The Thirty-Seventh AAAI Conference on Artificial Intelligence (AAAI-23)

Fast Saturating Gate for Learning Long Time Scales with Recurrent Neural Networks

Kentaro Ohno, Sekitoshi Kanai, Yasutoshi Ida

NTT

{kentaro.ohno.tf, sekitoshi.kanai.fu, yasutoshi.ida.yc}@hco.ntt.co.jp

Abstract

Gate functions in recurrent models, such as an LSTM and GRU, play a central role in learning various time scales in modeling time series data by using a bounded activation function. However, it is difficult to train gates to capture extremely long time scales due to gradient vanishing of the bounded function for large inputs, which is known as the saturation problem. We closely analyze the relation between saturation of the gate function and efficiency of the training. We prove that the gradient vanishing of the gate function can be mitigated by accelerating the convergence of the saturating function, i.e., making the output of the function converge to 0 or 1 faster. Based on the analysis results, we propose a gate function called fast gate that has a doubly exponential convergence rate with respect to inputs by simple function composition. We empirically show that our method outperforms previous methods in accuracy and computational efficiency on benchmark tasks involving extremely long time scales.

Introduction

Recurrent neural networks (RNNs) are models suited to processing sequential data in various applications, e.g., speech recognition (Ling et al. 2020) and video analysis (Zhu et al. 2020). The most widely used RNNs are a long short-term memory (LSTM) (Hochreiter and Schmidhuber 1997) and gated recurrent unit (GRU) (Cho et al. 2014), which has a gating mechanism. The gating mechanism controls the information flow in the state of RNNs via multiplication with a gate function bounded to a range $(0, 1)$. For example, when the forget gate takes a value close to 1 (or 0 for the update gate in the GRU), the state preserves the previous information. On the other hand, when it gets close to the other boundary, the RNN updates the state by the current input. Thus, in order to represent long temporal dependencies of data involving hundreds or thousands of time steps, it is crucial for the forget gate to take values near the boundaries (Tallec and Ollivier 2018; Mahto et al. 2021).

However, it is difficult to train RNNs so that they have the gate values near the boundaries. Previous studies hypothesized that this is due to gradient vanishing for the gate function called *saturation* (Chandar et al. 2019; Gu et al. 2020), i.e., the gradient of the gate function near the boundary is too small to effectively update the parameters. To avoid the saturation problem, a previous study used unbounded activation functions (Chandar et al. 2019). However, this makes training unstable due to the gradient explosion (Pascanu, Mikolov, and Bengio 2013). Another study introduced residual connection for a gate function to push the output value toward boundaries, hence mitigating the saturation problem (Gu et al. 2020). However, it requires additional computational cost due to increasing the number of parameters for another gate function. For broader application of gated RNNs, a more efficient solution is necessary.

To overcome the difficulty of training, we propose a novel activation function for the forget gate based on the usual sigmoid function, which we call the *fast gate*. Modification of the usual sigmoid gate to the fast gate is simple and easy to implement since it requires only one additional function composition. To this end, we analyze the relation between the saturation and gradient vanishing of the bounded activation function. Specifically, we focus on the convergence rate of the activation function to the boundary, which we call the *order of saturation*. For example, the sigmoid function $\sigma(z) = 1/(1 + e^{-z})$ has the exponential order of saturation, i.e., $1 - \sigma(z) = O(e^{-z})$ (see Fig. 1), and the derivative also decays to 0 exponentially as z goes to infinity. When a bounded activation function has a higher order of saturation, the derivative decays much faster as the input grows. Since previous studies have assumed that the decaying derivative on the saturating regime causes the stuck of training (Ioffe and Szegedy 2015), it seems that a higher order of saturation would lead to poor training. Contrarily to this intuition, we prove that a higher order of saturation alleviates the gradient vanishing on the saturating regime through observation on a toy problem for learning long time scales. This result indicates that functions saturating superexponentially are more suitable for the forget gate to learn long time scales than the sigmoid function. On the basis of this observation, we explore a method of realizing such functions by composing functions which increase faster than the identity function (e.g., $\alpha(z) = z + z^3$) as $\sigma(\alpha(z))$. We find that the hyperbolic sinusoidal function is suitable for achieving a higher order of saturation in a simple way, and we obtain the fast gate. Since the fast gate has a doubly exponential order of saturation $O(e^{-e^z})$, it improves the trainability of gated RNNs for long time scales of sequential data. We evaluate

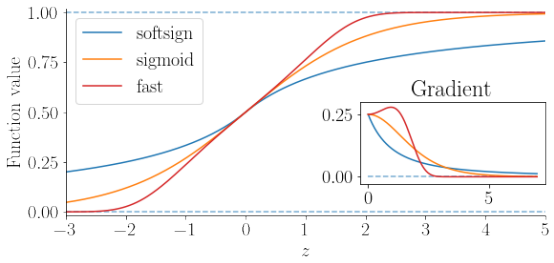

Figure 1: Function values and derivative of various bounded activation functions. Sigmoid function σ (orange) exponentially converges to 1 as $z \to \infty$, since $1 - \sigma(z) = \frac{1}{1+e^z} \approx e^{-z}$. Derivative also decays exponentially. Normalized version of softsign function $\mathrm{softsign}(z) = \frac{z}{1+|z|}$ (blue) converges to 1 more slowly. Fast gate (red) is proposed gate function. Although gradient decays faster than sigmoid function, it provably helps learning values near boundaries.

the computational efficiency and accuracy of a model with the fast gate on benchmark tasks, including synthetic tasks, pixel-by-pixel image classification, and language modeling, which involve a wide range of time scales. The model with the fast gate empirically outperforms a baseline LSTM and other variants recently proposed for tackling the saturation problem (Chandar et al. 2019; Gu et al. 2020) in terms of accuracy and the convergence speed of training while maintaining stability of training. Further visualization analysis of learning time scales shows that our theory fits the learning dynamics of actual models and that the fast gate can learn extremely long time scales of thousands of time steps.

