Stabilizing Gradients for Deep Neural Networks via Efficient SVD Parameterization

Jiong Zhang[1], Qi Lei[1] and Inderjit S. Dhillon[12]
[1]University of Texas at Austin. [2]Amazon/A9

Abstract

- Recurrent Neural Networks (RNN) has been widely used in different areas.
  - Handwritten (speech) recognition
  - Translation
  - Question answering
  - Image captioning
- RNN suffers from problems with long time dependency, esp. issues of gradient vanishing & exploding
- We propose an efficient parametrization of the transition matrix in RNN that
  - allows explicit control over its singular values to eliminate/reduce the gradient exploding/vanishing problem
  - loses no expressive power
  - share the similar time complexity as vanilla RNN
  - theoretically and empirically has better generalization and is easier to train

RNN with Vanishing/Exploding Gradient

- RNN with activation δ computes the next hidden state $h^{(t)}$ and output vector $o^{(t)} \in \mathbb{R}^{n_o}$ as:

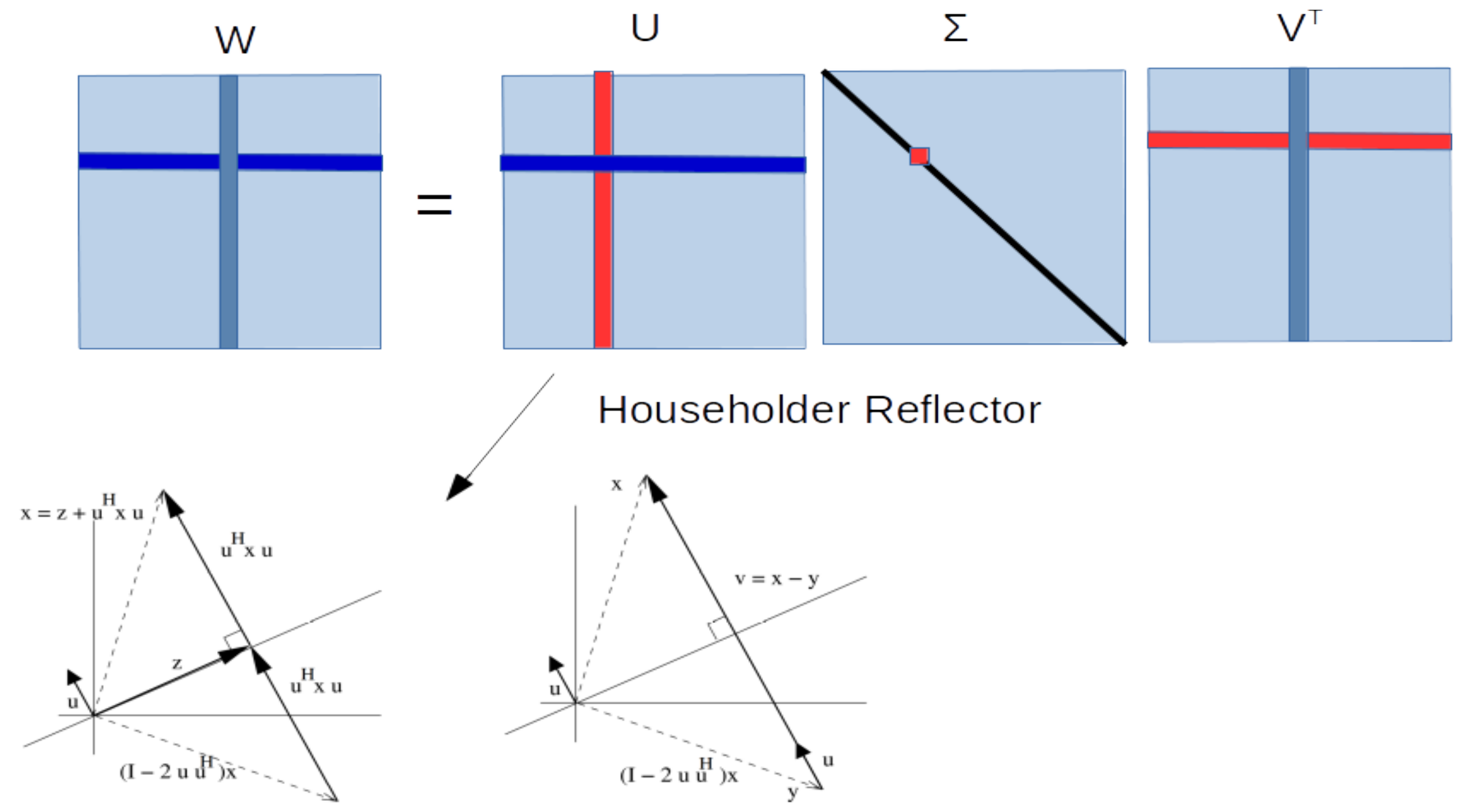
$$\begin{aligned} h^{(t)} &= \delta(Wh^{(t-1)} + Mx^{(t-1)} + b) \qquad (1) \\ o^{(t)} &= Yh^{(t)} \end{aligned}$$

