Mamba Arch

- **Introduction**

next slide →

02

Mamba vs Transformers

03

Models	Training	Inference	Context	Memory limit	Selective reasoning
Transformers	Fast	Slow	Bounded	Proportional to the sequence	Yes
RNNs	Slow	Fast	Unbounded	Proportional to hidden state	~Yes
CNNs	Fast	Fast	Bounded	Proportional to the sequence	~No
S6 (Mamba)	Fast	Fast	Unbounded	Proportional to hidden state	~Yes

04

Mamba ?

05

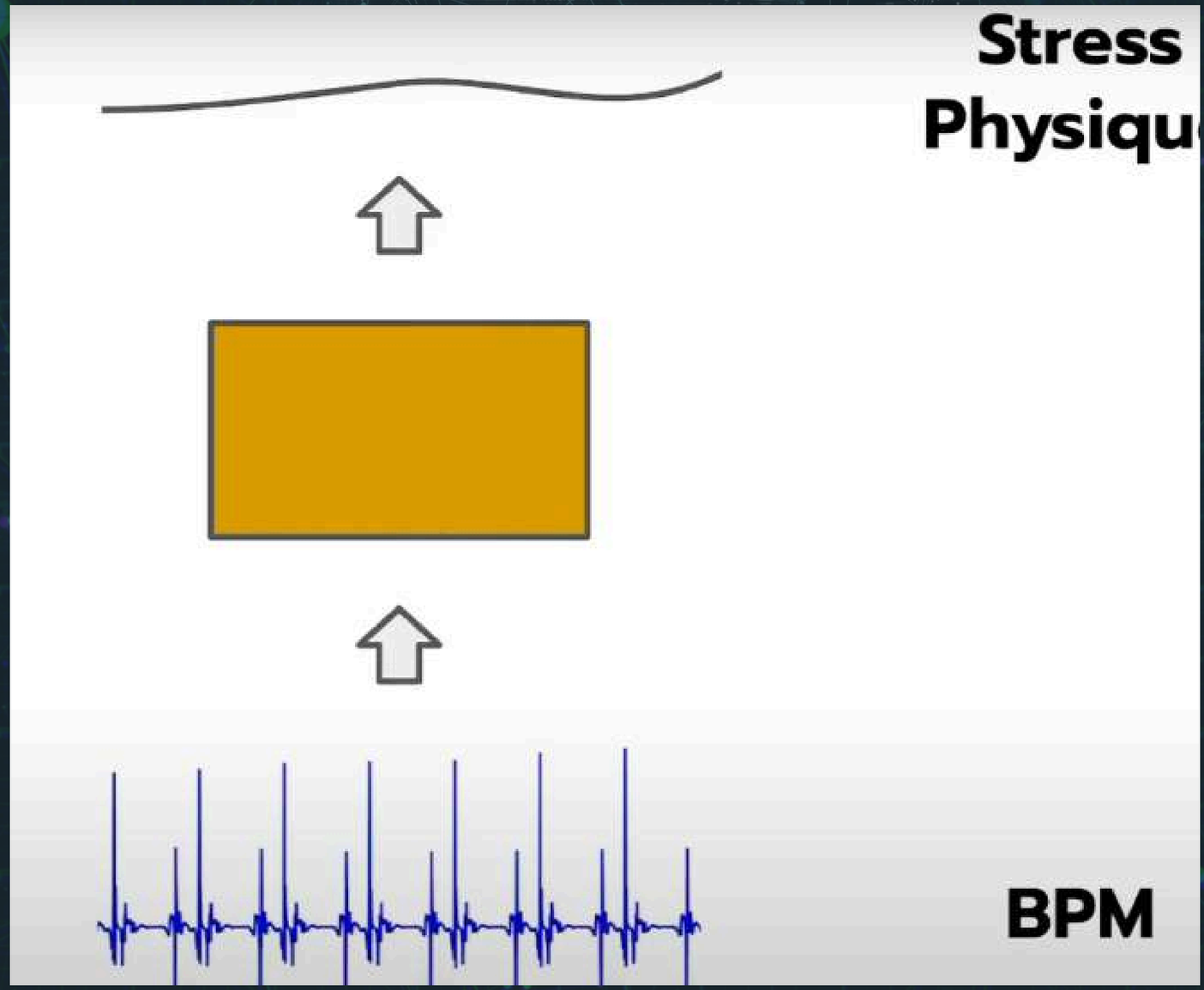
Mamba architecture

SSM-But de papier

- Créer un mécanisme ayant les propriétés des RNNs et CNNs
- Créer un mécanisme ayant une mémoire à long terme