Our major contributions are as follows:

- We prove that gate functions which saturate faster actually *accelerates* learning values near boundaries. The result indicates that fast saturation improves learnability of gated RNNs on data with long time scales.
- We propose the *fast gate* that saturates faster than the sigmoid function. In spite of its simplicity, the fast gate achieves a doubly exponential order of saturation, and thus effectively improves learning of long time scales.
- We evaluate the effectiveness of the fast gate against recently proposed methods such as an NRU (Chandar et al. 2019) and a refine gate (Gu et al. 2020). The results verify that the fast gate robustly improves the learnability for long-term dependencies in both synthetic and real data.

For appendices referred hereafter, we refer to the arXiv version of this paper (Ohno, Kanai, and Ida 2022).

Preliminaries

Time Scales in Gated RNNs

We review gated RNNs and their time scale interpretation (Tallec and Ollivier 2018). We begin with an LSTM (Hochreiter and Schmidhuber 1997), which is one of the most popular RNNs. An LSTM has a memory cell $c_t \in \mathbb{R}^n$ and hidden state $h_t \in \mathbb{R}^n$ inside, which are updated depending on the sequential input data x_t at each time step $t = 1, 2, \cdots$ by

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \tag{1}$$
$$h_t = o_t \odot \tanh(c_t) \tag{2}$$
$$f_t = \sigma(W_f x_t + U_f h_{t-1} + b_f) \tag{3}$$
$$i_t = \sigma(W_i x_t + U_i h_{t-1} + b_i) \tag{4}$$
$$\tilde{c}_t = \tanh(W_c x_t + U_c h_{t-1} + b_c) \tag{5}$$
$$o_t = \sigma(W_o x_t + U_o h_{t-1} + b_o) \tag{6}$$

where W_*, U_* and b_* are weight and bias parameters for each $* \in \{f, i, c, o\}$. The sigmoid function σ is defined as

$$\sigma(x) = \frac{1}{1 + e^{-x}}. \tag{7}$$

$f_t, i_t, o_t \in (0, 1)^n$ are called forget, input, and output gates, respectively. They were initially motivated as a binary mechanism, i.e., switching on and off, allowing information to pass through (Gers, Schmidhuber, and Cummins 2000). The forget gate has been reinterpreted as the representation for time scales of memory cells (Tallec and Ollivier 2018). Following that study, we simplify Eq. (1) by assuming $\tilde{c}_t = 0$ for an interval $t \in [t_0, t_1]$. Then, we obtain

$$c_{t_1} = f_{t_1} \odot c_{t_1 - 1} \tag{8}$$
$$= \bar{f}^{t_1 - t_0} \odot c_{t_1 - t_0}, \tag{9}$$

where $\bar{f} = (\prod_{s=t_0+1}^{t_1} f_s)^{\frac{1}{t_1 - t_0}}$ is the (entry-wise) geometric mean of the values of the forget gate. Through Eq. (8), the memory cell c_t loses its information on data up to time t_0 exponentially, and the entry of $\bar{f}$ represents its (averaged) decay rate. This indicates that, in order to capture long-term dependencies of the sequential data, the forget gate is desired to take values near 1 on average. We refer the associated time constant[1] $T = -1/\log \bar{f}$ as the *time scale* of units, which has been empirically shown to illustrate well the temporal behavior of LSTMs (Mahto et al. 2021).

The above argument applies not only to an LSTM, but also to general gated RNNs including a GRU (Cho et al. 2014) with state update of the form

$$h_t = f_t \odot h_{t-1} + i_t \odot \tilde{h}_t, \tag{10}$$

where h_t, f_t, i_t denotes the state, forget gate, and input gate, respectively, and $\tilde{h}_t$ is the activation to represent new information at time t. Here again, the forget gate f_t takes a role to control the time scale of each unit of the state.

Saturation in Gating Activation Functions

The sigmoid function $\sigma(z)$ in the gating mechanism requires large z to take a value near 1 as the output. On the other hand, the derivative $\sigma'(z)$ takes exponentially small values for $z \gg 0$ (Fig. 1). Thus, when a gated model needs to learn large gate values such as 0.99 with gradient methods, parameters in the gate cannot be effectively updated due to gradient vanishing. This is called *saturation* of bounded activation functions (Gulcehre et al. 2016). The behavior of

[1] An exponential function $F(t) = e^{-\alpha t}$ of time t decreases by a factor of $1/e$ in time $T = 1/\alpha$, which is called the time constant.

gate functions on the saturating regime is important for gated RNNs because forget gate values need to be large to represent long time scales as explained above. That is, gated RNNs must face saturation of the forget gate to learn long time scales. Thus, it is hypothesized that saturation causes difficulty in training gated RNNs for data with extremely long time scales (Chandar et al. 2019; Gu et al. 2020).

Related Work

We outline the most related studies here and provide discussion of other studies in Appendix A due to space limitation.

Several studies investigate the time scale representation of the forget gate function to improve learning on data involving long-term dependencies (Tallec and Ollivier 2018; Mahto et al. 2021). For example, performance of LSTM language models can be improved by fixing the bias parameter of the forget gate in accordance with a power law of time scale distribution, which underlies natural language (Mahto et al. 2021). Such techniques require us to know the appropriate time scales of data a priori, which is often difficult. Note that this approach can be combined with our method since it is complementary with our work.

