Advancing Training Efficiency of Deep Spiking Neural Networks through Rate-based Backpropagation

Chengting Yu[1], Lei Liu[1], Gaoang Wang[1], Erping Li[1], Aili Wang[1]

[1]Zhejiang University

Background & Motivation

Training Challenges in SNNs training

- Backpropagation through Time (BPTT)
- Complexity: SNNs require complex temporal-spatial computation graphs.
- Resource Intensive: High computational and memory demands limit scalability and practical application in large and deep networks.

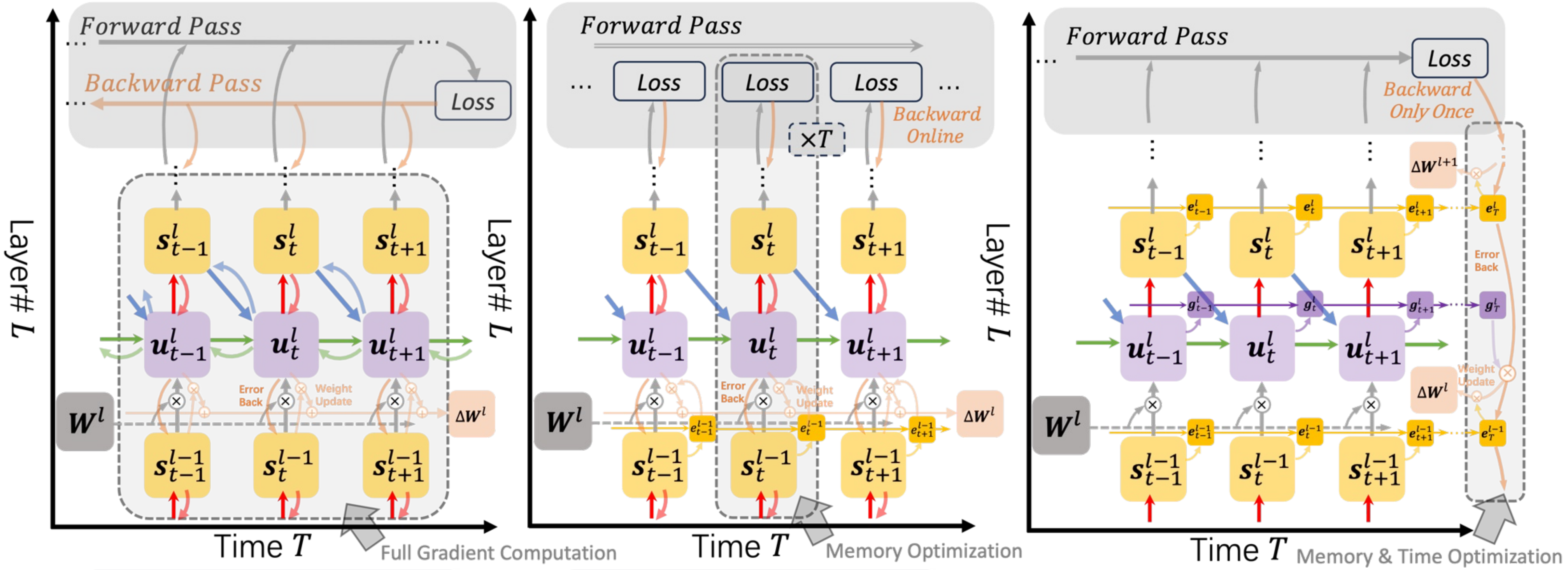
Rate Coding in spike representation

- Information encoding based on the frequency of neuronal spikes.
- Predominant form of data representation in SNNs.
- Rate Coding in BPTT: SNNs trained with BPTT on static benchmarks typically utilize rate coding, showing similarities with ANNs.

Innovation: Rate-based Backpropagation

- Leverage the effectiveness and dominance of rate coding.
- Targeted training only based rate could offer a high cost-effectiveness ratio.
- Method: Decouples BPTT by approximating rate coding, simplifying computations into a single spatial backpropagation.

Comparison of Training methods



(a)	(b)	(c)
Backward Memory: $\mathcal{O}(LT)$	Backward Memory: $\mathcal{O}(L)$	Backward Memory: $\mathcal{O}(L)$
Backward Time: $\mathcal{O}(LT)$	Backward Time: $\mathcal{O}(LT)$	Backward Time: $\mathcal{O}(L)$

(a) Standard BPTT Training

- Full gradient computations among the temporal and spatial dimensions.

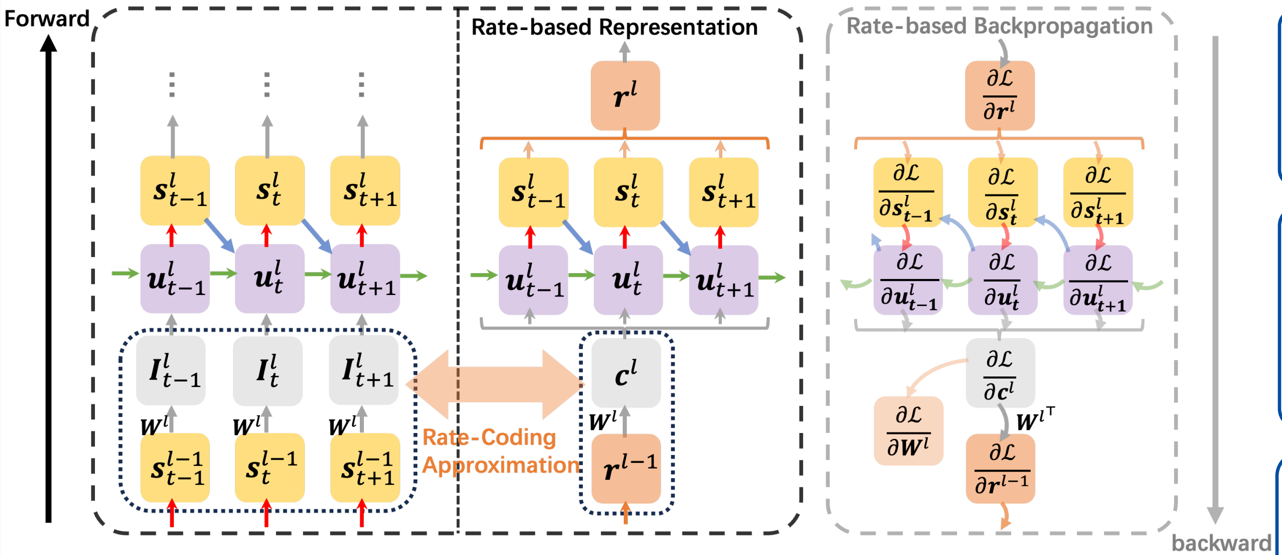
(b) Online Training

- Reduces memory costs by ignoring inter-temporal connections.

(c) Rate-based Backpropagation

- Simplifies training by conducting backward computations once, based solely on spatial dimensions.

Decoupling BPTT



Forward Pass of Spiking Neural Networks with the Standard Iterative LIF Neurons

$$u_t^l = \lambda(u_{t-1}^l - V_{\text{th}} s_{t-1}^l) + W^l s_t^{l-1}, \quad s_t^l = H(u_t^l - V_{\text{th}}) \qquad I_t^l = W^l s_t^{l-1}$$

Rate-based Representation

$$r^l = \mathbb{E}[s_t^l] = \frac{1}{T}\sum_{t<T} s_t^l.$$

$$c^l = \mathbb{E}[I_t^l] = \mathbb{E}[W^l s_t^{l-1}] = W^l \mathbb{E}[s_t^{l-1}] = W^l r^{l-1}.$$

Rate-coding Approximation

$$I_t^l \approx c^l \Rightarrow \frac{\partial I_t^l}{\partial c^l} = Id$$

Straight-Through Estimator (STE)

$$\frac{\partial c^l}{\partial r^{l-1}} = W^{l^\top} \cdot \boxed{c^L} \qquad \left(\frac{\partial r^l}{\partial c^l}\right)_{\text{rate}} \equiv \sum_\tau \left(\frac{\partial(\mathbb{E}[s_t^l])}{\partial I_\tau^l}\frac{\partial I_\tau^l}{\partial c^l}\right) = \frac{1}{T}\sum_t\sum_\tau\left(\frac{\partial s_t^l}{\partial I_\tau^l}\right) = \mathbb{E}[\varkappa_t^l]$$

$$\frac{\partial \mathcal{L}}{\partial c^L} \text{ from } \mathcal{L} = \ell\left(\boxed{\frac{1}{T}\sum_{t=1}^{T} o_t}, y\right)$$

Handling temporal dependency

$$\varkappa_t^l = \sum_\tau \frac{\partial s_\tau^l}{\partial I_t^l} = \left(\frac{\partial s_t^l}{\partial u_t^l} + \sum_{\tau>t}\frac{\partial s_\tau^l}{\partial u_\tau^l}\prod_{i=\tau-1}^{t}\left(\frac{\partial u_{i+1}^l}{\partial u_i^l} + \frac{\partial u_{i+1}^l}{\partial s_i^l}\frac{\partial s_i^l}{\partial u_i^l}\right)\right)$$