- Gradient for activations becomes:

$$\frac{\partial h^{(t)}}{\partial h^{(t-k)}} = \prod_{t\geq i\geq t-k} \frac{\partial h^{(i)}}{\partial h^{(i-1)}} = \prod_{t\geq i\geq t-k} W^\top \mathrm{diag}(\delta'(h^{(i-1)}))$$

- Spectral norm $\|W\|_2 > 1 \Longrightarrow$ Gradient will explode when t is large
- Spectral norm $\|W\|_2 < 1 \Longrightarrow$ Gradient will vanish when t is large

Preliminary: SVD Parameterization



$$U \Longrightarrow \begin{pmatrix} I_n - 2\frac{u_n u_n^\top}{\|u_n\|^2} \end{pmatrix} \cdot \begin{pmatrix} 1 & \\ & I_{n-1} - 2\frac{u_{n-1}u_{n-1}^\top}{\|u_{n-1}\|^2} \end{pmatrix} \cdots \begin{pmatrix} I_{n-k_1} & \\ & I_{k_1} - 2\frac{u_{k_1}u_{k_1}^\top}{\|u_{k_1}\|^2} \end{pmatrix}$$

- Singular value decomposition on W: maintain the transition matrix in its SVD form, i.e.

$$W = U\Sigma V^\top$$

- Further parameterization with products of Householder reflectors on U, V [1]:
  - For $u \in \mathbb{R}^k, k \leq n$, Householder reflector $\mathcal{H}_k^n(u)$ is defined as:

$$\mathcal{H}_k^n(u) = \begin{cases} \begin{pmatrix} I_{n-k} & \\ & I_k - 2\frac{uu^\top}{\|u\|^2} \end{pmatrix}, & u \neq \mathbf{0} \\ I_n, & \text{otherwise.} \end{cases}$$

  - $U \leftarrow \prod_{k=n-m_1+1}^{n} \mathcal{H}_k^n(u_k)$, and $V \leftarrow \prod_{k=n-m_2+1}^{n} \mathcal{H}_k^n(v_k)$

Our Proposal: svdRNN

- Expressive power of SVD Parameterization:
  - With $\sigma \in \mathbb{R}^n$ and $\{u_i\}_{i=k_1}^n, \{v_i\}_{i=k_2}^n, u_i, v_i \in \mathbb{R}^i$, we define the proposed SVD parametrization:

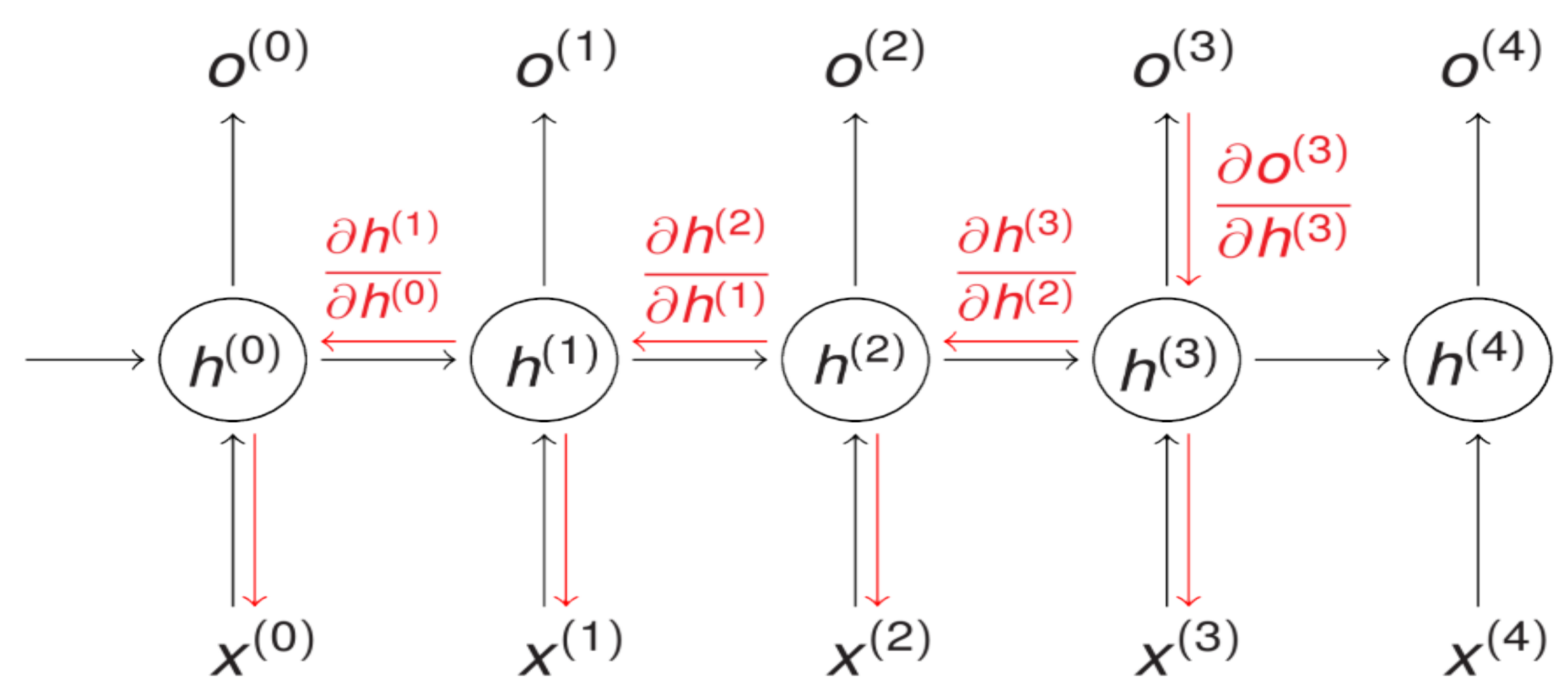
$$\begin{aligned} \mathcal{M}_{k_1,k_2} :& \mathbb{R}^{k_1} \times \ldots \times \mathbb{R}^n \times \mathbb{R}^{k_2} \times \ldots \times \mathbb{R}^n \times \mathbb{R}^n \mapsto \mathbb{R}^{n\times n} \\ & \{u_i\}_{i=k_1}^n, \{v_i\}_{i=k_2}^n, \sigma \mapsto \\ & \underbrace{\mathcal{H}_n(u_n)\ldots\mathcal{H}_{k_1}(u_{k_1})}_{U} \underbrace{diag(\sigma)}_{\Sigma} \underbrace{\mathcal{H}_{k_2}(v_{k_2})\ldots\mathcal{H}_n(v_n)}_{V^\top}. \qquad (2) \end{aligned}$$

