Learning to (Learn at Test Time): RNNs with Expressive Hidden States

PUBLISHED AT ICML 2025
AUTHORS: YU SUN ET AL.

KASHYAP SUTHAR & SERGIO RODRIGUEZ

Recurrent Neural Networks (RNNs):

- Processes one token at a time (sequential)
- Past information is compressed into a fixed-size hidden state
- Linear Complexity
- Struggles with long term dependencies

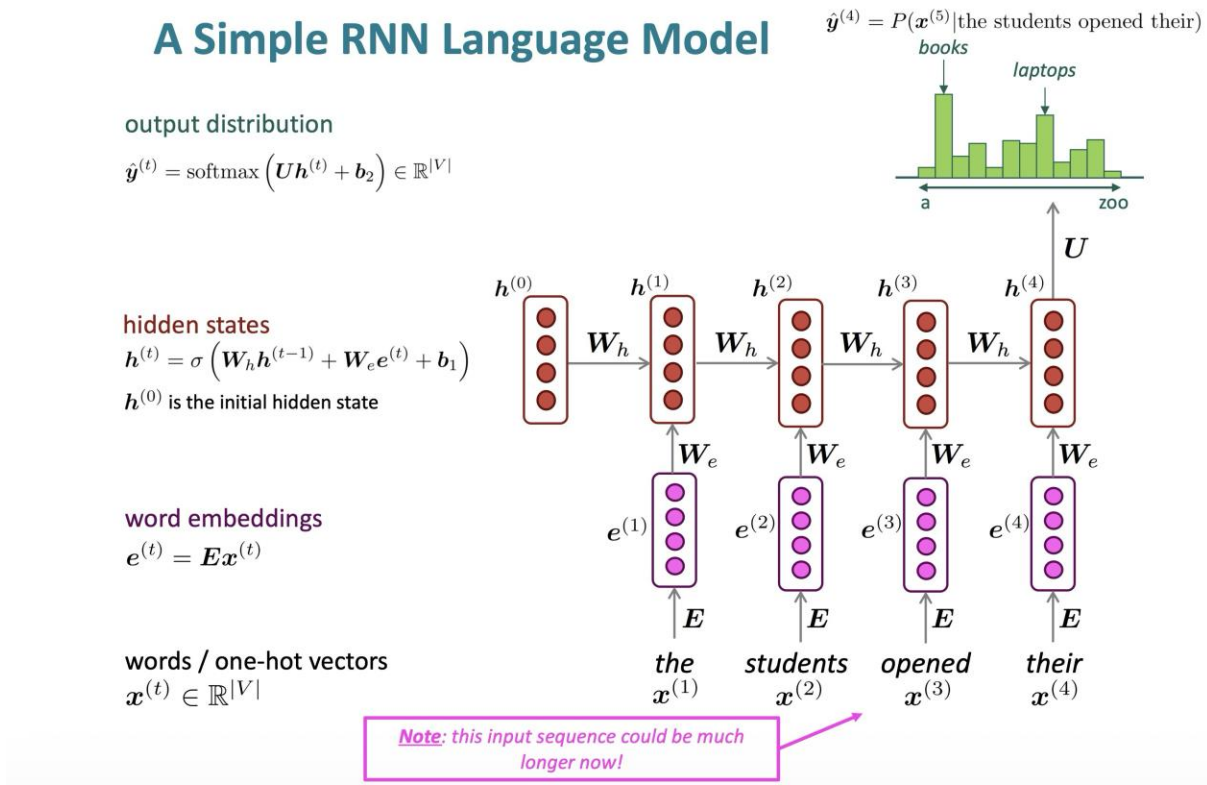


Figure adapted from Manning, 2024

Transformers

- Uses attention to captures long-context dependencies
- Quadratic Complexity
- Computationally expensive
- Parallelizable

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Q: Query
K: Key
V: Value
d_k : key dimension
softmax: converts scores into probabilities