Several modifications of the gate function have been proposed to tackle the saturation problem. The noisy gradient for a piece-wise linear gate function was proposed to prevent the gradient to take zero values (Gulcehre et al. 2016). This training protocol includes hyperparameters controlling noise level, which requires manual tuning. Furthermore, such a stochastic approach can result in unstable training due to gradient estimation bias (Bengio, Léonard, and Courville 2013). The refine gate (Gu et al. 2020) was proposed as another modification introducing a residual connection to push the gate value to the boundaries. It is rather heuristic and does not provide theoretical justification. It also requires additional parameters for the auxiliary gate, which increases the computational cost for both inference and training. In contrast, our method theoretically improves learnability and does not introduce any additional parameters. Another study suggests that omitting gates other than the forget gate makes training of models for long time scales easier (Van Der Westhuizen and Lasenby 2018). However, such simplification may lose the expressive power of the model and limit its application fields. Chandar et al. (2019) proposed an RNN with a non-saturating activation function to directly avoid the gradient vanishing due to saturation. Since its state and memory vector evolves in unbounded regions, the behavior of the gradient can be unstable depending on tasks. Our method mitigates the gradient vanishing by controlling the order of saturation, while maintaining the bounded state transition.

Analysis on Saturation and Learnability

We discuss the learning behavior of the forget gate for long time scales. First, we formulate a problem of learning long time scales in a simplified setting. Next, we relate the efficiency of learning on the problem to the saturation of the gate functions. We conclude that the faster saturation makes learning more efficient. All proofs for mathematical results below are given in Appendix C.

Problem Setting

Recall Eq. (8), which describes the time scales of the memory cell c_t of an LSTM via exponential decay. Let the memory cell at time t_1 be $c_{t_1} = \lambda c_{t_0}$ with $\lambda \in (0, 1)$. Requiring long time scales corresponds to getting λ close to 1. Therefore, we can consider a long-time-scale learning problem as minimizing a loss function L that measures discrepancy of c_{t_1} and $\lambda_* c_{t_0}$ where $\lambda_* \in (0, 1)$ is a desired value close to 1. We take L as the absolute loss for example. Then, we obtain

$$L = |c_{t_1} - \lambda_* c_{t_0}| \tag{11}$$
$$= |\bar{f}^{t_1 - t_0} c_{t_0} - \lambda_* c_{t_0}| \tag{12}$$
$$= c_{t_0} |\bar{f}^{t_1 - t_0} - \lambda_*|, \tag{13}$$

using Eq. (8). Let $z_t = W_f x_t + U_f h_{t-1} + b_f$, so that $f_t = \sigma(z_t)$. Since we are interested in the averaged value of f_t, we consider z_t to be time-independent, that is, $z_t = z$ in the same way as Tallec and Ollivier (2018). The problem is then reduced to a problem to obtain z that minimizes

$$L(z) = c_{t_0} |\sigma(z)^{t_1 - t_0} - \lambda_*|. \tag{14}$$

We consider this as the minimal problem to analyze the learnability of the forget gate for long time scales. Note that since the product $c_{t_1} = \bar{f}^{t_1 - t_0} c_{t_0}$ is taken element-wise, we can consider this as a one-dimensional problem. Furthermore, the global solution can be explicitly written as $z = \sigma^{-1}(\lambda_*^{1/(t_1 - t_0)})$ where σ^{-1} is an inverse of σ.

Next, we consider the learning dynamics of the model on the aforementioned problem Eq. (14). RNNs are usually trained with gradient methods. Learning dynamics with gradient methods can be analyzed considering learning rate $\to 0$ limit known as gradient flow (Harold and George 2003). Therefore, we consider the following gradient flow

$$\frac{dz}{d\tau} = -\frac{\partial L}{\partial z}, \tag{15}$$

using the loss function introduced above. Here, τ denotes a time variable for learning dynamics, which should not be confused with t representing the state transition. Our aim is to investigate the convergence rate of a solution of the differential equation Eq. (15) when σ in the forget gate is replaced with another function ϕ.

Order of Saturation

To investigate the effect of choice of gate functions on the convergence rate, we first define the candidate set $\mathcal{F}$ of bounded functions for the gate function.

Definition 0.1. Let $\mathcal{F}$ be a set of differentiable and strictly increasing surjective functions $\phi : \mathbb{R} \to (0, 1)$ such that the derivative ϕ' is monotone on $z > z_0$ for some $z_0 \geq 0$.

$\mathcal{F}$ is a natural class of gating activation functions including σ. To clarify the issue of gradient vanishing due to saturation when learning long time scales, we first show that saturation is inevitable regardless of the choice of $\phi \in \mathcal{F}$.

Proposition 0.2. $\lim_{z \to \infty} \phi'(z) = 0$ *holds for any* $\phi \in \mathcal{F}$.

