0.1 On the Benefit of Fastweight: A Case Study on Vanishing Gradients Alleviation

In this section, we analyze how NSM or any fastweight mechanism helps reduce the vanishing gradients problem [1, 5] in the interface network of a MANN. It is well known that MANN alleviates the gradients problem in the recurrent controller by storing the states into an external memory [3, 4]. However, for the interface network, current MANNs do not support any scheme to overcome the problem. It is rather straightforward to realize that:

Proposition 1. *The interface network in MANN suffers from the problem of vanishing gradients when* $\left\Vert W^{c}\right\Vert \left\Vert \mathbf{M}\right\Vert \rightarrow0$*.*

Proof. see Supplementary A.1. □

We remind that long-term dependencies in the interface are often required to generate correct memory operations. For instance, in the simple copy task, after writing the last object into the memory, the interface network needs to generate ξ_T that moves the reading head to the location of the first object in the memory, which was dictated by ξ_1. To learn that dependency, the norm $\left\Vert \frac{\partial\xi_{T}}{\partial\xi_{1}}\right\Vert$ should not decay exponentially fast with T. Moreover, the decay becomes worse as $\left\Vert W^{c}\right\Vert \left\Vert \mathbf{M}\right\Vert$ is very small close to zero (critical vanishing). Due to parameter updates via backpropagation and data-dependent values of $\mathbf{M}$, it is hard to control the value of $\left\Vert W^{c}\right\Vert \left\Vert \mathbf{M}\right\Vert$ or estimate its distribution. Thus, to analyze the benefit of using multiple programs we need to make no-knowledge assumption that during training, $\left\Vert W^{c}\right\Vert \left\Vert \mathbf{M}\right\Vert$ can reach any value in range $[0, b]$ with equal probability. For such a scenerio, introducing a dynamic W_t^c from NSM is a simple way to help alleviate the decay rate.

Theorem 2. *Assume during training,* $\left\Vert W^{c}\right\Vert \left\Vert \mathbf{M}\right\Vert \sim\mathcal{U}\left(0,\mathrm{b}\right)$ *where* $0<b<\infty$ *and the programs stored in NSM are independent, the chance for critical vanishing gradients (*$\left\Vert W^{c}\right\Vert \left\Vert \mathbf{M}\right\Vert \rightarrow0$*) happen when using multiple programs from NSM is smaller than that when using one program accross timesteps.*

Proof. see see Supplementary A.2. □

0.2 Theorem proofs

0.2.1 Proof of Proposition 1

Proof. For the sake of simplicity, the read vector at timestep t-th can be written as $r_t=\mathbf{M}^{\intercal}w_t^r=\mathbf{M}^{\intercal}D_r\xi_t$, where $\mathbf{M}$, w_t^r are the data memory and the read weight, respectively. The constant binary matrix D_r indicates the elements of ξ_t that will be allocated to form w_i^r. Hence, the output of the RNN controller reads $c_t=W_o\sigma\left(Wx_t+Uh_{t-1}+Vr_{t-1}\right)=W_o\sigma\left(Wx_t+Uh_{t-1}+V\mathbf{M}^{\intercal}D_r\xi_{t-1}\right)$. This leads to a recursive equation $\xi_t=W^cW_o\sigma\left(Wx_t+Uh_{t-1}+V\mathbf{M}^{\intercal}D_r\xi_{t-1}\right)$. Assuming the memory $\mathbf{M}^{\intercal}$ is constant $w.r.t$ ξ_{t-1}, $\frac{\partial\xi_t}{\partial\xi_{t-1}}=W^cW_o diag\left(\sigma'\right)V\mathbf{M}^{\intercal}D_r$. That means, $\left\Vert \frac{\partial\xi_{t}}{\partial\xi_{t-k}}\right\Vert \leq\left(\left\Vert W^{c}\right\Vert \left\Vert \mathbf{M}\right\Vert \gamma\right)^{k}$ where γ is the bound of $\left\Vert W\right\Vert \left\Vert diag\left(\sigma'\right)\right\Vert \left\Vert V\right\Vert \left\Vert D_{r}\right\Vert$.

