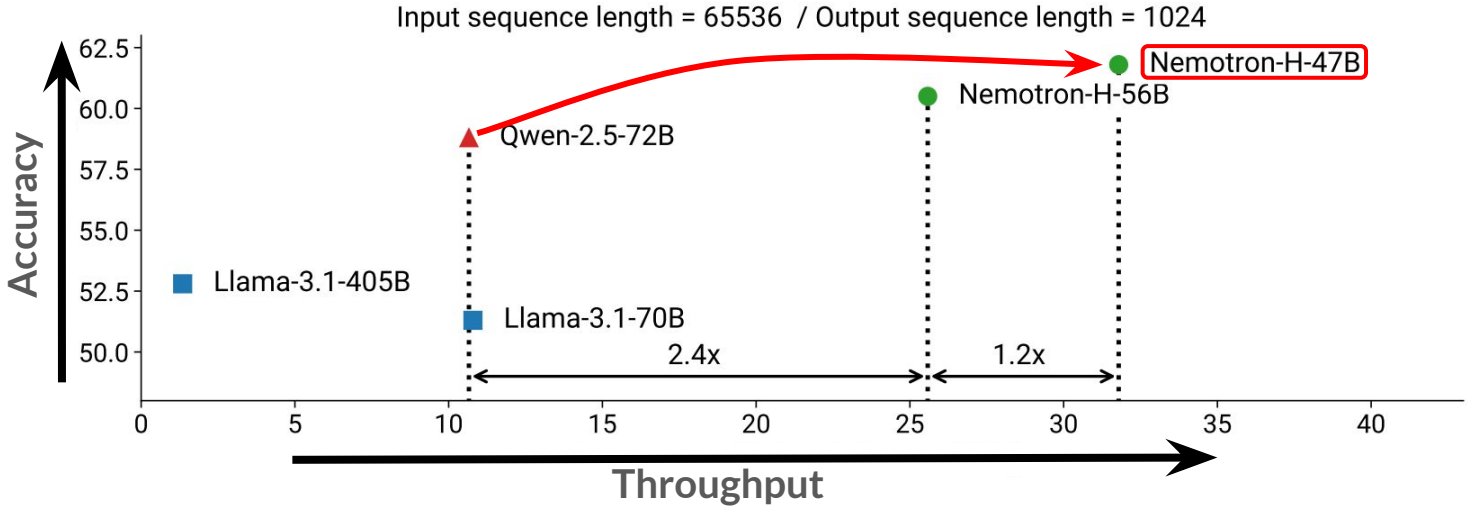
Efficiency-Expressivity Tradeoffs for Hybrid Models

