TU Delft

Deep activity propagation via weight initialization in spiking neural networks

Aurora Micheli, Olaf Booij, Jan van Gemert, Nergis Tomen

Introduction

- What is a good **weight initialization** for a spiking neural network (SNN)?

Introduction

- What is a good **weight initialization** for a spiking neural network (SNN)?

Unstable activations

Gradient issues

Training failure

Introduction

- What is a good **weight initialization** for a spiking neural network (SNN)?

- SNNs are often initialized following standard strategies designed for conventional networks (ANNs)

Xavier Glorot [1] → Sigmoid

Kaiming He [2] → ReLu

[1] X. Glorot and Y. Bengio. *Understanding the difficulty of training deep feedforward neural networks*
[2] K. He, X. Zhang, S. Ren, J. Sun. *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*

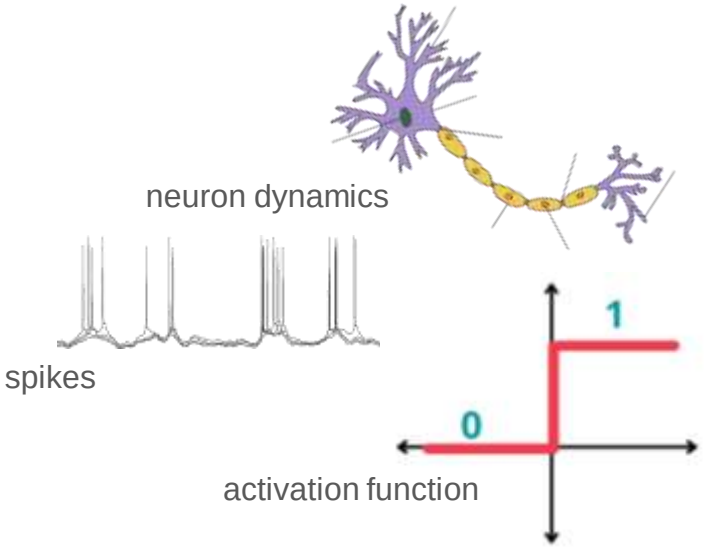
Introduction

- What is a good **weight initialization** for a spiking neural network (SNN)?

- SNNs are often initialized following standard strategies designed for conventional networks (ANNs)

Xavier Glorot [1] → Sigmoid

Kaiming He [2] → ReLu

- 

SNNs ≠ ANNs

[1] X. Glorot and Y. Bengio. *Understanding the difficulty of training deep feedforward neural networks*
[2] K. He, X. Zhang, S. Ren, J. Sun. *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*

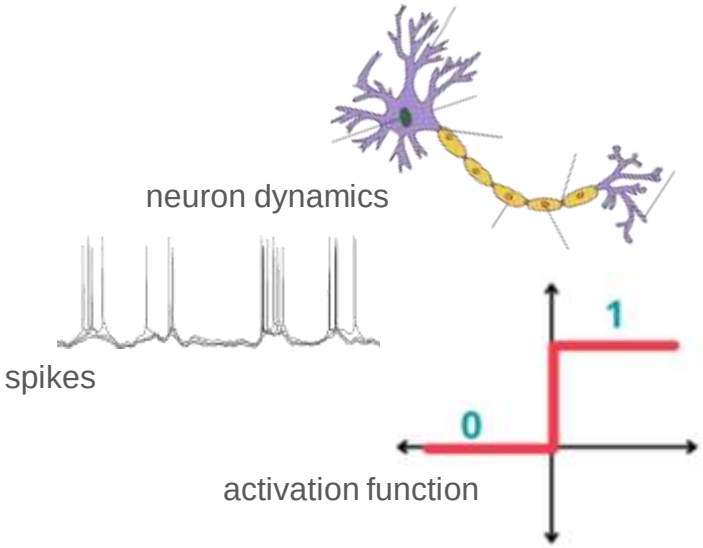
Introduction

- What is a good **weight initialization** for a spiking neural network (SNN)?

- SNNs are often initialized following standard strategies designed for conventional networks (ANNs)

  Xavier Glorot [1] → Sigmoid

  Kaiming He [2] → ReLu

- 

  SNNs ≠ ANNs

  ANN initialization schemes are **inadequate** for SNNs

[1] X. Glorot and Y. Bengio. *Understanding the difficulty of training deep feedforward neural networks*
[2] K. He, X. Zhang, S. Ren, J. Sun. *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*

In this paper

1. Analytically **derive** a **weight initialization** method tailored for SNNs

2. Empirically **validate** on simulations that our method ensures **stable spike propagation** in deep SNNs, preventing dissipation or amplification

3. Show that our weight initialization enhances **training speed** and improves **classification accuracy** on 4 different datasets

1. Proposed SNN initialization method

- Feed-forward fully-connected SNN initialized at t=0. For a generic layer l:

$$\boldsymbol{u}_l = \boldsymbol{w}_l \boldsymbol{x}_l$$
$$\boldsymbol{x}_l = f(\boldsymbol{u}_{l-1})$$

$$f(u_{l-1}) = \begin{cases} 1, & \text{if } u_{l-1} > \theta \\ 0, & \text{if } u_{l-1} < \theta \end{cases}$$

$\boldsymbol{u}_l \in \mathbb{R}^m$: membrane potentials
$\boldsymbol{w}_l \in \mathbb{R}^{m \times n}$: weight matrix
$\boldsymbol{x}_l \in \{0,1\}^n$: input spikes
f : activation function
θ : firing threshold

