A. Appendix

A.1. Proofs

Proposition 3. *Consider the ODE in Eq. 5 and assumptions on* $\mathbf{A}$ *described above. Suppose we have* $\|U\| < \alpha/2$, *and* $\phi(\cdot)$ *is 1-Lipshitz function, it follows that, for any given,* $\boldsymbol{\beta}$, $\mathbf{u}_m$, *an equilibrium point exists and is unique.*

To prove the proposition, we must find a solution to the non-linear equation $\mathbf{Az} + \mathbf{Bu}_m + \phi(\mathbf{Uz} + \mathbf{Wu}_m) = 0$ and show that it is unique. We do this by constructing a fixed-point iterate, and show that the iteration is contractive.

To this end, define $\Gamma(\mathbf{z}) = \mathbf{z} + \eta(\mathbf{Az} + \mathbf{Bu}_m + \phi(\mathbf{Uz} + \mathbf{Wu}_m))$, and note that for any two $\mathbf{z}, \mathbf{z}' \in \mathbb{R}^D$, we find,

$$\begin{aligned}
\|\Gamma(\mathbf{z}) - \Gamma(\mathbf{z}')\| &\leq \|(I + \eta\mathbf{A})(\mathbf{z} - \mathbf{z}')\| + \eta\|\phi(\mathbf{Uz} + \mathbf{Wu}_m)) - \phi(\mathbf{Uz}' + \mathbf{Wu}_m))\| \\
&\leq \|(I + \eta\mathbf{A})(\mathbf{z} - \mathbf{z}')\| + \eta\|\mathbf{U}(\mathbf{z} - \mathbf{z}')\| \\
&\leq \sigma_{\max}(I + \eta\mathbf{A})\|(\mathbf{z} - \mathbf{z}')\| + \eta\|\mathbf{U}(\mathbf{z} - \mathbf{z}')\| \\
&\leq \sigma_{\max}(I + \eta\mathbf{A})\|\mathbf{z} - \mathbf{z}'\| + \eta\|\mathbf{U}\|\|\mathbf{z} - \mathbf{z}'\| \\
\implies \|\Gamma(\mathbf{z}) - \Gamma(\mathbf{z}')\| &\leq \Big(\sigma_{\max}(I + \eta\mathbf{A}) + \eta\|\mathbf{U}\|\Big)\|\mathbf{z} - \mathbf{z}'\| = \gamma\|\mathbf{z} - \mathbf{z}'\| \qquad (7)
\end{aligned}$$

If the constant $\gamma < 1$, then the above inequality proves that Γ is a contraction. The result then follows by invoking the Banach fixed point theorem (contraction-mapping theorem). All that remains to show is that $\gamma = \sigma_{\max}(I + \eta\mathbf{A}) + \eta\|\mathbf{U}\| < 1$. From assumptions we have, $\sigma_{\max}(I + \eta\mathbf{A}) \leq 1 - \alpha\eta$ and $\|\mathbf{U}\| < \alpha$, where $\alpha > 0$; $\implies \gamma < 1 - \alpha\eta + \alpha\eta \leq 1$.

Proposition 4. *With the setup in Proposition 1, and regardless of* $\boldsymbol{\beta}$, *the equilibrium point is globally asymptotically stable, and the discrete Euler recursion converges to the equilibrium solution at a linear rate.*

Let $\mathbf{z}^*$ be the equilibrium solution, i.e. $\mathbf{Az}^* + \mathbf{Bu}_m + \phi(\mathbf{Uz}^* + \mathbf{Wu}_m) = 0$. We consider the Lyapunov function $V(\mathbf{z}(t)) = \|\mathbf{z}(t) - \mathbf{z}^*\|^2$ and show that it is monotonically decreasing along the ODE system trajectories. Observe that, as per our setup, components where $(\boldsymbol{\beta})_j = 0$ does not pose a problem, because those states remain frozen, and serve as an additional exogenous input in our ODE. Consequently, we can assume without loss of generality that $(\boldsymbol{\beta})_j = 1$ for all $j \in [D]$. The gradient of the Lyapunov function along the ODE system trajectories can be written as

