Carnegie Mellon University

Deep Equilibrium models

Shaojie Bai[1], J. Zico Kolter[1,2] and Vladlen Koltun[3]

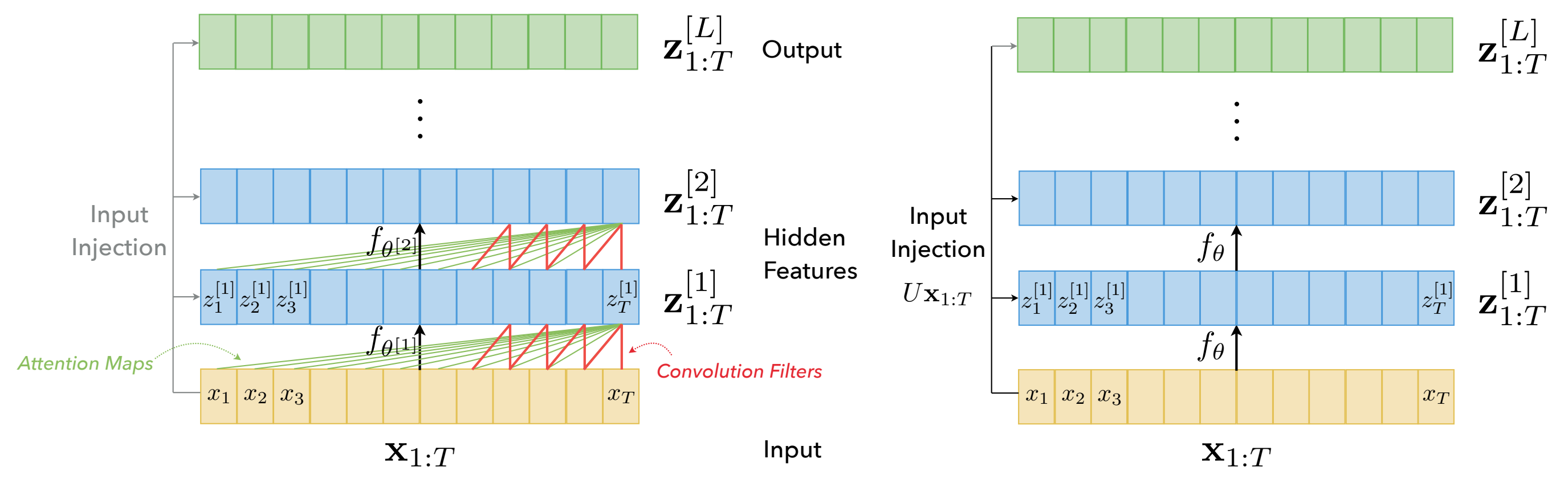
Carnegie Mellon University[1], Bosch Center for AI[2], Intel Labs[3]

ML MACHINE LEARNING DEPARTMENT

TL;DR: "One layer" is all you need! An implicit-depth (or infinite-depth) sequence model that solves for equilibriums and achieves SOTA, with only constant memory.

Introduction

- Deep networks have long been built on a core concept: *layers*.
- **The story we all tell**: Deep learning algorithms build *hierarchical* models of input data, where earlier layers extract "simple" features and later layers create high-level abstractions of the data.
- Layers are usually picked by model designers: e.g., ResNet-18/101.



(a) Conventional deep network (b) Weight-tied deep network

Figure 1: A conventional vs. a weight-tied deep network, where the same set of parameters is shared across layers.

- We focuses on sequence modeling, where a typical deep feed-forward network (e.g., Transformer) can be written as:

$$\mathbf{z}^{[i+1]}_{1:T} = f^{[i]}_\theta(\mathbf{z}^{[i]}_{1:T}; \mathbf{x}_{1:T}) \quad \text{for } i = 0, 1, 2, \dots, L-1$$

- Our work derives its motivation from surprising recent works that employ the *same* transformation in each layer (known as weight tying) and still achieve results competitive with the state-of-the-art:

$$\mathbf{z}^{[i+1]}_{1:T} = f_\theta(\mathbf{z}^{[i]}_{1:T}; \mathbf{x}_{1:T}), \quad \text{for } i = 0, 1, 2, \dots, L-1$$