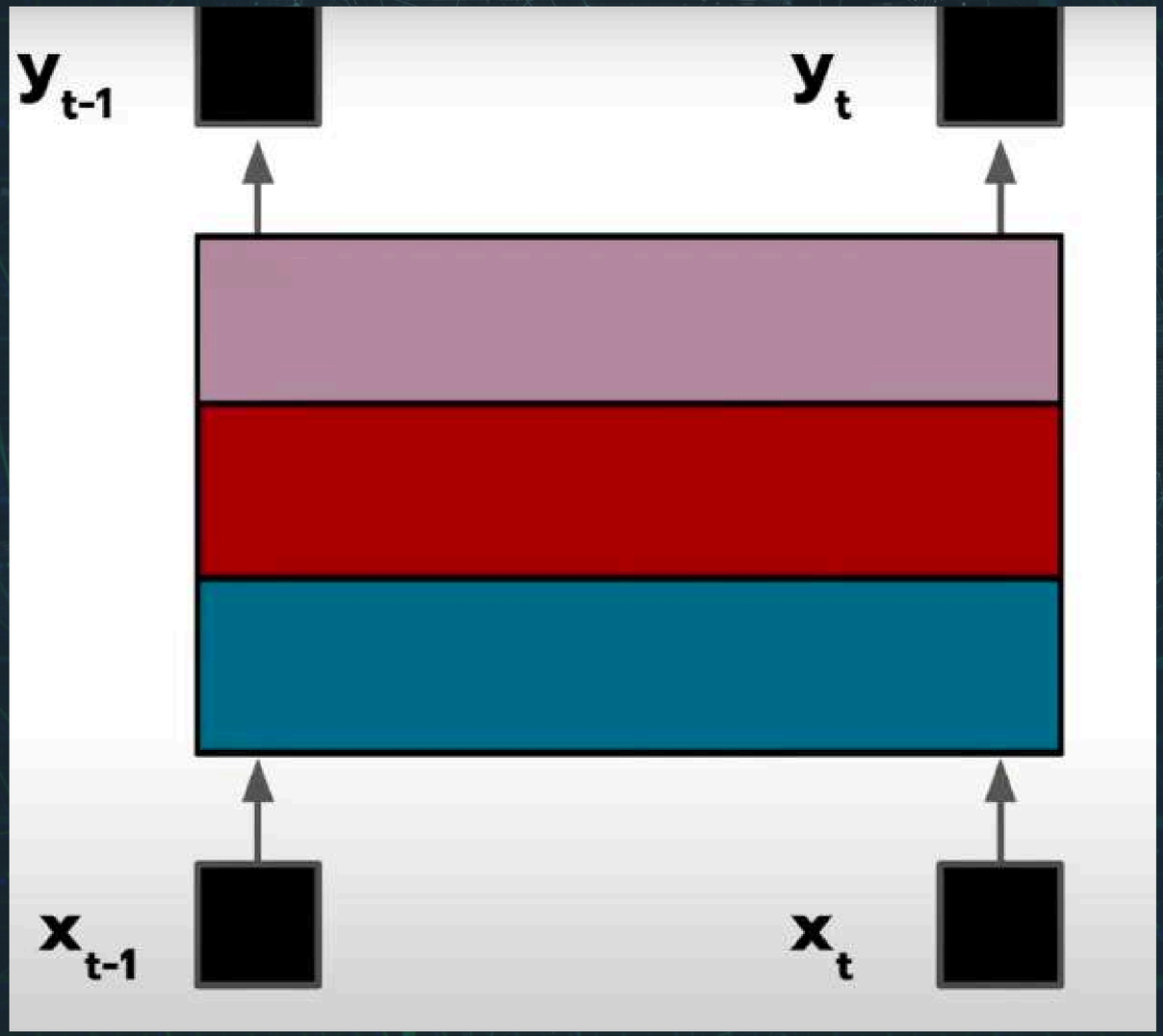
07

SSM



08

SSM



09

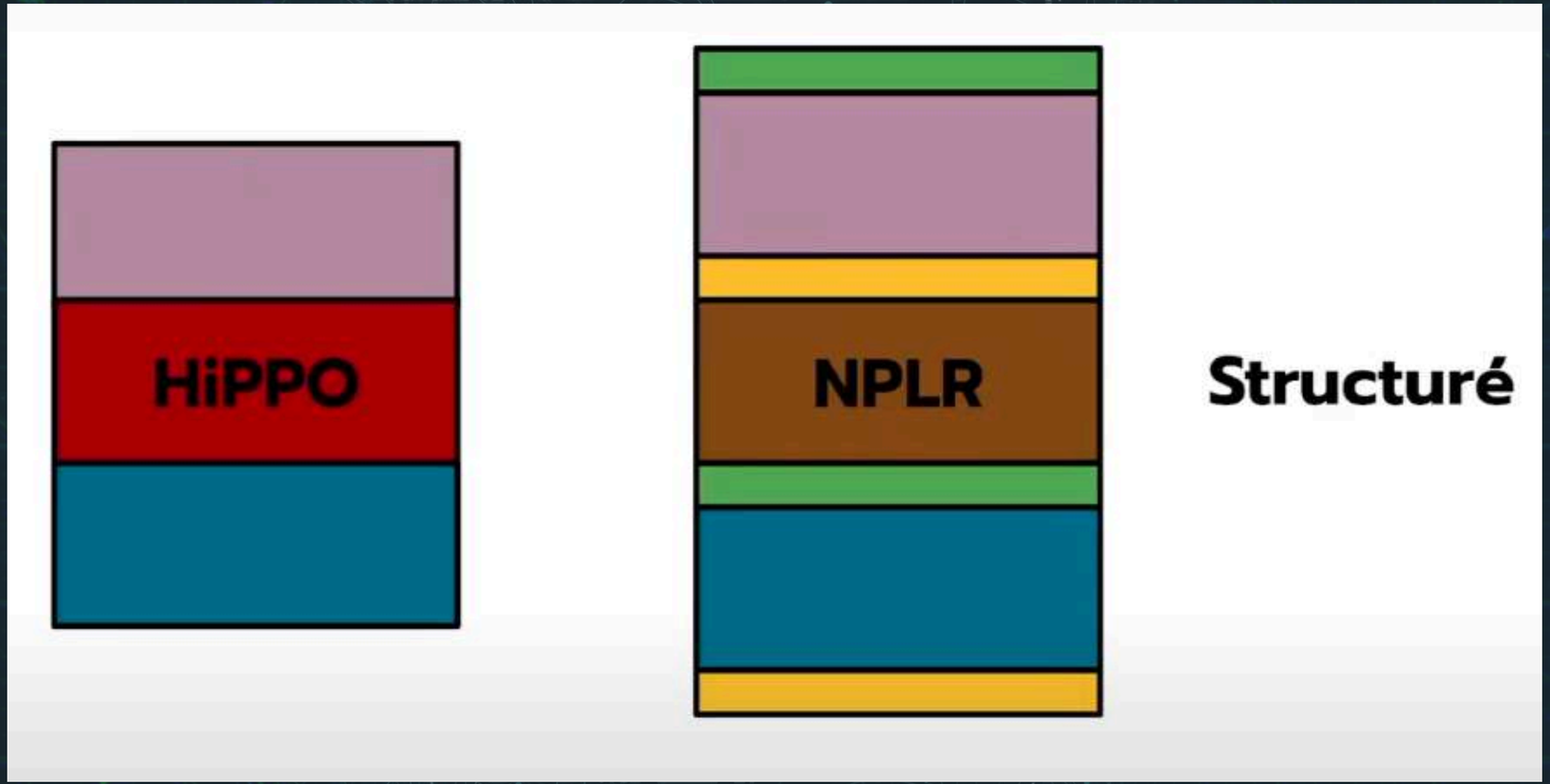
SSM-Propriétés

- SMM est linéaire
- Stateful
- parallélisable
- LIT: linear Time-Invariant

S4-But de papier

- Créer un mécanisme ayant une mémoire à long terme

S4



S4- Résultats

Table 3: Benchmarks vs. efficient Transformers

	Length 1024		Length 4096	
	Speed	Mem.	Speed	Mem.
Transformer	1×	1×	1×	1×
Performer	1.23×	0.43×	3.79×	0.086×
Linear Trans.	**1.58**×	**0.37**×	**5.35**×	**0.067**×
S4	**1.58**×	0.43×	5.19×	0.091×

Table 8: (**WikiText-103 language modeling**) S4 approaches the performance of Transformers with much faster generation. (*Top*) Transformer baseline which our implementation is based on, with attention replaced by S4. (*Bottom*) Attention-free models (RNNs and CNNs).

Model	Params	Test ppl.	Tokens / sec
Transformer	247M	**20.51**	0.8K (1×)
GLU CNN	229M	37.2	-
AWD-QRNN	151M	33.0	-
LSTM + Hebb.	-	29.2	-
TrellisNet	180M	29.19	-
Dynamic Conv.	255M	25.0	-
TaLK Conv.	240M	23.3	-