$$\begin{aligned}
\frac{dV(z(t))}{dt} &= (\dot{\mathbf{z}}(t))^\top(\mathbf{z}(t) - \mathbf{z}^*) + (\mathbf{z}(t) - \mathbf{z}^*)^\top\dot{\mathbf{z}}(t) \\
&= (\mathbf{Az}(t) + \mathbf{Bu}_m + \phi(\mathbf{Uz}(t) + \mathbf{Wu}_m))^\top(\mathbf{z}(t) - \mathbf{z}^*) \\
&+ (\mathbf{z}(t) - \mathbf{z}^*)^\top(\mathbf{Az}(t) + \mathbf{Bu}_m + \phi(\mathbf{Uz}(t) + \mathbf{Wu}_m)) \\
&= (\mathbf{A}(\mathbf{z}(t) - \mathbf{z}^*) + \mathbf{B}(\mathbf{u}_m - \mathbf{u}_m) + \phi(\mathbf{Uz}(t) + \mathbf{Wx}_m) - \phi(\mathbf{Uz}^* + \mathbf{Wu}_m))^\top(\mathbf{z}(t) - \mathbf{z}^*) \\
&\quad + (\mathbf{z}(t) - \mathbf{z}^*)^\top(\mathbf{A}(\mathbf{z}(t) - \mathbf{z}^*) + \mathbf{B}(\mathbf{u}_m - \mathbf{u}_m) + \phi(\mathbf{Uz}(t) + \mathbf{Wx}_m) - \phi(\mathbf{Uz}^* + \mathbf{Wu}_m)) \\
&= (\mathbf{A}(\mathbf{z}(t) - \mathbf{z}^*) + \phi(\mathbf{Uz}(t) + \mathbf{Wx}_m) - \phi(\mathbf{Uz}^* + \mathbf{Wu}_m))^\top(\mathbf{z}(t) - \mathbf{z}^*) \\
&\quad + (\mathbf{z}(t) - \mathbf{z}^*)^\top(\mathbf{A}(\mathbf{z}(t) - \mathbf{z}^*) + \phi(\mathbf{Uz}(t) + \mathbf{Wx}_m) - \phi(\mathbf{Uz}^* + \mathbf{Wu}_m)) \\
&= (\mathbf{z}(t) - \mathbf{z}^*)^\top(\mathbf{A} + \mathbf{A}^\top)(\mathbf{z}(t) - \mathbf{z}^*) + 2(\phi(\mathbf{Uz}(t) + \mathbf{Wu}_m) - \phi(\mathbf{Uz}^* + \mathbf{Wu}_m))^\top(\mathbf{z}(t) - \mathbf{z}^*)
\end{aligned}$$

We now invoke Cauchy-Schwartz inequality to bound the second term, namely,

$$\begin{aligned}
|(\phi(\mathbf{Uz}(t) + \mathbf{Wu}_m) - \phi(\mathbf{Uz}^* + \mathbf{Wu}_m))^\top(\mathbf{z}(t) - \mathbf{z}^*)| &\leq \|(\phi(\mathbf{Uz}(t) + \mathbf{Wu}_m) - \phi(\mathbf{Uz}^* + \mathbf{Wu}_m))\|\|\mathbf{z}(t) - \mathbf{z}^*\| \\
&\leq \|\mathbf{U}\|\|\mathbf{z}(t) - \mathbf{z}^*\|\|\mathbf{z}(t) - \mathbf{z}^*\| < \|\mathbf{z}(t) - \mathbf{z}^*\|^2
\end{aligned}$$

where in the last inequality we used the fact that $\phi(\cdot)$ is 1-Lipshitz and $\|\mathbf{U}\| < \alpha \leq 1$. As a result, we have,

$$\frac{dV(z(t))}{dt} < (\lambda_{\max}(\mathbf{A} + \mathbf{A}^T) + 1)\|\mathbf{z} - \mathbf{z}^*\|^2 \leq 0$$

where the last inequality follows because, we have $(\lambda_{\max}(\mathbf{A} + \mathbf{A}^T) \leq -1)$. This shows that the ODE is globally asymptotically stable and converges to a unique equilibrium point. To show a linear rate of convergence we note that K-fold iterations of the Euler method (see Prop 1), $\mathbf{z}^k = \Gamma(\mathbf{z}^{k-1}) = \mathbf{z}^{k-1} + \eta(\mathbf{Az}^{k-1} + \mathbf{Bu}_m + \phi(\mathbf{Uz}^{k-1} + \mathbf{Wu}_m))$, results in,

$$\|\mathbf{z}^K - \mathbf{z}^*\| \leq \gamma^K\|\mathbf{z}^0 - \mathbf{z}^*\|$$

which follows directly from the fact that $\mathbf{z}^K = \Gamma(\mathbf{z}^{K-1})$, $\mathbf{z}^* = \Gamma(\mathbf{z}^*)$, and Γ is a contraction as obtained by Eq. 7. This establishes the linear-rate of convergence.