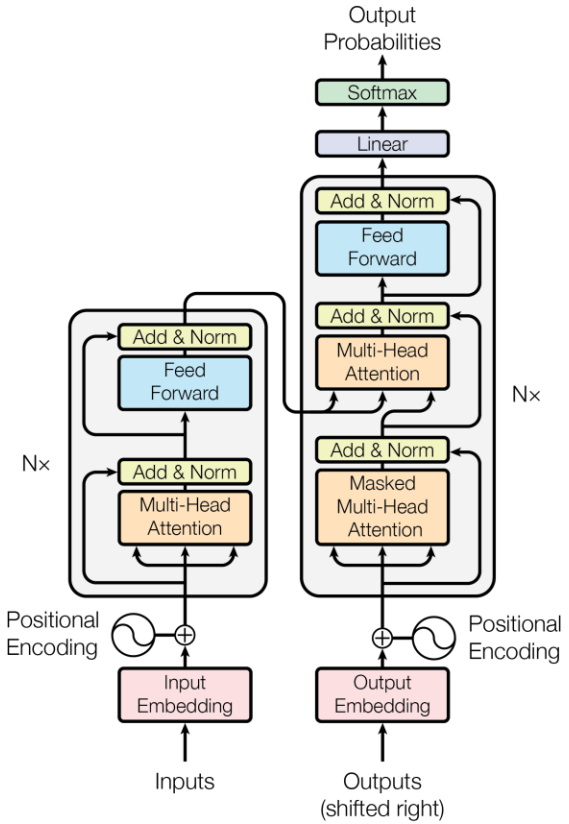


Figure adapted from Vaswani et al., 2017

Problem Statement & Motivation

Problem Statement:

- Existing RNNs have linear complexity, but their performance in long context is limited by the expressive power of its hidden states

To address this the author's propose a better compression algorithm that:

- Can preserve underlying structure and relationship over long sequences.
- Can effectively compress large sequences into a hidden state.
- Should remain efficient and expressive in long-context settings.

Proposed Method: Overview

Key Idea: Instead of making the hidden state a fixed vector, make it a learnable model that updates over time

- $s_t = W_t$, weights of a model *f*, which can be a linear model, a small neural network, or anything else.
- Theoretically equivalent to linear attention

Loss: $\ell(W; x_t) = \|f(\tilde{x}_t; W) - x_t\|^2$

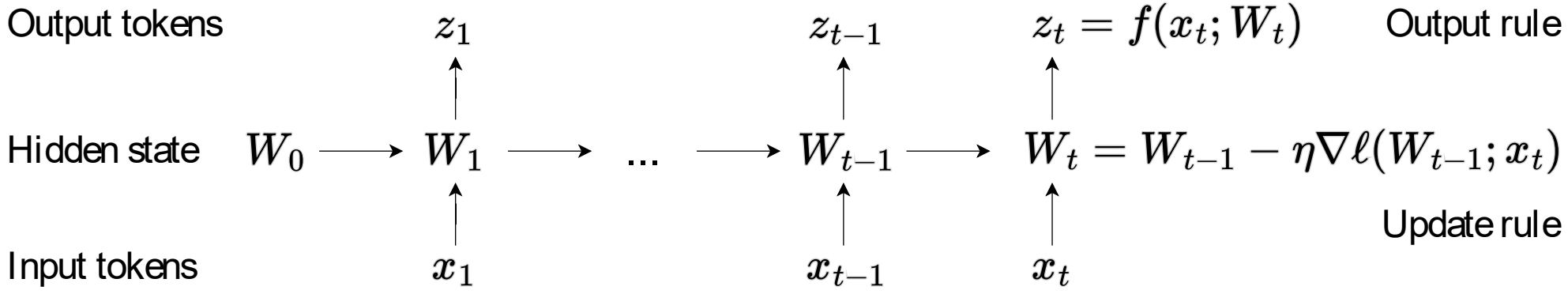


Figure adapted from Sun et al., 2024

Inner Loop vs Outer Loop in Test-Time Training (TTT)

Inner Loop:

- Trains the TTT Layer
- Updated at each time step using self-supervised loss
- Initialized from the outer loop parameters
- Happens during inference (test time)
- Fast (per token)

Outer loop:

- Trains the larger network
- Updates network parameters θ
- Happens during training only
- Updated across dataset
- Slow (across dataset)

Idea: Outer loop optimizes θ so that inner-loop updates of W are effective

Proposed Method: Learning a self-supervised task for TTT

Instead of handcrafting a self-supervised task we directly optimize the task for the final goal of next-token prediction.

We learn the task as part of the **outer loop** by creating outer-loop parameters, then creating low-level projections:

Training View: $k_t = \theta_K x_t$

Label View: $v_t = \theta_V x_t$

Test View: $q_t = \theta_Q x_t$

θ_K, θ_V, and θ_Q are learnable matrices from the outer loop.
Apply to our loss which gives us a new **self-supervised loss** and **output rule**:

$$\ell(W; x_t) = \|f(\theta_K x_t; W) - \theta_V x_t\|^2$$

$$z_t = f(\theta_Q x_t;\ W_t)$$

Proposed Method: f and W_0

Two variants of TTT layers:

1. TTT-Linear

   $$f_{lin} = Wx$$

2. TTT-Multilayer Perceptron (MLP)
   - Two layers, like MLPs in Transformers
   - Hidden dimension is 4x input dimension
   - GELU Activation

Always followed by a Layer Normalization and Residual Addition

The initial weights W_0 are shared across all sequences and are learned in outer loop rather than set to 0. This adds a negligible number of parameters but significantly improves training stability.

