Supplementary

1 SNN-crafted adversarial example

In this section, we develop a technique to produce adversarial examples from a Spiking Neural Network (SNN) utilizing the Fast Gradient Sign Method (FGSM) or its iterative variants. Suppose the source model is referred as M, the loss function of the model for input x and true labels y_{true}, is given by $J(x, y_{true})$. According to FGSM, the adversarial example is computed as,

$$x_{adv} = x + \epsilon \times sign\big(\nabla_x J\left(x, y_{true}\right)\big) \tag{1}$$

ϵ denotes the strength of the perturbation. FGSM adversarial examples require computing the gradient of loss with respect to input ($\nabla_x J(x, y_{true})$) obtained through backpropagation. Hence, it is essential to have a differentiable network from output all the way to input. However, SNNs have two sources of discontinuity in the gradient: (1) discontinuous gradients of the Leaky-Integrate-Fire (or Integrate-Fire) neurons in the hidden layers, (2) discretization at the input layer (Poisson encoding). We circumvent these two obstactles by the following approximations:

1. The transfer function of LIF (or IF) neuron is a step function, the derivative being discontinuous at the threshold point. We approximate the gradients of the neurons by some pseudo-derivatives like linear or exponential functions. This approach, known as surrogate-gradient technique, is applied to all the hidden layers, thereby making the network differentiable from output to the $1st$ hidden layer.

2. At the input layer, continuous-valued analog input is discretized into binary (0 or 1) spike train. The analog input can be approximated by the average of this spike train over the entire time-window, referred as the rate input, $\mathbf{X_{rate}}$. We assume that $\mathbf{X_{rate}}$, which is a continuous-valued signal, is the input to the $1st$ convolutional layer.

These two approximations make the SNN completely differentiable from output to input. Let us consider an SNN with analog input $\mathbf{X}$ and input spike-train $\{\mathbf{X_1}, \mathbf{X_2}, \cdots \mathbf{X_T}\}$, where T is the total number of timesteps. Each of $\mathbf{X_1}, \mathbf{X_2}, \cdots, \mathbf{X_T}$ has the same dimension as the analog input $\mathbf{X}$. $\mathbf{X_{rate}}$ describes the timed-average of the spike-train. The output of the $1st$ convolutional layer (weight matrix $\mathbf{W_{conv1}}$) passes through an LIF (or IF) neuron with membrane potential represented as $\mathbf{X_{conv1}}$ after T timesteps. We assume that

$$\mathbf{X} \approx \mathbf{X_{rate}} = \frac{\sum_{i=1}^{i=T} \mathbf{X_i}}{T} \tag{2}$$

and

$$\mathbf{X_{conv1}} \approx \mathbf{X_{rate}} \circledast \mathbf{W_{conv1}} \tag{3}$$

(The validity of the assumption in Eq. 3 is explained in Sec. 1.2.) Let us assume the loss function of the network is represented by J. Then the gradient of loss with respect to input is described using the chain rule of derivative,

$$\frac{\partial J}{\partial \mathbf{X}} \approx \frac{\partial J}{\partial \mathbf{X_{rate}}} = \frac{\partial J}{\partial \mathbf{X_{conv1}}} \times \frac{\partial \mathbf{X_{conv1}}}{\partial \mathbf{X_{rate}}} \tag{4}$$

$$= \frac{\partial J}{\partial \mathbf{X_{conv1}}} \circledast \mathbf{W}^{\mathbf{180rotated}}_{\mathbf{conv1}} \tag{5}$$

where $\mathbf{W}^{\mathbf{180rotated}}_{\mathbf{conv1}}$ is the $\mathbf{W_{conv1}}$ matrix rotated by 180°. The proof of Eq. 5 is provided in the next subsection. Hence, FGSM adversarial perturbation is computed as,

$$\epsilon \times sign(\frac{\partial J}{\partial \mathbf{X}}) \approx \epsilon \times sign(\frac{\partial J}{\partial \mathbf{X_{rate}}}) = \epsilon \times sign(\frac{\partial J}{\partial \mathbf{X_{conv1}}} \circledast \mathbf{W}^{\mathbf{180rotated}}_{\mathbf{conv1}}) \tag{6}$$

$\frac{\partial J}{\partial \mathbf{X_{conv1}}}$ is obatined by backpropagating the loss through the network with the help of the surrogate gradient technique. Then Eq. 6 is used to obtain the adversarial perturbation.