Nevertheless, choices of ϕ significantly affect the efficiency of the training. When the target λ_* takes an extreme value near boundaries, the efficiency of training should depend on the asymptotic behavior of $\phi(z)$ for $z \gg 0$, that is, the rate at which $\phi(z)$ converges as $z \to \infty$. We call the convergence rate of $\phi(z)$ as $z \to \infty$ as the *order of saturation*. More precisely, we define the notion as follows[2]:

Definition 0.3. Let $g : \mathbb{R} \to \mathbb{R}$ be a decreasing function. $\phi \in \mathcal{F}$ has the *order of saturation* of $O(g(z))$ if $\lim_{z\to\infty} \frac{g(az)}{1-\phi(z)} = 0$ for some $a > 0$. For $\phi, \tilde{\phi} \in \mathcal{F}$, ϕ has a *higher order of saturation* than $\tilde{\phi}$ if $\lim_{z\to\infty} \frac{1-\phi(z)}{1-\tilde{\phi}(az)} = 0$ holds for any $a > 0$ and $\tilde{\phi}^{-1}(\phi(z))$ is convex for $z \gg 0$.

Intuitively, the order of saturation of $O(g(z))$ means that the convergence rate of ϕ to 1 is bounded by the decay rate of g up to constant multiplication of z. For example, the sigmoid function σ satisfies $e^{-az}/(1-\sigma(z)) \to 0$ as $z \to \infty$ for any $a > 1$, thus has the exponential order of saturation $O(e^{-z})$. The convexity condition for a higher order of saturation is rather technical, but automatically satisfied for typical functions, see Appendix C.2. If ϕ has a higher order of saturation (or saturates *faster*) than another function $\tilde{\phi}$, then $\phi(z)$ converges faster than $\tilde{\phi}(z)$ as $z \to \infty$, and $\phi'(z)$ becomes smaller than $\tilde{\phi}'(z)$. In this sense, training with $\tilde{\phi}$ seems more efficient than ϕ in the above problem. However, this is not the case as we discuss in the next section.

Efficient Learning via Fast Saturation

To precisely analyze learning behavior, we trace the learning dynamics of the output value $f = \phi(z)$ since our purpose is to obtain the desired *output* value rather than the input z. We transform the learning dynamics (Eq. (15)) into that of f by

$$\frac{df}{d\tau} = \frac{dz}{d\tau}\frac{df}{dz} = -\phi'(z)\frac{\partial L}{\partial z} = -\phi'(z)^2\frac{\partial L}{\partial f}. \tag{16}$$

To treat Eq. (16) as purely of f, we define a function $g_\phi(f)$ of f by $g_\phi(f) := \phi'(\phi^{-1}(f))$, so that Eq. (16) becomes

$$\frac{df}{d\tau} = -g_\phi(f)^2\frac{\partial L}{\partial f}. \tag{17}$$

Our interest is in the dynamics of f near the boundary, i.e., the limit of $f \to 1$. We have the following result:

Theorem 0.4. *Let $\phi, \tilde{\phi} \in \mathcal{F}$. If ϕ has a higher order of saturation than $\tilde{\phi}$, then $g_\phi(f)/g_{\tilde{\phi}}(f) \to \infty$ as $f \to 1$.*

Theorem 0.4 indicates that a higher order of saturation accelerates the move of the output f near boundaries in accordance with Eq. (17) since $g_\phi(f)$ takes larger values. Thus, contrarily to the intuition in the previous section, a higher order of saturation leads to more efficient training for target values near boundaries. We demonstrate this effect using two activation functions, the sigmoid function $\sigma(z)$ and normalized softsign function $\sigma_{\mathrm{ns}}(z) = (\mathrm{softsign}(z/2) + 1)/2$ where $\mathrm{softsign}(z) = z/(1+|z|)$. σ_{ns} is the softsign function modified so that $0 \le \sigma_{\mathrm{ns}}(z) \le 1$ and $\sigma'_{\mathrm{ns}}(0) = \sigma'(0)$. σ has a higher order of saturation than σ_{ns} since σ has the order of saturation of $O(e^{-z})$ and σ_{ns} has $O(z^{-1})$ (see Fig. 1). We plot the learning dynamics of gradient flow for the problem in Fig. 2. Since σ has a higher order of saturation than σ_{ns}, the gate value f of σ_{ns} converges slower to the boundary. Fig. 2 also shows the dynamics of gradient descent with the learning rate 1. While gradient descent is a discrete approximation of gradient flow, it behaves similar to gradient flow.

[2]Our definition for asymptotic order is slightly different from the usual one which adopts $\limsup_{z\to\infty} \frac{g(z)}{1-\phi(z)} < \infty$, since it is more suitable for analyzing training efficiency.

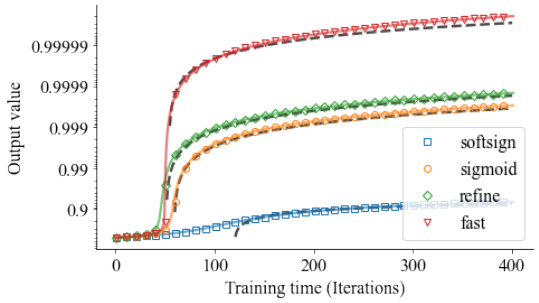

Figure 2: Learning curves for simplified long-time-scale learning problem with gradient descent (markers) and with gradient flow (solid lines). Gradient descent is done with learning rate 1. Time difference $t_1 - t_0$ is set to 10. Dashed lines are lower bounds given in Tab. 1 fitted to each learning curve with suitable translation. These lower bounds well approximate asymptotic convergence of gradient flow.

Explicit convergence rates. Beyond Theorem 0.4, we can explicitly calculate effective bounds of the convergence rate for the problem when the activation function is the sigmoid function $\sigma(z)$ or normalized softsign function $\sigma_{\mathrm{ns}}(z)$.

