Kalman Linear Attention: Parallel Bayesian Filters for Efficient Language Modelling and State Tracking

"Towards probabilistic LLMs - one Bayesian layer at a time"

Vaisakh Shaj, Cameron Barker, Aidan Scannell, Andras Szecsenyi, Elliot J. Crowley, Amos Storkey

THE UNIVERSITY of EDINBURGH

ICML International Conference On Machine Learning

MOTIVATION

Can we replace **SSM primitives** with an **uncertainty-aware Kalman filter (KF)** - at **no extra compute cost**?

Kalman updates are recursive and nonlinear
→ **exact parallel scans, O(log T) depth**

Bayesian primitives don't fit LLM pipelines
→ **KLA: a drop-in KF-based SSM block**

*Interpretable? State tracking? Zero-shot common-sense reasoning? → **RESULTS***

NOTATIONS: KLA vs MAMBA

$\mathbf{z}_t$:	latent state at time t
$\mathbf{o}_t$:	observed token / features at time t
$\mathbf{A}_t$:	state transition matrix
$\mathbf{B}_t$:	control / input matrix (Mamba)
$\mathbf{C}_t$:	observation (emission) matrix
$\mathbf{Q}_t$:	process noise covariance
Σ_t^{obs} :	observation noise covariance
(μ_t, Σ_t) :	belief over z_t (prior $-$ / posterior $+$)

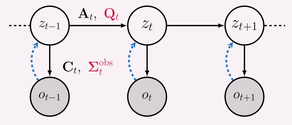
Red = Parameters **not** in Mamba SSM

Generative Model: If we knew the hidden state, the solid arrows show how z_t stochastically evolves and emits observations o_t e.g. $z_{t+1} \mid z_t \sim \mathcal{N}(\mathbf{A}_t z_t, \mathbf{Q}_t)$ and $o_t \mid z_t \sim \mathcal{N}(\mathbf{C}_t z_t, \boldsymbol{\Sigma}_t^{\text{obs}})$

Inference Model / Filter Update: in practice we only see the o_t 's; the blue dashed arrows indicate running this story in reverse to keep a Gaussian belief (prior/ posterior) over the state, given observations $o_{1:t}$.

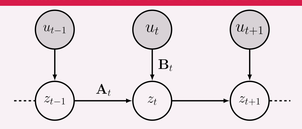
Two philosophies on reading tokens

KLA · TOKENS ARE NOISY OBSERVATIONS



*tokens are **noisy glimpses** of a hidden state - the filter **(blue arrows)** tracks a **guess + its confidence** (μ, Σ)*

MAMBA · TOKENS ARE CONTROL INPUTS



*tokens are **knobs** that push the state around - one point estimate, **no notion of doubt***

TAKE A PHOTO OF THIS

- Bayesian filters are **secretly attention maps**
- **Non-linear recurrence** (more expressive) yet GPU-friendly and parallelizable
- Gating **for free from Bayes' rule**
- **~350x faster** training with CUDA scans
- **Uncertainty neurons** - a path to interpretable, safer language models

Paper + code: QR at bottom right.

	Softmax Attention	SSMs / GLA	KLA
Expressivity	Nonlinear	Linear	Fractional lin. (Möbius)
Training eff.	$\mathcal{O}(T^2)$	$\mathcal{O}(T)$	$\mathcal{O}(T)$
Inference eff.	$\mathcal{O}(T)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$
Seq. uncertainty	✗	✗	✓
Parallel training	✓	✓	✓

KLA - KALMAN LINEAR ATTENTION

Kalman Filter Parallelization

Theorem 1 (Informal): Kalman updates in information form are fractional linear (Möbius) transforms.

Mamba / SSM Update

$$z_t = a_t z_{t-1} + b_t$$

Simple **affine** map in 1D (a_t, b_t from a neural net).

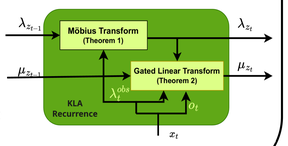
Kalman Attention Update

$$z_t = \frac{a_t z_{t-1} + b_t}{c_t z_{t-1} + d_t}$$

Fractional linear map in 1D (aka Möbius transform of z_{t-1}).

Corollary 1: Kalman updates are **associative** and hence can be implemented as log(T) depth **parallel scans**.