Derivation of Rate-base Gradients

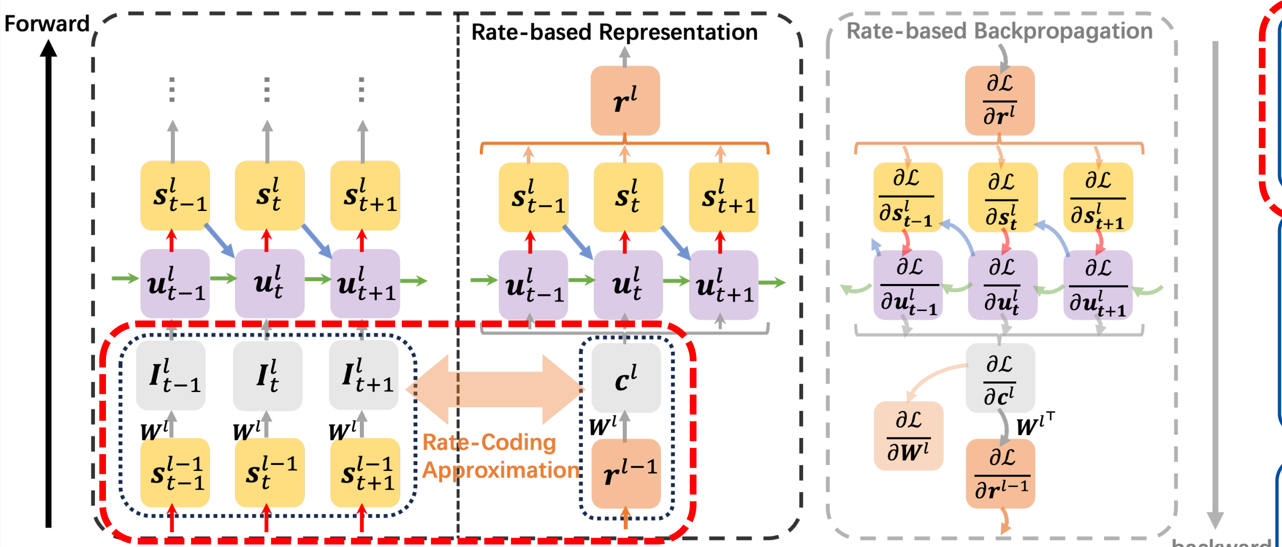
Rate-based Backpropagation

Error Back: $$\left(\frac{\partial \mathcal{L}}{\partial c^l}\right)_{\text{rate}} = \left(\frac{\partial \mathcal{L}}{\partial c^L}\prod_{i=L-1}^{l}\left(\frac{\partial c^{i+1}}{\partial r^i}\left(\frac{\partial r^i}{\partial c^i}\right)_{\text{rate}}\right)\right) = \left(\frac{\partial \mathcal{L}}{\partial c^L}\prod_{i=L-1}^{l}\left(W^{i^\top}\mathbb{E}[\varkappa_t^l]\right)\right)$$

Weights Descent: $$(\nabla_{W^l}\mathcal{L})_{\text{rate}} \equiv \frac{\partial \mathcal{L}}{\partial c^l}\frac{\partial c^l}{\partial W^l} = \frac{\partial \mathcal{L}}{\partial c^l} r^{l-1^\top}$$

Independent of the Temporal Dimension!!!

Decoupling BPTT



Forward Pass of Spiking Neural Networks with the Standard Iterative LIF Neurons

$$u_t^l = \lambda(u_{t-1}^l - V_{\text{th}} s_{t-1}^l) + W^l s_t^{l-1}, \quad s_t^l = H(u_t^l - V_{\text{th}}) \qquad I_t^l = W^l s_t^{l-1}$$

Rate-based Representation

$$r^l = \mathbb{E}[s_t^l] = \frac{1}{T}\sum_{t<T} s_t^l.$$

$$c^l = \mathbb{E}[I_t^l] = \mathbb{E}[W^l s_t^{l-1}] = W^l \mathbb{E}[s_t^{l-1}] = W^l r^{l-1}.$$

Rate-coding Approximation

$$I_t^l \approx c^l \Rightarrow \frac{\partial I_t^l}{\partial c^l} = Id$$

Straight-Through Estimator (STE)

$$\frac{\partial c^l}{\partial r^{l-1}} = W^{l\top} \cdot c^L \qquad \left(\frac{\partial r^l}{\partial c^l}\right)_{\text{rate}} \equiv \sum_\tau \left(\frac{\partial(\mathbb{E}[s_t^l])}{\partial I_\tau^l}\frac{\partial I_\tau^l}{\partial c^l}\right) = \frac{1}{T}\sum_t\sum_\tau\left(\frac{\partial s_t^l}{\partial I_\tau^l}\right) = \mathbb{E}[\varkappa_t^l]$$

$$\frac{\partial \mathcal{L}}{\partial c^L} \text{ from } \mathcal{L} = \ell\left(\frac{1}{T}\sum_{t=1}^{T} o_t, y\right)$$

Handling temporal dependency

$$\varkappa_t^l = \sum_\tau \frac{\partial s_t^l}{\partial I_\tau^l} = \left(\frac{\partial s_t^l}{\partial u_t^l} + \sum_{\tau>t}\frac{\partial s_\tau^l}{\partial u_\tau^l}\prod_{i=\tau-1}^{t}\left(\frac{\partial u_{i+1}^l}{\partial u_i^l} + \frac{\partial u_{i+1}^l}{\partial s_i^l}\frac{\partial s_i^l}{\partial u_i^l}\right)\right)$$

Derivation of Rate-base Gradients

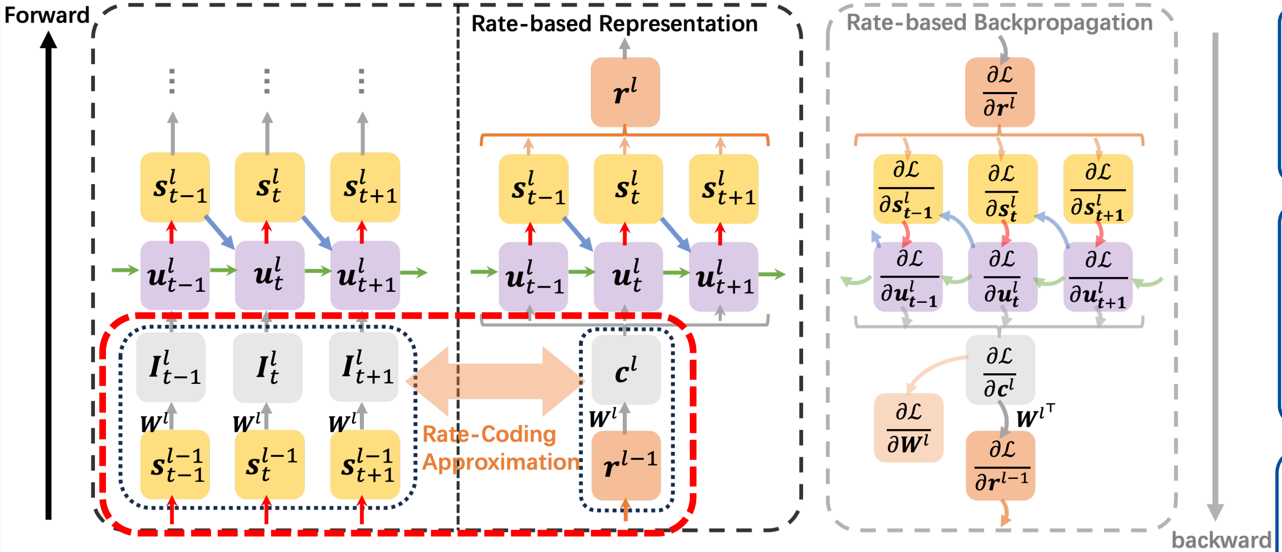
Rate-based Backpropagation

Error Back: $\left(\frac{\partial \mathcal{L}}{\partial c^l}\right)_{\text{rate}} = \left(\frac{\partial \mathcal{L}}{\partial c^L}\prod_{i=L-1}^{l}\left(\frac{\partial c^{i+1}}{\partial r^i}\left(\frac{\partial r^i}{\partial c^i}\right)_{\text{rate}}\right)\right) = \left(\frac{\partial \mathcal{L}}{\partial c^L}\prod_{i=L-1}^{l}\left(W^{i\top}\mathbb{E}[\varkappa_t^l]\right)\right)$

Weights Descent: $(\nabla_{W^l}\mathcal{L})_{\text{rate}} \equiv \frac{\partial \mathcal{L}}{\partial c^l}\frac{\partial c^l}{\partial W^l} = \frac{\partial \mathcal{L}}{\partial c^l} r^{l-1\top}$

Independent of the Temporal Dimension!!!