Proposition 0.5. *Consider the problem in Section with the absolute loss $L = |f^{t_1-t_0} - \lambda_*|$ with $\lambda_* = 1$. For the sigmoid function $f = \sigma(z)$, the convergence rate for the problem is bounded as $1 - f = O(\tau^{-1})$. Similarly, for the normalized softsign function $f = \sigma_{\mathrm{ns}}(z)$, the convergence rate is bounded as $1 - f = O(\tau^{-1/3})$.*

Proposition 0.5 shows the quantitative effect of difference in the order of saturation on the convergence rates. We fit the bounds to the learning curves with the gradient flow in Fig. 2. The convergence rates of the learning are well approximated by the bounds. These asymptotic analysis highlights that choices of the function ϕ significantly affects efficiency of training for long time scales.

Proposed Method

On the basis of the analysis in Section , we construct the fast gate, which is suitable for learning long time scales.

Desirable Properties for Gate Functions

We consider modification of the usual sigmoid function to another function $\phi \in \mathcal{F}$ for the forget gate in a gated RNN. Function ϕ should satisfy the following conditions.

(i) ϕ has a higher order of saturation than σ,

(ii) $\phi(z) \approx \sigma(z)$ for $z \approx 0$,

(iii) ϕ is symmetric in a sense that $\phi(-z) = 1 - \phi(z)$.

Condition (i) comes from the argument in the previous section that fast saturating functions learn values near boundaries efficiently. Conditions (ii) and (iii) indicate that the function $\phi(z)$ behaves similarly to $\sigma(z)$ around $z = 0$. In order to avoid possible harmful effects due to the modification, we do not want to change the behavior of the function away from the saturating regime. Hence, we require these conditions. The requirements are analogous to those by Gu et al. (2020, Section 3.4) for the gate adjustment. The first condition can be viewed as a theoretical refinement of their heuristic modification.

Fast Gate

We explore gate functions satisfying the above conditions. Recall that the sigmoid function $\sigma(z)$ has the exponential order of saturation. From condition (i) in the previous section, we explore functions saturating superexponentially. Since any superexponential order can be written as $O(e^{-\alpha(z)})$ with a function satisfying $\alpha(z) > z$ for large z, it is enough to consider a function of the form $\phi(z) = \sigma(\alpha(z))$ for such α. The desirable properties in Section are rephrased as follows in terms of α: (i) $\alpha(z) \gg z$ for $z \gg 0$, (ii) $\alpha'(0) = 1$, and (iii) $\alpha(-z) = -\alpha(z)$ for $z \in \mathbb{R}$. Such functions can be found as examples in the form $\alpha(z) = z + p(z)$ where p is a polynomial consisting of only odd higher degree terms, such as $\alpha(z) = z + z^3$. Since a higher degree term has a larger effect on the order of saturation, it mitigates gradient vanishing of the gate function more in accordance with Theorem 0.4. Thus, we take a limit of the degree to infinity, which leads to a simple function expression

$$\alpha(z) = z + \frac{z^3}{3!} + \frac{z^5}{5!} + \cdots \tag{18}$$

$$= \sinh(z) := \frac{e^z - e^{-z}}{2}. \tag{19}$$

Therefore, we adopt $\alpha(z) = \sinh(z)$ for the alternative function $\phi = \sigma \circ \alpha$ and obtain the fast gate

$$\phi(z) := \sigma(\sinh(z)). \tag{20}$$

This simple expression enables us to implement it with only one additional function. Note that there are infinitely many possible choices satisfying the desirable properties, and the above particular form is one of the simplest choices. For discussion of other candidates, see Appendix D.

Comparison with Other Gate Functions

We analyze the fast gate $\phi(z)$ and compare it with other gate functions. First, the order of saturation of the fast gate is $O(e^{-e^z})$ since $e^{-e^{az}}/(1 - \phi(z)) \to 0$ as $z \to \infty$ for any $a > 1$. We briefly describe the method of the refine gate (Gu et al. 2020), which was proposed to avoid gradient vanishing of the gate function. This method exploits an auxiliary gate $r_t \in (0, 1)^n$ to modify the forget gate value f_t to $g_t = r_t(1-(1-f_t)^2)+(1-r_t)f_t^2$. When a large output value is desired for the forget gate value, the auxiliary gate is expected to take $r_t \approx 1$ to push f_t to $g_t \approx 1-(1-f_t)^2$. Therefore, from the asymptotic view point, this method modifies the order of saturation of the gate function from $O(e^{-z})$ to $O(e^{-2z})$. Compared with the refine gate, the fast gate has a much higher order of saturation. We also analyze the asymptotic convergence rates of solving the toy problem defined in the previous section. We summarize the results in Tab. 1. See Appendix C.3 for detailed derivation. Since the fast gate has a doubly exponential order of saturation, the order of convergence rate of learning long time scales is faster than the sigmoid and refine gates which have an exponential order of saturation (see also Fig. 2 for comparison of the convergence rates). In addition, the fast gate does not require additional parameters whereas the refine gate does. Therefore, the fast gate is computationally more efficient than the refine gate.

	Softsign	Sigmoid	Refine	Fast
Saturation order	$O(z^{-1})$	$O(e^{-z})$	$O(e^{-2z})$	$O(e^{-e^z})$
Convergence rate	$O(\tau^{-\frac{1}{3}})$	$O(\tau^{-1})$	$O(\tau^{-1})$	$O(F(\tau))$
Additional parameters	No	No	Yes	No

Table 1: Comparison of order of saturation and convergence rate for simplified long-time-scale learning problem (see also Fig. 2). "Fast" denotes our method. Function $F(\tau) := W^2(c\tau^{-\frac{1}{2}})$ is asymptotically large than τ^{-1}, where $c > 0$ is some constant and $W^2(\cdot)$ is square of Lambert's function W defined as inverse of map $z \mapsto ze^z$.