*John Cooper**, Ilias Diakonikolas, Mingchen Ma*, Frederic Sala

WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON

Motivation - Large Scale Hybrid Models

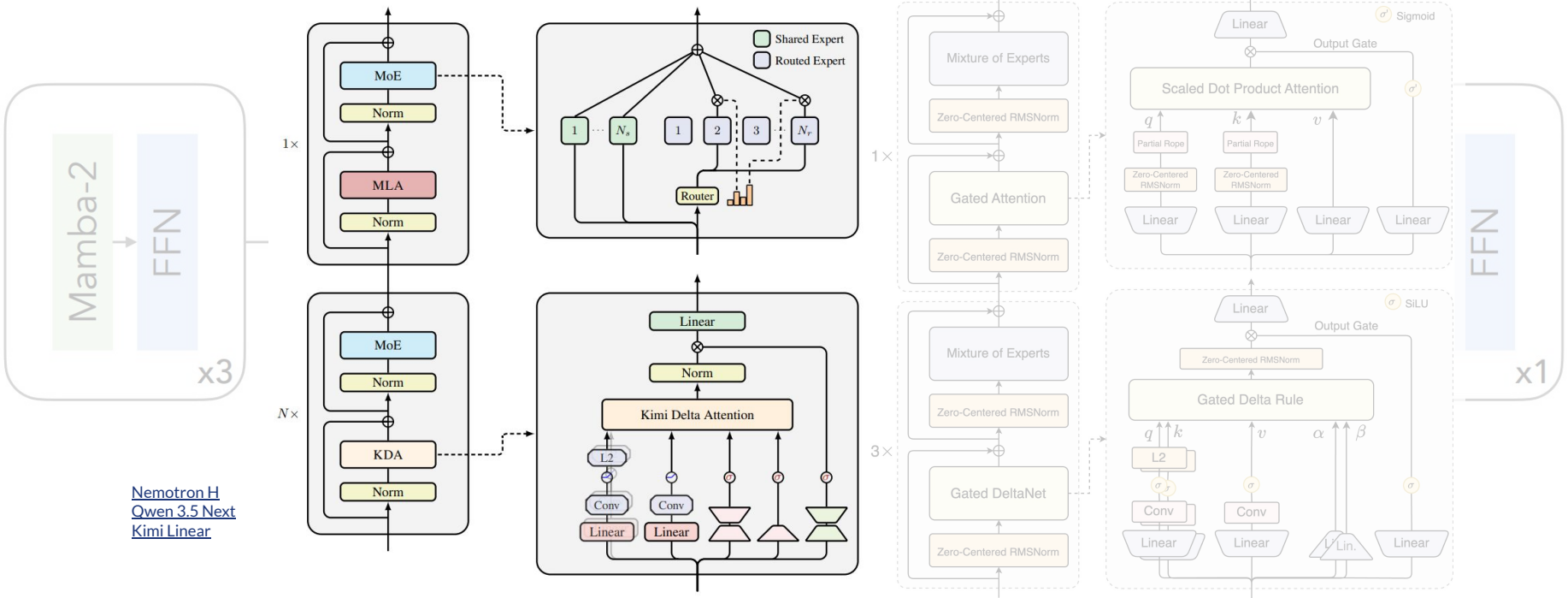


*What is causing **hybrid models** to be more efficient **and** more expressive?*

[Nemotron-H]
[Kimi-Linear]

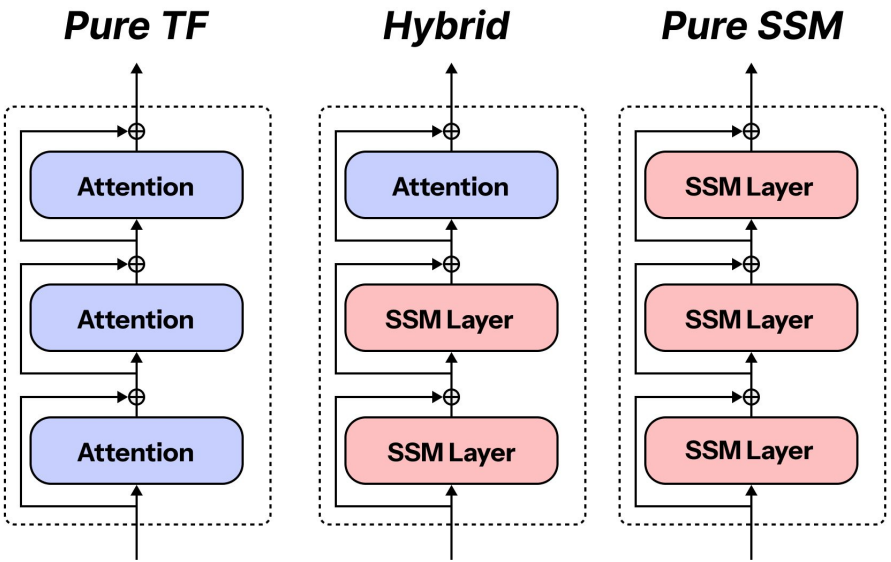
Motivation - Large Scale Hybrid Models

Many hybrids are being trained at scale with **various architectures**.



[Nemotron H](#)
[Qwen 3.5 Next](#)
[Kimi Linear](#)

Motivation - What is a hybrid?

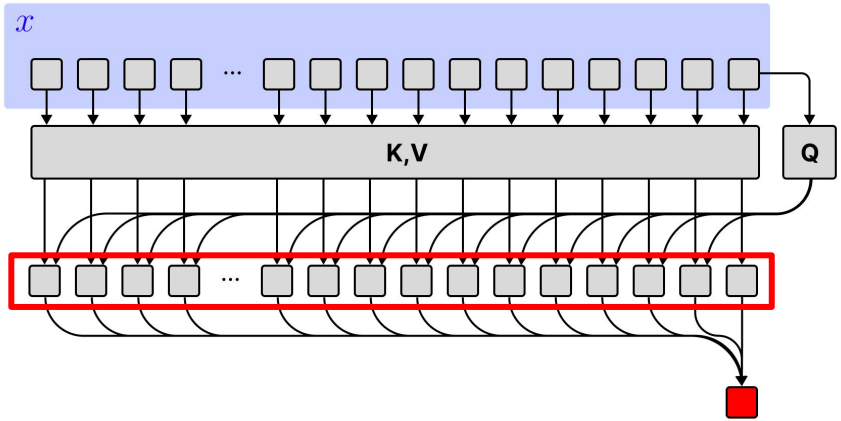


They combine SSM and Attention layers.

Attention gives **expressivity**.

SSMs buy **efficiency**.