  - **Explicit singular values**: $\mathcal{M}_{k_1,k_2}(\{u_i\}_{i=k_1}^n, \{v_i\}_{i=k_2}^n, \sigma)$ is a $n \times n$ real matrix with singular values σ.
  - **Full expressivity**: The image of $\mathcal{M}_{1,1}$ is the set of $n \times n$ real matrices.
  - **Orthogonal expressivity**: The image of $\mathcal{M}_{k_1,k_2}$ covers the set of $n \times n$ orthogonal matrices if $k_1 + k_2 \leq n + 2$.
- In svdRNN we parametrize the transition matrix $W \in \mathbb{R}^{n\times n}$ in Eqn. (1) using $m_1 + m_2$ Householder reflectors as:

$$\begin{aligned} W &= \mathcal{M}_{n-m_1+1,n-m_2+1}(u_{n-m_1+1}, \ldots, u_n, v_{n-m_2+1}, \ldots, v_n, \sigma) \\ &= \underbrace{\mathcal{H}_n(u_n)\ldots\mathcal{H}_{n-m_1+1}(u_{n-m_1+1})}_{U} \underbrace{diag(\sigma)}_{\Sigma} \underbrace{\mathcal{H}_{n-m_2+1}(v_{n-m_2+1})\ldots\mathcal{H}_n(v_n)}_{V^\top} \end{aligned}$$

- Update and control σ: $\sigma_i = 2r(sigmoid(\hat{\sigma}_i) - 0.5) + 1, \ i \in [n]$

Training Algorithm


$o^{(0)}$
$o^{(1)}$
$o^{(2)}$
$o^{(3)}$
$o^{(4)}$
$\frac{\partial o^{(3)}}{\partial h^{(3)}}$
$\frac{\partial h^{(1)}}{\partial h^{(0)}}$
$\frac{\partial h^{(2)}}{\partial h^{(1)}}$
$\frac{\partial h^{(3)}}{\partial h^{(2)}}$
$h^{(0)}$
$h^{(1)}$
$h^{(2)}$
$h^{(3)}$
$h^{(4)}$
$x^{(0)}$
$x^{(1)}$
$x^{(2)}$
$x^{(3)}$
$x^{(4)}$

- Forward propagation:
  The only different aspect from a regular RNN in the forward propagation is the computation of Wh. In our case, to evaluate:

$$\begin{aligned} Wh =& \mathcal{H}_n(u_n)\ldots\mathcal{H}_{n-m_1+1}(u_{n-m_1+1})diag(\sigma) \\ & \mathcal{H}_{n-m_2+1}(v_{n-m_2+1})\ldots\mathcal{H}_n(v_n)h \end{aligned}$$

  This can be done efficiently through $m_1 + m_2$ vector inner product and vector additions. For each reflector:

$$\mathcal{H}_k(u_k)h = \left(I_n - \frac{2u_k u_k^\top}{u_k^\top u_k}\right) h = h - 2\frac{u_k^\top h}{u_k^\top u_k}u_k$$

- Backward propagation:
  Let $L(\{u_i\}, \{v_i\}, \sigma, M, Y, b)$ be the loss or objective function. Gradients between adjacent layers can also be computed iteratively by computing the gradient of each Householder matrix at a time.
  Write $h^+ = \mathcal{H}_k(u_k)h$ and $g = \frac{\partial L}{\partial h^+}$, we have

$$\begin{aligned} \frac{\partial L}{\partial h} &= \left[\frac{\partial h^+}{\partial h}\right]^\top \frac{\partial L}{\partial h^+} = \left(I_n - \frac{2u_k u_k^\top}{u_k^\top u_k}\right) g = g - 2\frac{u_k^\top g}{u_k^\top u_k}u_k \\ \frac{\partial L}{\partial u_k} &= \left[\frac{\partial h^+}{\partial u_k}\right]^\top \frac{\partial L}{\partial h^+} = -2\left(\frac{u_k^\top h}{u_k^\top u_k}I_n + \frac{1}{u_k^\top u_k}hu_k^\top - 2\frac{u_k^\top h}{(u_k^\top u_k)^2}u_k u_k^\top\right) g \\ &= -2\frac{u_k^\top h}{u_k^\top u_k}g - 2\frac{u_k^\top g}{u_k^\top u_k}h + 4\frac{u_k^\top h}{u_k^\top u_k}\frac{u_k^\top g}{u_k^\top u_k}u_k \end{aligned}$$

  Thus backward propagation can also be done in $O((m_1 + m_2)n)$ time.