Proof of Theorem 1

Note that, when $\beta_i = 0$, $\mathbf{s}_m(i) = \mathbf{s}_{m-1}(i)$. On the other hand when $\beta_i > 0$, the system is in equilibrium, and for those components, j, we have

$$(\dot{\mathbf{z}}(t))_j = (F(\mathbf{z}(t), \mathbf{u}_m))_j = 0, \text{ where } F(\mathbf{z}(t), \mathbf{u}_m) = \beta(\mathbf{u}_m) \circ (\mathbf{A}\mathbf{z}(t) + \mathbf{B}\mathbf{u}_m + \phi(\mathbf{U}\mathbf{z}(t) + \mathbf{W}\mathbf{u}_m))$$

Now $(F(\mathbf{s}_m, \mathbf{u}_m))_k = 0$ regardless of β_k. This is because if $\beta_k(\mathbf{u}_m) > 0$ we reach equilibrium, and $\dot{\mathbf{z}}(t) = 0$, and on the other have if $\beta_k = 0$ then $(F(\mathbf{s}_m, \mathbf{u}_m))_k = 0$ in any case. With this in mind, define $D = \text{diag}[\mathbf{1}_{\beta_j(\mathbf{u}_m)>0}]$. We then write the vector $\mathbf{s}_m = D\mathbf{s}_m + (I - D)\mathbf{s}_{m-1}$. Let $J_{m,m-1}$ denote the Jacobian of $\mathbf{s}_m$ with respect to s_{m-1}. Taking derivatives we get,

$$\begin{aligned} 0 = \nabla F(\mathbf{s}_m, \mathbf{u}_m) =& \beta(\mathbf{u}_m) \circ (\mathbf{A}(DJ_{m,m-1} + (I - D)) + \mathbf{B}_2) \\ &+ \beta(\mathbf{u}_m) \circ (\nabla\phi(\mathbf{U}(DJ_{m,m-1} + (I - D)) + \mathbf{W}_2)) \\ &+ D\nabla\sigma(\mathbf{U}_s\mathbf{s}_{m-1} + \mathbf{W}_x\mathbf{x}_m)(\mathbf{A}\mathbf{s}_m + \mathbf{B}\mathbf{u}_m + \phi(\mathbf{U}\mathbf{s}_m + \mathbf{W}\mathbf{u}_m)) \end{aligned}$$

First, note that the third term is always zero, due to the fact we noted earlier, namely, if a component is active, then the corresponding state reaches equilibrium, and there is nothing to do if the component is otherwise inactive. Now noting that $\mathbf{A} = \mathbf{B}_2$ and $\mathbf{U} = \mathbf{W}_2$, we get,

$$\nabla F(\mathbf{s}_m, \mathbf{u}_m) = \beta(\mathbf{u}_m) \circ (\mathbf{A}D)(J_{m,m-1} - I) + \nabla\phi(\cdot)\mathbf{U}D(J_{m,m-1} - I)$$

Collecting the common terms, we have,

$$\nabla F(\mathbf{s}_m, \mathbf{u}_m) = \beta(\mathbf{u}_m) \circ (\mathbf{A} - \nabla\phi U)D(J_{m,m-1} - I)$$

Now for the case in hand, $\|\nabla\phi U\| < 1$, and since $\|\mathbf{A}\| \geq 1$, the middle term is non-zero. This implies that for all the active components, $(J_{m,m-1})_{kk} = 1$.