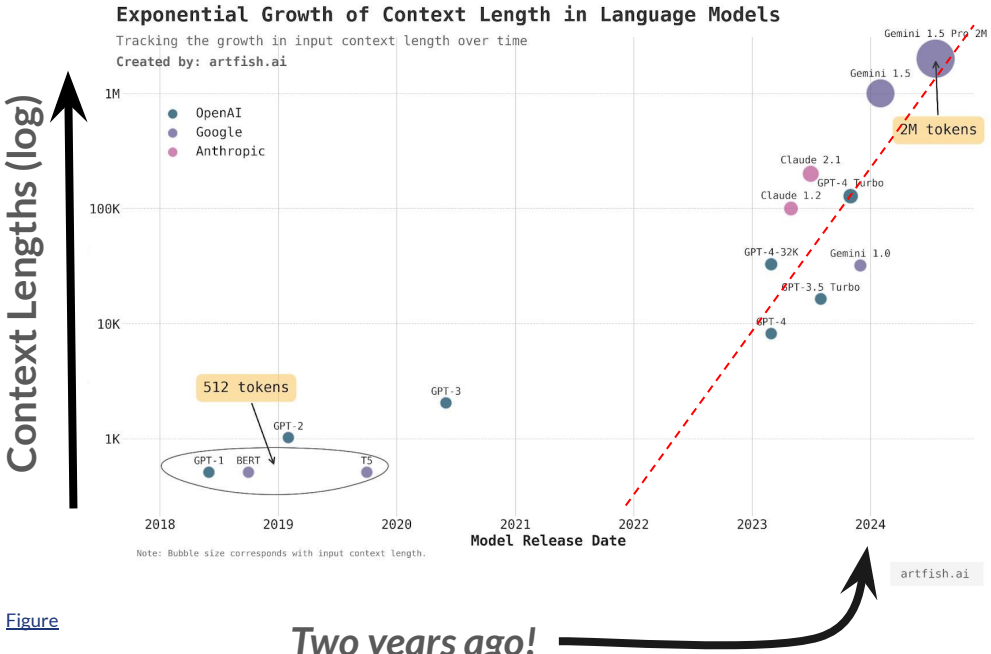
Background - Attention and its Drawbacks



Backbone of most modern LLMs.
Very **expressive.** ↑

The **output** requires **linear memory** for KV.
Not **efficient.** ↓

Background - Large Scale Attention Models



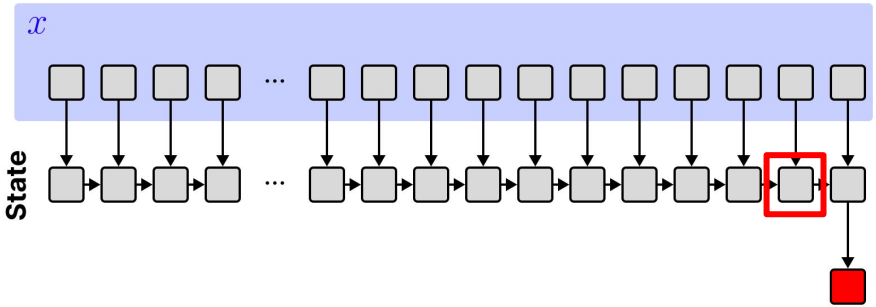
Figure

Two years ago!

Recent models have very large **context lengths**.

Attention is **inefficient**.

Background - SSMs and their Drawbacks



Needs only **constant** memory. **More efficiency**!

In practice, these models tend to be **less expressive** in many domains.

Main Question

*When does combining **SSMs** with **Attention** achieve capabilities that neither have on their own?*

Main Question

*When does combining **SSMs** with **Attention** achieve capabilities that neither have on their own?*

What tasks are best for hybrid models?

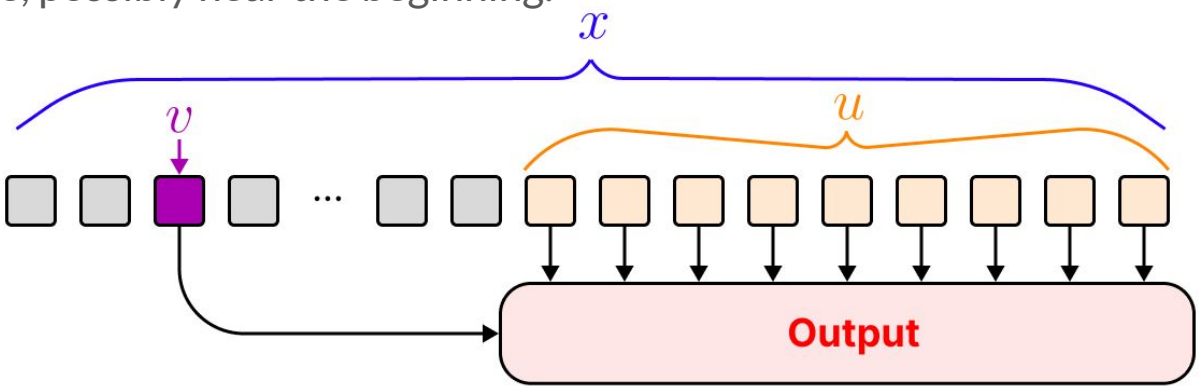
Function-Composition (S2FC) Tasks

These tasks show a separation, requiring what both models do well!

Compute **F**(**u**(**x**), **v**(**x**))

u: Dense, at the end.

v: Sparse, possibly near the beginning.