Proposed Method: Learnable η

Key Idea:

- Make the learning rate adaptable instead of fixed
- Learning rate becomes a function of input

$$\eta(x) = \eta_{\text{base}} \cdot \sigma\left(\theta_{\text{lr}} \cdot x\right)$$

Where:

- θ_{lr}: learned in outer loop
  - η_{base} initializations:
    - 1 for TTT-Linear
    - 0.1 for TTT-MLP
- η(x): base learning rate
- σ is the sigmoid function

Result:

- Learning rate changes over time
- Improves adaptability of updates

Proposed Method: Parallelization with Mini-batch TTT

To make the update step parallelized we do:

$$W_t = W_{t-1} - \eta G_t = W_0 - \eta \sum_{s=1}^{t} G_s$$

Where G_t is descent direction.

We propose using mini-batch gradient descent as follows:

$$G_t = \nabla \ell(W_{t'}; x_t)$$

Where t' = t – mod(t,*b*), *b* = TTT batch size, and t' is the last timestep of previous mini-batch.

This allows us to parallelize *b* gradient computations at a time.

TTT Layer Optimization: Primal vs. Dual Form

Primal Form:

- The standard update rule is inherently sequential this limits the parallelization and leads to poor hardware utilization on GPUs/TPUs, which increases the training time

Dual Form:

- Dual form avoids materializing intermediate W_t allowing for the computation of output tokens $z_1,\ldots,z_b$ using parallelized matrix multiplications, this sacrifices theoretical complexity for hardware utilization
- Equations: $W_b = W_0 - 2\eta(W_0X - X)X^T$

  $Z = W_0X - 2\eta\Delta \qquad \Delta = (W_0X - X)\,\mathrm{mask}(X^TX)$
- As a result, Dual Form is faster than Primal in this instance, while producing equivalent outputs

Experiment Framework

Baselines:

- Mamba: Modern RNN designed for efficient long-context modeling
- Transformers: Standard baseline for long-context performance comparison

Metrics:

- Perplexity: measures how well the model predicts the next token, lower value indicate better performance.
- Floating-Point Operations(FLOPs): computational cost of the model, lower value indicate better efficiency

Setup:

- Dataset: The Pile (2k, 8k) and Books3 (1k → 32k)
- Model sizes: 125M → 1.3B parameters
- Training: Identical batch sizes, tokens and learning rate

Backbones:

- Since TTT is a modular layer, it can be integrated into different backbone architectures.
- Mamba Backbone (M) and Transformer Backbone (T)

Experiment results - Short Context

Key observations:

- At 2k: All methods performed similar, except TTT-MLP(M) which performed slightly worse
- At 8k: Both TTT methods preformed better than Transformers and Mamba