Hence, the interface network will suffer from the vanishing gradients problem if $\|W^c\|\|\mathbf{M}\| < 1/\gamma$. The proof can be extended for GRU and LSTM controller. $\square$

0.2.2 Proof of Theorem 2

Lemma 3. *Given n IID random variables $X_1, X_2, ..., X_n \sim \mathcal{U}(0, \mathrm{b})$ then for $0 < z < b$,*

$$P(X_1 X_2 ... X_n \leq z) = \int_{x=0}^{z} \frac{\ln\left(\frac{b^n}{x}\right)^{n-1}}{b^n (n-1)!} dx$$

Proof. see [2]. $\square$

Lemma 4. *Given a random variables $X \sim \mathcal{U}(0, \mathrm{b})$ and an integer n then for $0 < z < b$,*

$$P(X^n < z) = \int_{x=0}^{z} \frac{x^{\left(\frac{1}{n}-1\right)}}{bn} dx$$

Proof. $P(X^n < z) = P(X < z^{1/n}) = \frac{z^{1/n}}{b} = \int_{x=0}^{z} \frac{x^{\left(\frac{1}{n}-1\right)}}{bn} dx.$ $\square$

When using one program, we can rewrite the vanishing gradients chance as,

$$P\left(\|W^c\|\|\mathbf{M}\| < 1/\gamma\right) = P\left(\left(\|W^c\|\|\mathbf{M}\|\right)^T < 1/\gamma^T\right)$$

When using multiple program, the condition for vanishing gradients problem can be written as,

$$\prod_{t=1}^{T} \|W^c\|\|\mathbf{M}\| < 1/\gamma^T$$

where T is the number of timesteps and each $\|W^c\|\|\mathbf{M}\| \sim \mathcal{U}(0, \mathrm{b})$. Let $P\left(\prod_{t=1}^{T} \|W^c\|\|\mathbf{M}\| < 1/\gamma^T\right)$ denote the chance for vanishing gradients happen in this case. According to Lemma 3 and 4, we have

$$P\left(\prod_{t=1}^{T} \|W^c\|\|\mathbf{M}\| < 1/\gamma^T\right) - P\left(\|W^c\|\|\mathbf{M}\| < 1/\gamma\right) = F\left(1/\gamma^T\right) = \int_{x=0}^{1/\gamma^T} \frac{\ln\left(\frac{b^n}{x}\right)^{n-1}}{b^n (n-1)!} - \frac{x^{\left(\frac{1}{n}-1\right)}}{bn} dx$$

If we examine the function $f(x) = \frac{\ln\left(\frac{b^n}{x}\right)^{n-1}}{b^n(n-1)!} - \frac{x^{\left(\frac{1}{n}-1\right)}}{bn}$ with $x > 0$, it is not hard to realize that $\forall b \in \mathbb{R}^+, n \in \mathbb{N}^+$: $\lim_{x\to 0} f(x) \to -\infty$ because $\ln\left(\frac{\ln\left(\frac{b^n}{x}\right)^{n-1}}{b^n(n-1)!}\right) =$

$o\left(\frac{x^{\left(\frac{1}{n}-1\right)}}{bn}\right)$ as $x \to 0$. This means the integral $F\left(1/\gamma^T\right) < 0$ for $0 < 1/\gamma^T < z_0$ and $z_0 \to \infty$ as $n, b \to \infty$. Threfore, the chance for critical vanishing gradients happen when using multiple programs from NSM is smaller than that when using one program accross timesteps. In practice, the interface network weight can have hundred of thousand elelments, resulting in a large values of b (often > 10). This magnifies the difference between two density functions, and thus increses the value of z_0, making the result holds true not only for critical cases but also for most of the cases.

References

[1] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.

[2] Carl P Dettmann and Orestis Georgiou. Product of n independent uniform random variables. *Statistics & probability letters*, 79(24):2501–2503, 2009.

[3] Caglar Gulcehre, Sarath Chandar, and Yoshua Bengio. Memory augmented neural networks with wormhole connections. *arXiv preprint arXiv:1701.08718*, 2017.

[4] Hung Le, Truyen Tran, and Svetha Venkatesh. Learning to remember more with less memorization. In *International Conference on Learning Representations*, 2019.

[5] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, pages 1310–1318, 2013.