1. Proposed SNN initialization method

- Feed-forward fully-connected SNN initialized at t=0. For a generic layer l:

$$\boldsymbol{u}_l = \boldsymbol{w}_l \boldsymbol{x}_l$$
$$\boldsymbol{x}_l = f(\boldsymbol{u}_{l-1})$$

$$f(u_{l-1}) = \begin{cases} 1, & \text{if } u_{l-1} > \theta \\ 0, & \text{if } u_{l-1} < \theta \end{cases}$$

$\boldsymbol{u}_l \in \mathbb{R}^m$: membrane potentials
$\boldsymbol{w}_l \in \mathbb{R}^{m \times n}$: weight matrix
$\boldsymbol{x}_l \in \{0,1\}^n$: input spikes
f : activation function
θ : firing threshold

[2] Good initialization method: avoid reducing or amplifying the magnitudes of the input signals …

[2] K. He, X. Zhang, S. Ren, J. Sun. *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*

1. Proposed SNN initialization method

- Feed-forward fully-connected SNN initialized at t=0. For a generic layer l:

$$\boldsymbol{u}_l = \boldsymbol{w}_l \boldsymbol{x}_l$$

$$\boldsymbol{x}_l = f(\boldsymbol{u}_{l-1})$$

$$f(u_{l-1}) = \begin{cases} 1, & \text{if } u_{l-1} > \theta \\ 0, & \text{if } u_{l-1} < \theta \end{cases}$$

$\boldsymbol{u}_l \in \mathbb{R}^m$: membrane potentials
$\boldsymbol{w}_l \in \mathbb{R}^{m \times n}$: weight matrix
$\boldsymbol{x}_l \in \{0,1\}^n$: input spikes
f : activation function
θ : firing threshold

[2] Good initialization method: avoid reducing or amplifying the magnitudes of the input signals …

$$\mathrm{Var}[u_l] = n_l \mathrm{Var}[w_l x_l] \quad \textbf{= constant}$$

n_l : input size

… by keeping the variance of the input to each layer constant

[2] K. He, X. Zhang, S. Ren, J. Sun. *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*

1. Proposed SNN initialization method

- Feed-forward fully-connected SNN initialized at t=0. For a generic layer l:

$$\boldsymbol{u}_l = \boldsymbol{w}_l \boldsymbol{x}_l$$

$$\boldsymbol{x}_l = f(\boldsymbol{u}_{l-1})$$

$$f(u_{l-1}) = \begin{cases} 1, & \text{if } u_{l-1} > \theta \\ 0, & \text{if } u_{l-1} < \theta \end{cases}$$

$\boldsymbol{u}_l \in \mathbb{R}^m$: membrane potentials
$\boldsymbol{w}_l \in \mathbb{R}^{m \times n}$: weight matrix
$\boldsymbol{x}_l \in \{0,1\}^n$: input spikes
f : activation function
θ : firing threshold

[2] Good initialization method: avoid reducing or amplifying the magnitudes of the input signals …

$$\text{Var}[u_l] = n_l \text{Var}[w_l] E[x_l^2] \quad \textbf{= constant}$$

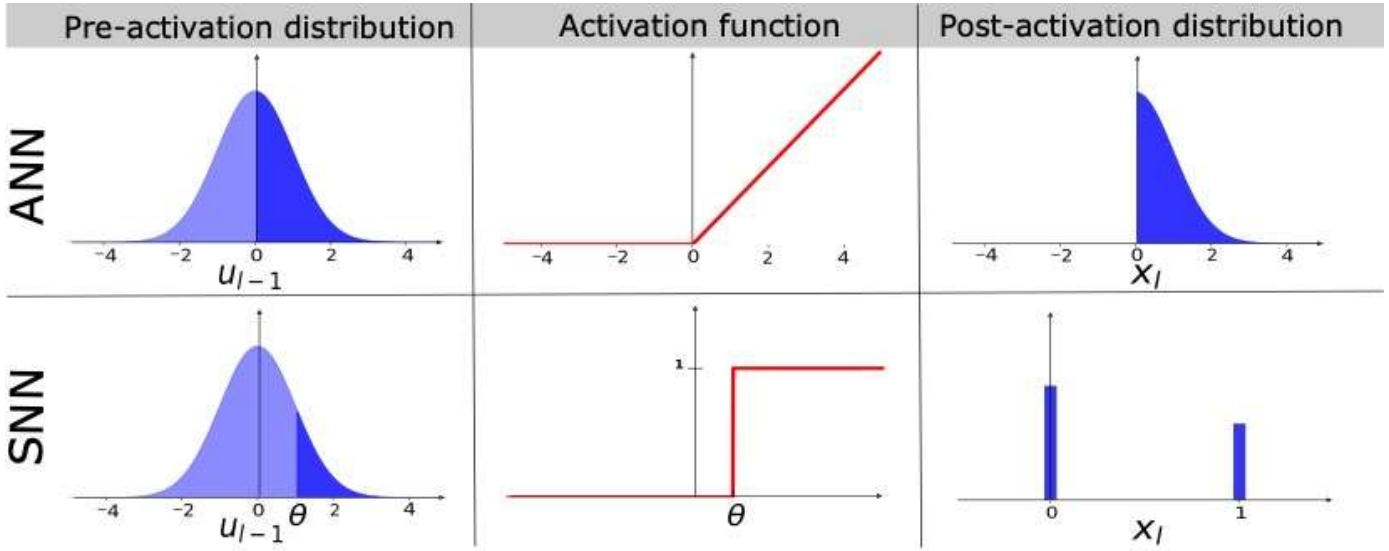
n_l : input size

… by keeping the variance of the input to each layer constant

[2] K. He, X. Zhang, S. Ren, J. Sun. *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*