- Note that we choose to explicitly model the passthrough connections from the input $\mathbf{x}_{1:T}$ (for reasons to be clear later).
- What is the limit of this process? How do we model it?
  We propose the deep equilibrium model (DEQ) that directly computes the eventual hidden layer values of this infinite limit:

$$\mathbf{z}^\star_{1:T} = f_\theta(\mathbf{z}^\star_{1:T}; \mathbf{x}_{1:T})$$

- DEQ offers constant memory cost, "infinite" layers, much faster convergence to (and backpropagation from) the fixed point, and results on par with (or better than) SOTA!
- Code at: `https://github.com/locuslab/deq`

Related Work

- Prior works have used gradient-checkpointing [4] and reversible networks [6] to reduce memory cost.
- Neural-ODE [3] focuses particularly on deep ResNets and models them as dynamic systems with latent, continuous trajectories.
- Implicit layers in deep learning can also trace back to some original work on training RNNs with recurrent backpropagation [1].

Deep Equilibrium Sequence Models

- We broadly consider the class of weight-tied, input-injected deep models:

$$\mathbf{z}^{[i+1]} = f_\theta(\mathbf{z}^{[i]}_{1:T}; \mathbf{x}), \quad i = 0, \dots, L-1, \quad \mathbf{z}^{[0]} = \mathbf{0}$$

- Isn't weight-tying a big restriction?
  1) Theoretically, no...
  Theorem[informal]: Any deep feedforward network can be represented by a weight-tied, input-injected network of equivalent depth.
  2) Empirically, no...
  Evidence: Recent works have well-demonstrated that weight-tied models achieve performances just as good (e.g., TrellisNet [2], Universal Transformer [5] and ALBERT [7]).
- Ideally, the network could have infinite depth (which is impossible in practice, due to limited memory availble in our GPUs).
- But *what is the limit of this process*? In practice, we observe these models tend to converge to a fixed point.

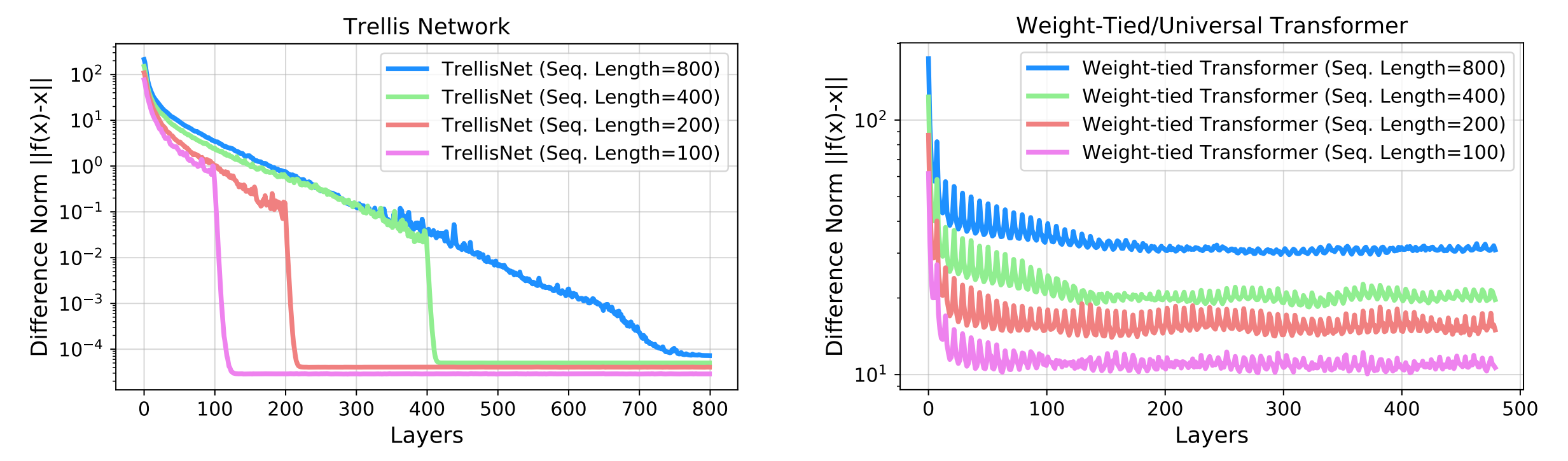


Figure 2: The convergence of intermediate activations in TrellisNet and weight-tied transformers.

