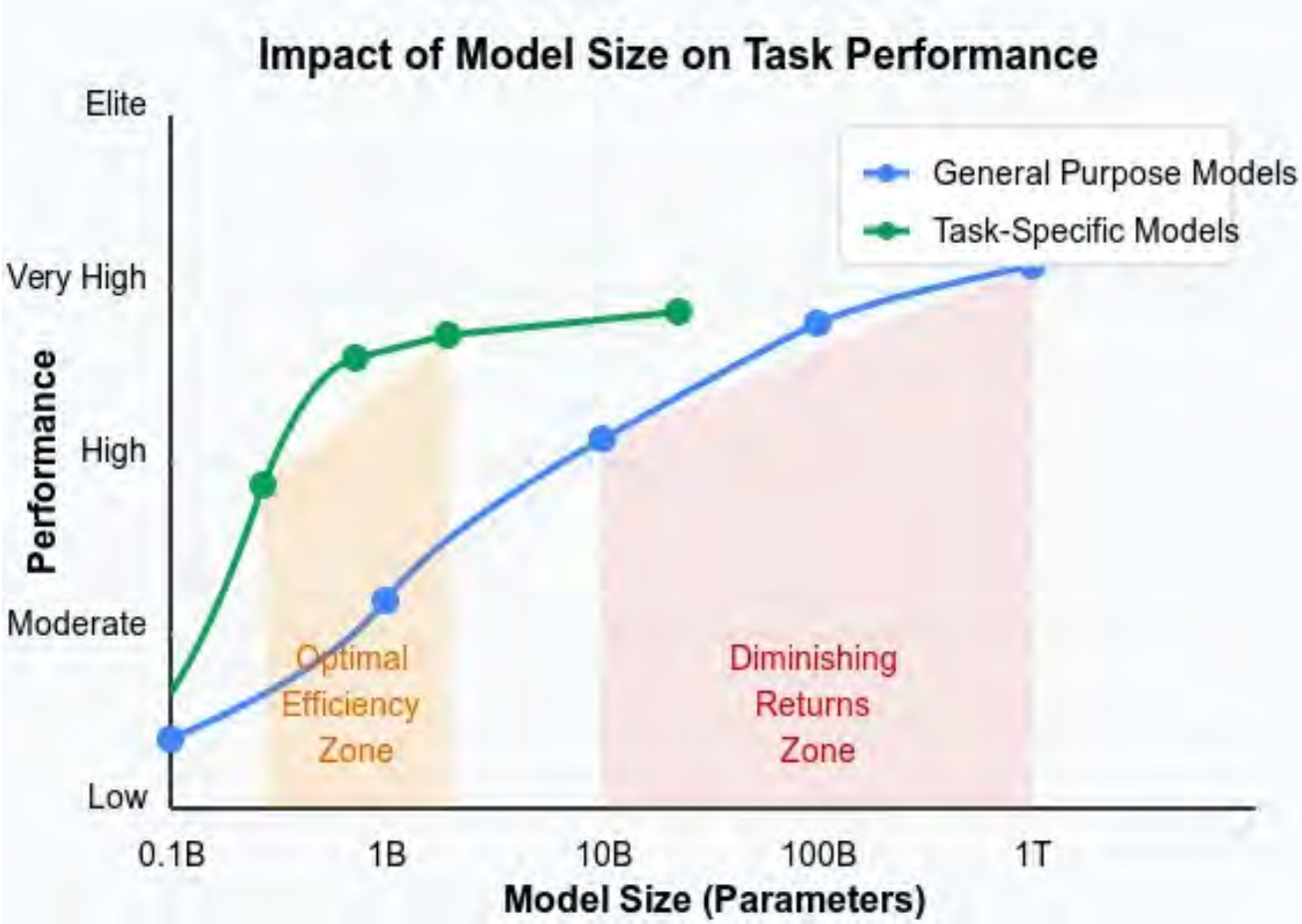
Closing the Gap Between Recurrent and In-Context Learning

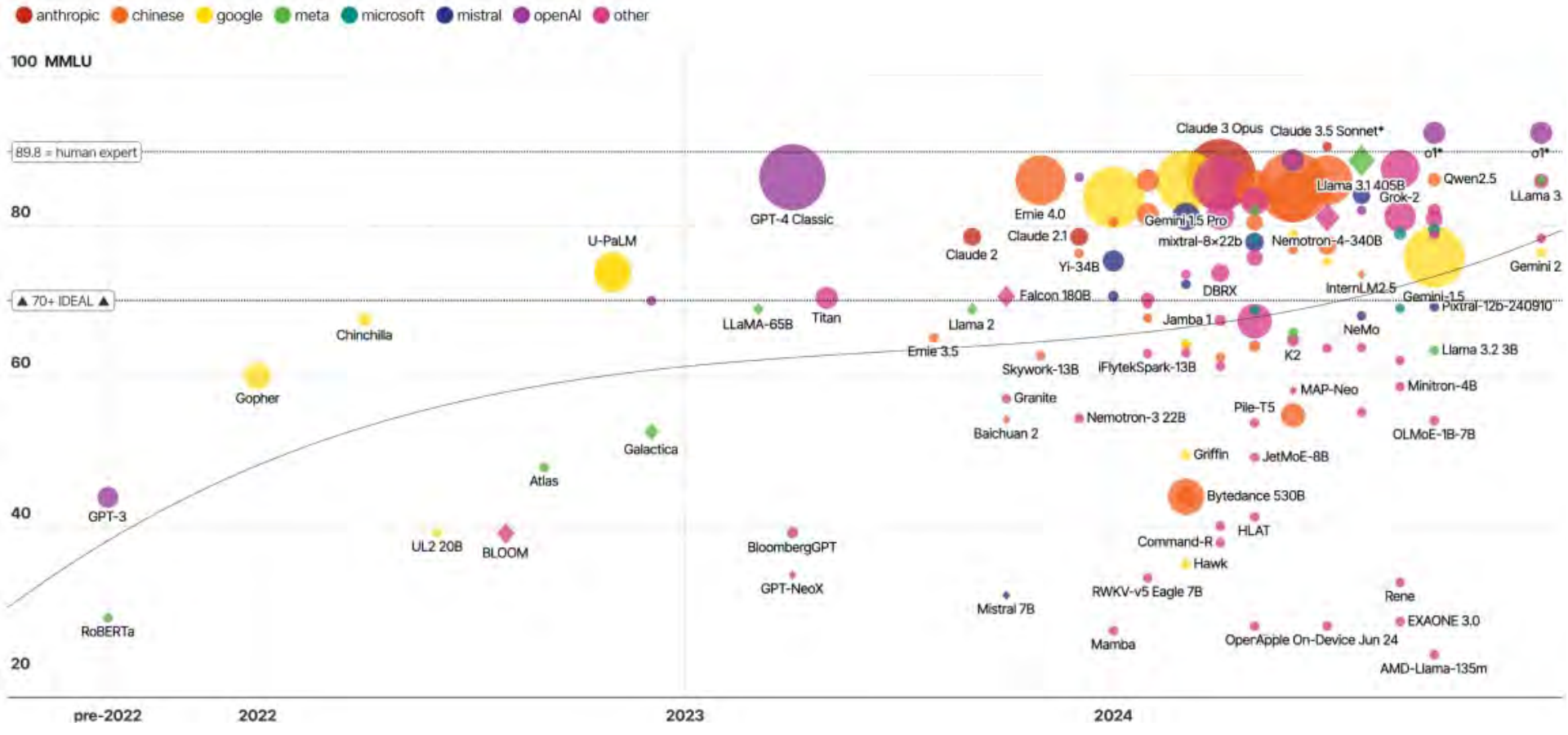
Sepp Hochreiter

JKU Institute for Machine Learning
JKU LIT Artificial Intelligence Lab
ellis LINZ unit
NXAI

Model Sizes Decrease



Model Sizes Decrease



Criteria	Small Language Models (SLMs)	Large Language Models (LLMs)
Inference Cost	✓ Low – cost-effective, great for scaling	✗ High – especially via API or at large scale
Hardware Needs	✓ Edge-friendly – can run on laptops or mobile devices	✗ Requires GPUs or cloud infrastructure
Privacy	✓ On-premise or local deployment	✗ Typically cloud-based; limited data control
Accuracy	✗ „Good-enough" for many tasks	✓ State-of-the-art performance in complex tasks
Fine-tuning	✓ Easy and affordable to customize	A Expensive and technically complex
Best Use Cases	Lightweight assistants, summarization, form-fil-	✓ Complex reasoning, multi-hop Q&A, creative tasks, coding ass.

xLSTM for Industrial AI

- Transformer technology is slow and memory inefficient
- LSTM is used in many industries, e.g. for time series prediction
- xLSTM extends LSTM to the performance of Transformers
- xLSTM is fast, memory efficient and energy saving

NXAI GmbH, Linz

xLSTM

xLSTM is faster at inference than Transformer:

- **Embedded** and **edge** applications
- **Robotics**
- **Self-driving** vehicles and **drones**
- **Autonomous production** systems

xLSTM

Transformer disadvantages

- quadratic in context length: compute intensive
- only pairwise interactions via dot products
  - ➢ no complex interactions
  - ➢ no abstractions that combine more sequence elements

xLSTM

LSTM has stood the test of time

- led to many deep learning success stories
- constituted the first Large Language Model (ELMo)
- linear in context length
- complex interactions and abstraction

Can we **scale LSTM** to billions of parameters?

