable.

On more memory bounded devices, DLM throughput > AR.

Why diffusion language models?

1. **Higher data potential**, i.e., given the same amount of data, it achieves a higher performance via repeating on it.
2. **More training and test-time scaling**.
3. **Much lower inference latency than AR.**
4. **Comparable or higher throughput** compared with AR.