- Intuitively, increasing depth brings "diminishing returns": each additional layer has a <u>smaller and smaller</u> contribution until the network reaches an equilibrium:

$$\lim_{i\to\infty} \mathbf{z}^{[i]}_{1:T} = \lim_{i\to\infty} f_\theta(\mathbf{z}^{[i]}_{1:T}; \mathbf{x}_{1:T}) \equiv f_\theta(\mathbf{z}^\star_{1:T}; \mathbf{x}_{1:T}) = \mathbf{z}^\star_{1:T}$$

- **Deep Equilibrium (DEQ) Model**: directly solve for this equilibrium/stable point via root-finding (e.g., Broyden's method), rather than just iterating the forward model; and directly differentiate *through* the equilibrium state.
- Define a single layer f_θ, and $g_\theta(\mathbf{z}_{1:T}; \mathbf{x}_{1:T}) := f_\theta(\mathbf{z}_{1:T}; \mathbf{x}_{1:T}) - \mathbf{z}_{1:T}$.

Forward pass: Given input $\mathbf{x}_{1:T}$, compute the equilibrium point $\mathbf{z}^\star_{1:T}$ such that $f_\theta(\mathbf{z}^\star_{1:T}; \mathbf{x}_{1:T}) - \mathbf{z}^\star_{1:T} = \mathbf{0}$; i.e.,

$$\mathbf{z}^\star_{1:T} = \text{RootFind}(g_\theta; \mathbf{x}_{1:T})$$

(via any black-box root-finding algorithms.)

Backward pass: Implicitly differentiate through the equilibrium state, directly using the Jacobian at the equilibrium:

$$\frac{\partial \ell}{\partial(\cdot)} = -\frac{\partial \ell}{\partial \mathbf{z}^\star_{1:T}} \left(J^{-1}_{g_\theta}\big|_{\mathbf{z}^\star_{1:T}}\right) \frac{\partial f_\theta(\mathbf{z}^\star_{1:T}; \mathbf{x}_{1:T})}{\partial(\cdot)}$$

regardless of the path/trajectory to this equilibrium.

Accelerating DEQ and Properties of DEQ Models

- One major challenge of scaling DEQ is the cost of computing/storing the exact inverse Jacobian $J^{-1}_{g_\theta}$ at every Newton step.
- In the forward pass, we propose to use Broyden's method (quasi-Newton) that makes low-rank updates to estimate $J^{-1}_{g_\theta}$:

$$J^{-1}_{g_\theta}\Big|_{\mathbf{z}^{[i+1]}_{1:T}} \approx B^{[i+1]}_{g_\theta} = B^{[i]}_{g_\theta} + \mathbf{u}^{[i]}\mathbf{v}^{[i]\top}$$

- In the backward pass, we alternatively solve the linear system

$$\left(J^\top_{g_\theta}\big|_{\mathbf{z}^\star_{1:T}}\right)\mathbf{x}^\top + \left(\frac{\partial \ell}{\partial \mathbf{z}^\star_{1:T}}\right)^\top = \mathbf{0}$$

where the first term (a vector-Jacobian product) can be efficiently computed without having to explicitly form the Jacobian (which can be prohibitively large on long and high-dimensional sequences).

- Memory cost of DEQ: DEQ provides a constant (i.e., $O(1)$) memory cost, as it only needs to store $\mathbf{z}^\star_{1:T}$, $\mathbf{x}_{1:T}$ and f_θ (model parameters).