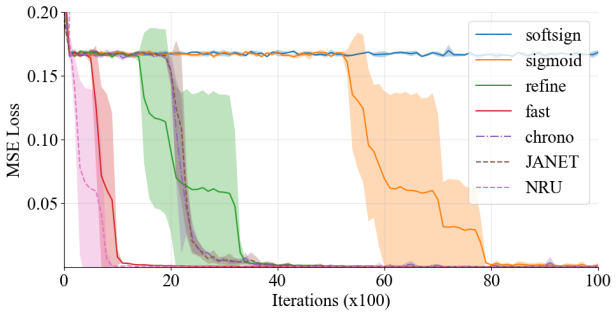

Figure 3: MSE loss for adding task of sequence length 5000

Experiments

Synthetic Tasks

We evaluate the learnability for long time scales across various methods on two synthetic tasks, adding and copy, following previous studies (Hochreiter and Schmidhuber 1997; Arjovsky, Shah, and Bengio 2016). While these tasks are simple and easy to solve for short sequences, they get extremely difficult for gated RNNs to solve when the sequence length grows to hundreds or thousands.

Setup. We compare the fast gate with the refine gate because Gu et al. (2020) reported that the refine gate achieved the best performance among other previous gate variants.

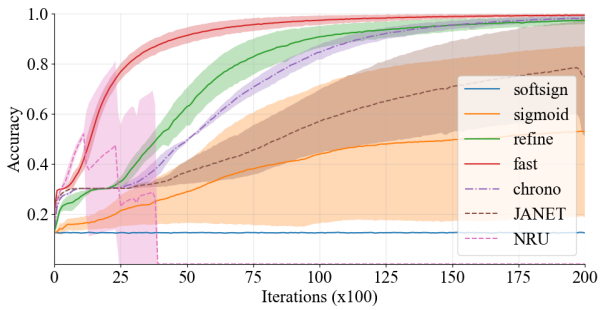

Figure 4: Accuracy for copy task of sequence length 500

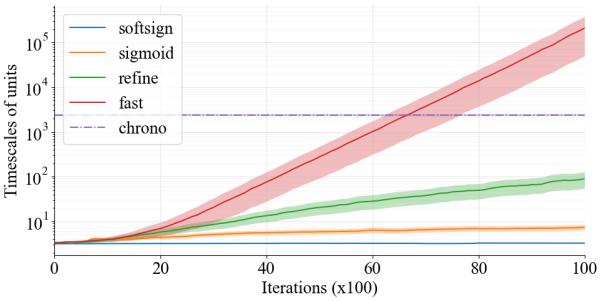

Figure 5: Growth of time scales of units over various gate functions on adding task. Lines and shaded areas represent mean and standard deviation (divided by 10 for visibility) of time scales over 128 units in memory cell. Fast gate learns exponentially larger magnitudes of time scales than others.

We also include the normalized softsign function $\sigma_{\mathrm{ns}}(z)$ as a referential baseline to test the compatibility of our theory. We use these gate functions in a single-layer LSTM. Since the initialization of the forget gate bias b_f is critical to model performance (Tallec and Ollivier 2018), we set it so that $\phi(b_f) = 1/(1+e^{-1})$ is satisfied for each gate function ϕ (with the bias for an additional gate function initialized by 0 for the refine gate), which amounts to $b_f = 1$ in the usual sigmoid case (Gers, Schmidhuber, and Cummins 2000; Greff et al. 2016). We also compare performance of these gate variants to that of **chrono**-initialization (Tallec and Ollivier 2018), a method to initialize parameters to represent long time scales for the sigmoid gate. In addition to the LSTM, we include **JANET** (Van Der Westhuizen and Lasenby 2018) and **NRU** (Chandar et al. 2019) as baselines. JANET is one of the simplest gated RNNs specialized to learn long time scales by omitting gates other than the forget gate and applying chrono-initialization. NRU uses non-saturating activation functions to write or erase to a memory cell. We train and evaluate each model three times by varying the random seed. See Appendix E.3 for detailed setting.

Results. The mean squared error on the adding task of sequence length 5000 and the accuracy on the copy task of sequence length 500 during training are shown in Fig. 3 and 4, respectively. While NRU requires the least number of parameter updates on the adding task, the training diverges on the copy tasks due to gradient explosion (Pascanu, Mikolov, and Bengio 2013). This is because the state in the NRU evolves on an unbounded region; thus, a small parameter update can drastically change the behavior of the model. We could not fix this instability even by reducing the clipping threshold for gradient by a factor of 10. We hypothesize that the training of the NRU tends to be more unstable on the copy task because this task has higher dimensional nature than the adding task in the sense of input dimension (10 vs 2) and the number of tokens to memorize (10 vs 2). Among the gate functions, the fast gate converges the fastest. This is due to the higher order of saturation: the fast gate has the order of saturation $O(e^{-e^z})$ whereas the refine gate has $O(e^{-2z})$, thus learns long time scales more efficiently. The normalized softsign gate completely fails to learn since it has a lower order of saturation $O(z^{-1})$ than the sigmoid function. Thus, the performance of the models is well explained by the difference in the order of saturation in Tab. 1. This indicates that our theoretical analysis matches practical settings despite the fact that it builds on a simplified learning problem. The result also shows that modification of the gate function is more effective for learning long-term dependencies than other methods such as chrono-initialization and JANET.