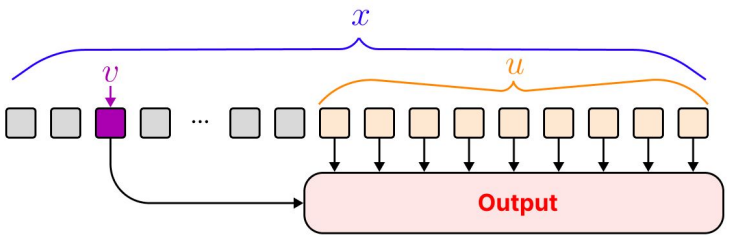


Function-Composition (S2FC) Tasks

These tasks show a separation, requiring what both models do well!

Compute **F(u(x), v(x))**

Example:



Deepseek

Question: If $a > 1$, then the sum of the real solutions of $\sqrt{a - \sqrt{a + x}} = x$ is equal to

Response: <think>

To solve the equation $\sqrt{a - \sqrt{a + x}} = x$, let's start by squaring both $\cdots$

$\left(\sqrt{a - \sqrt{a + x}}\right)^2 = x^2 \implies a - \sqrt{a + x} = x^2.$ ← v

Rearrange to isolate the inner square root term:

$(a - x^2)^2 = a + x \implies a^2 - 2ax^2 + (x^2)^2 = a + x \implies x^4 - 2ax^2 - x + (a^2 - a) = 0$

$\cdots$

Wait, wait. Wait. That's an aha moment I can flag here.

Let's reevaluate this step-by-step to identify if the correct sum can be $\cdots$

We started with the equation:

$\sqrt{a - \sqrt{a + x}} = x$

First, let's square both sides:

$a - \sqrt{a + x} = x^2 \implies \sqrt{a + x} = a - x^2$

Next, I could square both sides again, treating the equation: $\cdots$ ← u

$\cdots$

Why *Function Composition*?

*A **Transformer's** challenge*

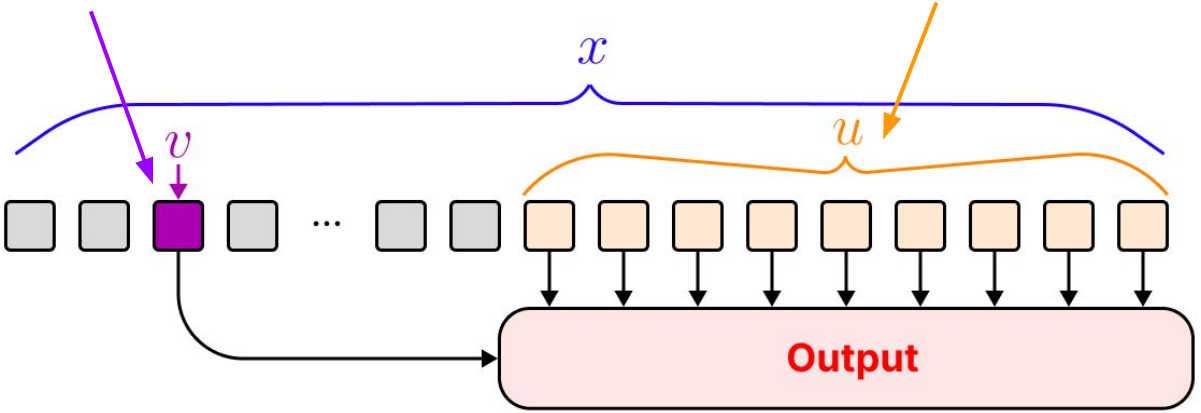
Tasks with **early dependencies**.

Captured by **v** in *Function Composition*.

*An **SSM's** challenge*

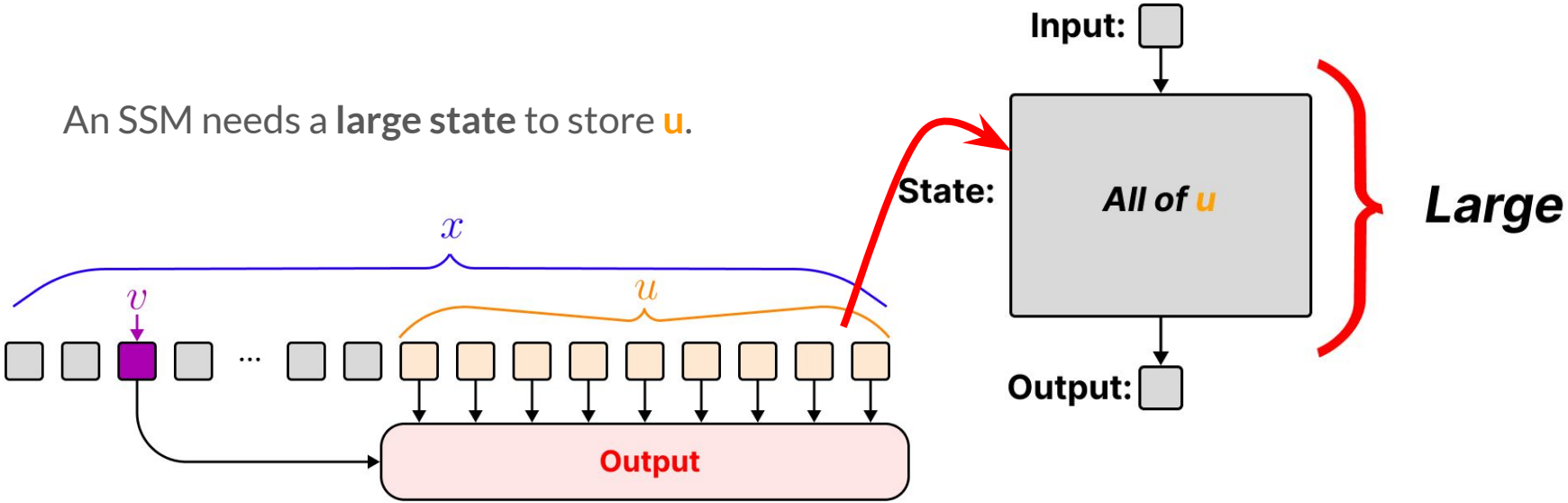
Tasks that **require a large memory**.