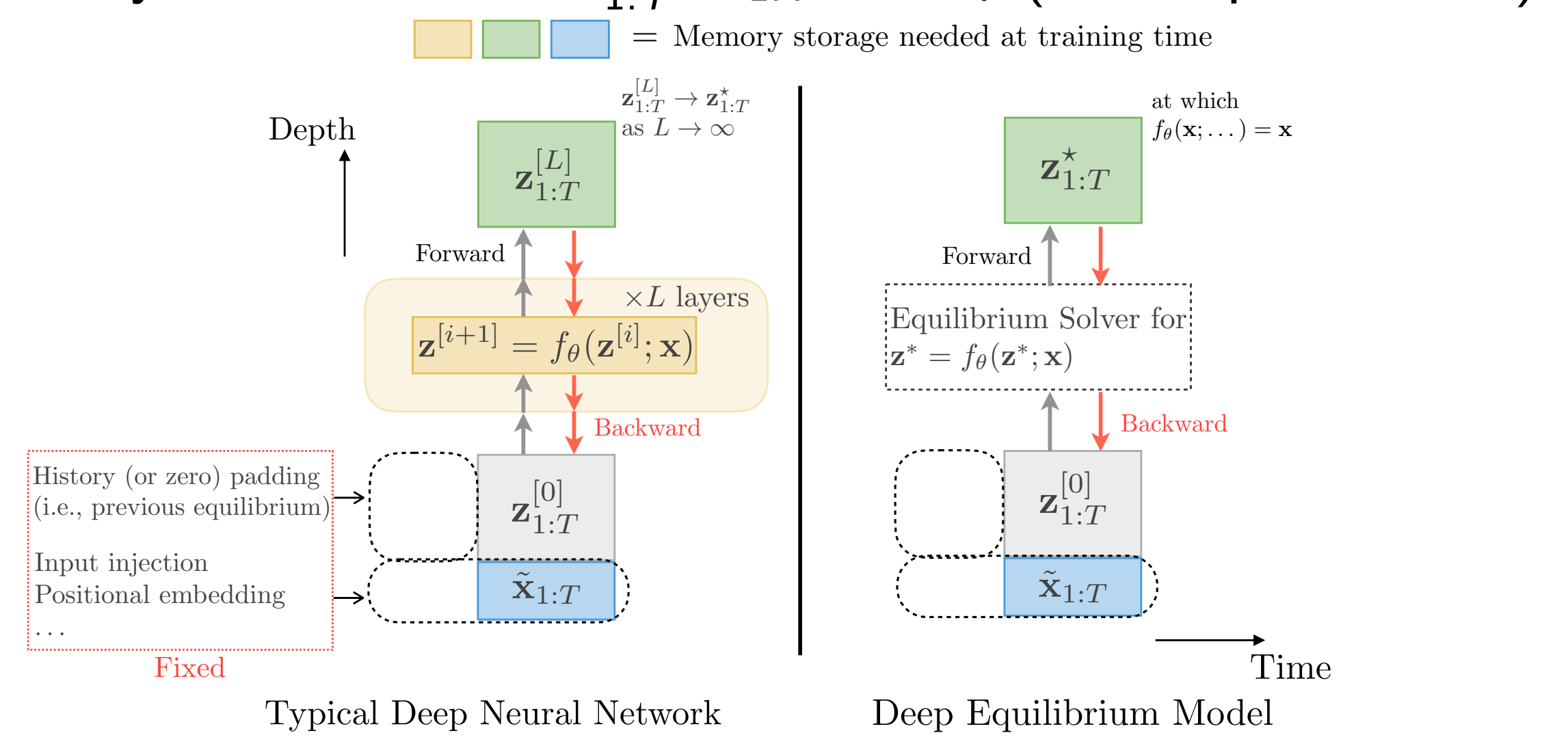

= Memory storage needed at training time
Depth
$\mathbf{z}^{[L]}_{1:T} \to \mathbf{z}^\star_{1:T}$ as $L \to \infty$
$\mathbf{z}^{[L]}_{1:T}$
Forward
$\times L$ layers
$\mathbf{z}^{[i+1]} = f_\theta(\mathbf{z}^{[i]}; \mathbf{x})$
Backward
History (or zero) padding (i.e., previous equilibrium)
Input injection
Positional embedding
...
Fixed
$\mathbf{z}^{[0]}_{1:T}$
$\tilde{\mathbf{x}}_{1:T}$
Typical Deep Neural Network
at which $f_\theta(\mathbf{x}; \dots) = \mathbf{x}$
$\mathbf{z}^\star_{1:T}$
Equilibrium Solver for $\mathbf{z}^* = f_\theta(\mathbf{z}^*; \mathbf{x})$
Time
Deep Equilibrium Model

Figure 3: DEQ operates with significantly less memory than deep nets due to an analytical backward pass.