LSTM Limitations

- Inability to **revise storage decisions**
  → Nearest Neighbor Search

- **Limited storage capacities**
  → Rare Token Prediction

- **No parallelization** for training

Nearest Neighbor Search:

Given the flag of Monaco 

Find most similar flag and remember the population size of that country

At sequence end give this population size

similarity score

[redacted]

population

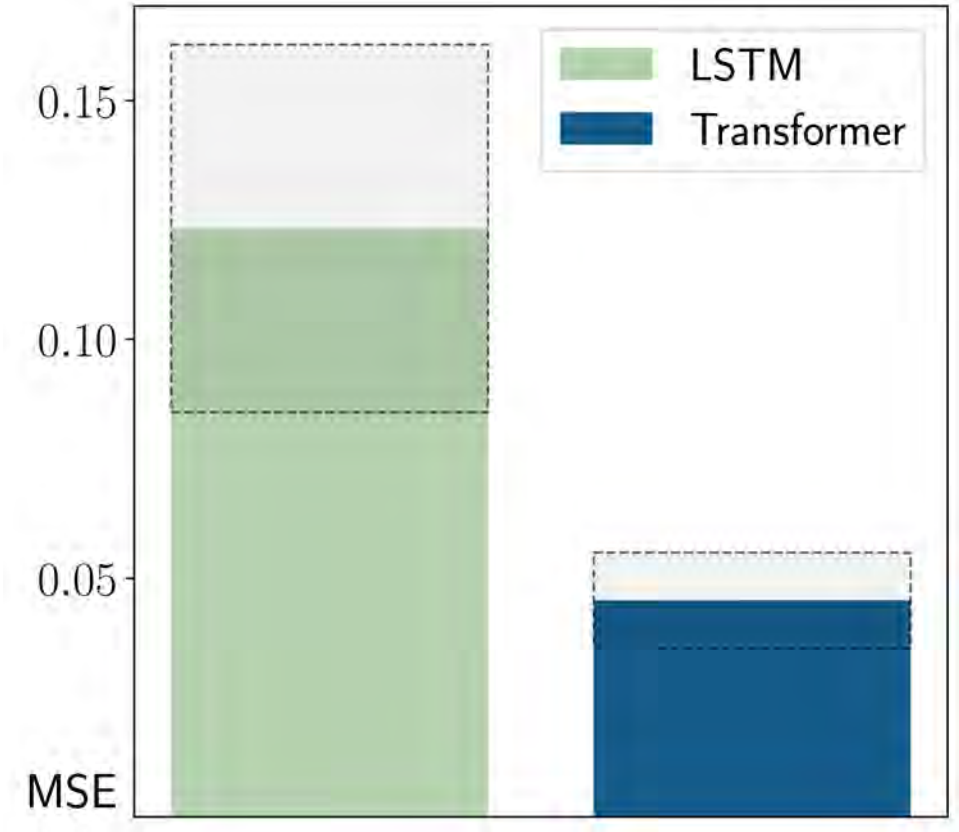
[redacted]

LSTM Limitations

Nearest Neighbor Search

new most similar $\rightarrow$

revise both population size and largest similarity

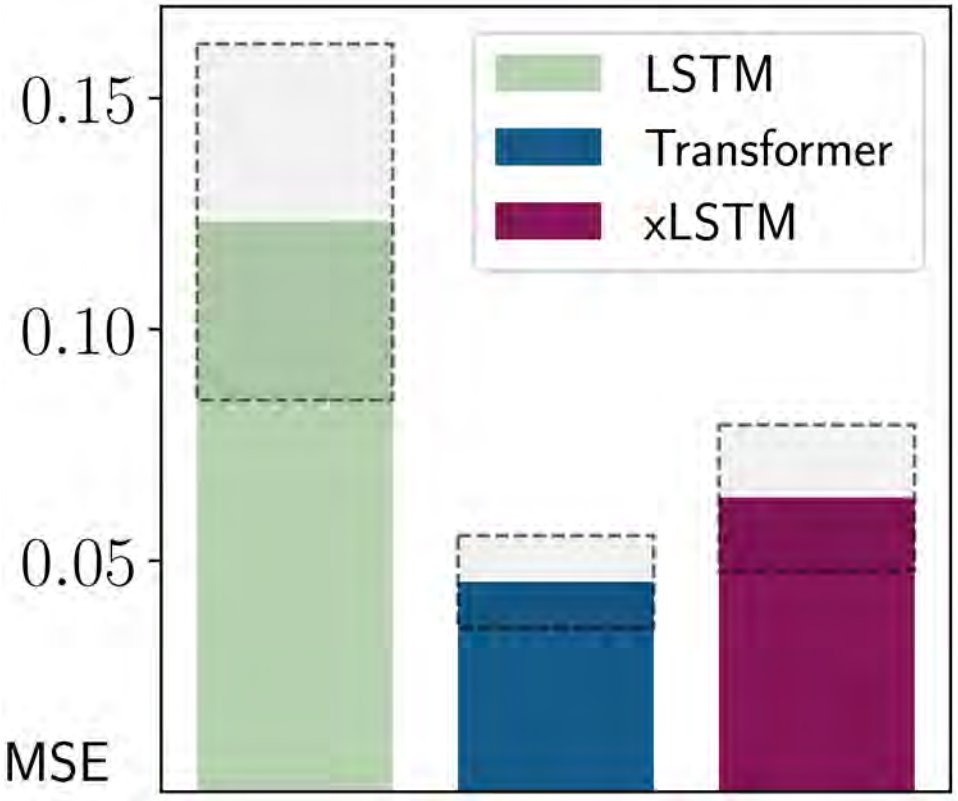


LSTM Limitations

Revise storage decisions

→ exponential gating

exponential input gate and normalization scales down old inputs



sLSTM

sLSTM (scalarLSTM)

$$
\begin{aligned}
c_t &= \mathrm{f}_t \, c_{t-1} + \mathrm{i}_t \, z_t \\
n_t &= \mathrm{f}_t \, n_{t-1} + \mathrm{i}_t \\
h_t &= \mathrm{o}_t \, \tilde{h}_t \,, \quad \tilde{h}_t = c_t / n_t \\
\mathrm{i}_t &= \exp\left(\tilde{\mathrm{i}}_t\right)
\end{aligned}
$$

classical LSTM

$$
\begin{aligned}
c_t &= \mathrm{f}_t \, c_{t-1} + \mathrm{i}_t \, z_t \\
h_t &= \mathrm{o}_t \, \tilde{h}_t \,, \quad \tilde{h}_t = \tanh\left(c_t\right) \\
\mathrm{i}_t &= \sigma\left(\tilde{\mathrm{i}}_t\right)
\end{aligned}
$$

Revise storage decisions:

$$
\mathrm{i}_t = \infty \,, \quad \tilde{h}_t = z_t
$$

LSTM Limitations

- Inability to **revise storage decisions**
  → Nearest Neighbor Search

exponential gating

- **Limited storage capacities**
  → Rare Token Prediction

- **No parallelization** for training

In mathematics and more specifically in topology, a **homeomorphism** (from Greek roots meaning "similar shape", named by Henri Poincaré),[2][3] also called **topological isomorphism**, or **bicontinuous function**, is a bijective and continuous function between topological spaces that has a continuous inverse function. Homeomorphisms are the isomorphisms in the category of topological spaces—that is, they are the mappings that preserve all the topological properties of a given space. Two spaces with a homeomorphism between them are called **homeomorphic**, and from a topological viewpoint they are the same.

Very roughly speaking, a topological space is a geometric object, and a homeomorphism results from a continuous deformation of the object into a new shape. Thus, a square and a circle are homeomorphic to each other, but a sphere and a torus are not. However, this description can be misleading. Some continuous deformations do not result into homeomorphisms such as the deformation of a line into a point. Some homeomorphisms do not result from continuous deformations, such as the homeomorphism between a trefoil knot and a circle. Homotopy and isotopy are precise definitions for the informal concept of *continuous deformation*.

An often-repeated mathematical joke is that topologists cannot tell the difference between a coffee mug and a donut;[1] since a sufficiently pliable donut could be reshaped to the form of a coffee mug by creating a dimple and progressively enlarging it, while preserving the donut hole in the mug's handle. This illustrates that a coffee mug and a donut (torus) are homeomorphic.

Definition [edit]

A function $f : X \to Y$ between two topological spaces is a **homeomorphism** if it has the following properties:

- f is a bijection (one-to-one and onto),
- f is continuous,
- the inverse function f^{-1} is continuous (f is an open mapping).

A homeomorphism is sometimes called a *bicontinuous* function. If such a function exists, X and Y are **homeomorphic**. A **self-homeomorphism** is a homeomorphism from a topological space onto itself. Being "homeomorphic" is an equivalence relation on topological spaces. Its equivalence classes are called **homeomorphism classes**.

JKU

Rare Token Prediction

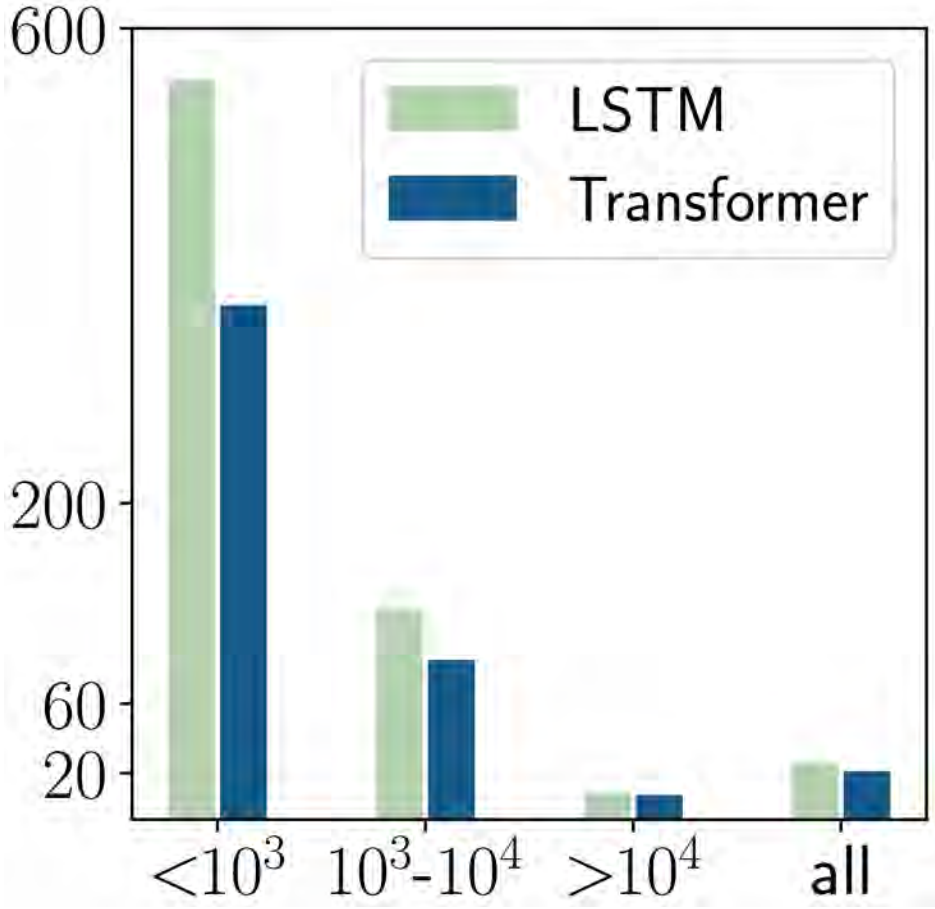
Wikitext 103

LSTM Limitations

Rare Token Prediction:

- perplexity (PPL) of token prediction on Wikitext 103
- partitions of token according to frequency

rare tokens must be memorized



LSTM Limitations

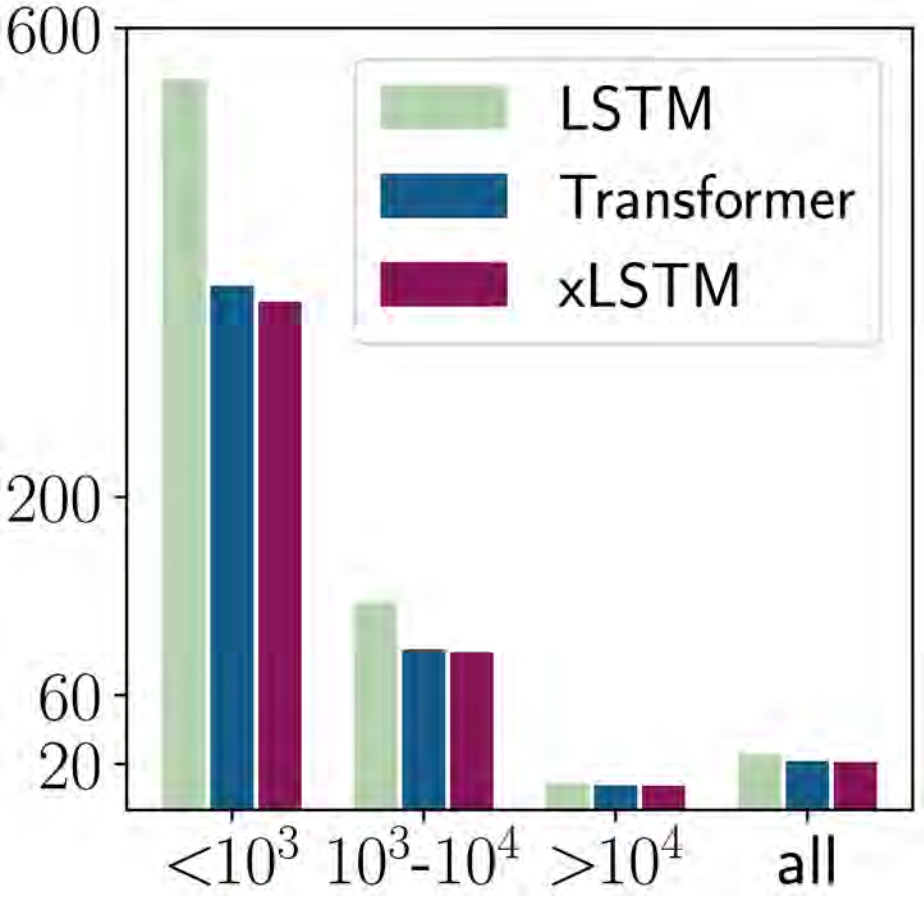
Limited storage capacities

New memory structure:

- matrix memory
- covariance update rule

mLSTM (matrixLSTM)

Results on Wikitext 103



mLSTM

mLSTM

$$
\begin{aligned}
\boldsymbol{C}_t &= \mathrm{f}_t\, \boldsymbol{C}_{t-1} + \mathrm{i}_t\, \boldsymbol{v}_t\, \boldsymbol{k}_t^\top \\
\boldsymbol{n}_t &= \mathrm{f}_t\, \boldsymbol{n}_{t-1} + \mathrm{i}_t\, \boldsymbol{k}_t \\
\boldsymbol{h}_t &= \mathrm{o}_t \odot \tilde{\boldsymbol{h}}_t \\
\tilde{\boldsymbol{h}}_t &= \boldsymbol{C}_t \boldsymbol{q}_t \;/\; \max\left\{\left|\boldsymbol{n}_t^\top \boldsymbol{q}_t\right|, 1\right\} \\
\mathrm{i}_t &= \exp\left(\tilde{\mathrm{i}}_t\right)
\end{aligned}
$$

classical LSTM

$$
\begin{aligned}
c_t &= \mathrm{f}_t\, c_{t-1} + \mathrm{i}_t\, z_t \\
h_t &= \mathrm{o}_t\, \tilde{h}_t\,, \qquad \tilde{h}_t = \tanh\left(c_t\right) \\
\mathrm{i}_t &= \sigma\left(\tilde{\mathrm{i}}_t\right)
\end{aligned}
$$

Classical Hopfield network with Hebb rule and Gates

Parallelization: gates, keys, values and queries are all independent of $\boldsymbol{h}_{t-1}$

LSTM Limitations

- Inability to **revise storage decisions**
  → Nearest Neighbor Search

exponential gating

- **Limited storage capacities**
  → Rare Token Prediction

matrix memory & covariance update

- **No parallelization** for training

update is memory independent

xLSTM

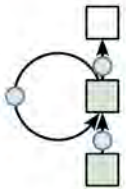
LSTM

Memory Cells

- → Constant Error Carousel
- → Sigmoid Gating
- → Recurrent Inference
- → Recurrent Training

$$c_t = f_t\, c_{t-1} + i_t\, z_t$$

$$h_t = o_t\, \psi(c_t)$$



Memory Cells

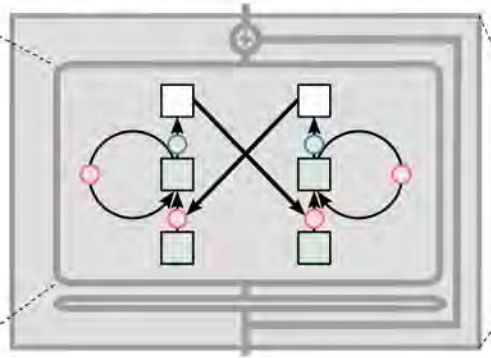
sLSTM

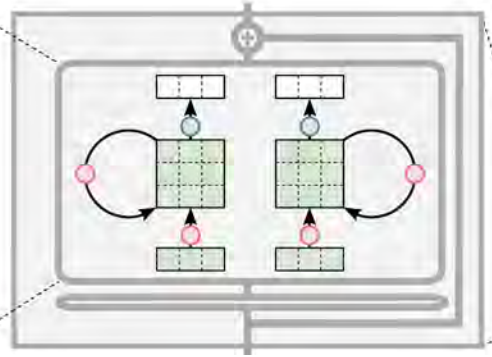
+ Exponential Gating
+ New Memory Mixing

mLSTM

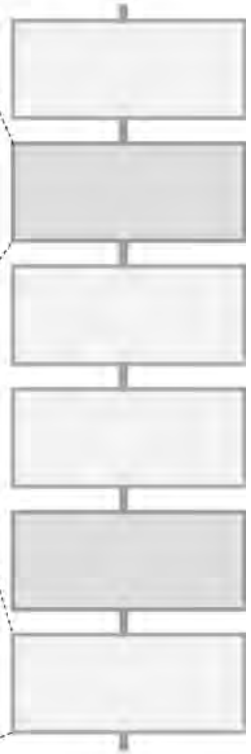
+ Exponential Gating
+ Matrix Memory
+ Parallel Training
+ Covariance Update Rule

xLSTM Blocks

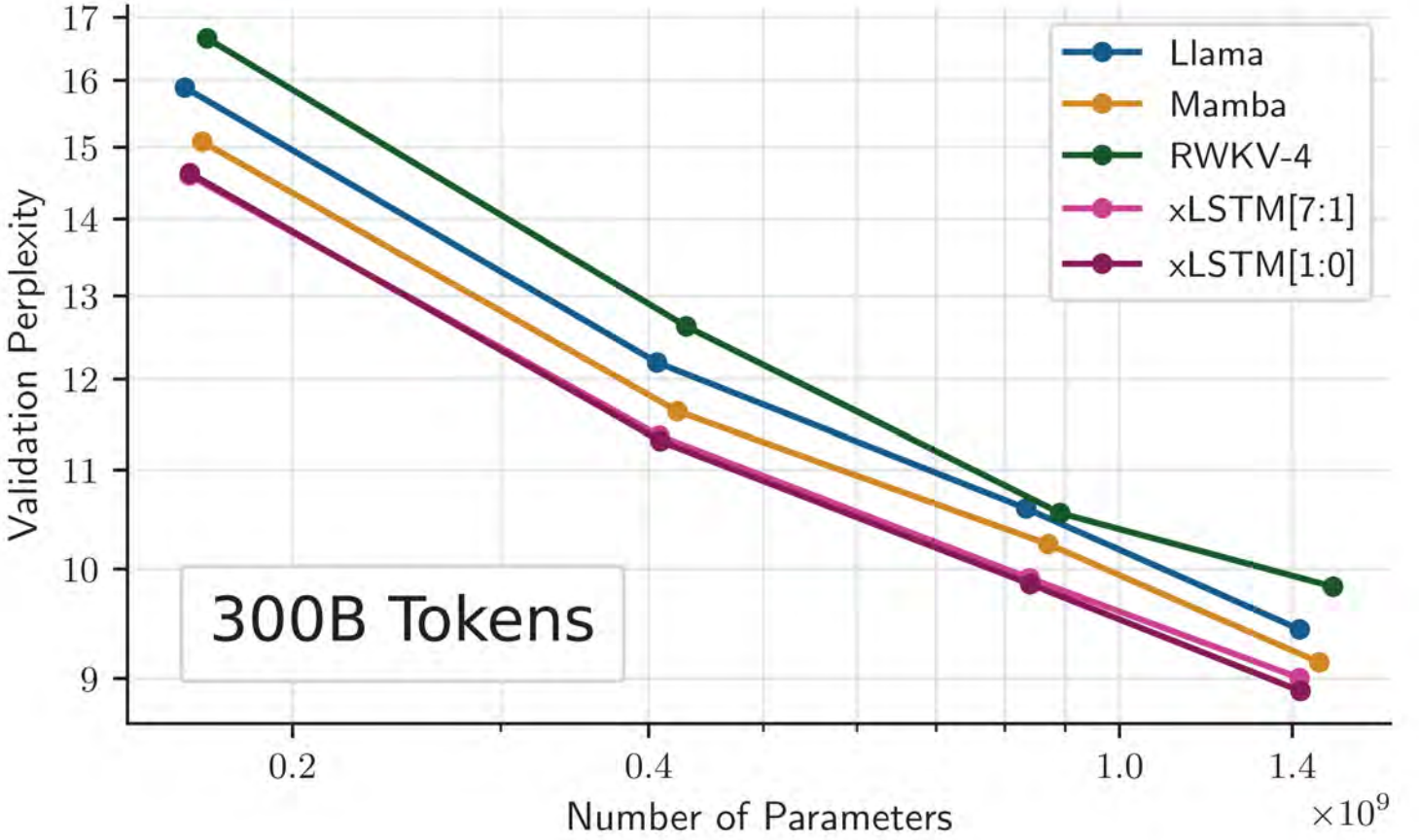




xLSTM



xLSTM Scaling Laws



Trained on 256 A100 GPUs on 32 nodes.