S4	249M	**20.95**	**48K (60×)**

S6-But de papier

- Donner au S4 la capacité de raisonner en fonction de contexte

S6-Propriétés

- S6 est linéaire
- Stateful
- Structuré
- optimisable
- parallélisable
- Non-LIT: Context-Aware

S6- Résultats

Model	Token.	Pile ppl ↓	LAMBADA ppl ↓	LAMBADA acc ↑	HellaSwag acc ↑	PIQA acc ↑	Arc-E acc ↑	Arc-C acc ↑	WinoGrande acc ↑	Average acc ↑
Hybrid H3-130M	GPT2	—	89.48	25.77	31.7	64.2	44.4	24.2	50.6	40.1
Pythia-160M	NeoX	29.64	38.10	33.0	30.2	61.4	43.2	24.1	**51.9**	40.6
Mamba-130M	NeoX	**10.56**	**16.07**	**44.3**	**35.3**	**64.5**	**48.0**	**24.3**	**51.9**	**44.7**
Hybrid H3-360M	GPT2	—	12.58	48.0	41.5	68.1	51.4	24.7	54.1	48.0
Pythia-410M	NeoX	9.95	10.84	51.4	40.6	66.9	52.1	24.6	53.8	48.2
Mamba-370M	NeoX	**8.28**	**8.14**	**55.6**	**46.5**	**69.5**	**55.1**	**28.0**	**55.3**	**50.0**
Pythia-1B	NeoX	7.82	7.92	56.1	47.2	70.7	57.0	27.1	53.5	51.9
Mamba-790M	NeoX	**7.33**	**6.02**	**62.7**	**55.1**	**72.1**	**61.2**	**29.5**	**56.1**	**57.1**