- Choice of f_θ: The analyses above do not depend on the internal structure of f_θ. Generally, f_θ should be stable and constrained so as to solve for the equilibrium in a reliable and efficient manner.
- If one DEQ is good, do we benefit from deeply "stacking" DEQs?
  Theorem[informal]: Given transformations $f_{\theta^{[1]}}$ and $\nu_{\theta^{[2]}}$ (of potentially different function classes), $\exists$ a transformation Γ_Θ (where $\Theta = \theta^{[1]} \cup \theta^{[2]}$) s.t. $\text{DEQ}(\Gamma_\Theta; \mathbf{x}_{1:T}) = \text{DEQ}(\nu_{\theta^{[2]}}; \text{DEQ}(f_{\theta^{[1]}}; \mathbf{x}_{1:T}))$.
  (In other words, stacking multiple DEQs does **not** create extra representational power over a single DEQ.)

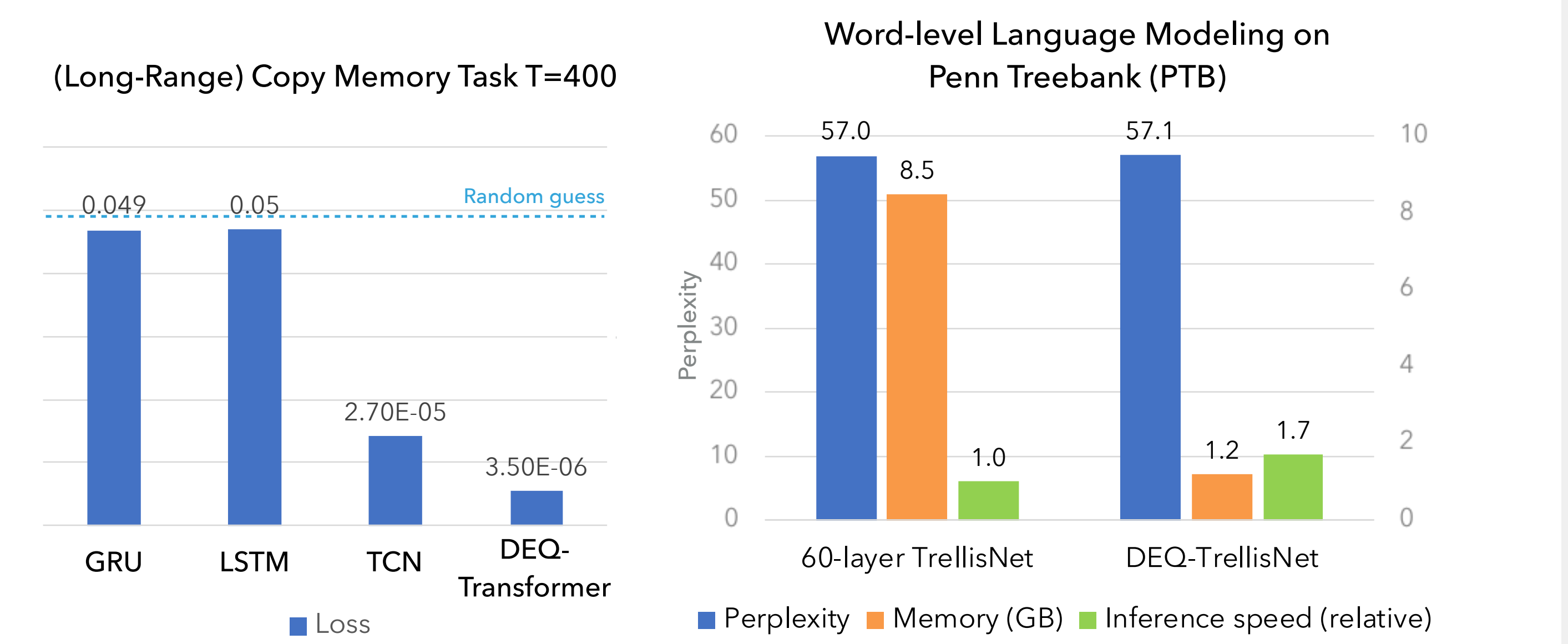
Instantiations of DEQ

We highlight two instantiations of f_θ based on very different SOTA sequence models.

- DEQ-TrellisNet [2] (weight-tied temporal convolutions that also generalize RNNs).
- DEQ-Transformer [5] (weight-tied multi-head self-attention).