Theoretical analysis

For linear svdRNN,

- For a linear Gaussian model,

$$y = A\mathrm{vec}(\mathcal{X}) + \epsilon, \qquad \epsilon \sim \mathcal{N}(0, \delta^2 I)$$

  consider choosing W to minimize quadratic loss:

$$\mathbb{R}_{\mathcal{X}\sim\mathcal{D}}\left[\|Y\mathcal{W}\mathrm{vec}(\mathcal{X}) - y\|_2^2\right]$$

- If $\lambda_{\min}(W) \geq e > 0$, every stationary point of the quadratic loss is a global optimum:

$$\begin{aligned} & \nabla_W \mathbb{R}_{\mathcal{X}\sim\mathcal{D}}\left[\|\mathcal{W}\mathrm{vec}(\mathcal{X}) - y\|_2^2\right] = 0 \\ \Rightarrow \quad & A = Y\mathcal{W} \end{aligned}$$

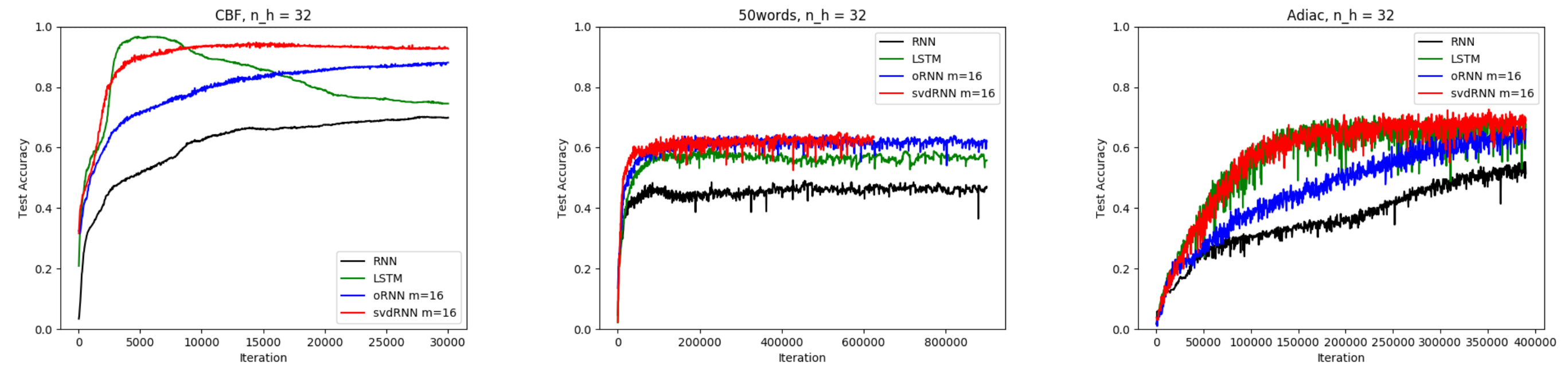
For svdMLP:

- Generalization of MLP is bounded by its Lipschitz constant L [3]
  - svdMLP guarantees $L \leq (1+\epsilon)^t$, if we do hard thresholding on W s.t. $\|W\|_2 \leq 1 + \epsilon$
- Weight matrices are Parseval tight frames $\Longrightarrow$ robust predictions [2]
  - svdMLP guarantees near orthogonal transition matrix

Experiments on Time Series Classification

In time series classification problem, time series are fed into RNN sequentially, which then tries to predict the right class upon receiving the sequence end. The dataset we choose is the largest public collection of class-labeled time-series with widely varying length, namely, the UCR time-series collection.

`http://www.cs.ucr.edu/~eamonn/time_series_data/`



(a) (b) (c)

Figure: Performance comparisons of the RNN based models on three UCR datasets.