Captured by **u** in *Function Composition*.



Lower Bounds - SSM (Theorem 3.3)

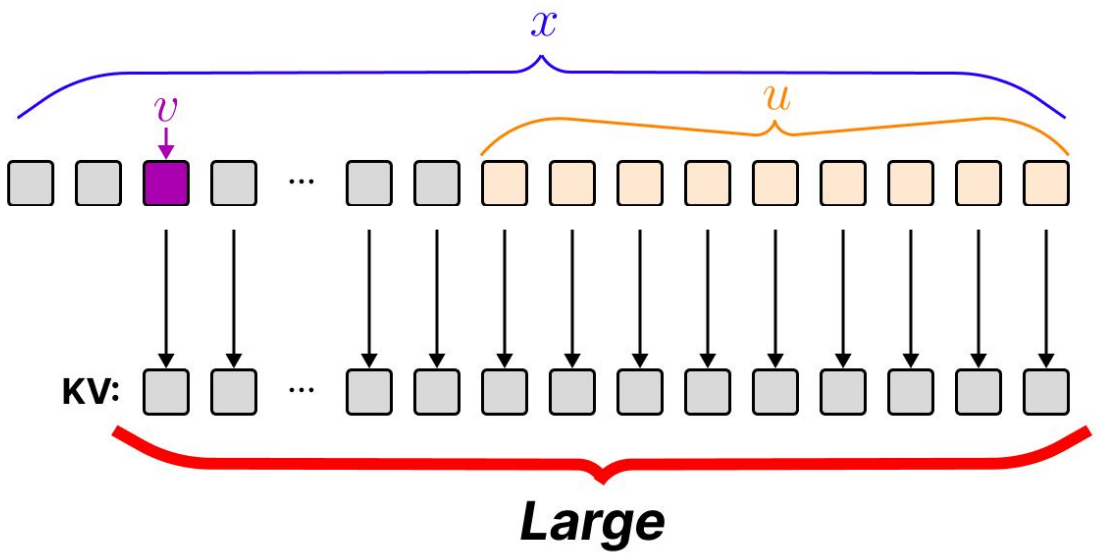
An SSM needs a **large state** to store **u**.



Theorem 3.3. *Let F be a function defined as in Definition 3.1 that satisfies Assumption 3.2. There is a distribution D over the input (u, v) such that any model M that is a composition of k state space layers SSM_i, with state space $\mathcal{S}_i, i \in [k]$ that can compute F with probability $1/2$ must satisfy* $\sum_{i=1}^{k} \log(|\mathcal{S}_i|) \geq \Omega(m \log(|\mathcal{V}|) - q \log(|\mathcal{Y}|))$.

Lower Bounds - TF (Theorem 3.7)

A Transformers needs a **large window** to capture **v**.



Theorem 3.7. *Let F be a function that satisfies Assumption 3.6. There is a distribution D over the input such that any model M that is a composition of k Transformer layers $\mathrm{TF}_1, \ldots, \mathrm{TF}_k$ that can compute F with probability $2/3$ must satisfy $\sum_{i=1}^{k} W_i \geq R$.*

Main Question

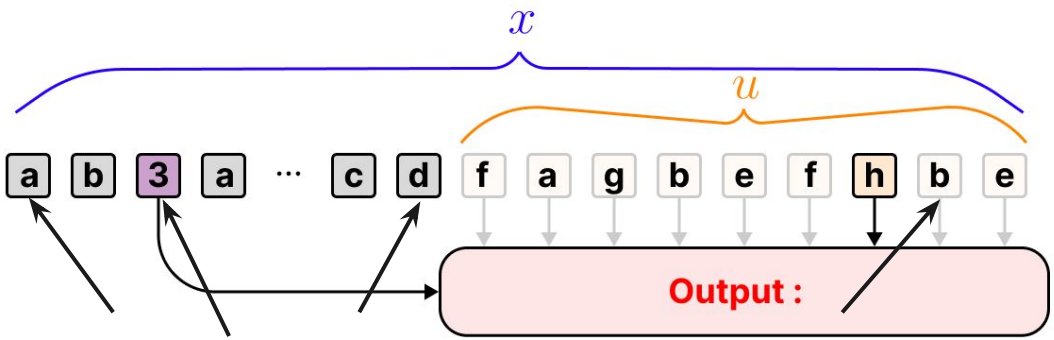
So far, we see that pure models need lots of memory to solve *function composition* tasks.
But the goal is to show **hybrids** are better!