For the other case, the proof follows in an identical manner. Specifically, for the non-zero rows of $\mathbf{B}$ the proof is identical, and the claims hold for those associated state components. For the rows with zero rows since,

A.2. Implementation Details

We acquired the publicly available code for the baselines except Antisymmetric RNN [10] and Incremental RNN[26]. We write the RNN cell implemntation for Antisymmetric RNN and Incremental RNNs from the pseudo code provided in their papers. Before running our grid search, we ensured that we were able to reproduce the publicly reported results. Following which we run our experiments for suggested hidden states as per the previous works for each dataset.

In order to avoid non-determinism in the experiments, we initialize both the numpy and tensorflow random library with the same seed number, 1234. Our parameter matrices are initialized with a random normal initializer with mean 0 and standard deviation 0.1 while our time-constant biases are initialized with -3.0 and remaining biases are initialized with 0.

We provide the pseudo code in Algorithm 1 to generate the hidden states of the TARNN . In order to implement this routine on a deep learning framework, we need to elaborate a bit more about the ODESolve function. We implement the Euler iterations described in the practical implementations in the method section. Following the recommendation from [26] and the fact that many of these datasets are slowly time varying, we use the $K = 5$ in the Euler recursions to reach the equilibrium. Table 5 provides the number of hidden units used for different datasets.

Our experiments use hidden size as suggested by [31, 10] i.e. 128. We point out that this is not the setting used by [27, 34] as their best results are achieved with much larger state space i.e. 512 state dimension, thus requiring much larger models. Thus, in order to provide fair comparison we only allow state space as 128 dimensions.

In order to enable grid search on the baseline methods, we use the method specific hyper-parameter values suggested in the respective baselines. We allow the methods to pick the non-linearity from the set $\{$ ReLU, tanh, sigmoid $\}$. For Antisymmetric RNN, as per their recommendation we step size from the set $\{0.01, 0.1, 1\}$ and diffusion parameter $\gamma \in \{0.001.0.01, 0.1, 1.0\}$. For nnRNN and expRNN methods, we follow the hyper-parameter search grid as suggested in [27].

We use grid search for tuning the hyper-parameters for the methods. We used the values $[4.5E-6, 4.5E-5, 4.5E-4, 1E-6, 1E-5, 1E-4]$ for L2 regularization. We searched over $[1e-2, 1e-3, 1e-4]$ as the base learning rates which are halved after each $\tau = [5, 10, 20]$ epochs have passed. We allowed the methods to train for $[30, 50, 100, 300]$ epochs. We

Table 5: Various hyper-parameters to reproduce results

Dataset	Hidden Dimension (hr)	Learning Rate (hr)	L2 regularization	Init η	Epochs τ		Batch Size
Pixel-MNIST	128	$1e^{-2}$	$4.5e^{-6}$	0.08	30	5	128
Permuted-MNIST	128	$1e^{-2}$	$4.5e^{-6}$	0.0008	30	5	128
Noisy-MNIST	128	$1e^{-2}$	$4.5e^{-5}$	0.0008	30	5	512
Noisy-CIFAR	128	$1e^{-2}$	$4.5e^{-5}$	0.001	30	5	256
Addition Task	128	$1e^{-2}$	$1.0e^{-5}$	0.001	2	-	128
Copying Task	128	$1e^{-2}$	$1.0e^{-6}$	0.45	-	-	128
PTB	256	-	-	0.001	100	-	

use ReLU as the non-linearity for all of our experiments except in Copy and PTB tasks where we use tanh as the non-linearity (performs better than ReLU).

We point out that we set $\mathbf{A} = -I$ for all our experiments except Pixel-MNIST/Permute-MNIST tasks where we use $\mathbf{A}$ to be the blocked triangular identity matrix as mentioned in the analysis Section 3.1. This allows us to couple the linear part resulting in better performance on these tasks in comparison to the $\mathbf{A} = -I$ configuration.

Note that the settings used for PTB dataset corresponds to the small configuration with 300 as the sequence length. We piggy back on the configuration changes used in [31, 26, 53] which describes the learning rate along with the learning rate schedule and the number of epochs all the methods are trained. Thus, we do not list these hyper-parameters in the table 5.

A.3. Unitary RNNs do not solve vanishing gradients.