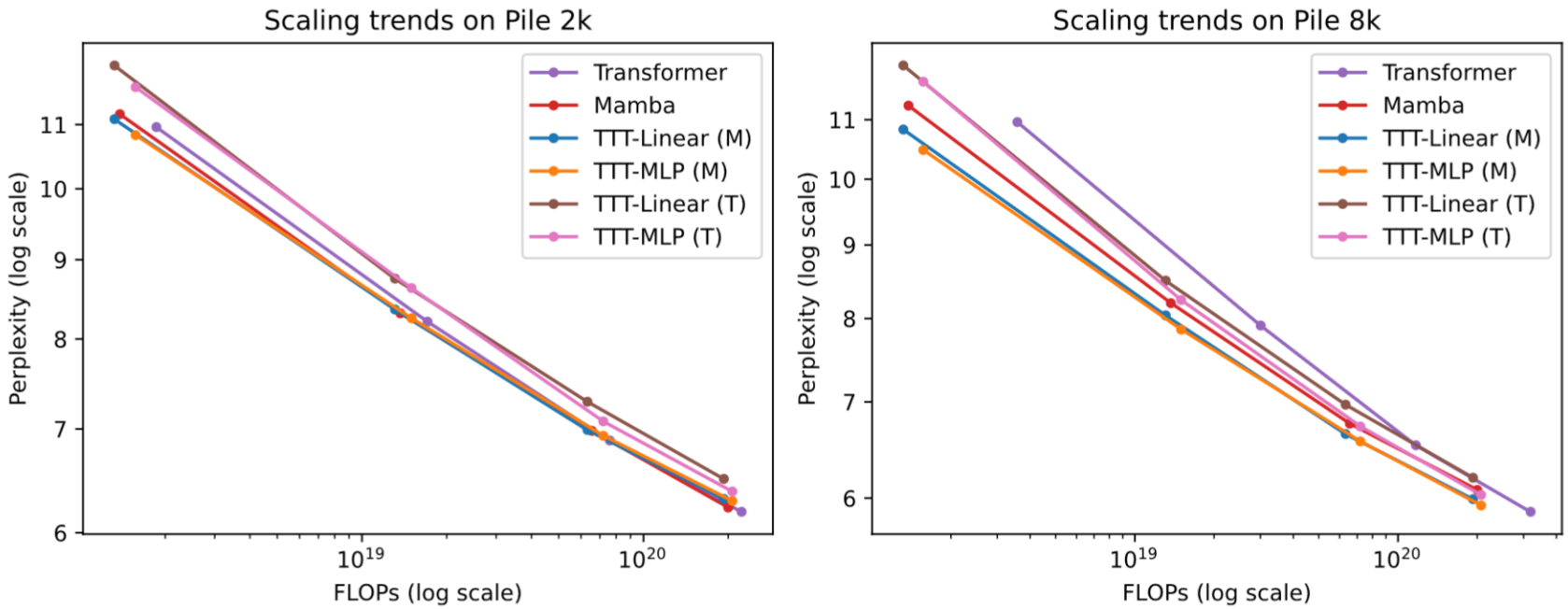
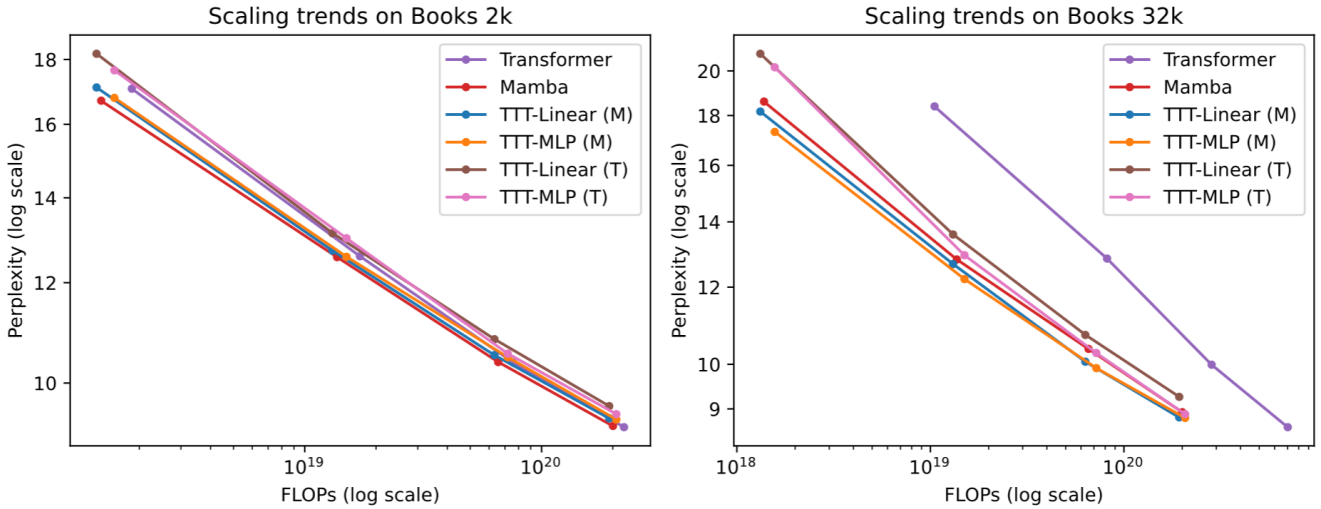