We **explicitly construct hybrid models** that use less memory!
We need to pick a specific task for this.

What tasks are best for hybrid models?

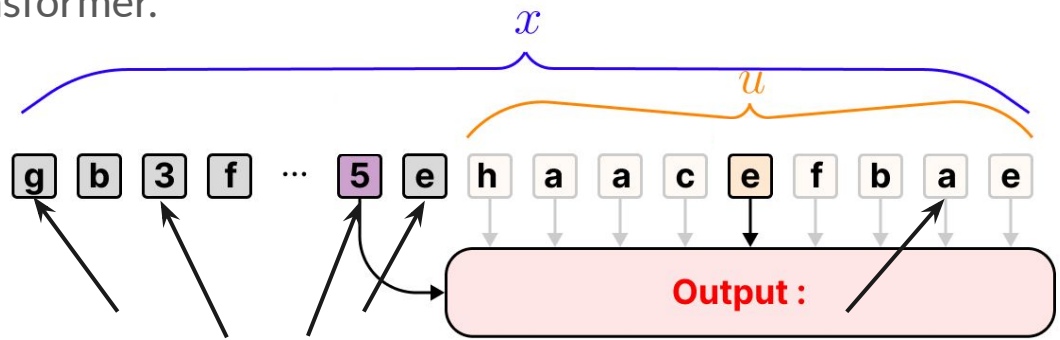
Selective Copying

A concrete instantiation of *function composition*.



u can be large -> Difficult for an SSM.

v can be early -> Difficult for a Transformer.



Selective Copying

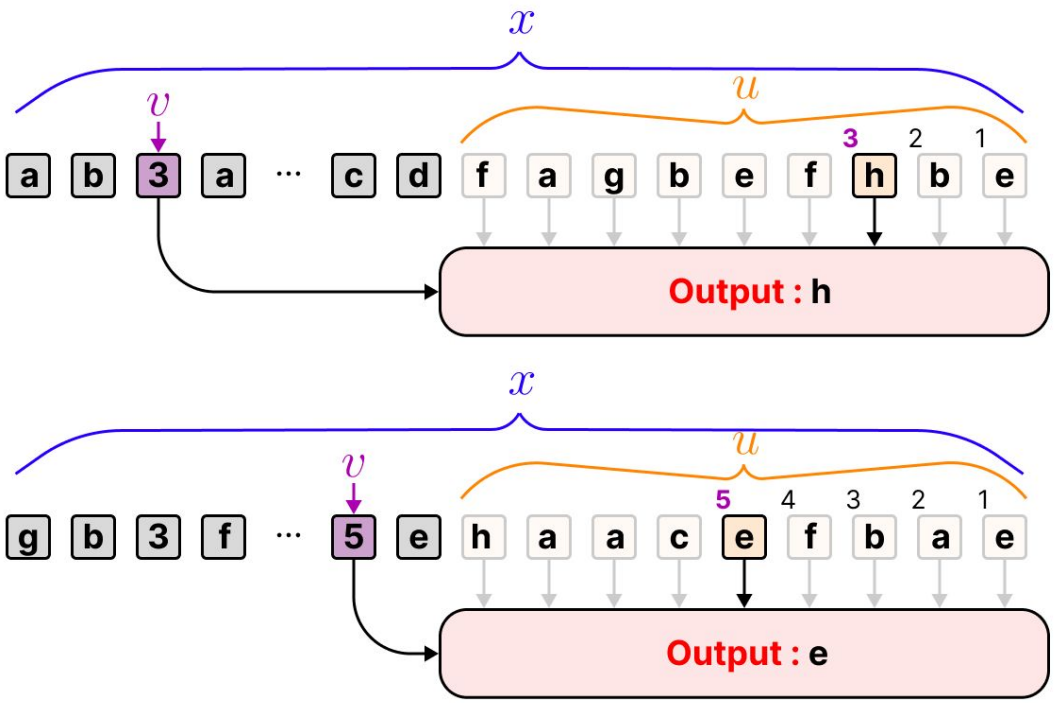
We want a simple algorithm.

Step 1: Find the **last number**.

Easy for an SSM!