1.1 Derivation of $\frac{\partial J}{\partial X}$ when $Y = X \circledast W$

Suppose the input image is represented as $\mathbf{X}$, $\mathbf{W}$ denotes the weight matrix of the convolutional layer and $\mathbf{Y}$ is the output of the convolution operation.

$$\mathbf{Y} = \mathbf{X} \circledast \mathbf{W} \tag{7}$$

Assume $\mathbf{X}$ is a 32×32 matrix and the kernel size $\mathbf{W}$ for convolution is 3×3. Convolution is performed with a padding of 1 and stride of 1. Then output $\mathbf{Y}$ will maintain a size of 32×32 (input dimension + $(2 \times$ padding$)$ − kernel size + 1).

$$\mathbf{X} = \begin{pmatrix} X_{1,1} & X_{1,2} & \cdots & X_{1,32} \\ X_{2,1} & X_{2,2} & \cdots & X_{2,32} \\ \vdots & \vdots & \ddots & \vdots \\ X_{32,1} & X_{32,2} & \cdots & X_{32,32} \end{pmatrix}, \mathbf{W} = \begin{pmatrix} W_{1,1} & W_{1,2} & W_{1,3} \\ W_{2,1} & W_{2,2} & W_{2,3} \\ W_{3,1} & W_{3,2} & W_{3,3} \end{pmatrix}$$

$$\mathbf{Y} = \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & X_{1,1} & X_{1,2} & \cdots & X_{1,32} & 0 \\ 0 & X_{2,1} & X_{2,2} & \cdots & X_{2,32} & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & X_{32,1} & X_{32,2} & \cdots & X_{32,32} & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 \end{pmatrix} \circledast \begin{pmatrix} W_{1,1} & W_{1,2} & W_{1,3} \\ W_{2,1} & W_{2,2} & W_{2,3} \\ W_{3,1} & W_{3,2} & W_{3,3} \end{pmatrix} \tag{8}$$

$$\begin{aligned} Y_{1,1} &= W_{2,2}X_{1,1} + W_{2,3}X_{1,2} + W_{3,2}X_{2,1} + W_{3,3}X_{2,2} \\ Y_{1,2} &= W_{2,1}X_{1,1} + W_{2,2}X_{1,2} + W_{2,3}X_{1,3} + W_{3,1}X_{2,1} + W_{3,2}X_{2,2} + W_{3,3}X_{2,3} \\ &\vdots \\ Y_{32,32} &= W_{1,1}X_{31,31} + W_{1,2}X_{31,32} + W_{2,1}X_{32,31} + W_{2,2}X_{32,32} \end{aligned}$$

Let us assume the loss function is represented by J. The gradient of J with respect to $\mathbf{X}$ is also a 32×32 matrix, each element described by the chain rule of derivative:

$$\frac{\partial J}{\partial X_{1,1}} = \frac{\partial J}{\partial Y_{1,1}}\frac{\partial Y_{1,1}}{\partial X_{1,1}} + \frac{\partial J}{\partial Y_{1,2}}\frac{\partial Y_{1,2}}{\partial X_{1,1}} + \ldots + \frac{\partial J}{\partial Y_{32,32}}\frac{\partial Y_{32,32}}{\partial X_{1,1}} \tag{9}$$

$$= \frac{\partial J}{\partial Y_{1,1}}W_{2,2} + \frac{\partial J}{\partial Y_{1,2}}W_{2,1} + \frac{\partial J}{\partial Y_{2,1}}W_{1,2} + \frac{\partial J}{\partial Y_{2,2}}W_{1,1} \tag{10}$$

$$\frac{\partial J}{\partial X_{1,2}} = \frac{\partial J}{\partial Y_{1,1}}W_{2,3} + \frac{\partial J}{\partial Y_{1,2}}W_{2,2} + \frac{\partial J}{\partial Y_{1,3}}W_{2,1} + \frac{\partial J}{\partial Y_{2,1}}W_{1,3} + \frac{\partial J}{\partial Y_{2,2}}W_{1,2} + \frac{\partial J}{\partial Y_{2,3}}W_{1,1} \tag{11}$$

$$\vdots \tag{12}$$

$$\frac{\partial J}{\partial X_{32,32}} = \frac{\partial J}{\partial Y_{31,31}}W_{3,3} + \frac{\partial J}{\partial Y_{31,32}}W_{3,2} + \frac{\partial J}{\partial Y_{32,31}}W_{2,3} + \frac{\partial J}{\partial Y_{32,32}}W_{2,2} \tag{13}$$