1. Proposed SNN initialization method

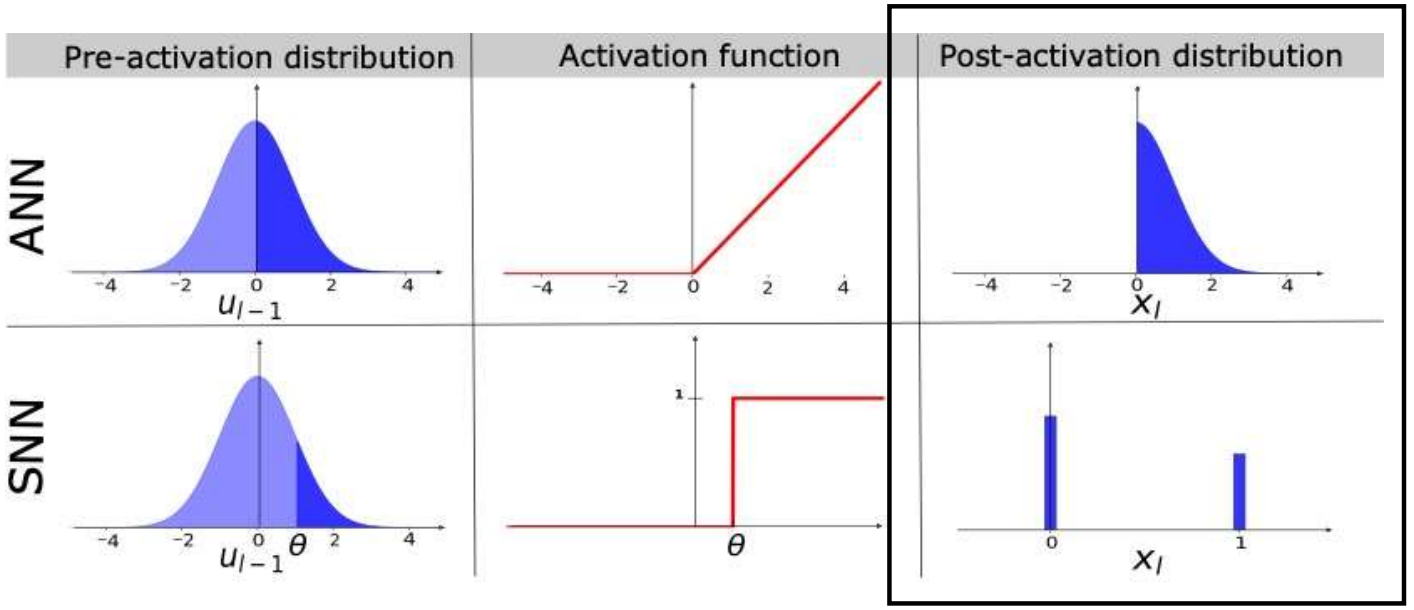
$$\mathrm{Var}[u_l] = n_l \mathrm{Var}[w_l] E[x_l^2]$$ [2]



[2] K. He, X. Zhang, S. Ren, J. Sun. *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*

1. Proposed SNN initialization method

$$\mathrm{Var}[u_l] = n_l \mathrm{Var}[w_l] E[x_l^2]$$ [2]

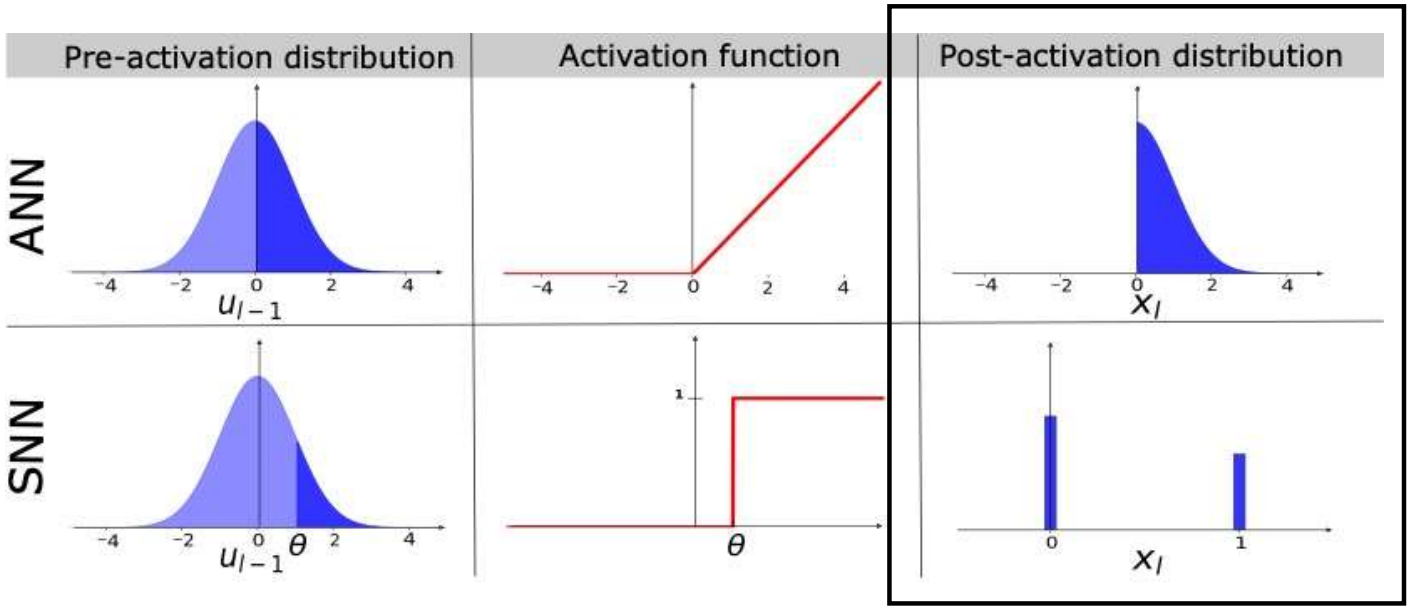


#

[2] K. He, X. Zhang, S. Ren, J. Sun. *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*