GPT-Neo 1.3B	GPT2	—	7.50	57.2	48.9	71.1	56.2	25.9	54.9	52.4
Hybrid H3-1.3B	GPT2	—	11.25	49.6	52.6	71.3	59.2	28.1	56.9	53.0
OPT-1.3B	OPT	—	6.64	58.0	53.7	72.4	56.7	29.6	59.5	55.0
Pythia-1.4B	NeoX	7.51	6.08	61.7	52.1	71.0	60.5	28.5	57.2	55.2
RWKV-1.5B	NeoX	7.70	7.04	56.4	52.5	72.4	60.5	29.4	54.6	54.3
Mamba-1.4B	NeoX	**6.80**	**5.04**	**64.9**	**59.1**	**74.2**	**65.5**	**32.8**	**61.5**	**59.7**
GPT-Neo 2.7B	GPT2	—	5.63	62.2	55.8	72.1	61.1	30.2	57.6	56.5
Hybrid H3-2.7B	GPT2	—	7.92	55.7	59.7	73.3	65.6	32.3	61.4	58.0
OPT-2.7B	OPT	—	5.12	63.6	60.6	74.8	60.8	31.3	61.0	58.7
Pythia-2.8B	NeoX	6.73	5.04	64.7	59.3	74.0	64.1	32.9	59.7	59.1
RWKV-3B	NeoX	7.00	5.24	63.9	59.6	73.7	67.8	33.1	59.6	59.6
Mamba-2.8B	NeoX	**6.22**	**4.23**	**69.2**	**66.1**	**75.2**	**69.7**	**36.3**	**63.5**	**63.3**
GPT-J-6B	GPT2	–	4.10	68.3	66.3	75.4	67.0	36.6	64.1	63.0
OPT-6.7B	OPT	–	4.25	67.7	67.2	76.3	65.6	34.9	65.5	62.9
Pythia-6.9B	NeoX	6.51	4.45	67.1	64.0	75.2	67.3	35.5	61.3	61.7
RWKV-7.4B	NeoX	6.31	4.38	67.2	65.5	76.1	67.8	37.5	61.0	62.5