Experiments: the MNIST classification from a Sequence of Pixels

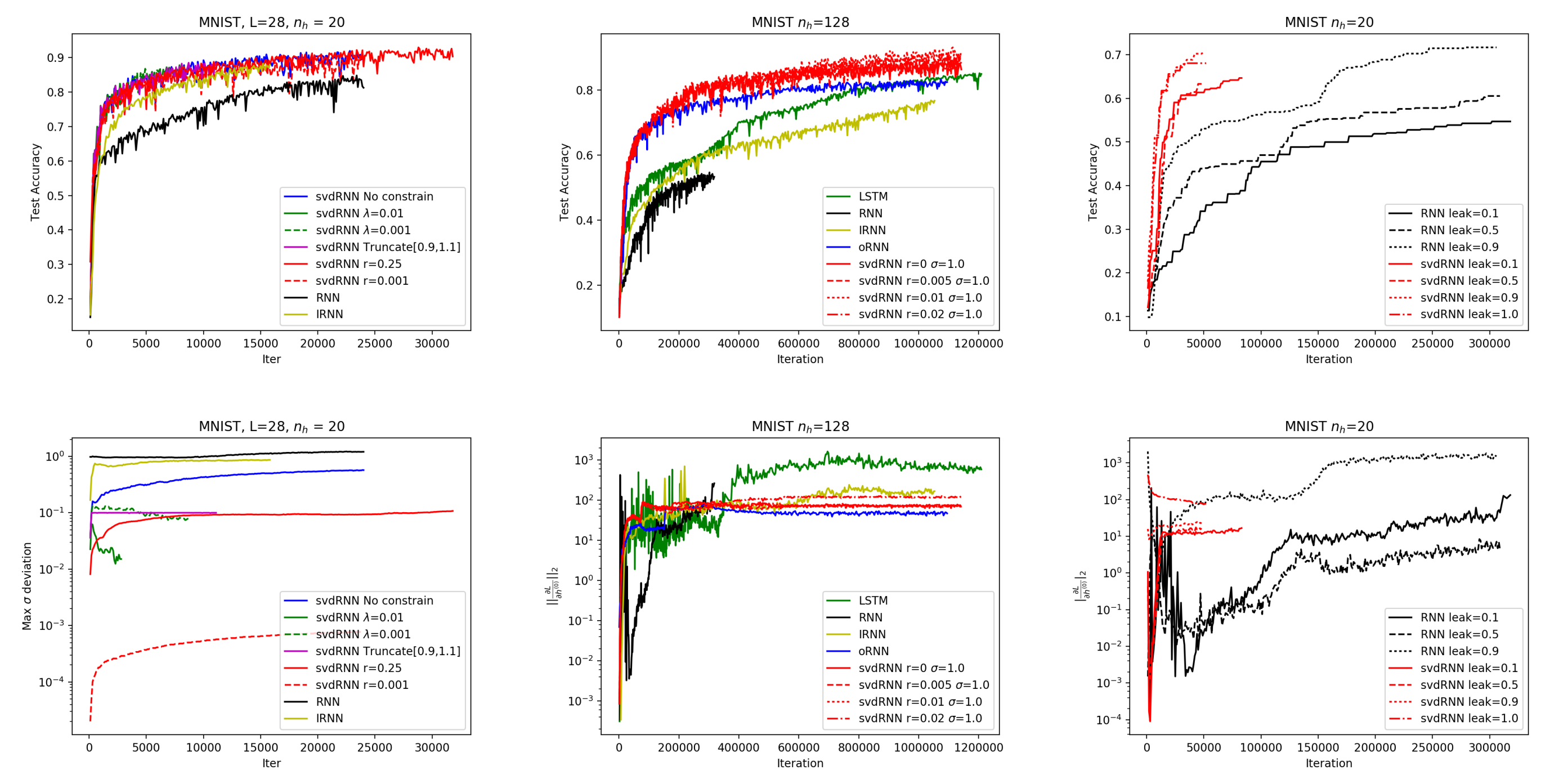


Figure: Variants of RNN models on MNIST

References

1. Mhammedi, Zakaria and Hellicar, Andrew and Rahman, Ashfaqur and Bailey, James. "Efficient Orthogonal Parametrisation of Recurrent Neural Networks Using Householder Reflections", ICML 2017.
2. Cisse, Moustapha and Bojanowski, Piotr and Grave, Edouard and Dauphin, Yann and Usunier, Nicolas. "Parseval networks: Improving robustness to adversarial examples". ICML 2017.
3. Bartlett, Peter and Foster, Dylan J and Telgarsky, Matus. "Spectrally-normalized margin bounds for neural networks". NIPS 2017

Mail: {zhangjiong724@,leiqi@ices.,inderjit@cs.}utexas.edu