1. Proposed SNN initialization method

$$\mathrm{Var}[u_l] = n_l \mathrm{Var}[w_l] E[x_l^2]$$ [2]



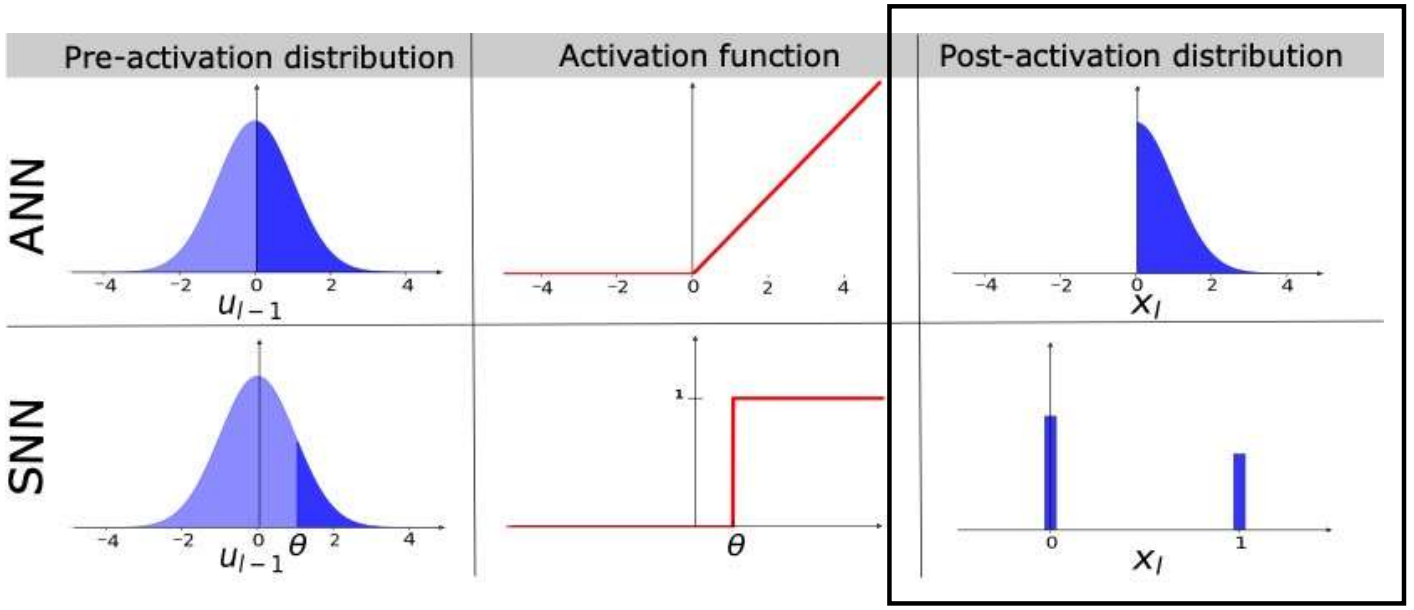
#

$$\mathrm{Var}[w_l] = \frac{1}{n_l P(u_{l-1} > \theta)}$$

[2] K. He, X. Zhang, S. Ren, J. Sun. *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*

1. Proposed SNN initialization method

$$\mathrm{Var}[u_l] = n_l \mathrm{Var}[w_l] E[x_l^2]$$ [2]



#

$$\mathrm{Var}[w_l] = \frac{1}{n_l P(u_{l-1} > \theta)}$$

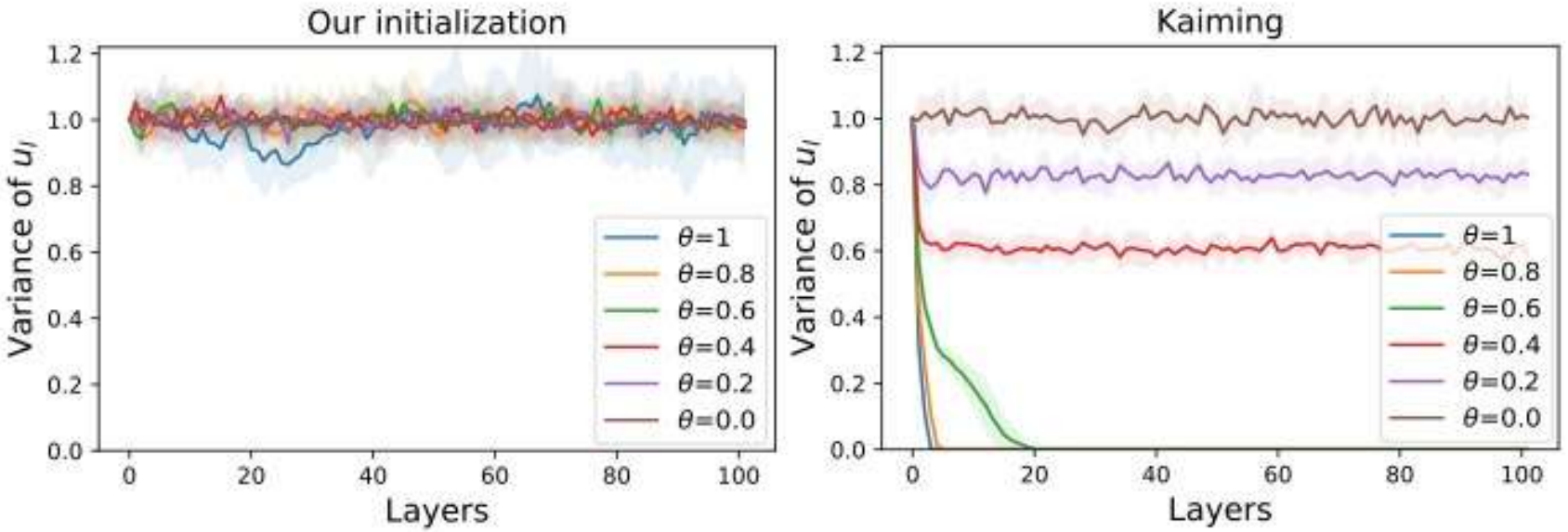
$$\frac{1}{n_l P(u_{l-1} > \theta)} > \frac{2}{n_l}$$