✓ Non-linear
✓ Associative
✓ Parallelizable



`torch._higher_order_ops.associative_scan()`

Gates from Bayes' Rule

Mean (μ) Recursion: a gated RNN-like update

$$\mu_t = \underbrace{\bar{a}_t\left(1 - \frac{k_t^2\,\Lambda_t^v}{\lambda_t}\right)}_{\textit{forget gate}} \odot \mu_{t-1} + \underbrace{\frac{k_t\,\Lambda_t^v}{\lambda_t}}_{\textit{input gate}} v_t$$

KLA GATE

$$f_t = \bar{a}_t\left(1 - \frac{k_t^2\,\Lambda_t^v}{\lambda_t}\right)$$

✓ from Bayes' rule - uncertainty ratios
✓ history dependent

vs

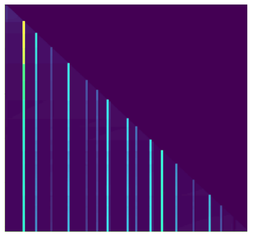
MAMBA GATE

$$f_t = 1 - \sigma(\text{Linear}(o_t))$$

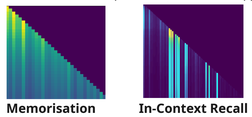
✗ ad-hoc parameterisation
✗ sees the current token only

Gating isn't a heuristic - it's Bayes' rule doing the bookkeeping.

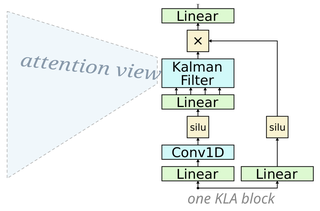
KLA Block: Secretly Attention



a learned attention pattern (Selective Copy)



Memorisation **In-Context Recall**



Scaffolding (same as Mamba)
- 1D-Conv
- SiLU
- State Expansions

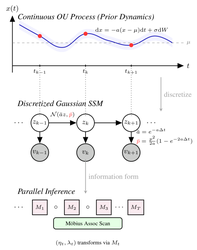
Dynamics: OU prior - stable, decaying memory.

KLA Sequence Transformation Matrix

$$\mathbf{W} = \begin{pmatrix} k_1\Lambda_1^v & 0 & \cdots & 0 \\ f_2\,k_1\Lambda_1^v & k_2\Lambda_2^v & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \prod_{s=2}^{T} f_s\,k_1\Lambda_1^v & \prod_{s=3}^{T} f_s\,k_2\Lambda_2^v & \cdots & k_T\Lambda_T^v \end{pmatrix}$$

red = forget gates · blue = input uncertainty terms

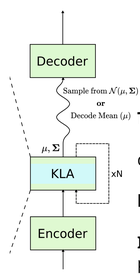
From OU Dynamics to Parallel Inference



ODE prior (Mamba) → SDE prior (KLA)

the mean-reverting OU prior keeps dynamics bounded - stack 12-24 KLA blocks with no gradient explosions

Language Modelling with KLA Blocks

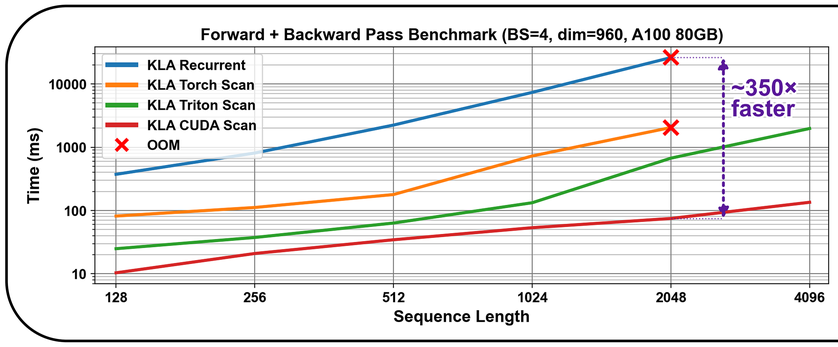


Training and Loss:
Cross Entropy Loss (mean decoding)
Marginal negative log likelihood (if sampling)

Inference: For now masked parallel decoding.

EMPIRICAL RESULTS

Probabilistic Inference at Long Context: Parallel Scans + Fused Kernels = 350× Speedup



Math (parallel scan **reparameterisation)**

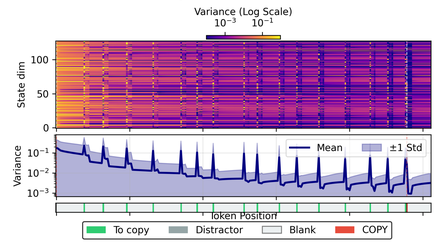
+

Hardware Awareness (fused SRAM-resident kernel)

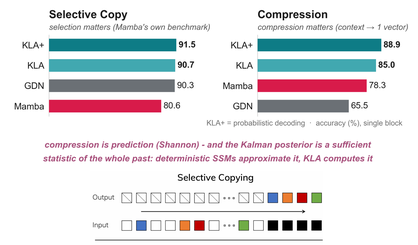
Each buys an order of magnitude faster training, matching the scan-parallel profile of modern SSMs at scale.