	sMNIST	psMNIST	sCIFAR	Time
Softsign	97.50 ± 0.58	91.71 ± 0.33	59.21 ± 0.39	17.7
Sigmoid	98.88 ± 0.12	95.71 ± 0.02	69.14 ± 0.39	14.3
Refine	98.94 ± 0.03	95.93 ± 0.16	69.55 ± 0.50	22.7
Fast	99.05 ± 0.04	96.18 ± 0.14	70.06 ± 0.38	14.7
chrono	98.83 ± 0.09	94.37 ± 0.69	60.36 ± 0.51	14.3
JANET	98.59 ± 0.03	93.85 ± 0.23	60.99 ± 0.51	10.6
NRU	98.73 ± 0.27	94.76 ± 0.35	62.32 ± 0.30	35.0

Table 2: Test accuracy on image classification tasks and processing time (min.) per epoch on psMNIST

We further observe the growth of time-scale distribution of the memory cell in the LSTM on the adding task. The time scale of i-th unit in the memory cell is measured using the bias term of the forget gate by $-1/\log \phi(b_{f,i})$ (Tallec and Ollivier 2018; Mahto et al. 2021). We show the statistics of time scales over all 128 units at each iteration of the training in Fig. 5. The fast gate represents much longer time scales than other gate functions after training, which validates our method. While chrono-initialization set time scales so that they are uniformly distributed within the range $[1, 5000]$, they do not change at all during training due to saturation of the usual sigmoid function $\sigma(z)$. Since the fast gate can learn to adapt to even longer time scales than such initialization, it is effective to approximate arbitrary desired time scales which is usually unknown a priori.

Pixel-by-pixel Image Recognition

Next, we evaluate the fast gate on the sequential image recognition task, where a pixel value is applied into recurrent models at each time step (Le, Jaitly, and Hinton 2015).

Setup. We use the usual order sequential MNIST (sMNIST) task and permuted order version (psMNIST) to introduce more complex and long-range dependencies. We also use the sequential CIFAR-10 (sCIFAR) task, which involves higher dimensional inputs and longer sequence length (i.e.,

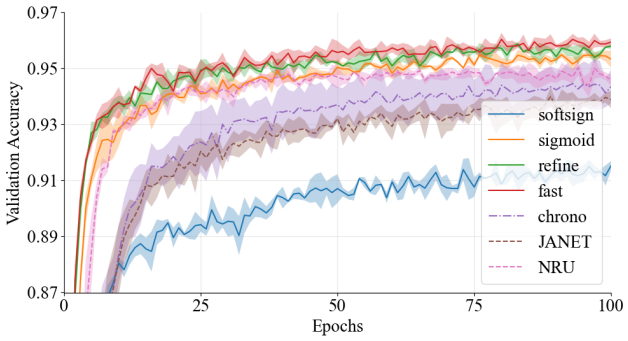

Figure 6: Validation accuracy for psMNIST

	$>$ 10K	1K-10K	100-1K	$<$ 100	All	Time
Sigmoid	7.25	27.72	170.25	2026.87	60.22	130
Refine	7.61	28.01	166.46	1936.02	60.50	185
Fast	7.43	27.70	166.68	1975.51	60.09	138

Table 3: Test perplexity for tokens across different frequency bins on Penn Treebank with training time (sec.) per epoch

3×1024) than MNIST. We train the LSTMs with the various gates, JANET, and NRU. See Appendix E.4 for training details. We set the hidden dimension as 512 on all models and the dimension of the memory cell in NRU as 144 following the original paper. We evaluate the averaged test accuracy with the standard deviation over three random seeds. We also report the computational time to train each model on the psMNIST task to compare the computational efficiency of the LSTM with the fast gate to other models.

Results. The results are shown in Tab. 2. The fast gate performs the best among various gates while the normalized softsign gate poorly performs. This is because the order of saturation of activation functions in the forget gate directly affects the learnability for long time scales. The LSTM with the fast gate also outperforms all the other baselines. Note that as chrono-initialization gives too heavy-tailed time scale distribution (Gu et al. 2020), the chrono-LSTM and JANET performs worse than simply initializing the gate bias as $b_f = 1$ (Sigmoid in the table). Since large model size tends to result in high accuracy on these tasks (Voelker, Kajić, and Eliasmith 2019; Erichson et al. 2021), we provide results of smaller models in Appendix F.3 for comparison. While the NRU achieves the highest accuracies on psMNIST with smaller model size, it performs worse than the LSTMs in Tab. 2. Therefore, performance of the LSTM seems to scale better than the NRU in model size. The refine gate requires more than 1.5 times longer processing time than the fast gate for training since it involves an auxiliary gate computation. The NRU requires longer processing time than the LSTMs due to its complicated state update rule. The learning curve for the psMNIST task in Fig. 6 shows that the fast gate reaches high accuracy in as few epochs as the refine gate. Combining with the results on the processing time per epoch, we conclude that the fast gate learns complex time scales the most efficiently among the gate functions.

Language Modeling

In natural language processing, performance of language models can suffer from difficulty in learning long time scales because predicting statistically rare words involves long time scales (Mahto et al. 2021). It is expected that using a forget gate function with a higher order of saturation improves learning to predict such rare words. We validate this effect in the following experiment.

Setup. We train and evaluate three-layer LSTM language models following a previous study (Mahto et al. 2021)[3] on the Penn Treebank (PTB) dataset, replacing every sigmoid forget gate in the baseline LSTM with the fast gate. We compare the LSTM with the fast gate against the LSTM with the sigmoid and refine gates and also NRU by replacing all LSTM modules with NRU modules under the same experimental setting. To evaluate the model performance on data involving different ranges of time scales, the test dataset is divided into four bins depending on their frequencies in the training dataset: more than 10,000, 1000-10,000, 100-1000, and fewer than 100 occurrences.