SNN **ANN**

[2] K. He, X. Zhang, S. Ren, J. Sun. *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification*

2. Empirical validation: t=0

- 100 layers fully-connected SNN with Leaky Integrate-and-Fire (LIF) neurons
- Input I_0 drawn from $\mathcal{N}(\mu = 0, \sigma^2 = 1)$, $\theta \in [0,1]$

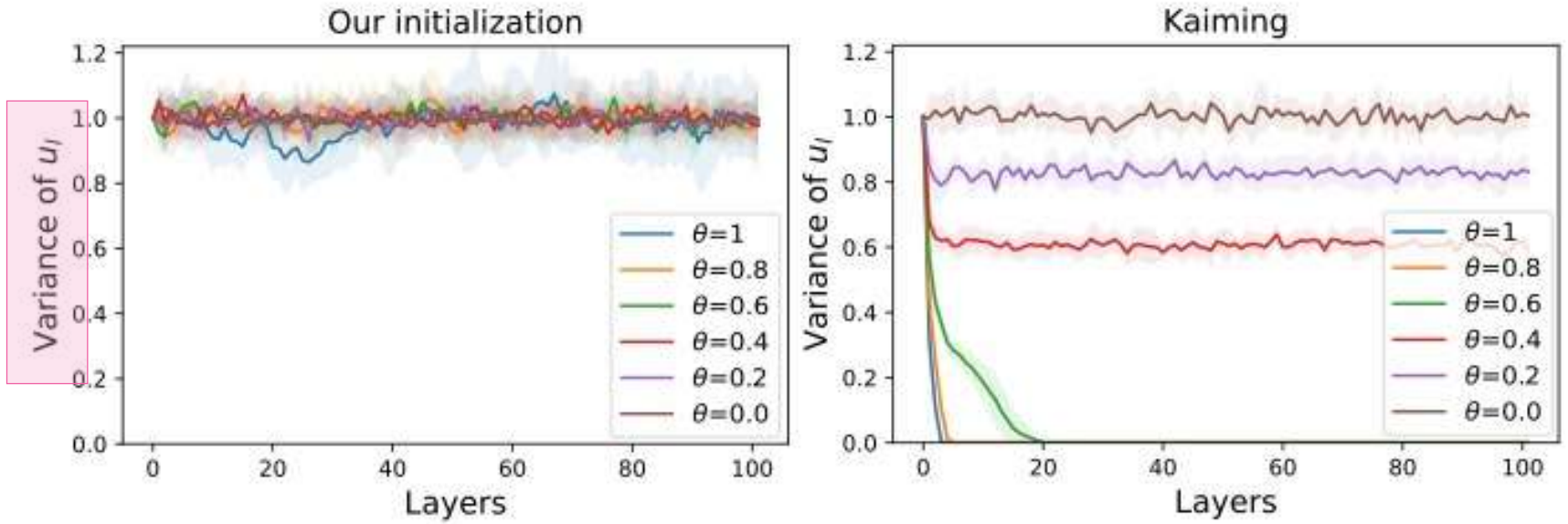


With the proposed method, unlike Kaiming, the variance of the membrane potentials stays constant across layers

2. Empirical validation: t=0

- 100 layers fully-connected SNN with Leaky Integrate-and-Fire (LIF) neurons
- Input I_0 drawn from $\mathcal{N}(\mu = 0, \sigma^2 = 1)$, $\theta \in [0,1]$

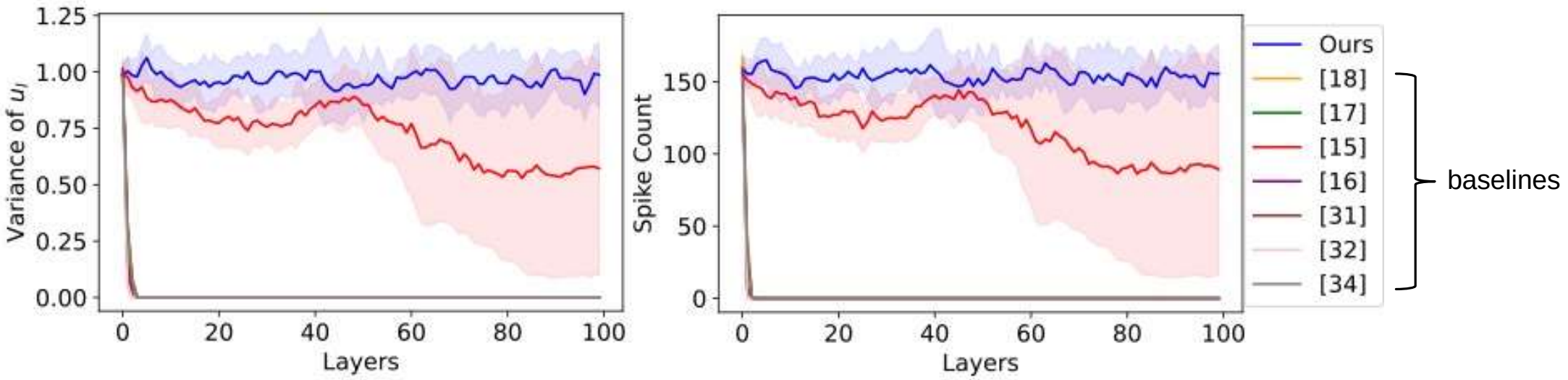
constant



With the proposed method, unlike Kaiming, the variance of the membrane potentials stays constant across layers