Decoupling BPTT



Forward Pass of Spiking Neural Networks with the Standard Iterative LIF Neurons

$$u_t^l = \lambda(u_{t-1}^l - V_{\text{th}} s_{t-1}^l) + W^l s_t^{l-1}, \quad s_t^l = H(u_t^l - V_{\text{th}}) \qquad I_t^l = W^l s_t^{l-1}$$

Rate-based Representation

$$r^l = \mathbb{E}[s_t^l] = \frac{1}{T}\sum_{t<T} s_t^l.$$

$$c^l = \mathbb{E}[I_t^l] = \mathbb{E}[W^l s_t^{l-1}] = W^l \mathbb{E}[s_t^{l-1}] = W^l r^{l-1}.$$

Rate-coding Approximation

$$I_t^l \approx c^l \Rightarrow \frac{\partial I_t^l}{\partial c^l} = Id$$

Straight-Through Estimator (STE)

$$\boxed{\frac{\partial c^l}{\partial r^{l-1}} = W^{l\top}} \quad c^L \qquad \left(\frac{\partial r^l}{\partial c^l}\right)_{\text{rate}} \equiv \sum_\tau \left(\frac{\partial(\mathbb{E}[s_t^l])}{\partial I_\tau^l}\frac{\partial I_\tau^l}{\partial c^l}\right) = \frac{1}{T}\sum_t\sum_\tau\left(\frac{\partial s_t^l}{\partial I_\tau^l}\right) = \mathbb{E}[\varkappa_t^l]$$

$\frac{\partial \mathcal{L}}{\partial c^L}$ from $\mathcal{L} = \ell\left(\frac{1}{T}\sum_{t=1}^{T} o_t, y\right)$

Handling temporal dependency

$$\varkappa_t^l = \sum_\tau \frac{\partial s_\tau^l}{\partial I_t^l} = \left(\frac{\partial s_t^l}{\partial u_t^l} + \sum_{\tau>t}\frac{\partial s_\tau^l}{\partial u_\tau^l}\prod_{i=\tau-1}^{t}\left(\frac{\partial u_{i+1}^l}{\partial u_i^l} + \frac{\partial u_{i+1}^l}{\partial s_i^l}\frac{\partial s_i^l}{\partial u_i^l}\right)\right)$$

Derivation of Rate-base Gradients

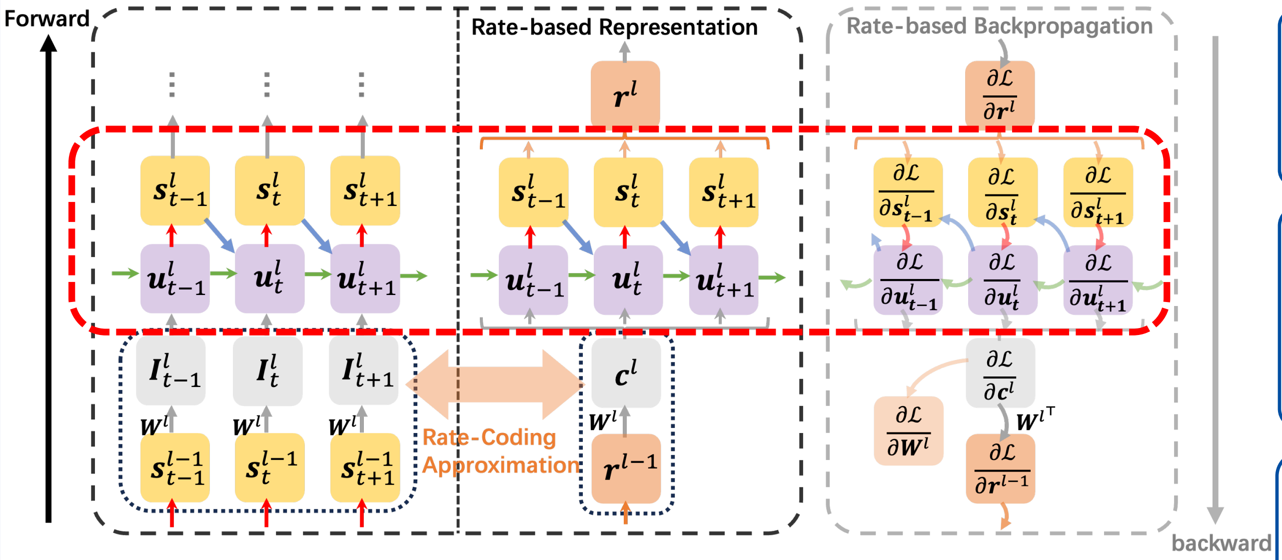
Rate-based Backpropagation

Error Back: $\left(\frac{\partial\mathcal{L}}{\partial c^l}\right)_{\text{rate}} = \left(\frac{\partial \mathcal{L}}{\partial c^L}\prod_{i=L-1}^{l}\left(\frac{\partial c^{i+1}}{\partial r^i}\left(\frac{\partial r^i}{\partial c^i}\right)_{\text{rate}}\right)\right) = \left(\frac{\partial\mathcal{L}}{\partial c^L}\prod_{i=L-1}^{l}\left(W^{i\top}\mathbb{E}[\varkappa_t^l]\right)\right)$

Weights Descent: $(\nabla_{W^l}\mathcal{L})_{\text{rate}} \equiv \frac{\partial\mathcal{L}}{\partial c^l}\frac{\partial c^l}{\partial W^l} = \frac{\partial \mathcal{L}}{\partial c^l} r^{l-1\top}$

Independent of the Temporal Dimension!!!

Decoupling BPTT



Forward Pass of Spiking Neural Networks with the Standard Iterative LIF Neurons

$$u_t^l = \lambda(u_{t-1}^l - V_{\text{th}} s_{t-1}^l) + W^l s_t^{l-1}, \quad s_t^l = H(u_t^l - V_{\text{th}}) \qquad I_t^l = W^l s_t^{l-1}$$

Rate-based Representation

$$r^l = \mathbb{E}[s_t^l] = \tfrac{1}{T}\textstyle\sum_{t<T} s_t^l.$$

$$c^l = \mathbb{E}[I_t^l] = \mathbb{E}[W^l s_t^{l-1}] = W^l \mathbb{E}[s_t^{l-1}] = W^l r^{l-1}.$$

Rate-coding Approximation

$$I_t^l \approx c^l \Rightarrow \frac{\partial I_t^l}{\partial c^l} = Id$$

Straight-Through Estimator (STE)

Derivation of Rate-base Gradients

$$\frac{\partial c^l}{\partial r^{l-1}} = W^{l\top} \cdot c^L \qquad \left(\frac{\partial r^l}{\partial c^l}\right)_{\text{rate}} \equiv \sum_\tau \left(\frac{\partial(\mathbb{E}[s_t^l])}{\partial I_\tau^l}\frac{\partial I_\tau^l}{\partial c^l}\right) = \frac{1}{T}\sum_t\sum_\tau\left(\frac{\partial s_t^l}{\partial I_\tau^l}\right) = \mathbb{E}[\varkappa_t^l]$$

$\frac{\partial \mathcal{L}}{\partial c^L}$ from $\mathcal{L} = \ell\left(\frac{1}{T}\sum_{t=1}^{T} o_t, y\right)$

Handling temporal dependency

$$\varkappa_t^l = \sum_\tau \frac{\partial s_\tau^l}{\partial I_t^l} = \left(\frac{\partial s_t^l}{\partial u_t^l} + \sum_{\tau>t}\frac{\partial s_\tau^l}{\partial u_\tau^l}\prod_{i=\tau-1}^{t}\left(\frac{\partial u_{i+1}^l}{\partial u_i^l} + \frac{\partial u_{i+1}^l}{\partial s_i^l}\frac{\partial s_i^l}{\partial u_i^l}\right)\right)$$