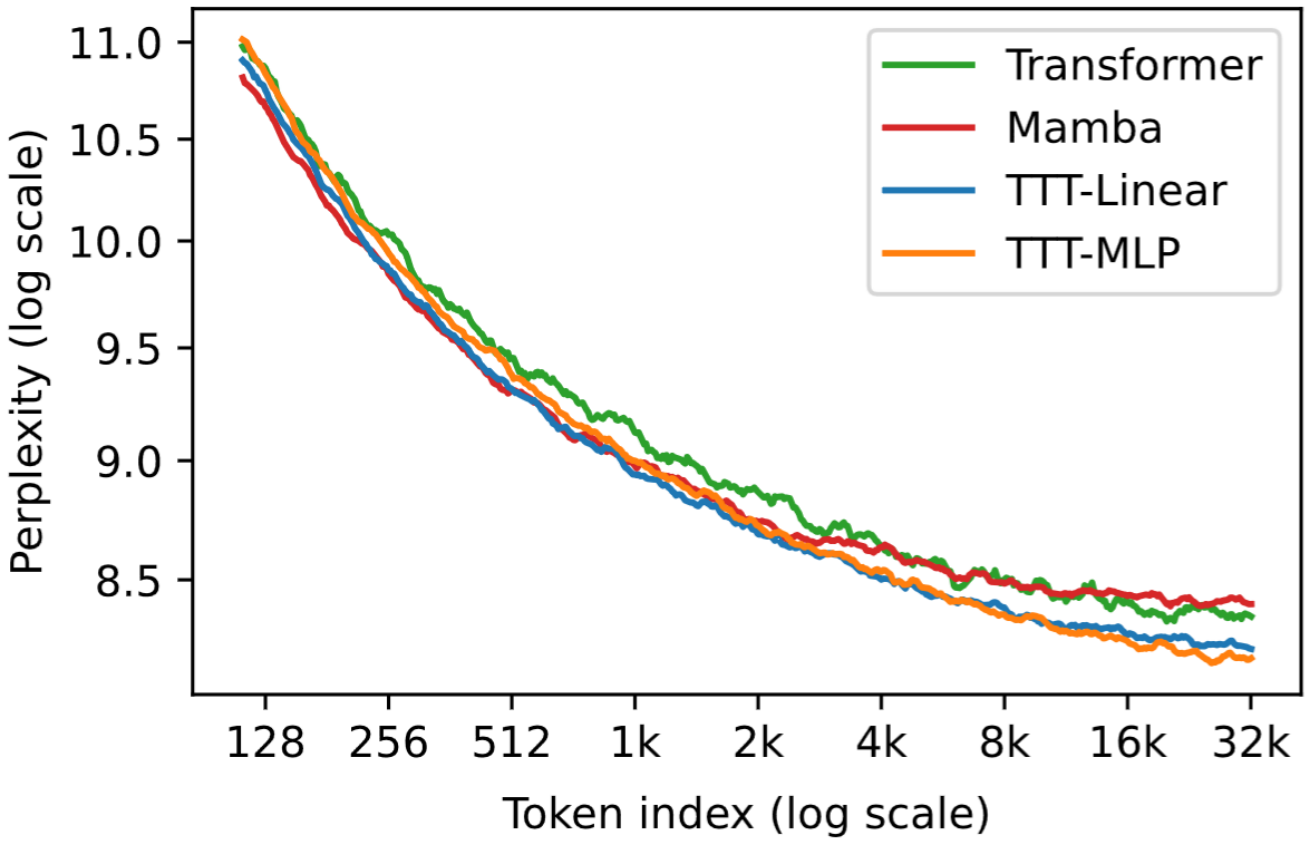
Figure adapted from Sun et al., 2024

Experiment results - Long Context



- 2K:
  - Earlier statements hold **except** that Mamba performed better than TTT-Linear
- 32K:
  - TTT-LINEAR (M) & TTT-MLP (M) performed better than Mamba
  - TTT-MLP (T) was slightly worse than TTT-MLP (M) at 1.3B params
- OVERALL:
  - TTT-MLP (T) suited for larger models & longer context

Figure adapted from Sun et al., 2024



Experiment results- Long Context

- Observations
  - Mamba(Red line) perplexity plateau after 16k, TTT models continues to decrease
  - TTT-MLP(Orange line) performs the best at long range, likely due to its larger hidden state capacity

Figure adapted from Sun et al., 2024

Strengths & Limitations

Strengths:

- Linear time complexity with strong long context performance.
- Compatible with different backbone architectures(Mamba and Transformer).
- It learns directly from the input sequence at test time, without the need of labels (Self-supervised).

Limitations:

- Increased memory usage as it must store all the weights in memory especially for TTT-MLP.
- Increase in time it takes to train the model (Wall-Clock-Time).
- TTT-MLP is better in long-context settings but is slower than TTT-Linear in practice.

Insights & Future Directions

- No real-world evaluations, everything was measured in perplexity or FLOPs.
- It would have been interesting to see how well TTTs work in other RNNs.
- Future works should explore how the use of different backbones will affect the performance, like Mamba 2 and Gated DeltaNet.
- Further improvements in system optimizations for efficiency.
- Scaling to longer context and larger models to see how they perform.
- More expressive inner models, like Convolutional Neural Networks.