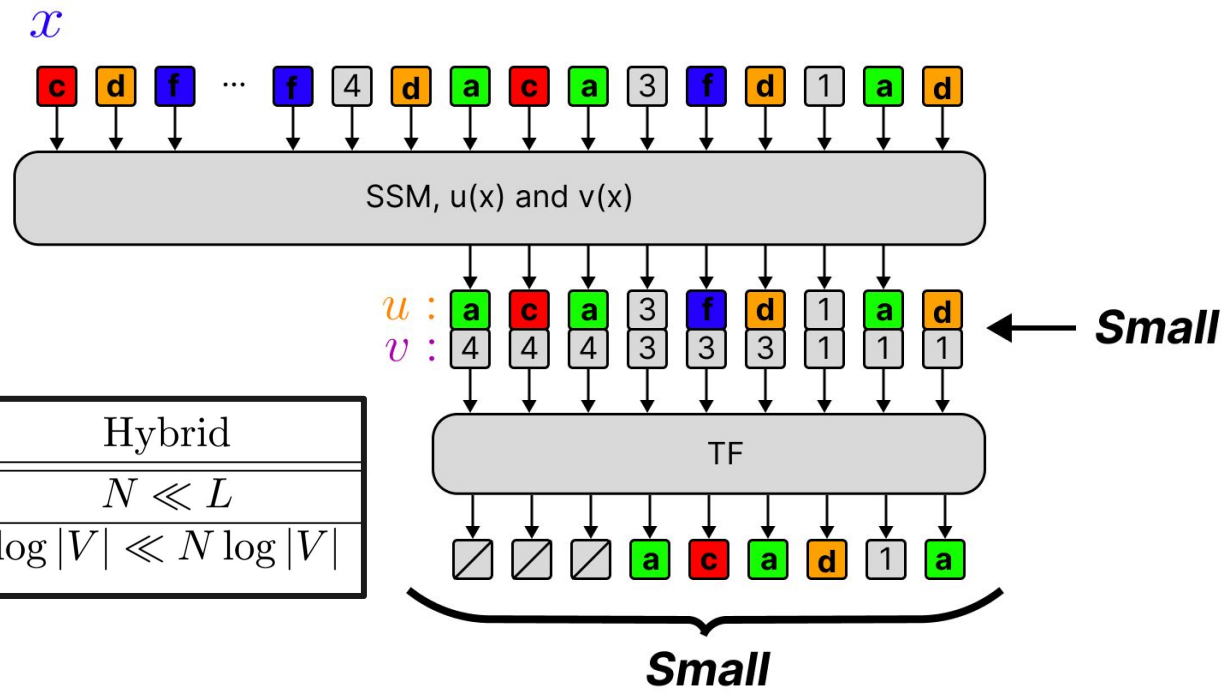
Step 2: Copy the **correct token**.

Easy for a Transformer!



Selective Copying Construction

We write the **explicit weights** for this hybrid model!



Model	SSM	TF	Hybrid
Window	—	L	$N \ll L$
State Size	$N \log \lvert V \rvert$	—	$\log \lvert V \rvert \ll N \log \lvert V \rvert$

Main Question

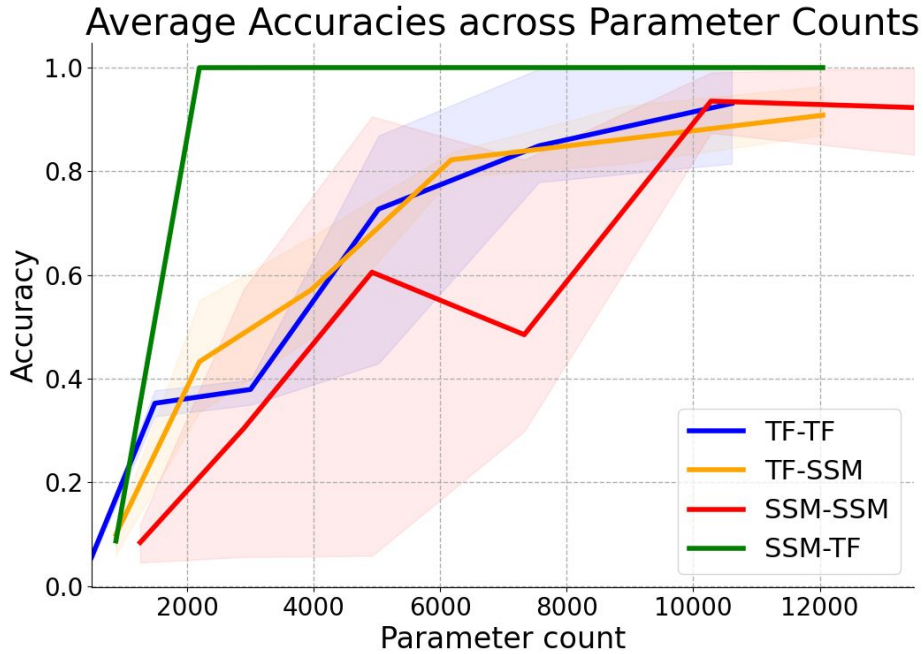
Hybrids are **theoretically** more capable with less memory than pure models!

Does this translate to **learned hybrids**?

What tasks are best for hybrid models?

Learnability - Selective Copying

Yes! Learned hybrids outperform pure learned models on selective copying!