2. Empirical validation: t=0

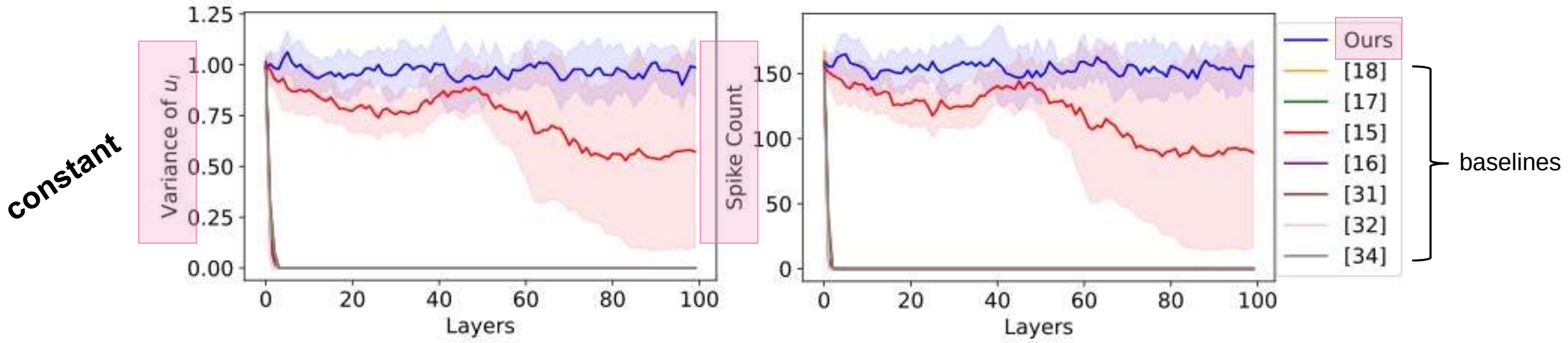
- 100 layers fully-connected SNN with Leaky Integrate-and-Fire (LIF) neurons
- Input I_0 drawn from $\mathcal{N}(\mu = 0, \sigma^2 = 1)$



Unlike other commonly used schemes, our initialization enables spiking activity to propagate from input to output

2. Empirical validation: t=0

- 100 layers fully-connected SNN with Leaky Integrate-and-Fire (LIF) neurons
- Input I_0 drawn from $\mathcal{N}(\mu = 0, \sigma^2 = 1)$



Unlike other commonly used schemes, our initialization enables spiking activity to propagate from input to output