$$\frac{\partial J}{\partial \mathbf{X}} = \begin{pmatrix} 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & \frac{\partial J}{\partial Y_{1,1}} & \frac{\partial J}{\partial Y_{1,2}} & \cdots & \frac{\partial J}{\partial Y_{1,32}} & 0 \\ 0 & \frac{\partial J}{\partial Y_{2,1}} & \frac{\partial J}{\partial Y_{2,2}} & \cdots & \frac{\partial J}{\partial Y_{2,32}} & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & \frac{\partial J}{\partial Y_{32,1}} & \frac{\partial J}{\partial Y_{32,2}} & \cdots & \frac{\partial J}{\partial Y_{32,32}} & 0 \\ 0 & 0 & 0 & \cdots & 0 & 0 \end{pmatrix} \circledast \begin{pmatrix} W_{3,3} & W_{3,2} & W_{3,1} \\ W_{2,3} & W_{2,2} & W_{2,1} \\ W_{1,3} & W_{1,2} & W_{1,1} \end{pmatrix} \tag{14}$$

$$\therefore \frac{\partial J}{\partial \mathbf{X}} = \frac{\partial J}{\partial \mathbf{Y}} \circledast \mathbf{W}^{180rotated}, \quad \text{when} \quad \mathbf{Y} = \mathbf{X} \circledast \mathbf{W} \tag{15}$$

1.2 Validity of the assumption $X_{conv1} \approx X_{rate} \circledast W_{conv1}$

In an SNN, the membrane potential $\mathbf{X_{conv1}}$ at the final timestep T, after passing through an IF neuron (leak factor $\lambda = 1$), is computed as,

$$\mathbf{X_{conv1}} = \sum_{i=1}^{i=T} \left(\mathbf{X_i} \circledast \mathbf{W_{conv1}} - \mathbf{V_{reset,i}}\right) \tag{16}$$

$\mathbf{V_{reset,i}}$ is the reset voltage at *i-th* timestep (in case of soft reset). $\mathbf{X_i}$ is the *i-th* spike input. Suppose $V_{threshold}$ is the threshold voltage of the neuron and $\mathbf{X_{conv1,i}}$ is the membrane potential at *i-th* timestep.

$$V_{reset,i} = \begin{cases} 0, & \text{if } X_{conv1,i} < V_{threshold} \\ V_{threshold}, & \text{otherwise} \end{cases} \tag{17}$$

Let us assume that the membrane potential reaches threshold n times over the time-window T.

$$\mathbf{X_{conv1}} = \sum_{i=1}^{i=T} \left(\mathbf{X_i} \circledast \mathbf{W_{conv1}}\right) - nV_{threshold} \tag{18}$$

$$= (\sum_{i=1}^{i=T} \mathbf{X_i}) \circledast \mathbf{W_{conv1}} - nV_{threshold}, \quad \text{distributive property of convolution} \tag{19}$$

$$= T\mathbf{X_{rate}} \circledast \mathbf{W_{conv1}} - nV_{threshold}, \quad \text{T is the total number of timesteps} \tag{20}$$

Using the chain rule of derivative,

$$\frac{\partial J}{\partial \mathbf{X_{rate}}} = \frac{\partial J}{\partial \mathbf{X_{conv1}}} \times \frac{\partial \mathbf{X_{conv1}}}{\partial \mathbf{X_{rate}}} \tag{21}$$

Assuming the gradient of the second term in Eq. 20 (*i.e.* $\frac{\partial(nV_{threshold})}{\partial \mathbf{X_{rate}}}$) to be negligibly small, we obtain the gradient of J w.r.t. $\mathbf{X_{rate}}$ from Eq. 15 as,

$$\frac{\partial J}{\partial \mathbf{X_{rate}}} \approx T(\frac{\partial J}{\partial \mathbf{X_{conv1}}} \circledast \mathbf{W}^{\mathbf{180rotated}}_{\mathbf{conv1}}) \tag{22}$$

Since T is a positive constant value,

$$sign(\frac{\partial J}{\partial \mathbf{X_{rate}}}) = sign(\frac{\partial J}{\partial \mathbf{X_{conv1}}} \circledast \mathbf{W}^{\mathbf{180rotated}}_{\mathbf{conv1}}) \tag{23}$$

which complies with our solution at Eq. 6, obtained from the approximation $X_{conv1} \approx X_{rate} \circledast W_{conv1}$.