Fine-Tuning Mamba

Install Ans Import

16

Install

```
pip install -q datasets
pip install -q trl
pip install -q peft
```

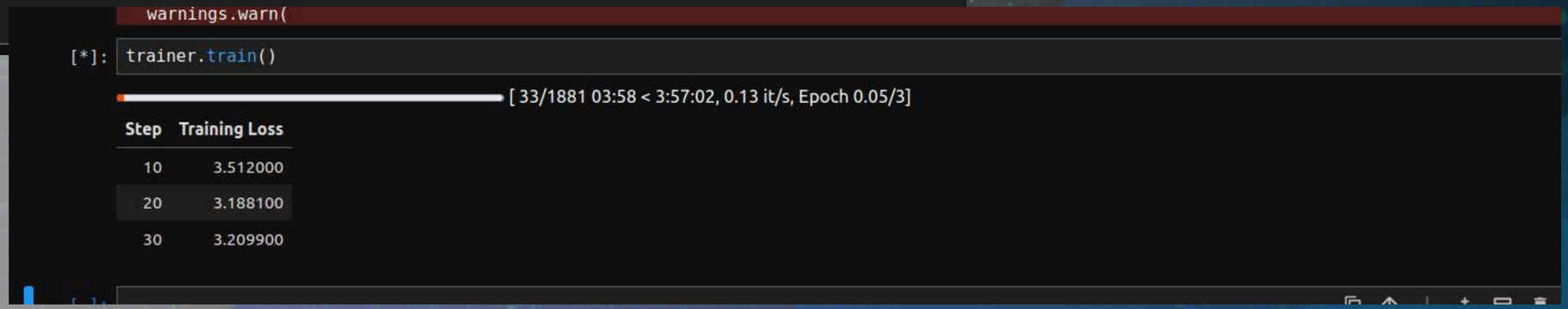
Import

```
from datasets import load_dataset
from trl import SFTTrainer
from peft import LoraConfig
from transformers import AutoTokenizer, AutoModelForCausalLM, TrainingArguments
```

Nothing New

```python
tokenizer = AutoTokenizer.from_pretrained("state-spaces/mamba-130m-hf")
model = AutoModelForCausalLM.from_pretrained("state-spaces/mamba-130m-hf")
dataset = load_dataset("Abirate/english_quotes", split="train")

training_args = TrainingArguments(
    output_dir="./results",
    num_train_epochs=3,
    per_device_train_batch_size=4,
    logging_dir='./logs',
    logging_steps=10,
    learning_rate=2e-3
)
lora_config =  LoraConfig(
        r=8,
        target_modules=["x_proj", "embeddings", "in_proj", "out_proj"],
        task_type="CAUSAL_LM",
        bias="none"
)
```



KPI: Key Performance Indicator

Accuracy and performance metrics :

- Perplexity
- Accuracy
- BLEU score
- ROUGE score (ROUGE-N, ROUGE-L)

Bias and fairness metrics :

- Demographic parity
- Equal opportunity
- Counterfactual fairness

Other metrics :

- Fluency
- Coherence
- Factuality


Thank You!