FSDP parallelism.

xLSTM

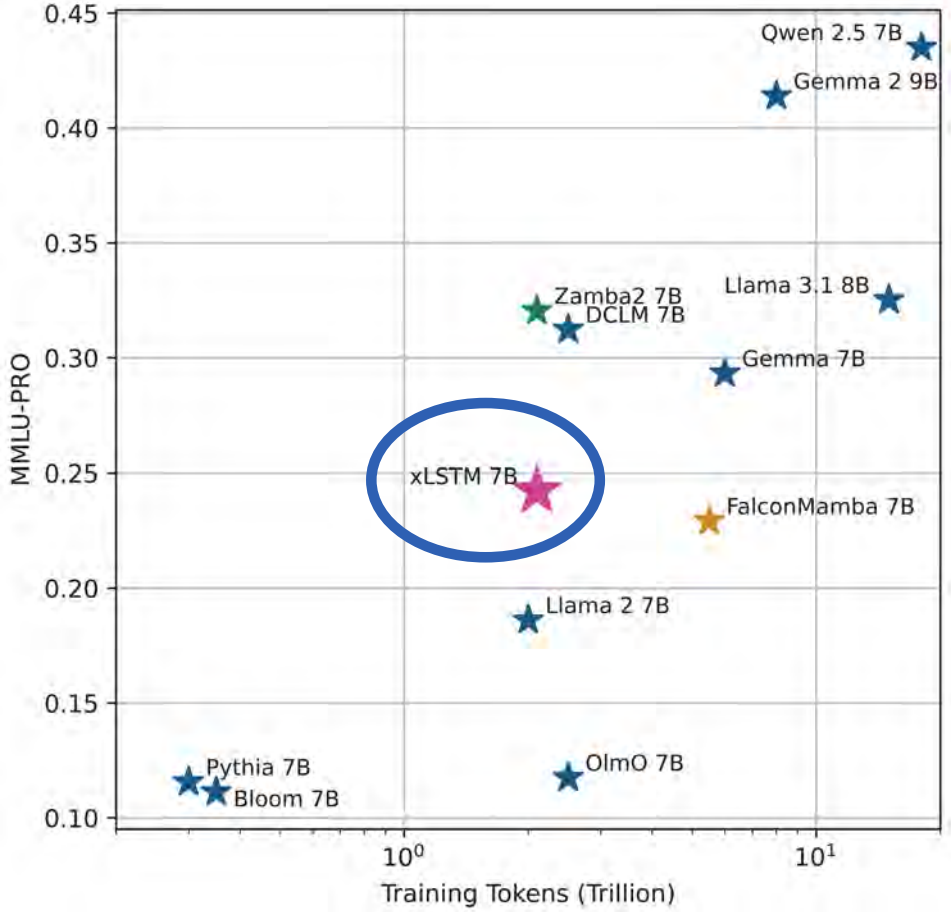
7B model downstream MMLU-PRO.

Performance vs. trained tokens.

Best alternative to attention.

Trained on 256 H100 GPUs with 32 nodes and 8 GPUs per node.

Implemented in JAX. FSDP parallelism.

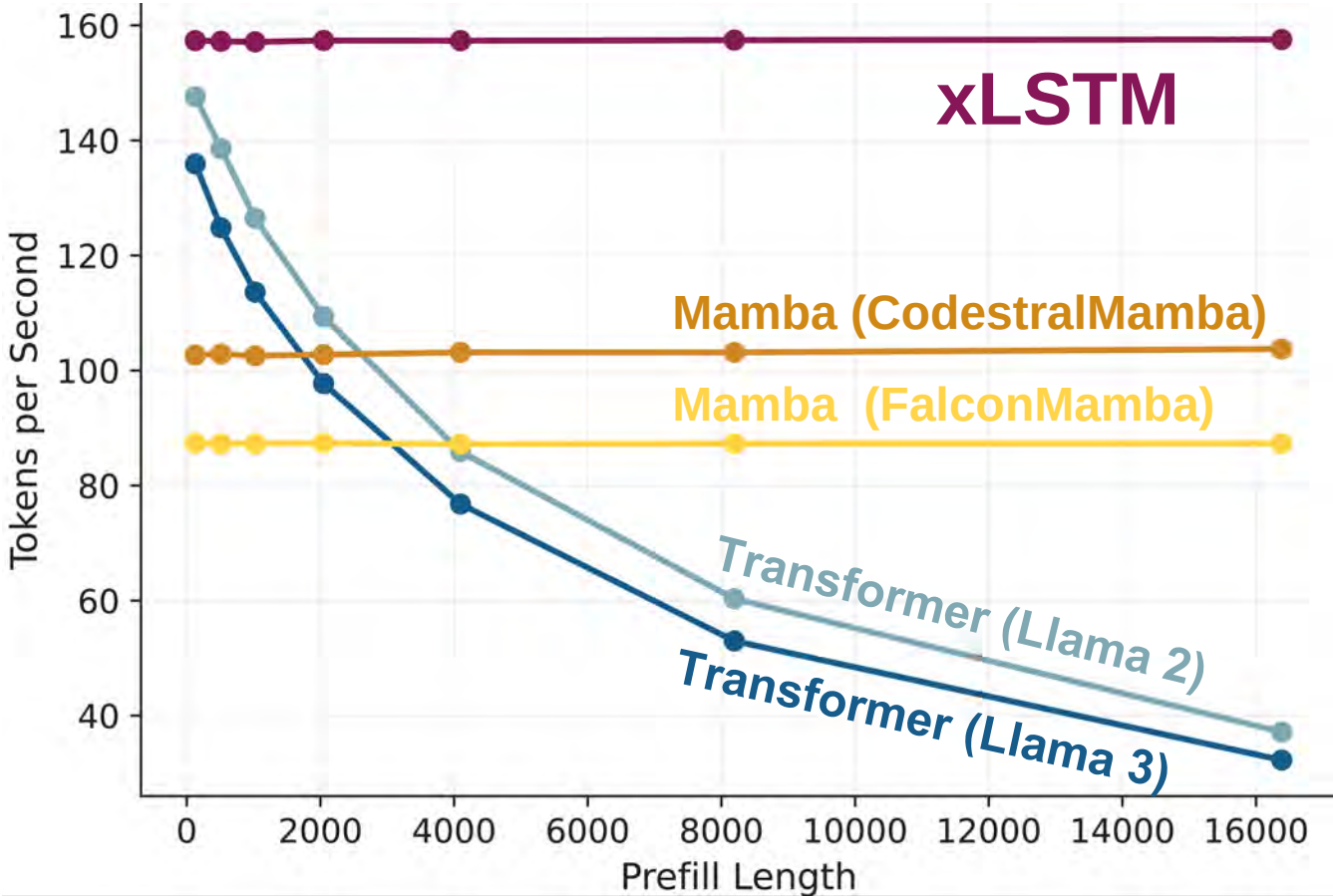


xLSTM: Inference

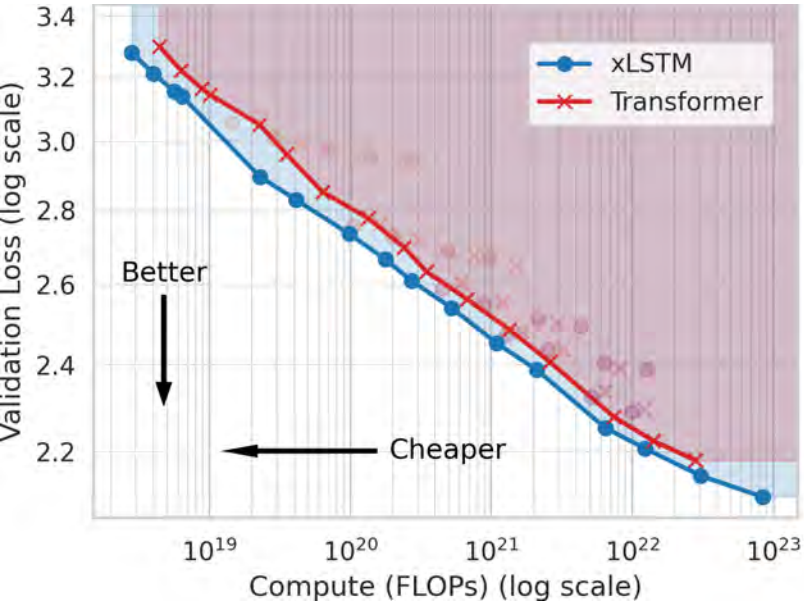
7B model: speed comparison.