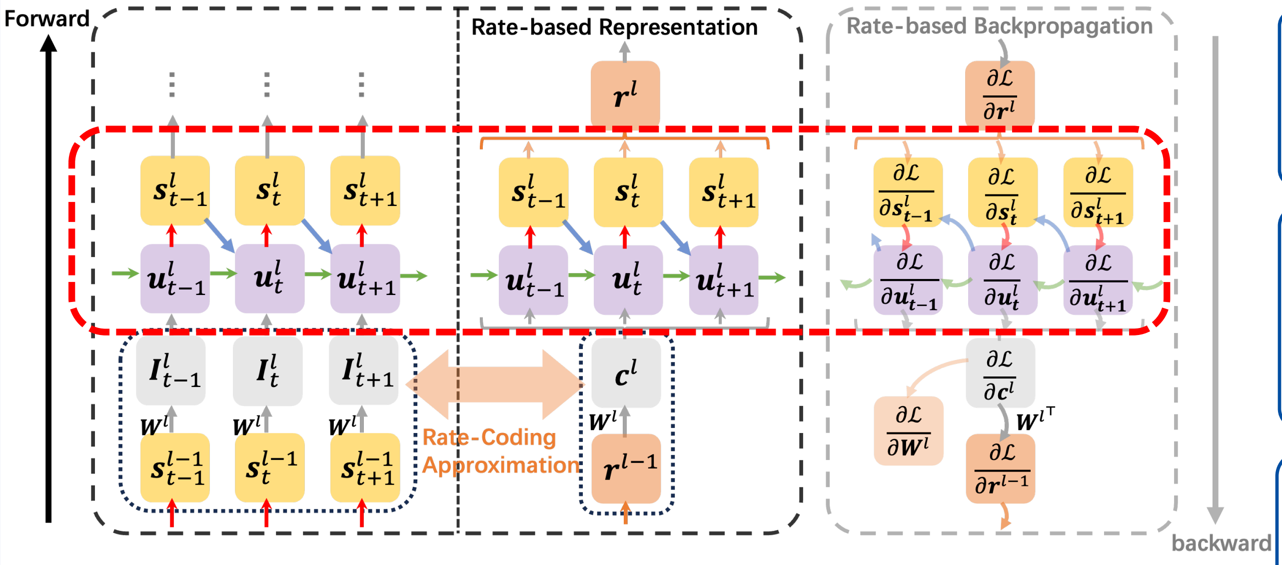
Rate-based Backpropagation

Error Back: $\left(\frac{\partial \mathcal{L}}{\partial c^l}\right)_{\text{rate}} = \left(\frac{\partial \mathcal{L}}{\partial c^L}\prod_{i=L-1}^{l}\left(\frac{\partial c^{i+1}}{\partial r^i}\left(\frac{\partial r^i}{\partial c^i}\right)_{\text{rate}}\right)\right) = \left(\frac{\partial \mathcal{L}}{\partial c^L}\prod_{i=L-1}^{l}\left(W^{i\top}\mathbb{E}[\varkappa_t^l]\right)\right)$

Weights Descent: $(\nabla_{W^l}\mathcal{L})_{\text{rate}} \equiv \frac{\partial \mathcal{L}}{\partial c^l}\frac{\partial c^l}{\partial W^l} = \frac{\partial \mathcal{L}}{\partial c^l} r^{l-1\top}$

Independent of the Temporal Dimension!!!

Decoupling BPTT



Forward Pass of Spiking Neural Networks with the Standard Iterative LIF Neurons

$$\boldsymbol{u}_t^l = \lambda(\boldsymbol{u}_{t-1}^l - V_{\text{th}}\boldsymbol{s}_{t-1}^l) + \boldsymbol{W}^l \boldsymbol{s}_t^{l-1}, \quad \boldsymbol{s}_t^l = H(\boldsymbol{u}_t^l - V_{\text{th}}) \qquad \boldsymbol{I}_t^l = \boldsymbol{W}^l \boldsymbol{s}_t^{l-1}$$

Rate-based Representation

$$\boldsymbol{r}^l = \mathbb{E}[\boldsymbol{s}_t^l] = \frac{1}{T}\sum_{t<T} \boldsymbol{s}_t^l.$$

$$\boldsymbol{c}^l = \mathbb{E}[\boldsymbol{I}_t^l] = \mathbb{E}[\boldsymbol{W}^l \boldsymbol{s}_t^{l-1}] = \boldsymbol{W}^l \mathbb{E}[\boldsymbol{s}_t^{l-1}] = \boldsymbol{W}^l \boldsymbol{r}^{l-1}.$$

Rate-coding Approximation

$$\boldsymbol{I}_t^l \approx \boldsymbol{c}^l \Rightarrow \frac{\partial \boldsymbol{I}_t^l}{\partial \boldsymbol{c}^l} = Id$$

Straight-Through Estimator (STE)

$$\frac{\partial \boldsymbol{c}^l}{\partial \boldsymbol{r}^{l-1}} = \boldsymbol{W}^{l\top} \cdot \boxed{\boldsymbol{c}^L} \qquad \left(\frac{\partial \boldsymbol{r}^l}{\partial \boldsymbol{c}^l}\right)_{\text{rate}} \equiv \sum_\tau \left(\frac{\partial(\mathbb{E}[\boldsymbol{s}_t^l])}{\partial \boldsymbol{I}_\tau^l}\frac{\partial \boldsymbol{I}_\tau^l}{\partial \boldsymbol{c}^l}\right) = \frac{1}{T}\sum_t\sum_\tau\left(\frac{\partial \boldsymbol{s}_t^l}{\partial \boldsymbol{I}_\tau^l}\right) = \mathbb{E}[\boldsymbol{\varkappa}_t^l]$$

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{c}^L} \text{ from } \mathcal{L} = \ell\left(\boxed{\frac{1}{T}\sum_{t=1}^T \boldsymbol{o}_t}, \boldsymbol{y}\right)$$

Handling temporal dependency

$$\boldsymbol{\varkappa}_t^l = \sum_\tau \frac{\partial s_\tau^l}{\partial I_t^l} = \left(\frac{\partial s_t^l}{\partial u_t^l} + \sum_{\tau>t}\frac{\partial s_\tau^l}{\partial u_\tau^l}\prod_{i=\tau-1}^{t}\left(\frac{\partial u_{i+1}^l}{\partial u_i^l} + \frac{\partial u_{i+1}^l}{\partial s_i^l}\frac{\partial s_i^l}{\partial u_i^l}\right)\right)$$

Derivation of Rate-base Gradients

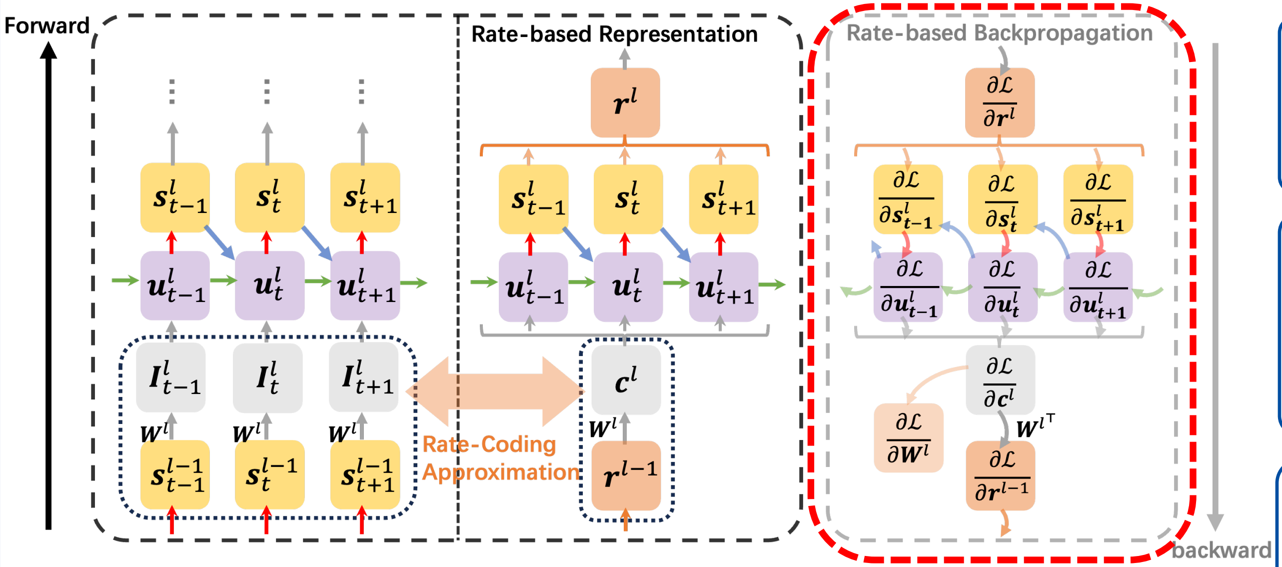
Rate-based Backpropagation

Error Back: $\left(\frac{\partial \mathcal{L}}{\partial \boldsymbol{c}^l}\right)_{\text{rate}} = \left(\frac{\partial \mathcal{L}}{\partial \boldsymbol{c}^L}\prod_{i=L-1}^{l}\left(\frac{\partial \boldsymbol{c}^{i+1}}{\partial \boldsymbol{r}^i}\left(\frac{\partial \boldsymbol{r}^i}{\partial \boldsymbol{c}^i}\right)_{\text{rate}}\right)\right) = \left(\frac{\partial \mathcal{L}}{\partial \boldsymbol{c}^L}\prod_{i=L-1}^{l}\left(\boldsymbol{W}^{i\top}\mathbb{E}[\boldsymbol{\varkappa}_t^l]\right)\right)$

Weights Descent: $(\nabla_{\boldsymbol{W}^l}\mathcal{L})_{\text{rate}} \equiv \frac{\partial \mathcal{L}}{\partial \boldsymbol{c}^l}\frac{\partial \boldsymbol{c}^l}{\partial \boldsymbol{W}^l} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{c}^l}\boldsymbol{r}^{l-1\top}$

Independent of the Temporal Dimension!!!