Results. The results are shown in Tab. 3. The result for the NRU is not in the table since the training diverges. Both refine and fast gates improve perplexity for less frequently observed words compared to the sigmoid gate. The model with the fast gate also achieves the lowest total perplexity. Since frequently observed words involve short-term dependencies, this result indicates that the fast gate improves model performance by learning a wide range of time scales that appear in practical tasks. Tab. 3 also shows the training time taken for one epoch. We observe that the refine gate has larger computational overhead than the fast gate, although the LSTM with the refine gate has the same number of parameters as other LSTMs by using the gate-tying trick (Appendix E.2). This is due to the two-stage computation for the gate value via the auxiliary gate, which cannot be parallelized, thus leads to slow computation. In summary, the fast gate can improve performance on data involving extremely long time scales without sacrificing the performance on data involving short time scales and with less computational overhead.

Conclusion

We analyzed the saturation problem in learning of gate functions in recurrent models. Against the common intuition that saturation of the activation function degrades training, we showed that strengthening the saturating behavior is effective in mitigating gradient vanishing of gate functions. We proposed the fast gate, which has a doubly exponential order of convergence with respect to inputs, by simply composing the hyperbolic sinusoidal function to the usual sigmoid function. We evaluated the trainability of the fast gate on data involving extremely long time scales. We empirically showed that the fast gate improves accuracy on benchmark tasks with little computational overhead. Our analytical approach is applicable to any other bounded activation functions that appear in the core of modules such as an attention mechanism. Thus, we expect that it can improve learnability of other neural networks beyond recurrent models.

[3] https://github.com/HuthLab/multi-timescale-LSTM-LMs.

References

Arjovsky, M.; Shah, A.; and Bengio, Y. 2016. Unitary evolution recurrent neural networks. In *International Conference on Machine Learning*, 1120–1128. PMLR.

Bengio, Y.; Léonard, N.; and Courville, A. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*.

Chandar, S.; Sankar, C.; Vorontsov, E.; Kahou, S. E.; and Bengio, Y. 2019. Towards non-saturating recurrent units for modelling long-term dependencies. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, 3280–3287.

Cho, K.; van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; and Bengio, Y. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1724–1734.

Erichson, N. B.; Azencot, O.; Queiruga, A.; Hodgkinson, L.; and Mahoney, M. W. 2021. Lipschitz recurrent neural networks. In *International Conference on Learning Representations*.

Gers, F. A.; Schmidhuber, J. A.; and Cummins, F. A. 2000. Learning to Forget: Continual Prediction with LSTM. *Neural computation*, 12(10): 2451–2471.

Greff, K.; Srivastava, R. K.; Koutník, J.; Steunebrink, B. R.; and Schmidhuber, J. 2016. LSTM: A search space odyssey. *IEEE Transactions on Neural Networks and Learning Systems*, 28(10): 2222–2232.

Gu, A.; Gulcehre, C.; Paine, T.; Hoffman, M.; and Pascanu, R. 2020. Improving the Gating Mechanism of Recurrent Neural Networks. In *International Conference on Machine Learning*, 3800–3809. PMLR.

Gulcehre, C.; Moczulski, M.; Denil, M.; and Bengio, Y. 2016. Noisy activation functions. In *International Conference on Machine Learning*, 3059–3068. PMLR.

Harold, K.; and George, Y. 2003. *Stochastic approximation and recursive algorithms and applications*, volume 35. Springer Science & Business Media.

Hochreiter, S.; and Schmidhuber, J. 1997. Long short-term memory. *Neural computation*, 9(8): 1735–1780.

Ioffe, S.; and Szegedy, C. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, 448–456. PMLR.

Le, Q. V.; Jaitly, N.; and Hinton, G. E. 2015. A simple way to initialize recurrent networks of rectified linear units. *arXiv preprint arXiv:1504.00941*.

Ling, S.; Liu, Y.; Salazar, J.; and Kirchhoff, K. 2020. Deep contextualized acoustic representations for semi-supervised speech recognition. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 6429–6433. IEEE.

Mahto, S.; Vo, V. A.; Turek, J. S.; and Huth, A. G. 2021. Multi-timescale representation learning in LSTM Language Models. *International Conference on Learning Representations*.

Ohno, K.; Kanai, S.; and Ida, Y. 2022. Fast Saturating Gate for Learning Long Time Scales with Recurrent Neural Networks. *arXiv preprint arXiv:2210.01348*.

Pascanu, R.; Mikolov, T.; and Bengio, Y. 2013. On the difficulty of training recurrent neural networks. In *International Conference on Machine Learning*, 1310–1318. PMLR.

Tallec, C.; and Ollivier, Y. 2018. Can recurrent neural networks warp time? In *International Conference on Learning Representations*.

Van Der Westhuizen, J.; and Lasenby, J. 2018. The unreasonable effectiveness of the forget gate. *arXiv preprint arXiv:1804.04849*.

Voelker, A.; Kajić, I.; and Eliasmith, C. 2019. Legendre memory units: Continuous-time representation in recurrent neural networks. *Advances in neural information processing systems*, 32.

Zhu, L.; Tran, D.; Sevilla-Lara, L.; Yang, Y.; Feiszli, M.; and Wang, H. 2020. FASTER recurrent networks for efficient video classification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 13098–13105.