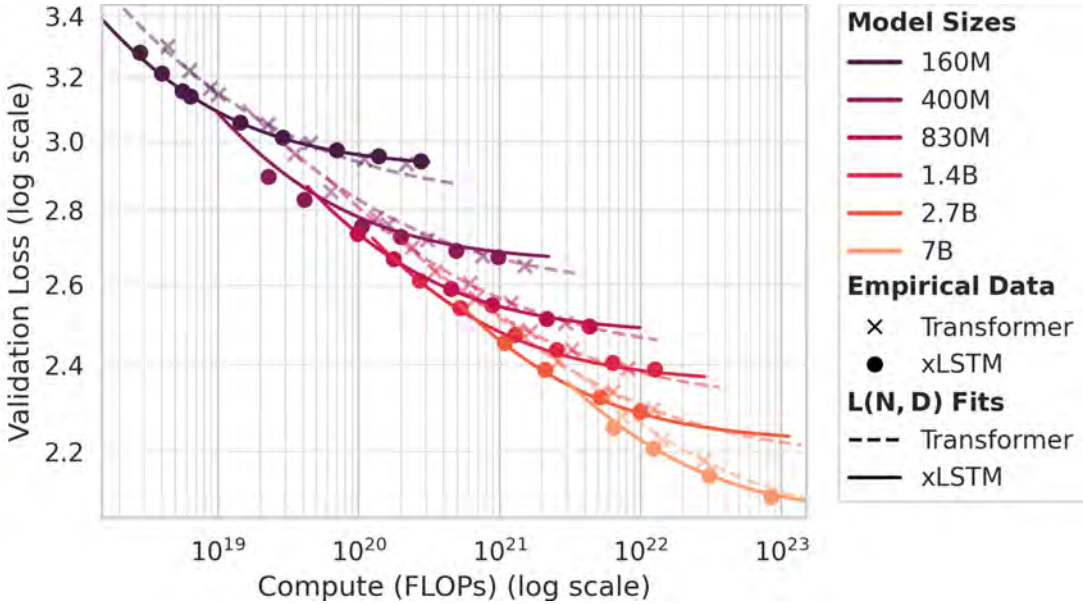
Different context sizes via prefill.

xLSTM is fastest model.



xLSTM Scaling Laws Training



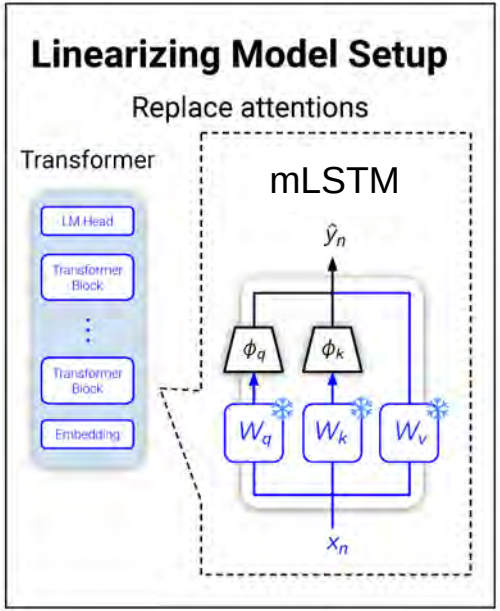


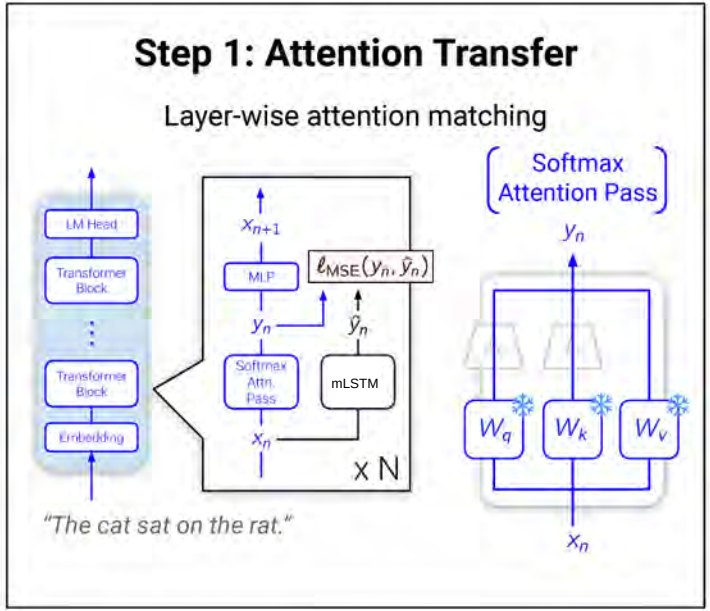
Validation loss over training compute.

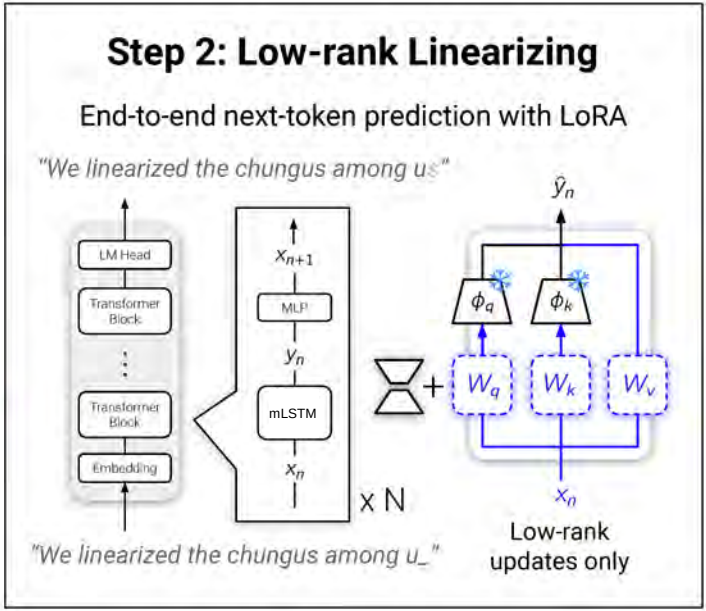
xLSTM is pareto-dominant over dense multi-head Transformers in terms of loss:

- fixed FLOP budget ⟶ xLSTM models are better.
- fixed validation loss ⟶ xLSTM models require less FLOPs.

xLSTM: Distillation







xLSTM

xLSTM is used in industry:

- **Festo uses xLSTM**
- **Spleenlab uses xLSTM for vision**

NXAI

JKU

xLSTM for vision in drones excels in depth estimation.

SPLEENLAB

xLSTM

xLSTM

- much faster than Transformer at inference,
- constant memory,
- faster than Transformer at training,
- better than Transformer for given compute,
- **has high potential for industrial AI**

(embedded applications: self-driving, robotics).

xLSTM

Model	Recurrence	Memory read-out
Linear Attention [14, 13]	$\boldsymbol{S}_t = \boldsymbol{S}_{t-1} + \boldsymbol{v}_t \boldsymbol{k}_t^\intercal$	$\boldsymbol{o}_t = \boldsymbol{S}_t \boldsymbol{q}_t$
+ Kernel	$\boldsymbol{S}_t = \boldsymbol{S}_{t-1} + \boldsymbol{v}_t \phi(\boldsymbol{k}_t)^\intercal$	$\boldsymbol{o}_t = \boldsymbol{S}_t \phi(\boldsymbol{q}_t)$
+ Normalization	$\boldsymbol{S}_t = \boldsymbol{S}_{t-1} + \boldsymbol{v}_t \phi(\boldsymbol{k}_t)^\intercal, \quad \boldsymbol{z}_t = \boldsymbol{z}_{t-1} + \phi(\boldsymbol{k}_t)$	$\boldsymbol{o}_t = \boldsymbol{S}_t \phi(\boldsymbol{q}_t) / (\boldsymbol{z}_t^\intercal \phi(\boldsymbol{q}_t))$
DeltaNet [24]	$\boldsymbol{S}_t = \boldsymbol{S}_{t-1}(\boldsymbol{I} - \beta_t \boldsymbol{k}_t \boldsymbol{k}_t^\intercal) + \beta_t \boldsymbol{v}_t \boldsymbol{k}_t^\intercal$	$\boldsymbol{o}_t = \boldsymbol{S}_t \boldsymbol{q}_t$
Gated RFA [20]	$\boldsymbol{S}_t = g_t \boldsymbol{S}_{t-1} + (1 - g_t) \boldsymbol{v}_t \boldsymbol{k}_t^\intercal, \quad \boldsymbol{z}_t = g_t \boldsymbol{z}_{t-1} + (1 - g_t) \boldsymbol{k}_t$	$\boldsymbol{o}_t = \boldsymbol{S}_t \boldsymbol{q}_t / (\boldsymbol{z}_t^\intercal \boldsymbol{q}_t)$
S4 [9, 26]	$\boldsymbol{S}_t = \boldsymbol{S}_{t-1} \odot \exp(-(\boldsymbol{\alpha} \boldsymbol{1}^\intercal) \odot \exp(\boldsymbol{A})) + \boldsymbol{B} \odot (\boldsymbol{v}_t \boldsymbol{1}^\intercal)$	$\boldsymbol{o}_t = (\boldsymbol{S}_t \odot \boldsymbol{C}) \boldsymbol{1} + \boldsymbol{d} \odot \boldsymbol{v}_t$
H3 [7]	$\boldsymbol{S}_t = \boldsymbol{S}_{t-1} \odot \boldsymbol{A} + \boldsymbol{B} \odot \boldsymbol{v}_t \boldsymbol{k}_t^\intercal$	$\boldsymbol{o}_t = \boldsymbol{S}_t \boldsymbol{q}_t + \boldsymbol{d} \odot \boldsymbol{v}_t$
ABC [21]	$\boldsymbol{S}_t^k = \boldsymbol{S}_{t-1}^k + \boldsymbol{k}_t \boldsymbol{\phi}_t^\intercal, \quad \boldsymbol{S}_t^v = \boldsymbol{S}_{t-1}^v + \boldsymbol{v}_t \boldsymbol{\phi}_t^\intercal$	$\boldsymbol{o}_t = \boldsymbol{S}_t^v \mathrm{softmax}\left(\boldsymbol{S}_t^k \boldsymbol{q}_t\right)$
DFW [16]	$\boldsymbol{S}_t = \boldsymbol{S}_{t-1} \odot (\boldsymbol{\beta}_t \boldsymbol{\alpha}_t^\intercal) + \boldsymbol{v}_t \boldsymbol{k}_t^\intercal$	$\boldsymbol{o}_t = \boldsymbol{S}_t \boldsymbol{q}_t$
RetNet [27]	$\boldsymbol{S}_t = \gamma \boldsymbol{S}_{t-1} + \boldsymbol{v}_t \boldsymbol{k}_t^\intercal$	$\boldsymbol{o}_t = \boldsymbol{S}_t \boldsymbol{q}_t$
Mamba [8]	$\boldsymbol{S}_t = \boldsymbol{S}_{t-1} \odot \exp(-(\boldsymbol{\alpha}_t \boldsymbol{1}^\intercal) \odot \exp(\boldsymbol{A})) + (\boldsymbol{\alpha}_t \odot \boldsymbol{v}_t) \boldsymbol{k}_t^\intercal$	$\boldsymbol{o}_t = \boldsymbol{S}_t \boldsymbol{q}_t + \boldsymbol{d} \odot \boldsymbol{v}_t$
GLA [28]	$\boldsymbol{S}_t = \boldsymbol{S}_{t-1} \odot (\boldsymbol{1} \boldsymbol{\alpha}_t^\intercal) + \boldsymbol{v}_t \boldsymbol{k}_t^\intercal = \boldsymbol{S}_{t-1} \mathrm{diag}(\boldsymbol{\alpha}_t) + \boldsymbol{v}_t \boldsymbol{k}_t^\intercal$	$\boldsymbol{o}_t = \boldsymbol{S}_t \boldsymbol{q}_t$
RWKV-6 [19]	$\boldsymbol{S}_t = \boldsymbol{S}_{t-1} \mathrm{diag}(\boldsymbol{\alpha}_t) + \boldsymbol{v}_t \boldsymbol{k}_t^\intercal$	$\boldsymbol{o}_t = (\boldsymbol{S}_{t-1} + (\boldsymbol{d} \odot \boldsymbol{v}_t) \boldsymbol{k}_t^\intercal) \boldsymbol{q}_t$
HGRN-2 [23]	$\boldsymbol{S}_t = \boldsymbol{S}_{t-1} \mathrm{diag}(\boldsymbol{\alpha}_t) + \boldsymbol{v}_t (1 - \boldsymbol{\alpha}_t)^\intercal$	$\boldsymbol{o}_t = \boldsymbol{S}_t \boldsymbol{q}_t$
mLSTM [2]	$\boldsymbol{S}_t = f_t \boldsymbol{S}_{t-1} + i_t \boldsymbol{v}_t \boldsymbol{k}_t^\intercal, \quad \boldsymbol{z}_t = f_t \boldsymbol{z}_{t-1} + i_t \boldsymbol{k}_t$	$\boldsymbol{o}_t = \boldsymbol{S}_t \boldsymbol{q}_t / \max\{1, \lvert \boldsymbol{z}_t^\intercal \boldsymbol{q}_t \rvert\}$
Mamba-2 [6]	$\boldsymbol{S}_t = \gamma_t \boldsymbol{S}_{t-1} + \boldsymbol{v}_t \boldsymbol{k}_t^\intercal$	$\boldsymbol{o}_t = \boldsymbol{S}_t \boldsymbol{q}_t$
Longhorn [15]	$\boldsymbol{S}_t = \boldsymbol{S}_{t-1} \odot \left(\boldsymbol{I} - \frac{\boldsymbol{\beta}_t}{1 + \boldsymbol{k}_t^\intercal \boldsymbol{k}_t \boldsymbol{\beta}_t} (\boldsymbol{k} \odot \boldsymbol{k})^\intercal\right) + \left(\frac{\boldsymbol{\beta}_t}{1 + \boldsymbol{k}_t^\intercal \boldsymbol{k}_t \boldsymbol{\beta}_t} \odot \boldsymbol{x}_t\right) \boldsymbol{k}_t$	$\boldsymbol{o}_t = \boldsymbol{S}_t \boldsymbol{q}_t$
GSA [30]	$\boldsymbol{S}_t^k = \boldsymbol{S}_{t-1}^k \mathrm{diag}(\boldsymbol{\alpha}_t) + \boldsymbol{k}_t \boldsymbol{\phi}_t^\intercal, \quad \boldsymbol{S}_t^v = \boldsymbol{S}_{t-1}^v \mathrm{diag}(\boldsymbol{\alpha}_t) + \boldsymbol{v}_t \boldsymbol{\phi}_t^\intercal$	$\boldsymbol{o}_t = \boldsymbol{S}_t^v \mathrm{softmax}\left(\boldsymbol{S}_t^k \boldsymbol{q}_t\right)$
MetaLA [5]	$\boldsymbol{S}_t = \boldsymbol{S}_{t-1} \mathrm{diag}(\boldsymbol{\alpha}_t) + \boldsymbol{v}_t (1 - \boldsymbol{\alpha}_t)^\intercal$	$\boldsymbol{o}_t = (\boldsymbol{S}_{t-1} + (\boldsymbol{d} \odot \boldsymbol{v}_t) \boldsymbol{k}_t^\intercal) \boldsymbol{q}_t$
Gated DeltaNet [29]	$\boldsymbol{S}_t = \boldsymbol{S}_{t-1} \left(\alpha_t (\boldsymbol{I} - \beta_t \boldsymbol{k}_t \boldsymbol{k}_t^\intercal)\right) + \beta_t \boldsymbol{v}_t \boldsymbol{k}_t^\intercal$	$\boldsymbol{o}_t = \boldsymbol{S}_t \boldsymbol{q}_t$