2. Empirical validation: t=0

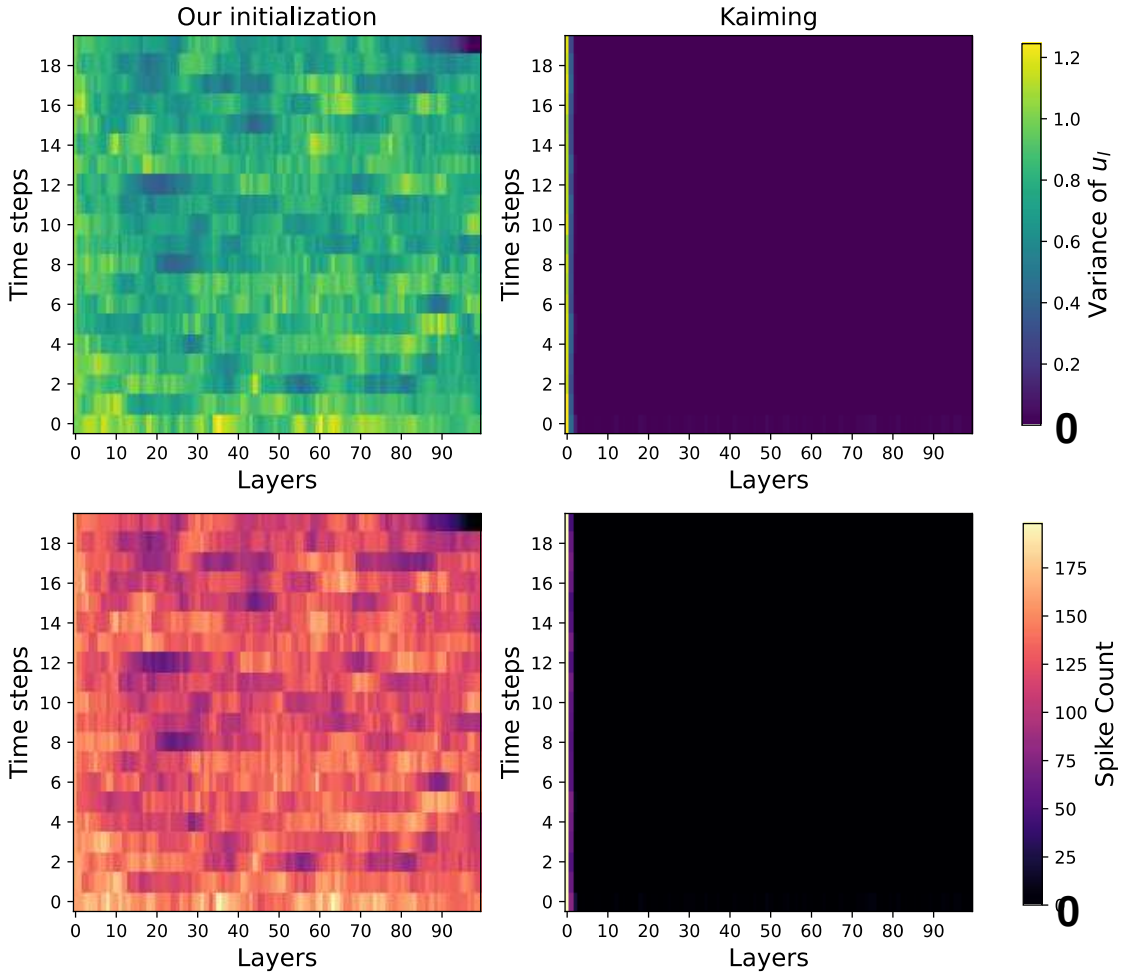
2. Empirical validation: ~~t=0~~ t>0

2. Empirical validation: t>0

- 100 layers fully-connected SNN of Leaky Integrate-and-fire (LIF) neurons with soft reset
- Input I_0 drawn from $\mathcal{N}(\mu = 0, \sigma^2 = 1)$
- Time steps T=20

While our method doesn't explicitly account for time,

it approximately conserves variance and spike count

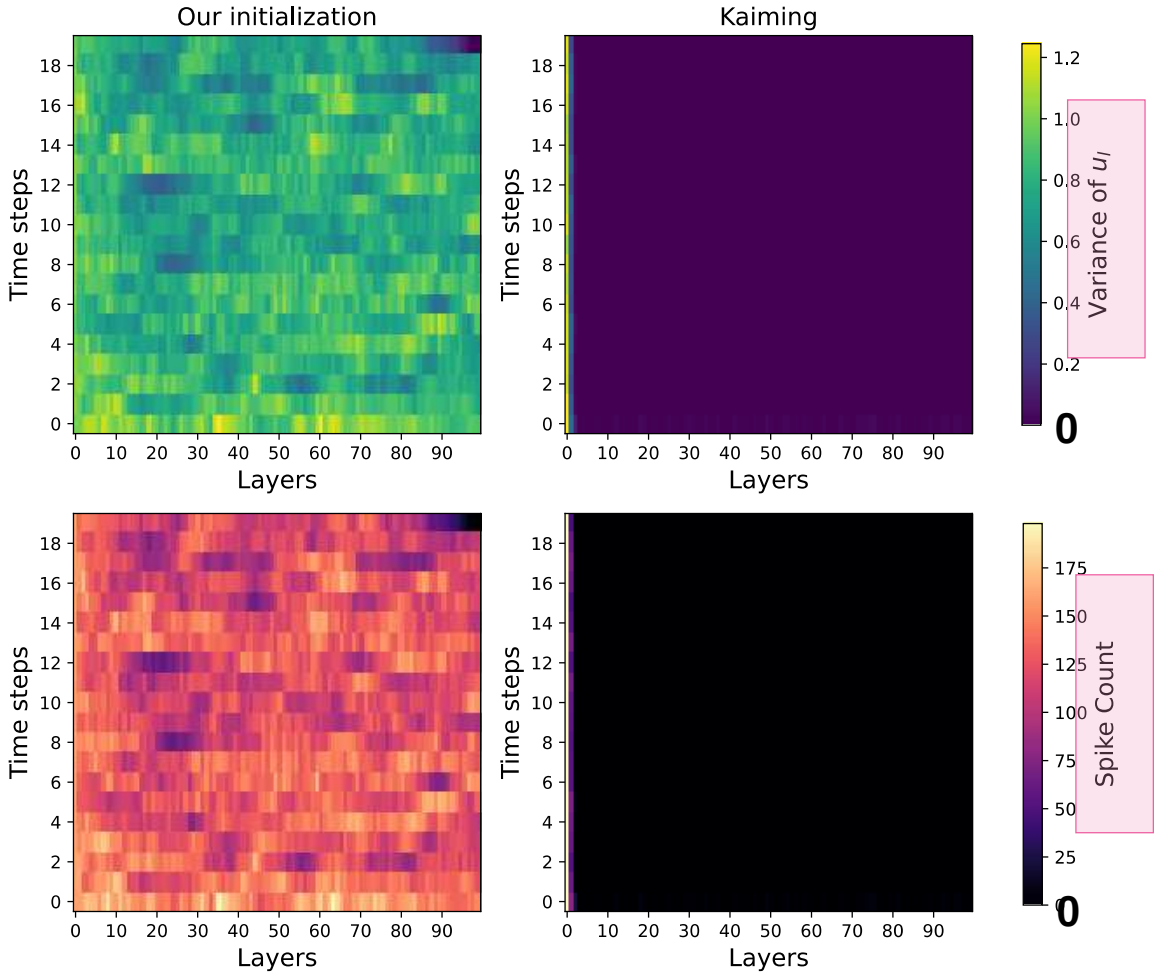


2. Empirical validation: t>0

- 100 layers fully-connected SNN of Leaky Integrate-and-fire (LIF) neurons with soft reset
- Input I_0 drawn from $\mathcal{N}(\mu = 0, \sigma^2 = 1)$
- Time steps T=20