Decoupling BPTT



Forward Pass of Spiking Neural Networks with the Standard Iterative LIF Neurons

$$u_t^l = \lambda(u_{t-1}^l - V_{\text{th}} s_{t-1}^l) + W^l s_t^{l-1}, \quad s_t^l = H(u_t^l - V_{\text{th}}) \qquad I_t^l = W^l s_t^{l-1}$$

Rate-based Representation

$$r^l = \mathbb{E}[s_t^l] = \frac{1}{T}\sum_{t<T} s_t^l.$$

$$c^l = \mathbb{E}[I_t^l] = \mathbb{E}[W^l s_t^{l-1}] = W^l \mathbb{E}[s_t^{l-1}] = W^l r^{l-1}.$$

Rate-coding Approximation

$$I_t^l \approx c^l \Rightarrow \frac{\partial I_t^l}{\partial c^l} = Id$$

Straight-Through Estimator (STE)

$$\frac{\partial c^l}{\partial r^{l-1}} = W^{l^\top} \qquad \left(\frac{\partial r^l}{\partial c^l}\right)_{\text{rate}} \equiv \sum_\tau \left(\frac{\partial(\mathbb{E}[s_t^l])}{\partial I_\tau^l}\frac{\partial I_\tau^l}{\partial c^l}\right) = \frac{1}{T}\sum_t\sum_\tau\left(\frac{\partial s_t^l}{\partial I_\tau^l}\right) = \mathbb{E}[\varkappa_t^l]$$

$$\frac{\partial \mathcal{L}}{\partial c^L} \text{ from } \mathcal{L} = \ell\left(\frac{1}{T}\sum_{t=1}^{T} o_t, y\right)$$

Handling temporal dependency

$$\varkappa_t^l = \sum_\tau \frac{\partial s_\tau^l}{\partial I_t^l} = \left(\frac{\partial s_t^l}{\partial u_t^l} + \sum_{\tau>t}\frac{\partial s_\tau^l}{\partial u_\tau^l}\prod_{i=\tau-1}^{t}\left(\frac{\partial u_{i+1}^l}{\partial u_i^l} + \frac{\partial u_{i+1}^l}{\partial s_i^l}\frac{\partial s_i^l}{\partial u_i^l}\right)\right)$$

Derivation of Rate-base Gradients

Rate-based Backpropagation

Error Back: $$\left(\frac{\partial\mathcal{L}}{\partial c^l}\right)_{\text{rate}} = \left(\frac{\partial\mathcal{L}}{\partial c^L}\prod_{i=L-1}^{l}\left(\frac{\partial c^{i+1}}{\partial r^i}\left(\frac{\partial r^i}{\partial c^i}\right)_{\text{rate}}\right)\right) = \left(\frac{\partial\mathcal{L}}{\partial c^L}\prod_{i=L-1}^{l}\left(W^{i^\top}\mathbb{E}[\varkappa_t^l]\right)\right)$$

Weights Descent: $$(\nabla_{W^l}\mathcal{L})_{\text{rate}} \equiv \frac{\partial\mathcal{L}}{\partial c^l}\frac{\partial c^l}{\partial W^l} = \frac{\partial\mathcal{L}}{\partial c^l} r^{l-1^\top}$$

Independent of the Temporal Dimension!!!

Backwards with Local Eligibility Traces

Local Iterative Eligibility Traces

$$\boldsymbol{e}_t^l = \tfrac{1}{t}((t-1)\boldsymbol{e}_{t-1}^l + \boldsymbol{s}_t^l)$$
$$\boldsymbol{g}_t^l = \tfrac{1}{t}((t-1)\boldsymbol{g}_{t-1}^l + \frac{\partial \boldsymbol{s}_t^l}{\partial \boldsymbol{u}_t^l}\boldsymbol{\rho}_t)$$
$$\boldsymbol{\rho}_t^l = 1 + \boldsymbol{\rho}_{t-1}^l \left(\frac{\partial \boldsymbol{u}_t^l}{\partial \boldsymbol{u}_{t-1}^l} + \frac{\partial \boldsymbol{u}_t^l}{\partial \boldsymbol{s}_{t-1}^l}\frac{\partial \boldsymbol{s}_{t-1}^l}{\partial \boldsymbol{u}_{t-1}^l} \right)$$

Eligibility Traces for Rate-based Representation

$$\boldsymbol{e}_t^l = \tfrac{1}{t}((t-1)\boldsymbol{e}_{t-1}^l + \boldsymbol{s}_t^l) \Rightarrow \boldsymbol{r}^l = \boldsymbol{e}_T^l$$
$$\boldsymbol{g}_t^l = \tfrac{1}{t}((t-1)\boldsymbol{g}_{t-1}^l + \frac{\partial \boldsymbol{s}_t^l}{\partial \boldsymbol{u}_t^l}\boldsymbol{\rho}_t) \Rightarrow \boldsymbol{g}_T^l = \mathbb{E}[\frac{\partial \boldsymbol{s}_t^l}{\partial \boldsymbol{u}_t^l}\boldsymbol{\rho}_t^l] = \mathbb{E}[\boldsymbol{\varkappa}_t^l]$$

$$\sum_t \boldsymbol{\varkappa}_t^l = \sum_t \left(\frac{\partial \boldsymbol{s}_t^l}{\partial \boldsymbol{u}_t^l} + \sum_{\tau>t} \left(\frac{\partial \boldsymbol{s}_\tau^l}{\partial \boldsymbol{u}_\tau^l} \prod_{i=\tau-1}^{t} \left(\frac{\partial \boldsymbol{u}_{i+1}^l}{\partial \boldsymbol{u}_i^l} + \frac{\partial \boldsymbol{u}_{i+1}^l}{\partial \boldsymbol{s}_i^l}\frac{\partial \boldsymbol{s}_i^l}{\partial \boldsymbol{u}_i^l} \right) \right) \right)$$
$$= \sum_t \left(\frac{\partial \boldsymbol{s}_t^l}{\partial \boldsymbol{u}_t^l} \left(1 + \sum_{\tau<t} \prod_{i=t-1}^{\tau} \left(\frac{\partial \boldsymbol{u}_{i+1}^l}{\partial \boldsymbol{u}_i^l} + \frac{\partial \boldsymbol{u}_{i+1}^l}{\partial \boldsymbol{s}_i^l}\frac{\partial \boldsymbol{s}_i^l}{\partial \boldsymbol{u}_i^l} \right) \right) \right) = \sum_t \left(\frac{\partial \boldsymbol{s}_t^l}{\partial \boldsymbol{u}_t^l}\boldsymbol{\rho}_t^l \right)$$

Backwards with Local Eligibility Traces

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{r}^l} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{c}^{l+1}} {\boldsymbol{W}^{l+1}}^\top$$
$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{c}^l} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{r}^l} \boldsymbol{g}_T^l$$
$$\nabla_{\boldsymbol{W}^l}\mathcal{L} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{c}^l} (\boldsymbol{e}_T^{l-1})^\top$$