TiRex

TiRex

Time series foundation model:

- pre-trained on all available time series
- zero-shot learning (no learning)
- in context learning (learning at data in context)
- past values as context (prompt)
- predicts future values

TiRex

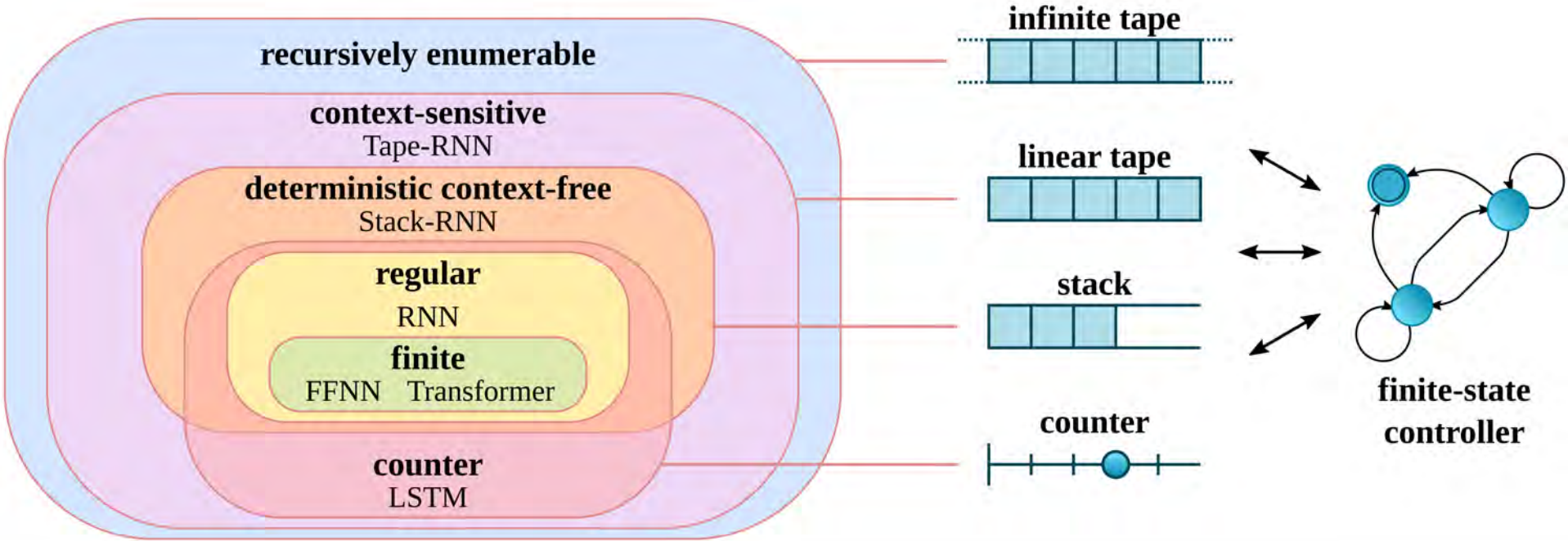
We used **xLSTM** to build a time-series foundation model: **TiRex**

TiRex can do state tracking in contrast to other models.

TiRex: More Powerful

xLSTM (sLSTM) is able to do state tracking and counting via recurrent matrix $\boldsymbol{R}$.

Chomsky hierarchy from "Neural Networks and the Chomsky Hierarchy", G. Delétang, 2022.

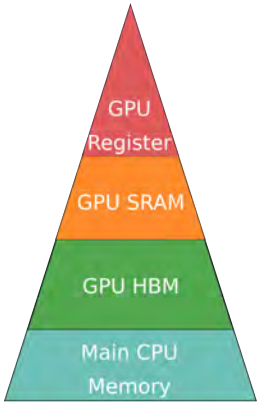


TiRex: Fast

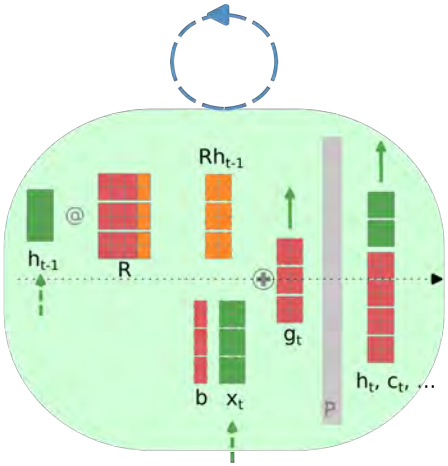
sLSTM is fast

- Caching of recurrent matrix R in registers
- Synchronization across multiple Streaming Multiprocessors needed (unlike attention)

Basic Memory Hierarchy in modern GPUs.

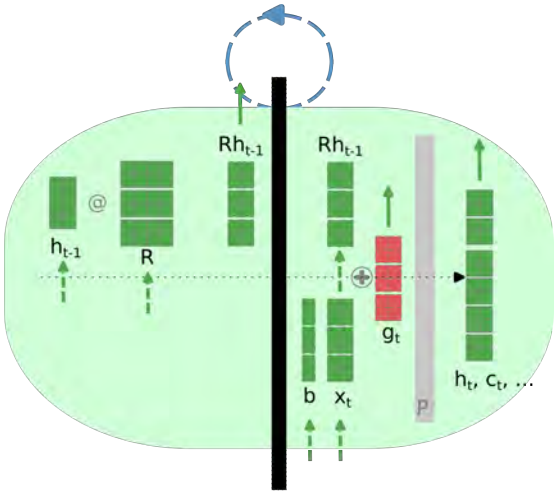


Fused Kernel (forward) leveraging all caching options for maximal speed.

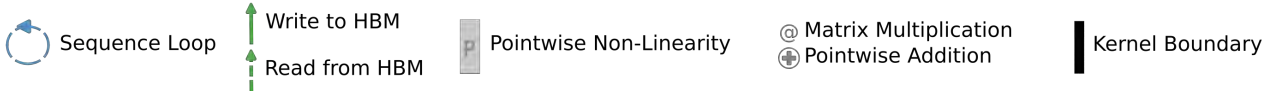


Fused

Alternating Kernels (forward) for maximum hidden sizes, with two kernel calls per time step.