Experiments

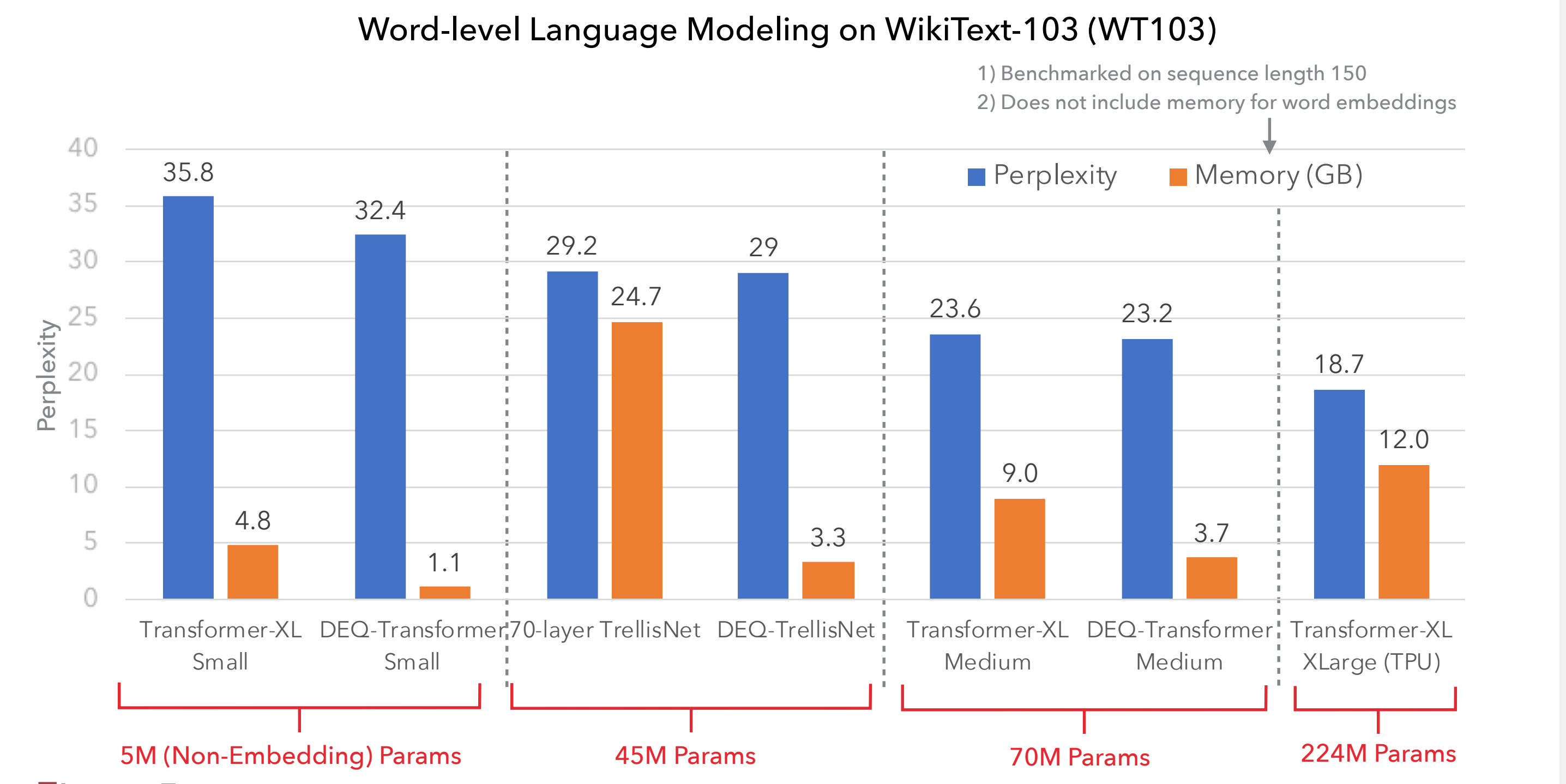
- We test both instantiations of DEQ on both synthetic (long-range copy memory test) and realistic, large-scale and high-dimensional (word-level language modeling on PTB and WT103) sequence benchmarks.



Model	Loss
GRU	0.049
LSTM	0.05
TCN	2.70E-05
DEQ-Transformer	3.50E-06

Model	Perplexity	Memory (GB)	Inference speed (relative)
60-layer TrellisNet	57.0	8.5	1.0
DEQ-TrellisNet	57.1	1.2	1.7

Figure 4: Left: DEQ demonstrates good memory retention over relatively long sequences. Right: DEQ achieves competitive performance on PTB corpus with > 80% reduction in memory cost.