Algorithm 1: Single Training Iteration of the Rate-based Backpropagation

---

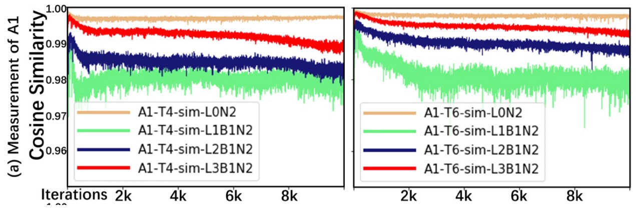
Input: Timesteps T; Network depth L; Trainable parameters $\{\boldsymbol{W}^l\}_{l\le L}$; Training Mini-batch $\{(\boldsymbol{x}_t^0, \boldsymbol{y})\}$; Training Mode $\boldsymbol{rate}_S$ or $\boldsymbol{rate}_M$.
Output: Updated parameters $\{\boldsymbol{W}^l\}_{l\le L}$
Initialize input spikes $\boldsymbol{s}_t^0 = \boldsymbol{x}_t^0$ for all $t \in [1, T]$.
Initialize $\boldsymbol{\rho}_0^l = 0$, $\boldsymbol{g}_0^l = 0$, $\boldsymbol{e}_0^l = 0$ for all $l \in [1, L]$.
for $t = 1$ *to* T **do**
  for $l = 1$ *to* L **do**
    Compute input currents through linear operators $\boldsymbol{I}_t^l = \boldsymbol{W}^l \boldsymbol{s}_t^{l-1}$;
    Initialize $\boldsymbol{\rho}_0^l = 0$, $\boldsymbol{g}_0^l = 0$, $\boldsymbol{e}_0^l = 0$;
    Compute output spikes $\boldsymbol{s}_t^l$ from $\boldsymbol{I}_t^l$ following neural dynamics in Eq. (1);
    Compute the eligibility trace $\boldsymbol{\rho}_t^l = 1 + \boldsymbol{\rho}_{t-1}^l \left(\frac{\partial \boldsymbol{u}_t^l}{\partial \boldsymbol{u}_{t-1}^l} + \frac{\partial \boldsymbol{u}_t^l}{\partial \boldsymbol{s}_{t-1}^l}\frac{\partial \boldsymbol{s}_{t-1}^l}{\partial \boldsymbol{u}_{t-1}^l} \right)$ in Eq. (8);
    Accumulate $\boldsymbol{e}_t^l = \frac{1}{t}((t-1)\boldsymbol{e}_{t-1}^l + \boldsymbol{s}_t^l)$;
    Accumulate $\boldsymbol{g}_t^l = \frac{1}{t}((t-1)\boldsymbol{g}_{t-1}^l + \frac{\partial \boldsymbol{s}_t^l}{\partial \boldsymbol{u}_t^l}\boldsymbol{\rho}_t)$;
    Save $\boldsymbol{u}_t^l$, $\boldsymbol{s}_t^l$ for neuron states;
    Save $\boldsymbol{g}_t^l$, $\boldsymbol{e}_t^l$, $\boldsymbol{\rho}_t^l$ as eligibility traces.
  end
end
Compute the outputs gradient $\frac{\partial \mathcal{L}}{\partial \boldsymbol{c}^L}$ from the objective function.
for $l = L-1$ *to* 1 **do**
  Compute error backpropagated through the linear part $\frac{\partial \mathcal{L}}{\partial \boldsymbol{r}^l} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{c}^{l+1}} {\boldsymbol{W}^{l+1}}^\top$;
  Compute error backpropagated through the neuron part $\frac{\partial \mathcal{L}}{\partial \boldsymbol{c}^l} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{r}^l}\boldsymbol{g}_T^l$;
  Compute the weight gradients $\nabla_{\boldsymbol{W}^l}\mathcal{L} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{c}^l}(\boldsymbol{e}_T^{l-1})^\top$;
  Update parameters $\{\boldsymbol{W}^l\}_{l\le L}$ based on the gradient-based optimizer.
end

Connecting Error Backward to BPTT

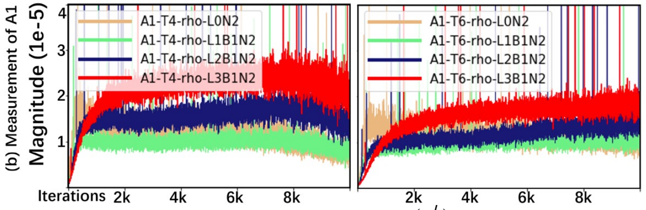
Equivalent Conditions

Theorem 1. *Given $\boldsymbol{\delta}_t^{(\boldsymbol{s}^l)} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{s}_t^l}$ that refers to gradients computed following the chain rule of BPTT in Eq. (2), and $\boldsymbol{\kappa}_t^l = \sum_\tau \frac{\partial \boldsymbol{s}_t^l}{\partial \boldsymbol{I}_\tau^l}$ (where $\mathbb{E}\left[\boldsymbol{\kappa}_t^l\right] = \mathbb{E}\left[\boldsymbol{\varkappa}_t^l\right]$ in Eq.(6-7)), if $\mathbb{E}\left[\boldsymbol{\delta}_t^{(\boldsymbol{s}^l)} \boldsymbol{\kappa}_t^l\right] = \mathbb{E}\left[\boldsymbol{\delta}_t^{(\boldsymbol{s}^l)}\right] \mathbb{E}\left[\boldsymbol{\kappa}_t^l\right]$ holds for $\forall l$, we have $\mathbb{E}\left[\boldsymbol{\delta}_t^{(\boldsymbol{s}^l)}\right] = \left(\frac{\partial \mathcal{L}}{\partial \boldsymbol{r}^l}\right)_{rate}$. Furthermore, given $\boldsymbol{\delta}_t^{(\boldsymbol{I}^l)} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{I}_t^l}$, if $\mathbb{E}\left[\boldsymbol{\delta}_t^{(\boldsymbol{I}^l)} \boldsymbol{s}_t^{l-1}\right] = \mathbb{E}\left[\boldsymbol{\delta}_t^{(\boldsymbol{I}^l)}\right] \mathbb{E}[\boldsymbol{s}_t^{l-1}]$ for $\forall l$, we then obtain $(\nabla_{\boldsymbol{W}^l}\mathcal{L})_{rate} = \frac{1}{T}(\nabla_{\boldsymbol{W}^l}\mathcal{L})$. Here, $\mathbb{E}[\boldsymbol{x}_t] = \frac{1}{T}\sum_t \boldsymbol{x}_t$ refers the mean value of tensor $\boldsymbol{x}_t$ over temporal dimension T.*

$$\mathbb{E}\left[\boldsymbol{\delta}_t^{(\boldsymbol{s}^l)} \boldsymbol{\kappa}_t^l\right] = \mathbb{E}\left[\boldsymbol{\delta}_t^{(\boldsymbol{s}^l)}\right] \mathbb{E}\left[\boldsymbol{\kappa}_t^l\right] \text{ (A1)}$$



$$cos\langle\mathbb{E}\left[\boldsymbol{\delta}_t^{(\boldsymbol{s}^l)} \boldsymbol{\kappa}_t^l\right], \mathbb{E}\left[\boldsymbol{\delta}_t^{(\boldsymbol{s}^l)}\right] \mathbb{E}\left[\boldsymbol{\kappa}_t^l\right]\rangle$$



$$\rho = \frac{COV(\boldsymbol{\kappa}_t, \boldsymbol{\delta}_t^{(\boldsymbol{s}^l)})}{\sqrt{\mathrm{var}(\boldsymbol{\kappa}_t)\mathrm{var}(\boldsymbol{\delta}_t^{(\boldsymbol{s}^l)})}}$$

Bounded Approximation Errors

Theorem 2. *For gradients $\boldsymbol{\delta}_t^{(\boldsymbol{s}^l)} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{s}_t^l}$ and $\boldsymbol{\kappa}_t^l = \sum_\tau \frac{\partial \boldsymbol{s}_t^l}{\partial \boldsymbol{I}_\tau^l}$, given the approximation error bound $\epsilon > 0$ s.t. $\left\|\mathbb{E}\left[\boldsymbol{\delta}_t^{(\boldsymbol{s}^l)} \boldsymbol{\kappa}_t^l\right] - \mathbb{E}\left[\boldsymbol{\delta}_t^{(\boldsymbol{s}^l)}\right] \mathbb{E}\left[\boldsymbol{\kappa}_t^l\right]\right\| \leq \epsilon(1 + \left\|\mathbb{E}\left[\boldsymbol{\delta}_t^{(\boldsymbol{s}^l)}\right]\right\|)$ for $\forall l$. Denote the stacked tensor $\boldsymbol{I}^l = [\boldsymbol{I}_1^l, ..., \boldsymbol{I}_T^l]$ and $\boldsymbol{s}^l = [\boldsymbol{s}_1^l, ..., \boldsymbol{s}_T^l]$. Assuming the backward procedure follows non-expansivity s.t. $\frac{\partial \boldsymbol{I}^{l+1}}{\partial \boldsymbol{I}^l} = \boldsymbol{W}^{l+1\top}\frac{\partial \boldsymbol{s}^l}{\partial \boldsymbol{I}^l}$ is 1-lipschitz continuous without loss of generality and the biases are bounded uniformly by B, i.e. $\left\|\boldsymbol{x}\frac{\partial \boldsymbol{I}^{l+1}}{\partial \boldsymbol{I}^l} - \hat{\boldsymbol{x}}\frac{\partial \boldsymbol{I}^{l+1}}{\partial \boldsymbol{I}^l}\right\| \leq \left\|\boldsymbol{x} - \hat{\boldsymbol{x}}\right\|$ for $\forall \boldsymbol{x}, \hat{\boldsymbol{x}}$. Define $\boldsymbol{\delta}_{rate}^l = \left(\frac{\partial \mathcal{L}}{\partial \boldsymbol{c}^l}\right)_{rate}$ as the error propagated through Eq. (7), and $\boldsymbol{\delta}_t^{(\boldsymbol{I}^l)} = \frac{\partial \mathcal{L}}{\partial \boldsymbol{I}_t^l}$ as the error propagated through BPTT, with $\boldsymbol{\delta}_{rate}^L = \mathbb{E}[\boldsymbol{\delta}_t^{(\boldsymbol{I}^L)}]$. We have the gradient difference bounded by $\left\|\boldsymbol{\delta}_{rate}^{L-k} - \mathbb{E}[\boldsymbol{\delta}_t^{(\boldsymbol{I}^{L-k})}]\right\| = \mathcal{O}(k^2\epsilon)$.*

$$\mathbb{E}\left[\boldsymbol{\delta}_t^{(\boldsymbol{I}^l)} \boldsymbol{s}_t^{l-1}\right] = \mathbb{E}\left[\boldsymbol{\delta}_t^{(\boldsymbol{I}^l)}\right] \mathbb{E}[\boldsymbol{s}_t^{l-1}] \text{ (A2)}$$