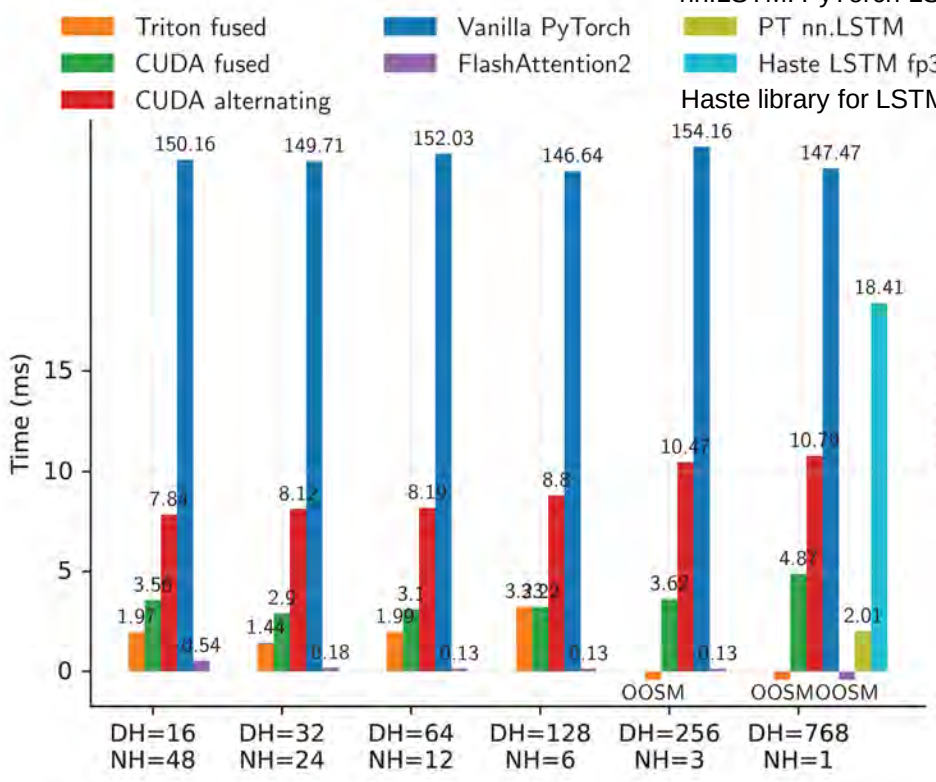
Alternating



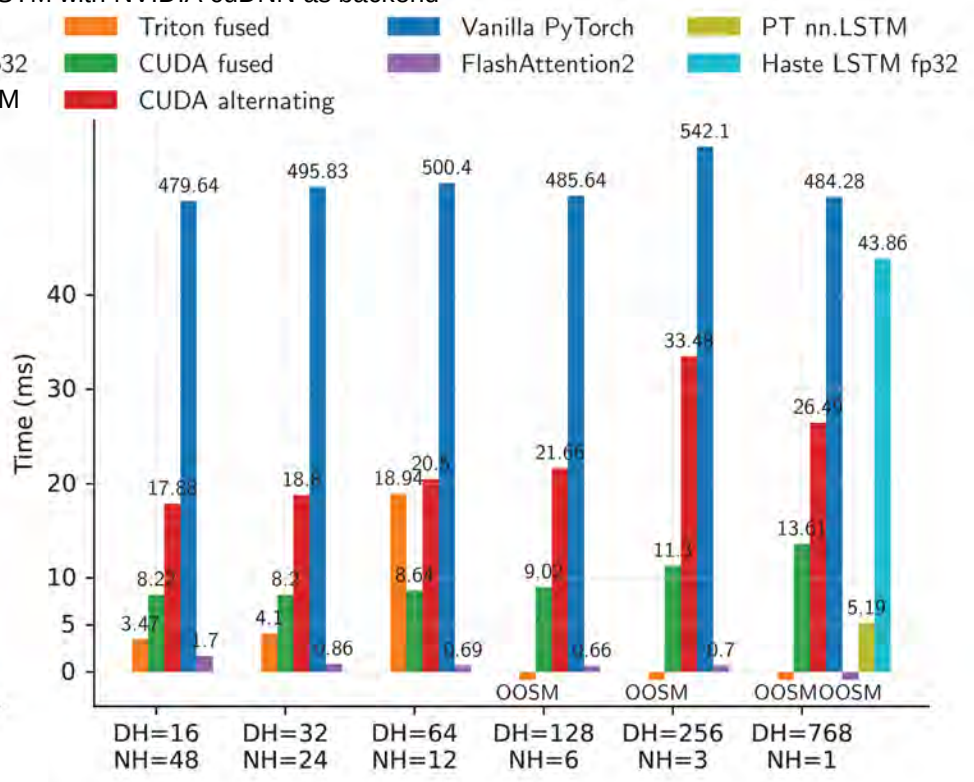
TiRex: Fast and low Memory

nn.LSTM: PyTorch LSTM with NVIDIA cuDNN as backend

Haste library for LSTM



Forward pass



Forward + backward pass

TiRex: Heading Leaderboards

There are two benchmark datasets for time-series foundation models with leaderboard on huggingface:

1. Chronos Benchmark II or Chronos ZS
   https://huggingface.co/spaces/autogluon/fev-leaderboard

2. GIFT-Eval Time Series Forecasting
   https://huggingface.co/spaces/Salesforce/GIFT-Eval

Hugging Face

TiRex is leading both

TiRex: Heading Leaderboards

Huggingface Chronos Benchmark II probabilistic forecast (WQL)

	Instit.	Model	error	rank	time	overlap
	NXAI	**TiRex**	**0.599**	**4.333**	**0.212**	**0**
	Amazon	chronos-bolt-base	0.624	6.778	0.406	0
	Salesforce	moirai-1.1-R-large	0.631	7.926	14.254	81.5
	Amazon	chronos-bolt-small	0.636	7.926	0.393	0
	Salesforce	moirai-1.1-R-base	0.637	8.333	6.712	81.5
	Amazon	chronos-t5-base	0.643	10.037	7.765	0
	Amazon	chronos-bolt-mini	0.644	8.37	0.386	0
	Amazon	chronos-t5-small	0.654	11.519	2.593	0
	Amazon	chronos-t5-large	0.656	9.222	26.827	0
	Amazon	chronos-t5-mini	0.665	11.926	2.023	0
	Amazon	chronos-bolt-tiny	0.668	9.667	0.385	0
	Google	timesfm-2.0-500m	0.7	7.815	2.253	14.8
	Amazon	chronos-t5-tiny	0.703	13.037	2.069	0
	Salesforce	moirai-1.1-R-small	0.703	11.778	3.779	81.5
	Google	timesfm-1.0-200m	0.708	10.148	0.396	14.8
	baseline	auto_arima	0.741	11.704	75.884	0
	baseline	auto_theta	0.793	11.556	23.892	0
	baseline	auto_ets	0.814	11.037	3.508	0
	baseline	seasonal_naive	1	18	0.096	0
	IBM-granite	granite-timeseries-ttm-r2	1.127	18.889	0.016	0

Huggingface Chronos Benchmark II point forecast (MASE)

	Instit.	Model	error	rank	time	over
	NXAI	**TiRex**	**0.78**	**4.74**	**0.21**	**0**
	Google	timesfm-2.0-500m-py	0.789	7.111	2.253	14.8
	Salesforce	moirai-1.1-R-large	0.791	7.148	14.254	81.5
	Amazon	chronos-bolt-base	0.795	7.185	0.406	0
	Amazon	chronos-t5-base	0.818	9.519	7.765	0
	Salesforce	moirai-1.1-R-base	0.819	8.852	6.712	81.5
	Amazon	chronos-bolt-small	0.823	9.852	0.393	0
	Amazon	chronos-t5-large	0.824	9.296	26.827	0
	Amazon	chronos-bolt-mini	0.827	10.67	0.386	0
	Amazon	chronos-t5-small	0.837	11.19	2.593	0
	Amazon	chronos-t5-mini	0.842	12.59	2.023	0
	Amazon	chronos-bolt-tiny	0.849	12.22	0.385	0
	baseline	auto_theta	0.859	9.259	23.892	0
	baseline	auto_arima	0.869	9.741	75.884	0
	Amazon	chronos-t5-tiny	0.874	13.3	2.069	0
	Google	timesfm-1.0-200m	0.882	11.41	0.396	14.8
	Salesforce	moirai-1.1-R-small	0.894	12.93	3.779	81.5
	baseline	auto_ets	0.943	9.296	3.508	0
	baseline	seasonal_naive	1	16	0.096	0
	IBM-granite	granite-timeseries-ttr	1.12	17.7	0.016	0

TiRex: Heading Leaderboards

Huggingface Chronos Benchmark II probabilistic forecast (WQL)