[34, 27] and others propose to "cheaply" design orthonormal transition matrices (OTM), appealing to [2] for justification. [2] (Eq. 4) only shows an upper-bound with ReLU + OTM. This solves exploding gradients, but the more pernicious vanishing gradients remains (RELU+OTM is discussed in [44] [PSG17]). In [2]'s notation with D_k binary diagonal arising from ReLU activations, W unitary, we would need, $\|\partial C/\partial h_T(\prod_{s=t}^{T-1} D_s W^\top)\| \geq \|\partial C/\partial h_T\|$. This is generally not true due to matrix non-commutativity. E.g. for $t = T-2$, this is possible if $\|D_{T-1}W^\top D_{T-2}W^\top\| = \|D_{T-1}W^\top D_{T-2}\| \geq 1$. Unless, $D_{T-1} = D_{T-2}$ is identity, $D_{T-1}WD_{T-2}$ is a submatrix of W, and generically has norm less than one.

A.4. Relationship to existing Recurrent architectures.

We will now briefly discuss other recurrent architectures in the literature to gain intuition into our framework. We will refer to the ODE Eq. 5

(a) Vanilla RNNs: Setting $\beta = 1$, $\mathbf{A} = -\mathbf{I}$, $\mathbf{B}^1 = \mathbf{0}; \mathbf{B}^2 = \mathbf{0}$, results in the ODE, $\dot{\mathbf{z}}(t) = -\mathbf{z}(t) + \phi(\mathbf{U}\mathbf{z}(t) + \mathbf{W}\mathbf{x}_m)$; $\mathbf{z}(t_0) = \mathbf{s}_{m-1}$. Euler discretization of this ODE with only one step results in Vanilla RNNs.

(b) Fast/Antisymmetric RNNs: Setting $\beta = 1$, $\mathbf{A} = \mathbf{0}$, $\mathbf{B}^1 = \mathbf{0}; \mathbf{B}^2 = \mathbf{0}$, results in the ODE, $\dot{\mathbf{z}}(t) = \phi(\mathbf{U}\mathbf{z}(t) + \mathbf{W}\mathbf{x}_m)$; $\mathbf{z}(t_0) = \mathbf{s}_{m-1}$. Euler discretization of this ODE with only one step results in [31, 10].

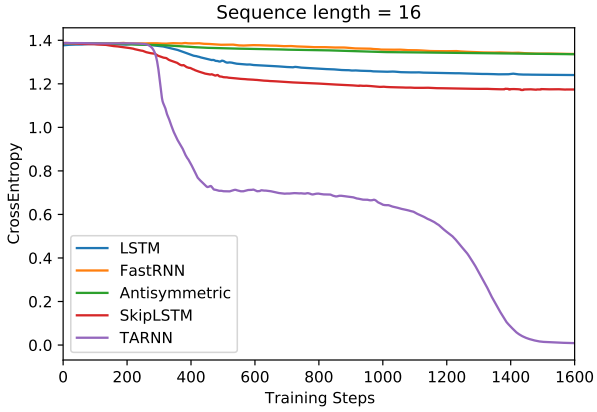
(c) Incremental RNNs: Setting $\beta = 1$, $\mathbf{A} = -\mathbf{I}$, $\mathbf{B}^1 = \mathbf{0}; \mathbf{B}^2 = \mathbf{I}$, results in the ODE, $\dot{\mathbf{z}}(t) = -\mathbf{z}(t) + \phi(\mathbf{U}\mathbf{z}(t) + \mathbf{W}\mathbf{x}_m)$; $\mathbf{z}(t_0) = \mathbf{s}_{m-1}$. Since the initial state of the ODE, $\mathbf{z}(t_0) = \mathbf{s}_{m-1}$, we can write it into $\dot{\mathbf{z}}(t) = -(\mathbf{z}(t) - \mathbf{s}_{m-1}) + \phi(\mathbf{U}(\mathbf{z}(t) - \mathbf{s}_{m-1}) + \mathbf{W}\mathbf{x}_m)$ with $\mathbf{z}(t_0) = 0$. This ODE is equivalent to [26].

A.5. Additional plots for Toy Example.