Why should I use a Bayesian Filter For Language Modelling ?

Interpretable In-Context Learning Through Explicit Uncertainty Neurons



MAD-Lab Synthetic LM (single block benchmarking)



Non-linear + Parallel Recurrence = State Tracking Models That Scale

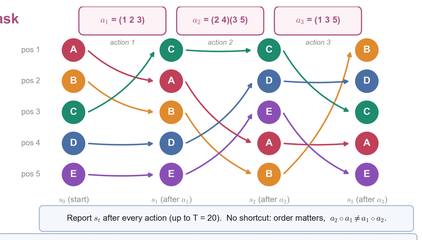
The A_5 state-tracking task

A minimal world model: 5 objects in 5 slots; each token is an action that shuffles them. Report the state s_t after every action.

Provably sequential. A_5 is NC^1-complete: no parallel shortcut.

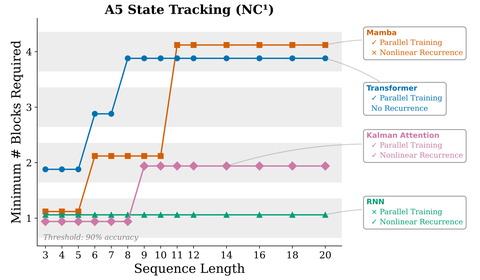
Linear recurrences fail. SSMs and attention need depth growing in T.

KLA: nonlinear yet associative - constant depth, parallel training.



Report s_t after every action (up to T = 20). No shortcut: order matters, $a_2 \circ a_1 \neq a_1 \circ a_2$.

Same problem, in the wild: **Chess:** where is every piece after a move sequence? **Code:** x[1], x[3] = x[3], x[1] - what is x now?
Stories: Alice and Carl trade coins - who holds what? ***All are permutation composition in disguise.*** (Merrill et al., 2024)



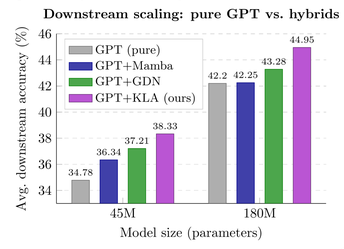
Add One KLA Layer to Your GPT: Better Downstream Accuracy Across the Board

Standalone: a competitive probabilistic LLM

First-generation primitive already matches Mamba/GDN on average - recall (LAMBADA) lags, the known fixed-state trade-off.

Model	LAMB. acc ↑	HellaS. acc_n ↑	PIQA acc ↑	Arc-E acc ↑	Arc-C acc_n ↑	WinoGr. acc ↑	OBQA acc ↑	BoolQ acc ↑	Average acc ↑
GPT-45M	**16.00**	28.60	58.00	40.20	20.60	50.00	**27.40**	37.40	34.78
Mamba-45M	9.20	**29.20**	**59.80**	37.20	**25.40**	48.80	25.40	61.60	37.08
GDN-45M	**16.00**	28.20	58.40	39.00	23.40	**52.60**	27.20	45.20	36.25
KLA-45M	7.60	28.80	58.60	40.60	23.20	52.20	**27.40**	**63.00**	37.68
GPT+KLA-45M	**16.00**	28.20	58.60	**42.80**	23.00	50.80	24.40	62.80	**38.33**
GPT-180M	**28.80**	32.20	63.40	51.00	27.00	49.20	31.20	54.80	42.20
Mamba-180M	24.00	30.60	63.00	51.00	27.20	**53.80**	**33.00**	55.80	42.30
GDN-180M	27.80	31.60	60.40	45.60	25.80	53.40	30.00	60.80	41.93
KLA-180M	22.40	34.00	63.00	50.00	26.00	53.60	32.40	60.80	42.78
GPT+KLA-180M	**30.20**	**34.40**	**64.20**	**55.80**	**29.00**	51.80	31.00	**63.20**	**44.95**

GPT + 1 KLA Layer: Keeps Recall. Improves Zero-Shot Reasoning.



FIRST ITERATION - THE HEADROOM: **Current:** diagonal LTI dynamics; mean-only propagation between layers.
Untapped: posterior covariance for OOD / hallucination detection, full-posterior stacking, continuous modalities.

References