Conclusion

Hidden states with a learnable model (TTT) updated by self-supervised learning:

- It is competitive with Mamba in long-context settings.
- It provides more efficient scaling when compared to Transformers.
- The dual form is faster than primal form, while giving equivalent outputs.

References

- Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., Presser, S., & Leahy, C. (2020). The Pile: An 800GB Dataset of Diverse Text for Language Modeling. *ArXiv:2101.00027 [Cs]*. https://arxiv.org/abs/2101.00027
- Manning, C. (2024). *Natural Language Processing with Deep Learning*.
- Sun, Y., Li, X., Dalal, K., Xu, J., Vikram, A., Zhang, G., Dubois, Y., Chen, X., Wang, X., Koyejo, S., Hashimoto, T., & Guestrin, C. (2024). Learning to (Learn at Test Time): RNNs with Expressive Hidden States. *ArXiv (Cornell University)*. https://doi.org/10.48550/arxiv.2407.04620
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017, June 12). *Attention Is All You Need*. Cornell University. https://arxiv.org/abs/1706.03762

Preemptive Slides

- Why compare against Mamba?
  - Mamba is a modern RNN-based model designed for efficient long-context modeling. It provides a baseline to evaluate the effectiveness of TTT Layers.
- How is this different from Transformers?
  - Transformers use attention to access past information, while TTT compresses past information into a model and updates it over time.
- What is the difference between views and projections?
  - Projections are transformations, and views are resulting representation

Input Examples

F.1 Pile-CC

pot trending topics and the coverage around them. First up, there's a bit of a visual redesign. Previously, clicking on a trending topic would highlight a story from one publication, and you'd have to scroll down past a live video section to view related stories. Facebook is replacing that system with a simple carousel, which does a better job of showing you different coverage options. To be clear, the change doesn't affect how stories are sourced, according to Facebook. It's still the same algorithm pickin

---

e public safety. He said the bridge saves commuters two or three minutes when trains pass – and those minutes could be vital.

"Two to three minutes may not mean much if you're just driving home from work, but if you're the one waiting for an ambulance to get to your home, if you're the one waiting for a fire truck to get to your home, if you're the one waiting for a police car to get to your home, those two to three minutes could mean the difference between life or death," Sharp said. "That's what this pro

F.3 Books3

cept of _forçage_ , 'a forcing of language enacted by the advent of an "other" language that is at once immanent and created', 44as Badiou puts it: this opens up vistas of a truly syntactic analysis of the poem, in which, again, Badiou would be close to his philosophical other, Deleuze, who, as we just saw, defines style through a-grammaticality and who tries to define what he calls an 'intensive line of syntax'.45

Nevertheless, the insistence on syntax as guarantee involves a _seventh paradox_ , the parad

---

rnment, before the Second World War there were 5,300 communities and two million burakumin. The BLL thinks there must be at least three million burakumin living in Japan today.

We visited a hall in Osaka where a taiko drum group, made up exclusively of young burakumin, were about to start their weekly rehearsal. The small gymnasium was filled with taiko drums of all sizes. The smallest was about the size of a snare drum, the largest about the size of a compact car. The Japanese drum group Kodo have made th

Proposed Method: Dual Form

*For TTT-Linear

Given:

$$z_t = f(x_t; W_t) = W_0 x_t - 2\eta \sum_{s=1}^{t} (W_0 x_s - x_s)\, x_s^T x_t$$