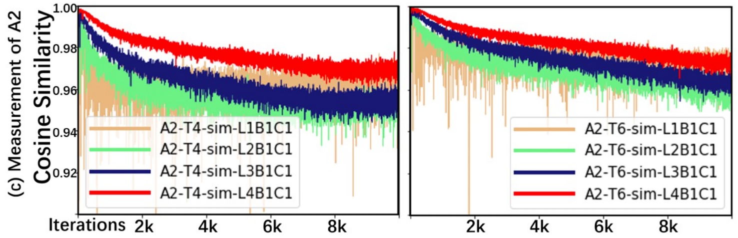
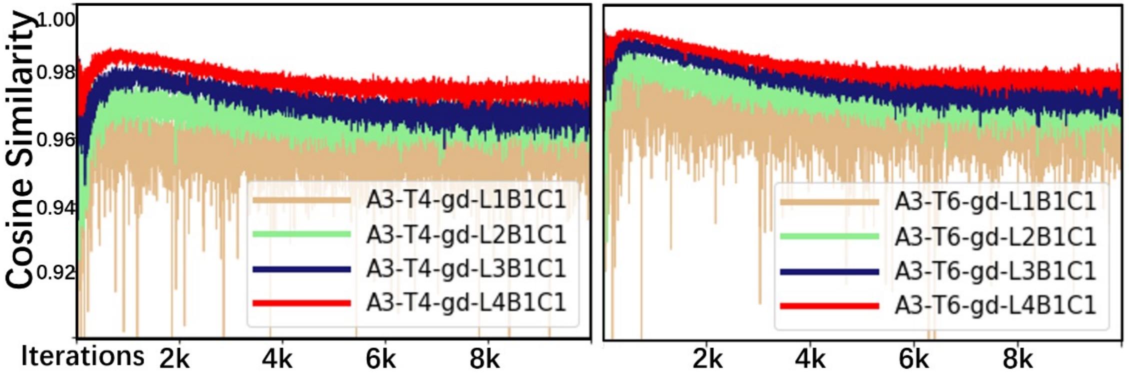


Figure 3: Empirical measurements conducted on the training procedure of BPTT. The experiments are carried out on the CIFAR-100 dataset using ResNet-18. Each subplot is labeled according to the naming convention "A{test#}-T{timesteps#}-{target}-L{layer#}B{block#}N{LIF#}/C{conv#}."

Results 1: Comparable to BPTT

Similarity of Weight Gradients with BPTT



Result 1: Rate-based backpropagation is comparable to BPTT on common visual benchmarks, with both methods exhibiting largely consistent gradient descent directions.

Performance Comparison with BPTT on CIFAR-10/100

Training	Model	Timesteps	Top-1 Acc (%)
BPTT_S	ResNet-18	2	95.02
		4	95.53
		6	95.68
	ResNet-19	2	96.12
		4	96.38
		6	96.57
	VGG-11	2	95.27
		4	95.61
		6	95.63
$\textbf{rate}_S$	ResNet-18	2	94.82±0.07(94.89)
		4	95.42±0.11(95.56)
		6	95.73±0.03(95.78)
	ResNet-19	2	96.11±0.05(96.18)
		4	96.32±0.04(96.38)
		6	96.38±0.06(96.45)
	VGG-11	2	95.44±0.02(95.46)
		4	95.57±0.08(95.68)
		6	95.64±0.12(95.76)
BPTT_M	ResNet-18	2	94.93
		4	95.64
		6	96.03
	ResNet-19	2	96.16
		4	96.49
		6	96.70
	VGG-11	2	95.31
		4	95.67
		6	95.64
$\textbf{rate}_M$	ResNet-18	2	94.75±0.05(94.82)
		4	95.61±0.02(95.64)
		6	95.90±0.07(96.01)
	ResNet-19	2	96.23±0.10(96.33)
		4	96.26±0.03(96.29)
		6	96.38±0.02(96.40)
	VGG-11	2	95.17±0.12(95.35)
		4	95.30±0.06(95.37)
		6	95.23±0.06(95.32)

Table 4: Performance comparison of rate-based backpropagation and BPTT on CIFAR-10.

Training	Model	Timesteps	Top-1 Acc (%)
BPTT_S	ResNet-18	2	76.24
		4	77.72
		6	78.65
	ResNet-19	2	79.33
		4	80.12
		6	80.77
	VGG-11	2	77.37
		4	77.82
		6	78.13
$\textbf{rate}_S$	ResNet-18	2	75.89±0.11(75.97)
		4	77.73±0.28(77.93)
		6	78.86±0.08(78.94)
	ResNet-19	2	79.71±0.02(79.74)
		4	80.41±0.14(80.54)
		6	80.75±0.05(80.79)
	VGG-11	2	77.34±0.04(77.37)
		4	77.87±0.35(78.13)
		6	78.23±0.03(78.27)
BPTT_M	ResNet-18	2	77.09
		4	77.93
		6	78.35
	ResNet-19	2	80.01
		4	81.07
		6	81.12
	VGG-11	2	77.42
		4	77.96
		6	78.25
$\textbf{rate}_M$	ResNet-18	2	75.97±0.20(76.27)
		4	78.26±0.12(78.38)
		6	79.02±0.11(79.16)
	ResNet-19	2	79.87±0.03(79.90)
		4	80.71±0.12(80.84)
		6	80.83±0.07(80.94)
	VGG-11	2	77.40±0.05(77.46)
		4	77.86±0.03(77.89)
		6	77.99±0.11(78.11)

Table 5: Performance comparison of rate-based backpropagation and BPTT on CIFAR-100.

Results 2: SOTA on benchmarks

Results on CIFAR-10 and CIFAR-100 Datasets

	Training	Method	Model	Timesteps	Top-1 Acc (%)
CIFAR10	QCFS [7]	ANN2SNN	ResNet-18	8	94.82
	DSR [47]	one-step	PreAct-ResNet-18	20	95.10±0.15
	SSF [68]	one-step	PreAct-ResNet-18	20	94.90
	BPTT_M	BPTT	ResNet-18	4	95.64
	$\textbf{rate}_M$ (ours)	one-step	ResNet-18	4	95.61±0.02(95.64)
	OTTT [76]	online	VGG-11*	6	93.52±0.06
	SLTT [48]	online	ResNet-18	6	94.44±0.21
	OS [89]	online	VGG-11 ResNet-19	4 4	94.35 95.20
	BPTT_S	BPTT	ResNet-18	4	95.53
			VGG-11	4	95.61
	$\textbf{rate}_S$ (ours)	one-step	ResNet-18	4	95.42±0.11(95.56)
			VGG-11	4	95.57±0.08(95.68)
CIFAR100	DSR [47]	one-step	PreAct-ResNet-18	20	78.50±0.12
	SSF [68]	one-step	PreAct-ResNet-18	20	75.48
	BPTT_M	BPTT	ResNet-18	4	77.93
	$\textbf{rate}_M$ (ours)	one-step	ResNet-18	4	78.26±0.12(78.38)
	OTTT [76]	online	VGG-11*	6	71.05±0.04
	SLTT [48]	online	ResNet-18	6	74.38±0.30
	OS [89]	online	VGG-11 ResNet-19	4 4	76.48 77.86
	BPTT_S	BPTT	ResNet-18	4	77.72
			VGG-11	4	77.82
	$\textbf{rate}_S$ (ours)	one-step	ResNet-18	4	77.73±0.28(77.93)
			VGG-11	4	77.87±0.35(78.13)

Table 1: Performance on ImageNet, and CIFAR10-DVS. Results are averaged over three runs of experiments, except for single crop evaluations on ImageNet. Models marked with (*) employ scaled weight standardization, adapting to normalizer-free architectures.