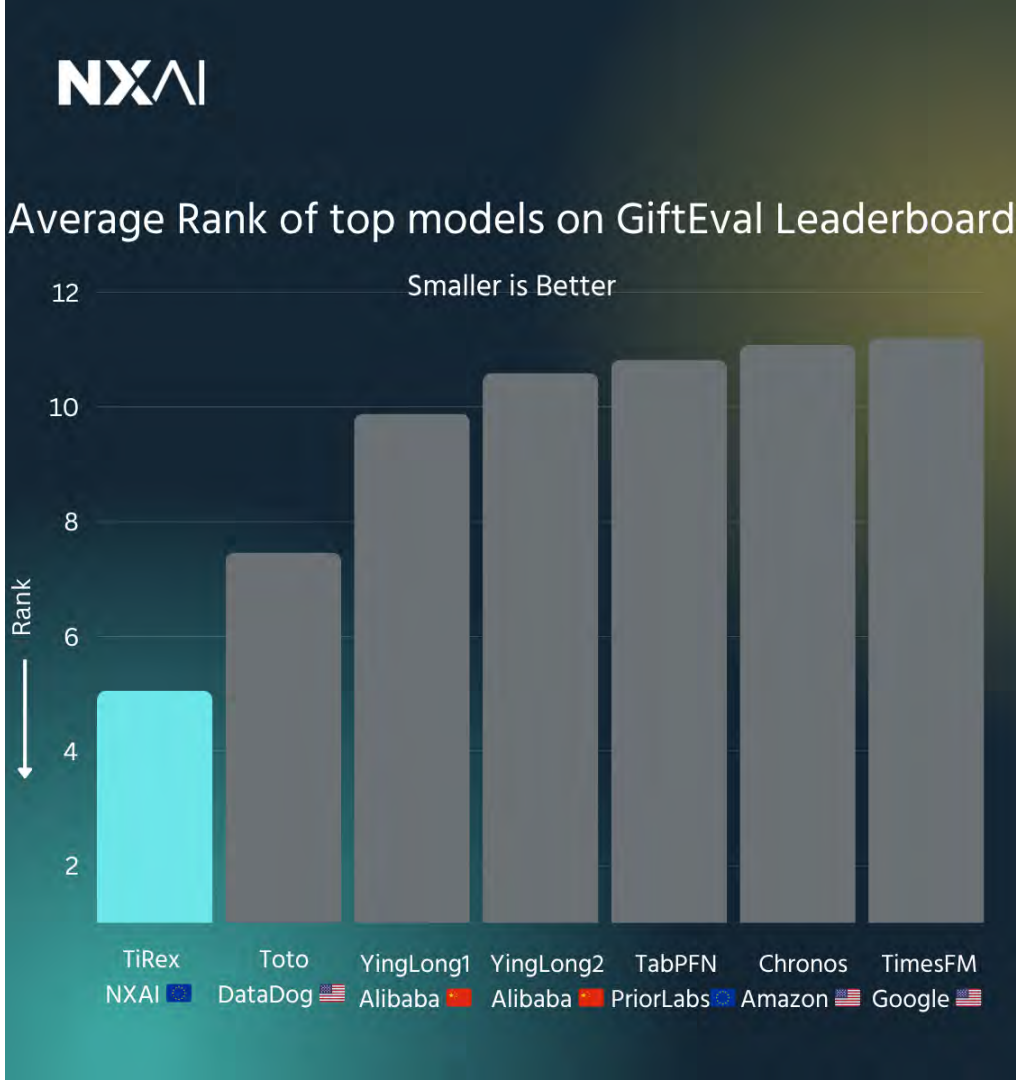
	Instit.	Model	error	rank	time	overlap
	NXAI	**TiRex**	**0.599**	**4.333**	**0.212**	**0**
	Amazon	chronos-bolt-base	0.624	6.778	0.406	0
	Salesforce	moirai-1.1-R-large	0.631	7.926	14.254	81.5
	Amazon	chronos-bolt-small	0.636	7.926	0.393	0
	Salesforce	moirai-1.1-R-base	0.637	8.333	6.712	81.5
	Amazon	chronos-t5-base	0.643	10.037	7.765	0
	Amazon	chronos-bolt-mini	0.644	8.37	0.386	0
	Amazon	chronos-t5-small	0.654	11.519	2.593	0
	Amazon	chronos-t5-large	0.656	9.222	26.827	0
	Amazon	chronos-t5-mini	0.665	11.926	2.023	0
	Amazon	chronos-bolt-tiny	0.668	9.667	0.385	0
	Google	timesfm-2.0-500m	0.7	7.815	2.253	14.8
	Amazon	chronos-t5-tiny	0.703	13.037	2.069	0
	Salesforce	moirai-1.1-R-small	0.703	11.778	3.779	81.5
	Google	timesfm-1.0-200m	0.708	10.148	0.396	14.8
	baseline	auto_arima	0.741	11.704	75.884	0
	baseline	auto_theta	0.793	11.556	23.892	0
	baseline	auto_ets	0.814	11.037	3.508	0
	baseline	seasonal_naive	1	18	0.096	0
	IBM-granite	granite-timeseries-ttm-r2	1.127	18.889	0.016	0

Huggingface Chronos Benchmark II point forecast (MASE)

	Instit.	Model	error	rank	time	over
	NXAI	**TiRex**	**0.78**	**4.74**	**0.21**	**0**
	Google	timesfm-2.0-500m-py	0.789	7.111	2.253	14.8
	Salesforce	moirai-1.1-R-large	0.791	7.148	14.254	81.5
	Amazon	chronos-bolt-base	0.795	7.185	0.406	0
	Amazon	chronos-t5-base	0.818	9.519	7.765	0
	Salesforce	moirai-1.1-R-base	0.819	8.852	6.712	81.5
	Amazon	chronos-bolt-small	0.823	9.852	0.393	0
	Amazon	chronos-t5-large	0.824	9.296	26.827	0
	Amazon	chronos-bolt-mini	0.827	10.67	0.386	0
	Amazon	chronos-t5-small	0.837	11.19	2.593	0
	Amazon	chronos-t5-mini	0.842	12.59	2.023	0
	Amazon	chronos-bolt-tiny	0.849	12.22	0.385	0
	baseline	auto_theta	0.859	9.259	23.892	0
	baseline	auto_arima	0.869	9.741	75.884	0
	Amazon	chronos-t5-tiny	0.874	13.3	2.069	0
	Google	timesfm-1.0-200m	0.882	11.41	0.396	14.8
	Salesforce	moirai-1.1-R-small	0.894	12.93	3.779	81.5
	baseline	auto_ets	0.943	9.296	3.508	0
	baseline	seasonal_naive	1	16	0.096	0
	IBM-granite	granite-timeseries-ttr	1.12	17.7	0.016	0

Huggingface GIFT-Eval Time Series Forecasting

Institution	Model	MASE	CRPS	Rank
NXAI	TiRex	0.65	0.42	5.155
Datadog	Toto_Open_Base_1.	0.673	0.437	7.577
Alibaba	YingLong_300m	0.716	0.463	10.113
Alibaba	YingLong_110m	0.726	0.471	10.897
PriorLabs	TabPFN-TS	0.692	0.46	11.144
Amazon	chronos_bolt_base_	0.725	0.485	11.371
Google	timesfm_2_0_500m	0.68	0.465	11.526
USC/Google	TEMPO_ensemble	0.773	0.434	11.711
Alibaba	YingLong_50m	0.738	0.479	11.866
Amazon	chronos_bolt_small_	0.738	0.487	12.423
Tsinghua U	sundial_base_128m	0.673	0.472	13.062
IBM-granite	TTM-R2-Finetuned_(	0.679	0.492	13.691
Salesforce	Moirai_large_(code)	0.785	0.506	14.021
Salesforce	Moirai_base_(code)	0.809	0.515	14.062
Princeton/I	PatchTST	0.762	0.496	14.247
Oxford/Goo	TFT	0.822	0.511	15.66
Alibaba	YingLong_6m	0.79	0.514	16.309
Salesforce	Moirai_small_(code	0.849	0.549	17.948
Amazon	Chronos_large_(cod	0.781	0.547	18.784
Amazon	Chronos_base_(cod	0.786	0.551	18.876
Google	TimesFM	0.967	0.575	18.887
Amazon	Chronos_small_(cod	0.8	0.56	19.979
Google/UCS	TIDE	0.98	0.652	23.928
Amazon Ge	DeepAR	1.206	0.721	24.072
Zhejiang,St	VisionTS	0.775	0.638	26.082
Shanghai -	Crossformer	2.31	1.383	26.237
Baseline	Auto_Arima	0.964	0.77	26.979
Element AI	N-BEATS	0.842	0.689	27.062
Morgan Sta	Lag-Llama	1.102	0.744	27.464
IBM-granite	TTM-R2-Zeroshot	0.915	0.738	28.804
U Hong Kor	DLinear	0.952	0.714	28.99
IBM-granite	TTM-R1-Zeroshot	0.969	0.753	29.289
Baseline	Auto_Theta	0.978	1.051	29.392
Baseline	Auto_ETS	1.088	6.327	30.443
Tsinghua U	Timer	1.019	0.82	31.32
Baseline	Seasonal_Naive	1	1	32.01
Baseline	Naive	1.26	1.383	33.928



TiRex: Heading Leaderboards

Francis Rafal • 2nd
AI Engineer & CEO @ Totoy | Tailored AI for your business | G...
1mo •

Connect

TiRex Demo:

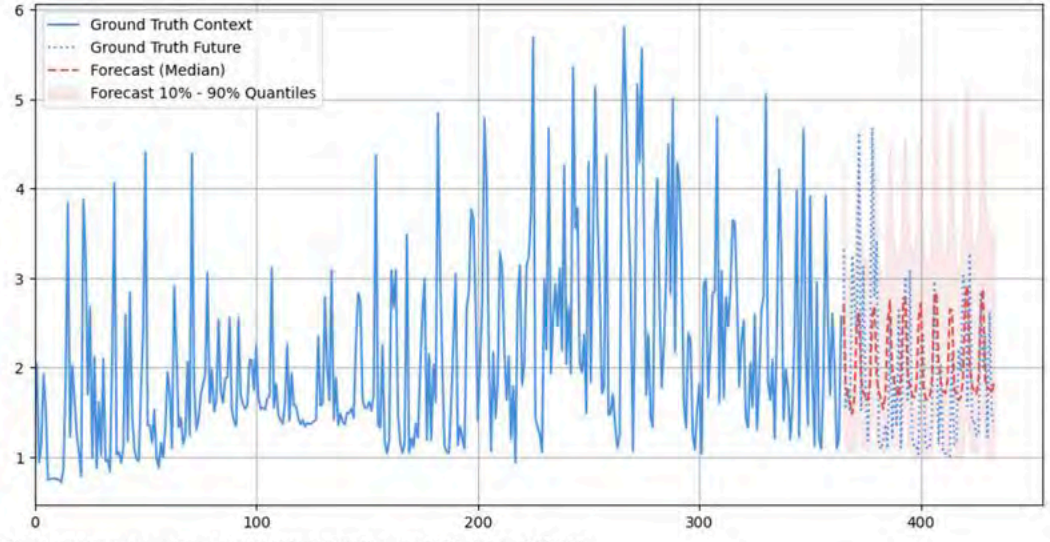
Saturday Experiment: Predicting Personal Electricity Consumption with the New TiRex Model from Austrian AI Developer NXAI

My test setup:
1. Download electricity consumption data from the Wiener Netze GmbH Smart Meter Portal (I have data from 434 days there)
2. Use the first 365 days as context
3. TiRex is supposed to predict the remaining 69 days (without knowing the real values)

Result:
1. TiRex Forecast: 139kWh
2. Actual consumption: 133 kWh
Pretty close! 💡
And all this without specific model training (= Zero-Shot Forecasting) AN
source!
Good job Sepp Hochreiter + team!

Here are a few ideas for what Time Series AI models could be used for in
- Financial forecasts
-Production planning
- Maintenance cycles for machines
- Personnel requirements planning
- Predict server utilization

Google Colab to try out for yourself in the comments.



xLSTM

- much faster than Transformer at inference,
- constant memory,
- faster than Transformer at training,
- better than Transformer for given compute,
- **has high potential for industrial AI**

(embedded applications: self-driving, robotics).

xLSTM

xLSTM landing page

Team: NXAI

From left to right:

Miks Mikelsons
Alex Pierer
Dennis Just
Albert Ortig
Benedikt Alkin
Felix Neusser
Sepp Hochreiter
Korbinian Pöppel
Sebastian Böck
Johannes Brandstetter
Richard Kurle
Matthias Dorfer
Tobias Kronlachner
Maximilian Beck
Maurits Bleeker
Patrick Blies
Thomas Meneder
Thomas Lichtenegger

END