Hybrids > SSMs
Hybrids > Transformers

SSM->TF > TF->SSM

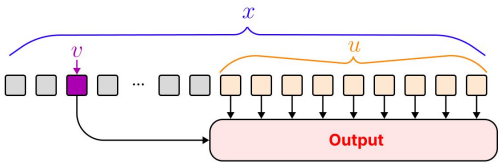
TF->SSM = SSMs
TF->SSM = Transformers

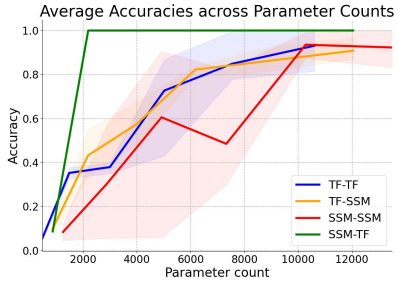
Final Takeaways

1. Hybrid models are **fundamentally** more expressive than pure models.
2. Task structure and **data dependencies drive hybrid performance**.
3. **Not all hybrid models perform well**! Architecture design is nuanced.

Theorem 3.3. *Let F be a function defined as in Definition 3.1 that satisfies Assumption 3.2. There is a distribution D over the input (u, v) such that any model M that is a composition of k state space layers SSM_i, with state space $\mathcal{S}_i, i \in [k]$ that can compute F with probability $1/2$ must satisfy* $\sum_{i=1}^{k} \log(|\mathcal{S}_i|) \geq \Omega(m \log(|\mathcal{V}|) - q \log(|\mathcal{Y}|))$.

Theorem 3.7. *Let F be a function that satisfies Assumption 3.6. There is a distribution D over the input such that any model M that is a composition of k Transformer layers $\mathrm{TF}_1, \ldots, \mathrm{TF}_k$ that can compute F with probability $2/3$ must satisfy* $\sum_{i=1}^{k} W_i \geq R$.





Coming Next!

When are Transformer-first hybrids the best choice?

How can *function composition* structure be detected in real data?

How do the dependencies of data in context change the optimal hybrid?

Where does the 3:1 or 7:1 SSM-to-TF ratio come from, theoretically?

What about deeper hybrids? How can more layers be used best?

Thank you!




Website:

Feel free to reach out!

Email - jfcooper2@cs.wisc.edu

Poster Session:

Location - Hall A

Time - 5:00 - 6:45pm Later!

Lower Bounds

- SSMs require sufficient state to remember all of **u**, which can be large.

Theorem 3.3. *Let F be a function defined as in Definition 3.1 that satisfies Assumption 3.2. There is a distribution D over the input (u, v) such that any model M that is a composition of k state space layers SSM_i, with state space $\mathcal{S}_i, i \in [k]$ that can compute F with probability $1/2$ must satisfy* $\sum_{i=1}^{k} \log(|\mathcal{S}_i|) \geq \Omega(m \log(|\mathcal{V}|) - q \log(|\mathcal{Y}|))$.

Total bits across all SSM layers

The width of **u**

Needed bits for a single token

Lower Bounds

- Transformers need enough window to capture **v**, which can be the whole sequence.

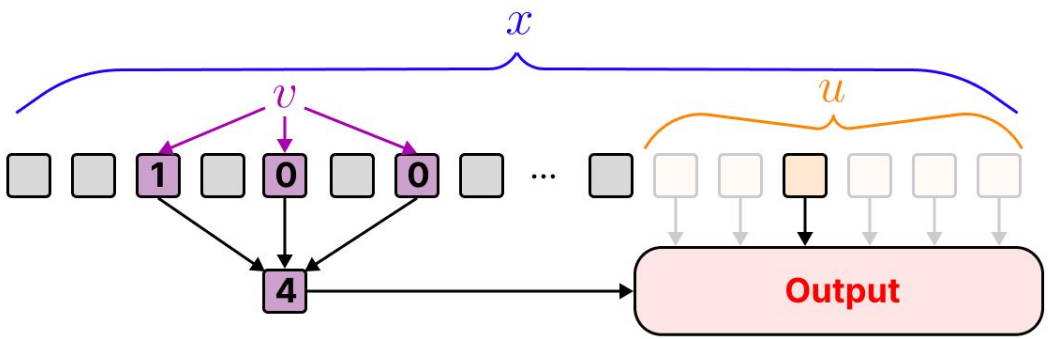
Theorem 3.7. *Let F be a function that satisfies Assumption 3.6. There is a distribution D over the input such that any model M that is a composition of k Transformer layers $\mathrm{TF}_1, \ldots, \mathrm{TF}_k$ that can compute F with probability $2/3$ must satisfy $\sum_{i=1}^{k} W_i \geq R$.*

Total of window sizes for each layer

Maximum distance to **v**

Associative Recall with Decoding

Another representative of *function composition*.



Vocabulary: bit tokens (0 and 1) and a vocabulary of number tokens.

Task: find the **last k bits, construct binary value from them**, and then perform **associative recall with this number**.

For a large vocabulary, **u** can be large -> Difficult for an SSM.

For certain data distributions, **v** can be early -> Difficult for a Transformer.

Learnability - Associative Recall with Decoding

In contrast to the Selective Copying experiments, this task is significantly harder.

The construction for task required 3 layers rather than 2.

However, we still see that the **hybrid** is the top performing model architecture.