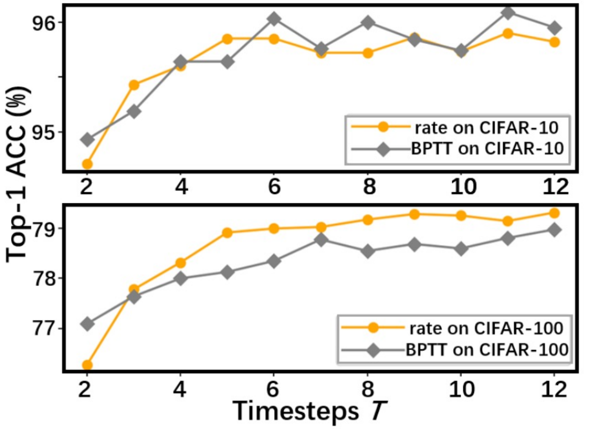
Results on ImageNet and CIFAR10-DVS Datasets

	Training	Method	Model	Timesteps	Top-1 Acc (%)
ImageNet	OTTT [76]	online	PreAct-ResNet-34*	6	65.15
	SLTT [48]	online	PreAct-ResNet-34*	6	66.19
	OS [89]	online	SEW-ResNet-34 PreAct-ResNet-34	4 4	64.14 67.54
	SEW-ResNet [20]	BPTT	SEW-ResNet-34	4	67.04
	$\textbf{rate}_S$ (ours)	one-step	SEW-ResNet-34 PreAct-ResNet-34	4 4	65.66 69.58
	$\textbf{rate}_M$ (ours)	one-step	SEW-ResNet-34 PreAct-ResNet-34	4 4	65.84 70.01
CIFAR10-DVS	DSR [47] SSF [68]	one-step	VGG-11 VGG-11	20 20	77.27±0.24 78.0
	OTTT [76] SLTT [48]	online	VGG-11* VGG-11	10 10	76.63±0.34 77.17±0.23
	BPTT_S BPTT_M	BPTT	VGG-11 VGG-11	10 10	76.73 76.86
	$\textbf{rate}_S$ (ours) **$\textbf{rate}_M$ (ours)**	one-step	VGG-11 VGG-11	10 10	76.48±0.23(76.71) 76.96±0.13(77.13)

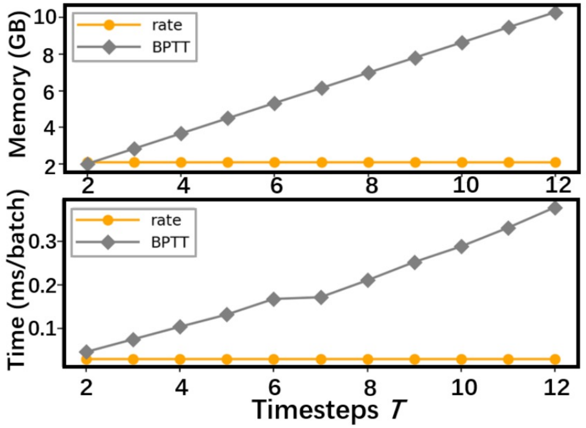
Result 2: The Rate-based backpropagation can surpasses results among all SNNs efficient training methodologies on CIFAR-10/100, ImageNet, and CIFAR10-DVS datasets.

Results 3: Memory and time efficiency

Accuracies and Training costs via Timesteps



(a) Comparison of Classification Performance



(b) Comparison of Training Costs

Result 3: The backward costs of the Rate-based Backpropagation are independent of the number of timesteps set, which reduces training overhead significantly both in terms of memory and time, .

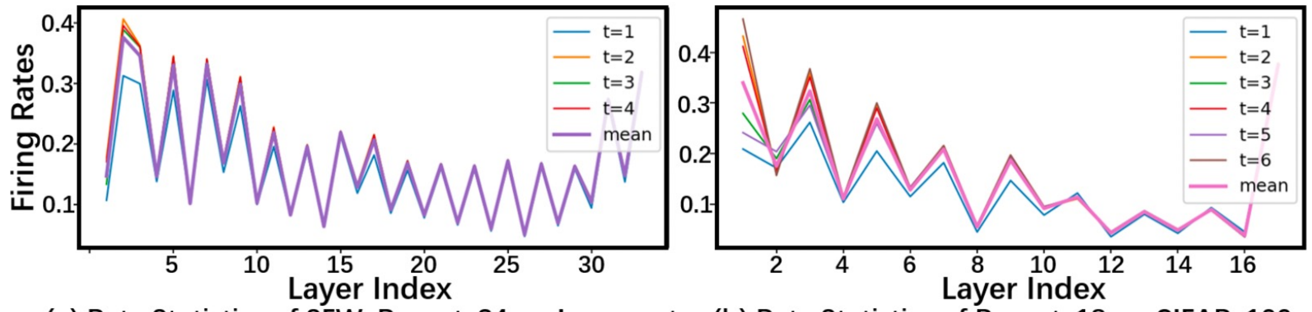
Comprehensive Evaluation of Training Costs

Datasets	Network	Method		Timesteps				
				T=1	T=2	T=4	T=8	T=16
CIFAR100	ResNet-18	rate_M	Time of Eligibility Track	0.003	0.004	0.007	0.015	0.027
			Time of Backward	0.034	0.035	0.036	0.034	0.036
			Time of both	0.037	0.039	0.043	0.049	0.063
			Memory Allocated	1.8492	1.8488	1.8473	1.8496	1.8483
			Top-1 Acc [%]	74.60	76.04	78.24	79.24	79.37
		BPTT_M	Time of Backward	0.023	0.044	0.098	0.199	0.564
			Memory Allocated	1.4272	2.4454	4.4804	8.0460	15.685
			Top-1 Acc [%]	74.38	76.65	78.49	78.35	
	ResNet-19	rate_M	Time of Eligibility Track	0.006	0.012	0.020	0.041	
			Time of Backward	0.083	0.083	0.082	0.083	
			Time of both	0.089	0.095	0.102	0.124	
			Memory Allocated [GB]	4.4787	4.4798	4.4788	4.4784	
			Top-1 Acc [%]	78.3	80.00	80.65	81.31	
		BPTT_M	Time of Backward	0.046	0.111	0.285	0.552	
			Memory Allocated [GB]	3.2556	5.6636	10.8978	20.3862	
			Top-1 Acc [%]	78.39	80.06	81.11	81.13	
	VGG11	rate_M	Time of Eligibility Track	0.003	0.003	0.006	0.011	0.020
			Time of Backward	0.017	0.017	0.017	0.017	0.018
			Time of both	0.020	0.020	0.023	0.028	0.038
			Memory Allocated [GB]	1.3624	1.3607	1.3619	1.3613	1.3601
			Top-1 Acc [%]	76.13	77.59	77.75	78.34	78.65
		BPTT_M	Time of Backward	0.010	0.021	0.054	0.135	0.384
			Memory Allocated [GB]	0.9911	1.6784	3.7363	6.6141	12.3768
			Top-1 Acc [%]	76.34	77.20	77.98	78.26	78.37
ImageNet	SEW-ResNet-34	rate_M	Time of Eligibility Track	0.012	0.014	0.023		
			Time of Backward	0.074	0.074	0.074		
			Time of both	0.086	0.088	0.097		
			Memory Allocated [GB]	5.7887	5.7898	5.7883		
		BPTT_M	Time of Backward	0.046	0.095	0.233		
			Memory Allocated [GB]	3.9858	6.8654	12.5597		
	PreAct-ResNet-34	rate_M	Time of Eligibility Track	0.007	0.009	0.020		
			Time of Backward	0.072	0.071	0.072		
			Time of both	0.079	0.080	0.092		
			Memory Allocated [GB]	5.4995	5.4982	5.4942		
		BPTT_M	Time of Backward	0.046	0.088	0.211		
			Memory Allocated [GB]	3.7017	6.4778	11.969		

Results 4: Rate-coding in statistics

Result 4: Results on spike statistics confirmed that rate-coding information is the predominant form of spike representation.

Firing Rates Statistics



(a) Rate Statistics of SEW-Resnet-34 on Imagenet (b) Rate Statistics of Resnet-18 on CIFAR-100

Experiments on Spikes Temporal Shuffle

Table 2: Performance w/o and w/ temporal shuffle for models trained by $\textbf{rate}_M$

Dataset	Model	Timesteps	Accuracy	Shuffled
CIFAR-10	ResNet-18	2	94.77	94.63±0.04
		4	95.51	95.50±0.04
		6	95.97	95.95±0.09
	VGG-11	2	95.13	95.10±0.05
		4	95.37	95.37±0.03
		6	95.77	95.79±0.05
CIFAR-100	ResNet-18	2	76.27	75.59±0.11
		4	78.32	77.72±0.15
		6	79.10	79.10±0.14
	VGG-11	2	77.46	77.21±0.12
		4	77.88	77.78±0.16
		6	77.97	78.02±0.09
ImageNet	SEW-ResNet-34	4	65.84	65.11±0.11
	PreAct-ResNet-34	4	70.01	69.78±0.10
CIFAR10-DVS	VGG-11	10	76.50	74.69±0.17

Take-home Message

Rate-based backpropagation,

A new SNNs training method that requires spatial backward only once:

- Demonstrates the pivotal role of rate-coding representation within current SNNs.
- Not alter the SNNs backbone, making backward costs independent of timestep T.
- Reduces memory and time costs while maintaining performance comparable to BPTT.
- Theoretical analysis and empirical validation confirm the optimization guarantees.
- Paves a way for more scalable and resource-efficient training of SNNs.

THANK YOU!

Contact Us

chengting.21@intl.zju.edu.cn

ailiwang@intl.zju.edu.cn

arXiv

GitHub