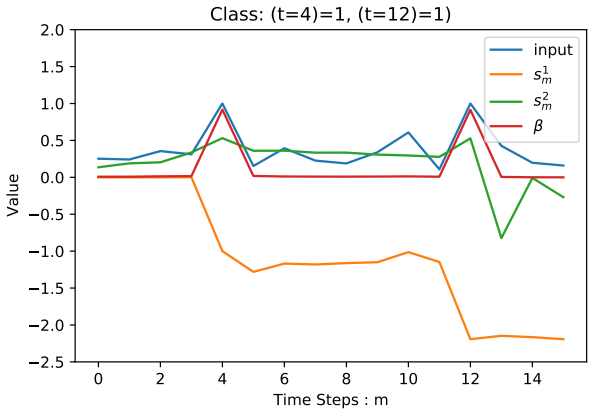
We add additional figures for the toy example in order to describe the following properties: (a) TARNN achieves faster convergence than the baselines, (b) TARNN time constants activate at the correct locations where the markers are placed and hence we get the hidden state transitions at these locations, and finally (c) we plot a the hidden state norms in order to demonstrate that SkipLSTM does focus at the input markers while TARNN ends up changing the hidden states at these locations.

A.6. Toy Example with larger state space.

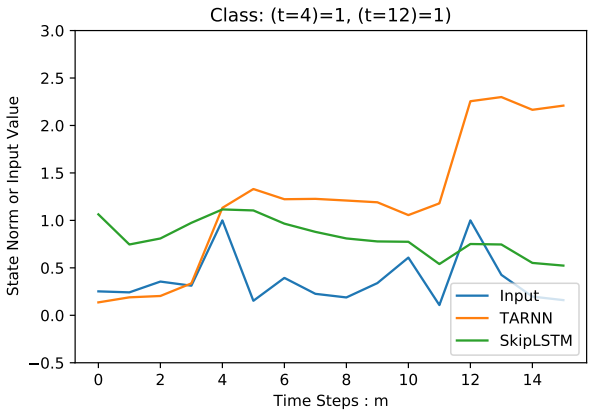
In our main experiments with Toy example we have used a very small state space to demonstrate that TARNN outperforms the baselines even with such small state space. Note that forcing training and inference on a small state dimension leads to a



(a)



(b)



(c)

Figure 3: Toy Example. (a) TARNN converges quickly to the 0.0 cross-entropy error. (b) shows time constant β along with the input, at locations $t = 4, 12$, both the input and time constants are in sync resulting in the state update while everywhere else the time constant does not allow the state to update (see s_m^1 state which captures the update or skip state part). (c) shows the norm of the hidden state for SkipLSTM and TARNN .

difficult problem. This is because in our setup we have 16-length real-valued input traces. As such a large state dimension could in principle commit the entire input trace to memory, and good performance would not be surprising (indeed as Table 6 shows). The purpose of our example was to motivate our key intuition (a) that trainability of an RNN is limited by vanishing/exploding gradients; and (b) Sequential data consists of uninformative/noisy data segments, which if not suppressed can lead to performance degradation. See Caption of Fig.1 in the main text paper. For good performance we need both lossless hidden-state gradients (for informative input segments), and skipping (uninformative inputs).

Table 6: Toy Example: Accuracy for various hidden state sizes.

Algorithm	Hidden Dimension					
	2	4	8	16	32	64
FastRNN	45	47	52	69	82	96
Antisymmetric	37	39	41	59	73	90
LSTM	45	54	67	82	96	100
SkipLSTM	60	66	72	91	98	100
TARNN	100	100	100	100	100	100

A.7. Gradient Norm Plot for Add-Task.

TARNN works exactly as in the toy example, on other datasets as well. As evidence, we plot gradient norm for the add-task in Figure 4, and as expected TARNN is able to better maintain gradient norms near unity. The plots for other datasets follow a similar trend.

A.8. Google-30, HAR-2 datasets

In order to verify that our method works well for IoT tasks, we use popular datasets from previous works ([31]). These datasets primarily focus on detecting activitiy embedded in a longer sequence. We pick two datasets namely: (a) HAR-2 [1], *i.e.* Human Activity Recognition from an accelerometer and gyroscope on a Samsung Galaxy S3 smartphone, and (b) Google-30 [50], *i.e.* detection of utterances of 30 commands plus background noise and silence. For these tasks, light footprint of the model also becomes extremely important given that these models are deployed on resource constrained IoT devices.