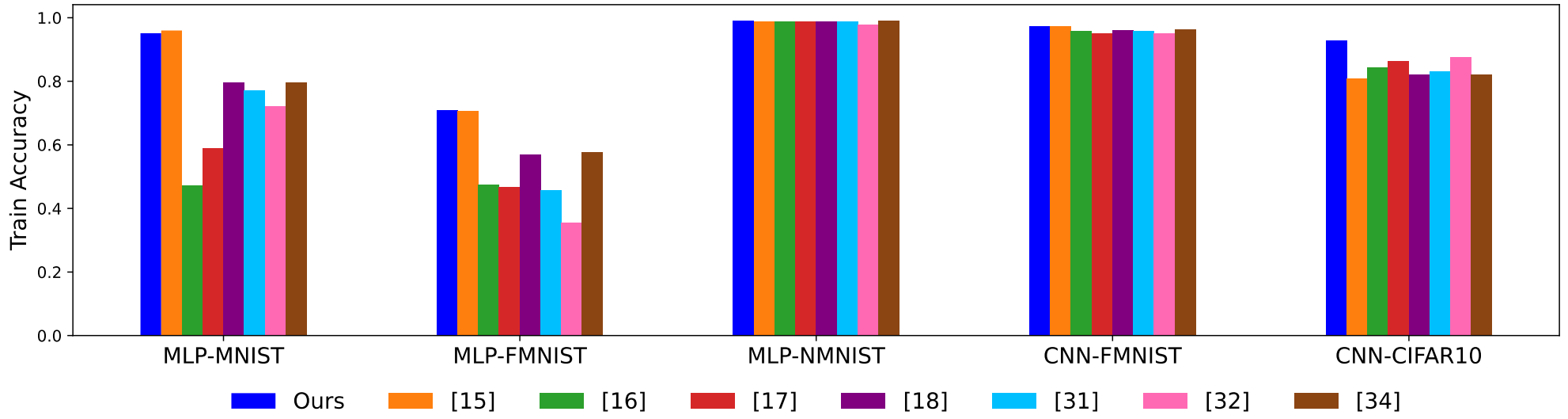
While our method doesn't explicitly account for time,

it approximately conserves variance and spike count



3. Evaluation on classification datasets

- Image classification on 4 datasets: MNIST, Fashion-MNIST, Neuromorphic-MNIST, CIFAR10
- Fully-connected (MLP) and Convolutional (CNN) SNNs trained with backpropagation



Our weight initialization can improve the classification accuracy with only T = 3 time steps

Conclusions

- Derive a new variance-conserving weight initialization method tailored to the activation function of SNNs
- Numerically validate that our method ensures stable spike propagation in deep SNNs, unlike baseline methods
- Show effectiveness of proper weight initialization on training across four image classification datasets

Conclusions

- Derive a new variance-conserving weight initialization method tailored to the activation function of SNNs
- Numerically validate that our method ensures stable spike propagation in deep SNNs, unlike baseline methods
- Show effectiveness of proper weight initialization on training across four image classification datasets

Limitations & future work

- Our method does not explicitly account for temporal dynamics → extend theory
- Validation on more complex architectures and datasets

Conclusions

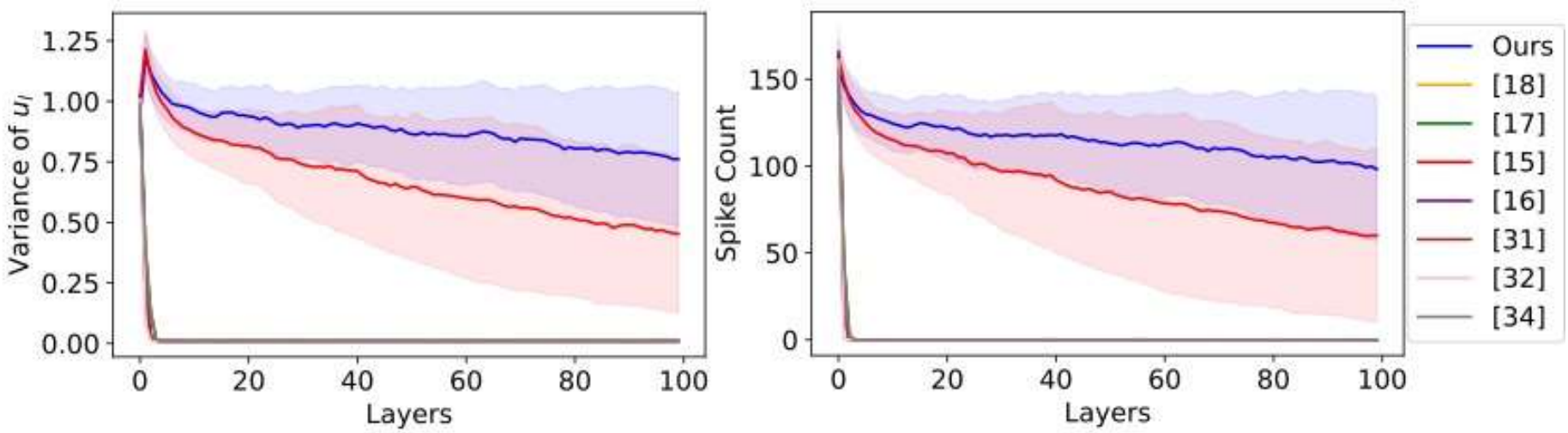
- Derive a new variance-conserving weight initialization method tailored to the activation function of SNNs
- Numerically validate that our method ensures stable spike propagation in deep SNNs, unlike baseline methods
- Show effectiveness of proper weight initialization on training across four image classification datasets

Limitations & future work

- Our method does not explicitly account for temporal dynamics → extend theory
- Validation on more complex architectures and datasets

2. Empirical validation: t>0

- 100 layers fully-connected SNN of Leaky Integrate-and-fire (LIF) neurons with soft reset
- Input I_0 drawn from $\mathcal{N}(\mu = 0, \sigma^2 = 1)$



While our method doesn't explicitly account for time, it approximately conserves variance and spike count