If we denote δ and the matrix Δ

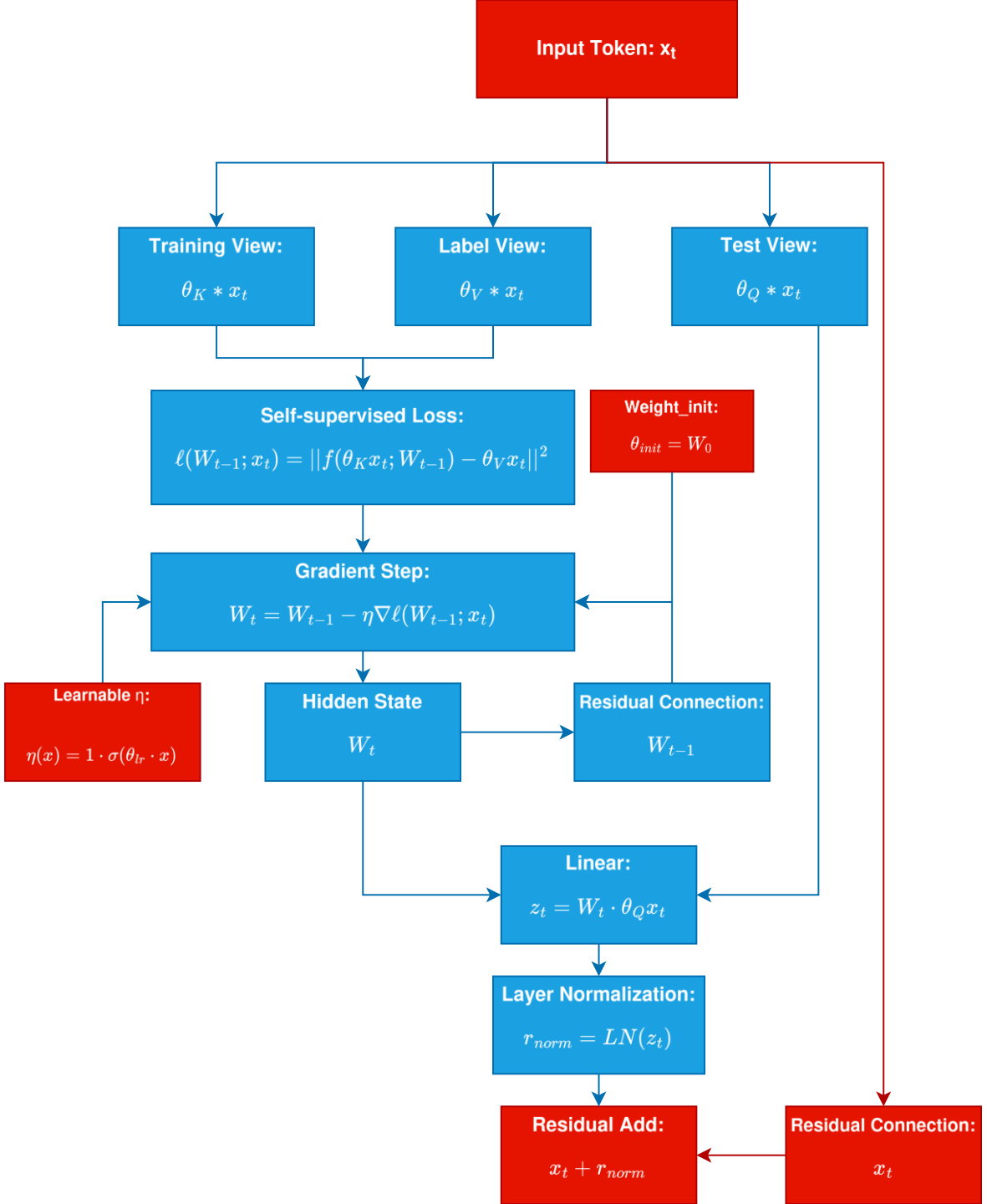
$$\delta_t = \sum_{s=1}^{t} (W_0 x_s - x_s)\, x_s^T x_t, \quad \Delta = [\delta_1, \ldots, \delta_b]$$

We can derive that:

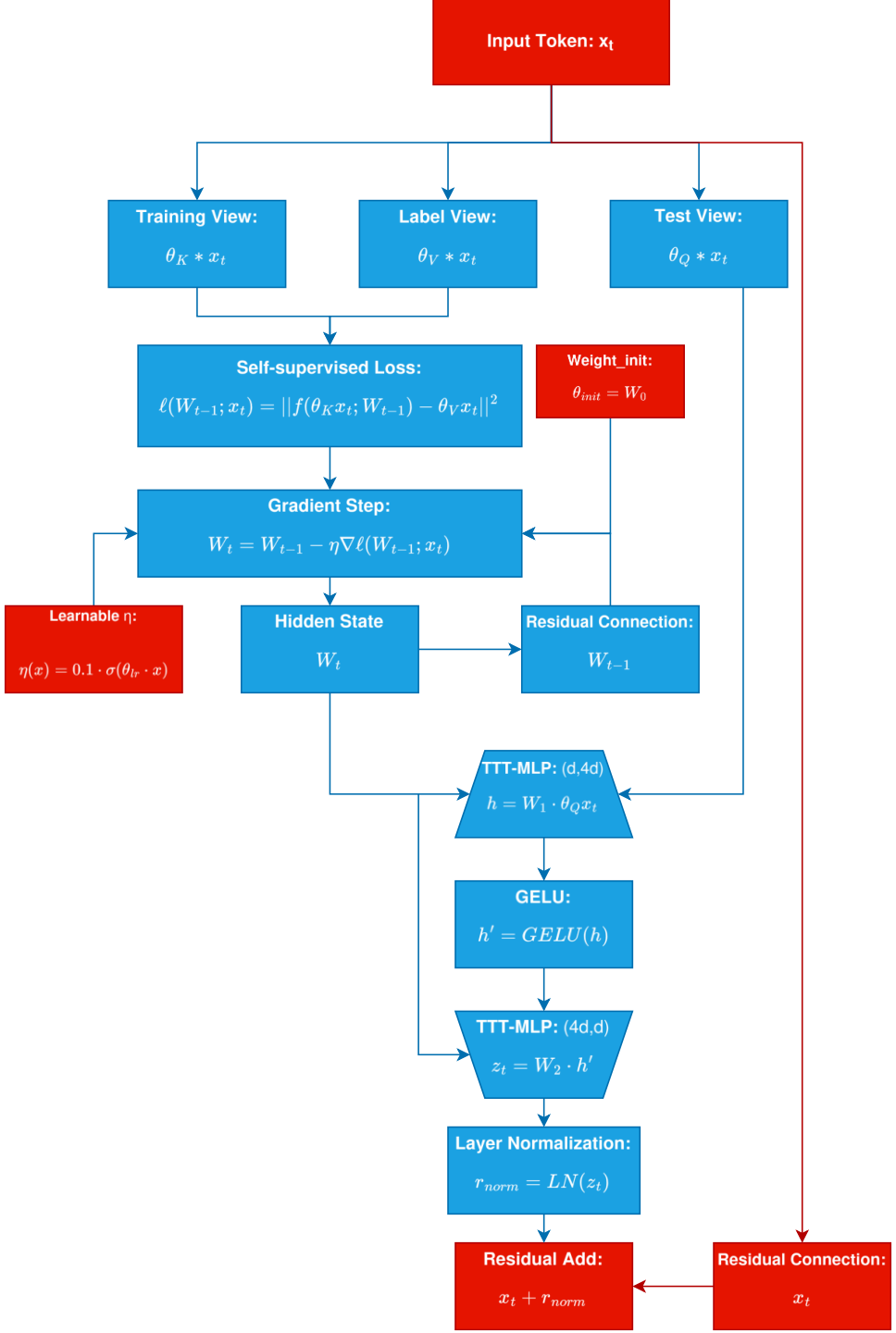
$$\Delta = (W_0 X - X)\, \mathrm{mask}(X^T X)$$

Then plugging Δ into original equation

$$Z = W_0 X - 2\eta \Delta$$


Input Token: x_t
Training View:
$\theta_K * x_t$
Label View:
$\theta_V * x_t$
Test View:
$\theta_Q * x_t$
Self-supervised Loss:
$\ell(W_{t-1}; x_t) = ||f(\theta_K x_t; W_{t-1}) - \theta_V x_t||^2$
Weight_init:
$\theta_{init} = W_0$
Gradient Step:
$W_t = W_{t-1} - \eta \nabla \ell(W_{t-1}; x_t)$
Learnable η:
$\eta(x) = 1 \cdot \sigma(\theta_{lr} \cdot x)$
Hidden State
W_t
Residual Connection:
W_{t-1}
Linear:
$z_t = W_t \cdot \theta_Q x_t$
Layer Normalization:
$r_{norm} = LN(z_t)$
Residual Add:
$x_t + r_{norm}$
Residual Connection:
x_t

TTT-Linear


Input Token: x_t
Training View:
$\theta_K * x_t$
Label View:
$\theta_V * x_t$
Test View:
$\theta_Q * x_t$
Self-supervised Loss:
$\ell(W_{t-1}; x_t) = ||f(\theta_K x_t; W_{t-1}) - \theta_V x_t||^2$
Weight_init:
$\theta_{init} = W_0$
Gradient Step:
$W_t = W_{t-1} - \eta \nabla \ell(W_{t-1}; x_t)$
Learnable η:
$\eta(x) = 0.1 \cdot \sigma(\theta_{lr} \cdot x)$
Hidden State
W_t
Residual Connection:
W_{t-1}
TTT-MLP: (d,4d)
$h = W_1 \cdot \theta_Q x_t$
GELU:
$h' = GELU(h)$
TTT-MLP: (4d,d)
$z_t = W_2 \cdot h'$
Layer Normalization:
$r_{norm} = LN(z_t)$
Residual Add:
$x_t + r_{norm}$
Residual Connection:
x_t

TTT-MLP