Table 7 shows accuracy, model size, training time, inference time, and the number of parameters. TARNN beats the

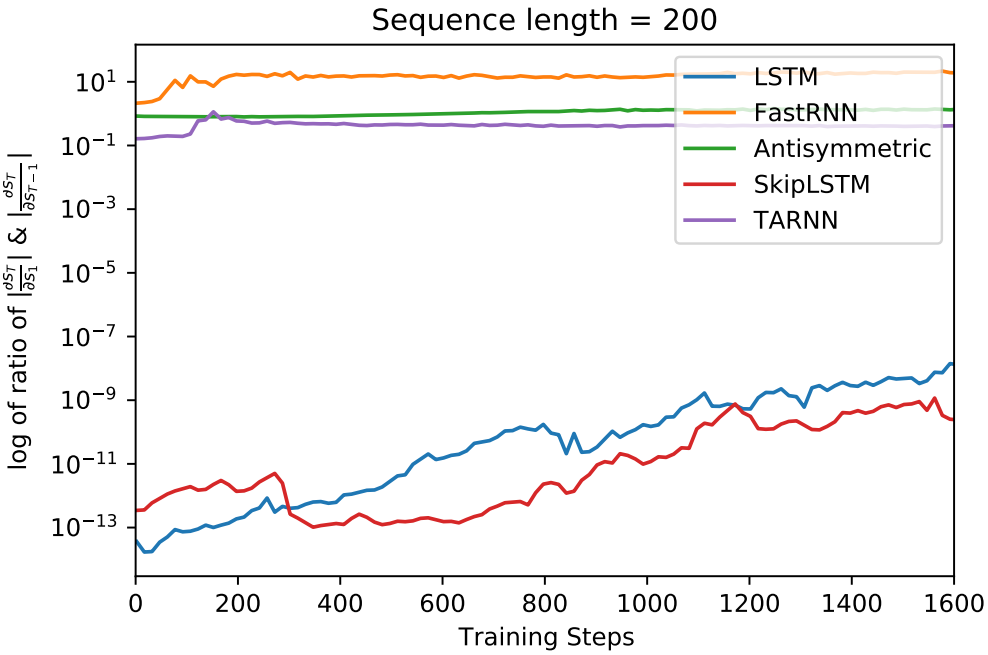


Figure 4: Add Task Gradient Norm for 200 length sequences.

baselines in terms of test accuracy. TARNN has smaller model size, while its inference time comparable to iRNN and hence well suited for IoT tasks.

Table 7: Results for Activity Recoginition (IoT) Datasets.

Data set	Algorithm	Accuracy (%)	Model Size (KB)	Train Time (hr)	Test Time (ms)	#Params
HAR-2	FastRNN	94.50	29	0.063	**0.01**	7.5k
	LSTM	93.65	74	0.183	0.04	16k
	Antisymmetric	93.15	29	0.087	**0.01**	7.5k
	iRNN	96.30	18	**0.018**	0.03	4k
	TARNN	**96.59**	**17**	0.03	0.02	**3.7k**
Google-30	FastRNN	91.60	96	1.30	**0.01**	18k
	LSTM	90.31	219	2.63	0.05	41k
	Antisymmetric	90.91	64	0.54	**0.01**	12k
	iRNN	94.23	45	0.44	0.05	**8.5k**
	TARNN	**94.93**	**20**	**0.38**	0.01	9k

A.9. Inference time

As the table 7 shows that the inference time for TARNN is similar to FastRNN and about at least one-half of the inference time for the LSTMs.

A.10. Impact of larger K on the results

Our choice of K is inspired by previous ODE discretization works [26]. Small K suffices for many datasets because inputs are slowly varying (small drift). Furthermore, our dynamical system is exponentially stable (Proposition 2), allowing for rapid convergence to equilibrium. As shown in Table 8, larger values of K lead to increased inference time, and there is an inherent tradeoff between seeking exact equilibria and inference time. We will elaborate this in the revision.

Table 8: PTB Language Modeling: Larger K values.

Algorithm	Hidden Dimension	K	Test Perplexity	Train Time (min)	Inference Time (ms)
TARNN	128	1	104.15	27	1
TARNN	128	3	102.42	40	1.7
TARNN	128	5	101.21	65	3.2
TARNN	128	7	101.01	91	5.3
TARNN	128	10	100.91	123	8.1