Model	Params	Perplexity	Memory (GB)
Transformer-XL Small	5M (Non-Embedding) Params	35.8	4.8
DEQ-Transformer Small	5M (Non-Embedding) Params	32.4	1.1
70-layer TrellisNet	45M Params	29.2	24.7
DEQ-TrellisNet	45M Params	29	3.3
Transformer-XL Medium	70M Params	23.6	9.0
DEQ-Transformer Medium	70M Params	23.2	3.7
Transformer-XL XLarge (TPU)	224M Params	18.7	12.0

Figure 5: DEQ significantly reduces memory while improving the results on the large-scale WikiText-103 dataset.

- Empirically, the DEQ models achieve results on par with (or better than) the SOTA, while paying only 10-30% of the memory cost.

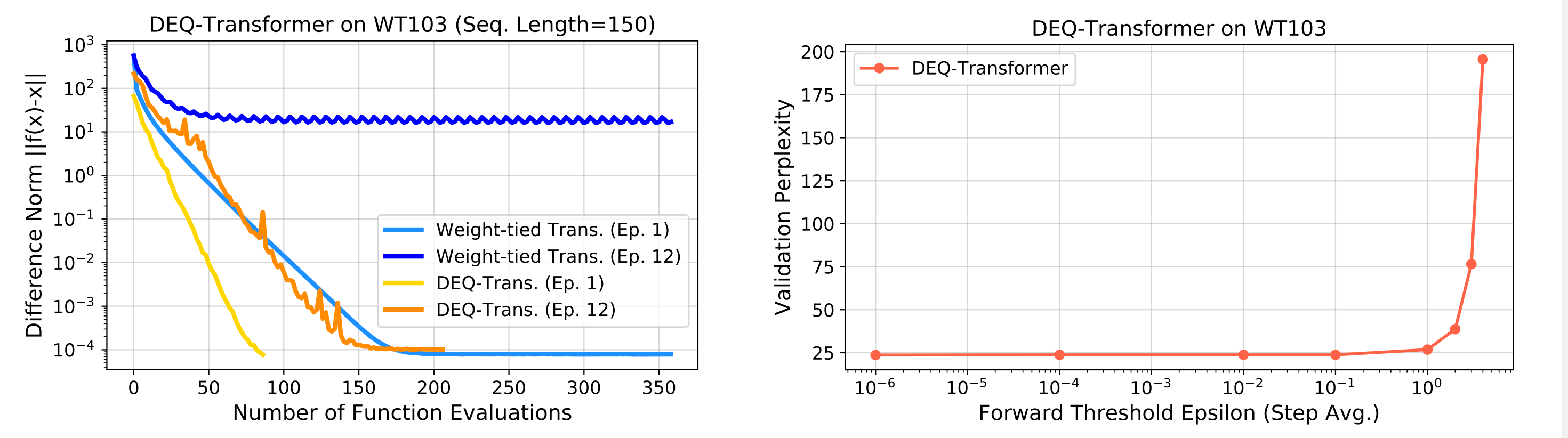


Figure 6: Left: DEQ-Transformer finds the equilibrium in a stable and efficient manner (whereas deep weight-tying could oscillate around the fixed point). Right: DEQ can be accelerated by leveraging higher tolerance ε without hurting the performance.

- Our current models represent the largest-scale practical application of implicit layers in deep learning of which we are aware.

[1] Luis B Almeida. A learning rule for asynchronous perceptrons with feedback in a combinatorial environment. In *Artificial Neural Networks*. 1990.
[2] Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. Trellis networks for sequence modeling. In *International Conference on Learning Representations (ICLR)*, 2019.
[3] Tian Qi Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. In *Neural Information Processing Systems*, 2018.
[4] Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. Training deep nets with sublinear memory cost. *arXiv:1604.06174*, 2016.
[5] Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. Universal transformers. *International Conference on Learning Representations (ICLR)*, 2019.
[6] Aidan N Gomez, Mengye Ren, Raquel Urtasun, and Roger B Grosse. The reversible residual network: Backpropagation without storing activations. In *Neural Information Processing Systems*, 2017.
[7] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.