IM-Loss: Information Maximization Loss for Spiking Neural Networks

Yufei Guo[*]**, Yuanpei Chen**[*]**, Liwen Zhang, Xiaode Liu, Yinglei Wang, Xuhui Huang, Zhe Ma**[†]
Intelligent Science & Technology Academy of CASIC
yfguo@pku.edu.cn, rop477@163.com, mazhe_thu@163.com

A Appendix

A.1 Proofs of Zero Conditional Entropy

Proof.

$$
\begin{aligned}
I(U;O) &= H(O) - H(O|U) && (1)\\
&= \sum_{u,o} p(u,o) log \frac{p(u,o)}{p(u)p(o)} && (2)\\
&= \sum_{u,o} p(u,o) log \frac{p(u,o)}{p(u)} - \sum_{u,o} p(u,o) logp(o) && (3)\\
&= \sum_{u,o} p(u) p_{O|U=u}(o) logp_{O|U=u}(o) - \sum_{u,o} p(u,o) logp(o) && (4)\\
&= \sum_{u} p(u) (\sum_{o} p_{O|U=u}(o) logp_{O|U=u}(o)) - \sum_{o} (\sum_{u} p(u,o)) logp(o)) && (5)\\
&= -\sum_{u} p(u) H(O|U=u) - \sum_{o} p(o) logp(o) && (6)\\
&= -H(O|U) + H(O) && (7)
\end{aligned}
$$

In above equations, $p(u)$, $p(o)$ and $p(u,o)$ are the marginal probability mass functions of the discrete variables U, O and their joint probability mass function. The conditional entropy $H(O|U)$ can be expressed as the below equation according to the Eq.5 and Eq.7.

$$
H(O|U) = \sum_{u} p(u) (\sum_{o} p_{O|U=u}(o) logp_{O|U=u}(o)) \quad (8)
$$

Since every u is corresponding to a fixed spike 0 or 1, $p_{O|U=u} = 0$ or 1. So we have

$$
H(O|U) = \sum_{u} p(u)(0 + 0 + \cdots + 0) = 0 \quad (9)
$$

Then maximizing the mutual information $I(U;O)$ is equivalent to maximizing the information entropy $H(O)$:

[*]Equal Contributions.

[†]Corresponding author.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

$$\arg\max_{U,O} I(U;O) = H(O) \tag{10}$$

A.2 Algorithm

The proposed training algorithm of an SNN is presented in Algo.1.

Algorithm 1 The proposed training algorithm of an SNN.

Input: Initialized SNN; training dataset; total training epochs, I; training iterations per epoch, J.
Output: The trained SNN.

for all $i = 1, 2, \ldots, I$-th epoch **do**
  for all $j = 1, 2, \ldots, J$-th iteration **do**
    Forward propagation:
    Compute classification loss $\mathcal{L}^j_{CE}$.
    for all $l = 1, 2, \ldots, (L-1)$-th layers (IM-Loss is not added for the last output layer.) **do**
      Compute $\bar{\mathbf{U}}_l$ for l-th layer.
    end for
    Compute IM-Loss $\mathcal{L}^j_{IM}$.
    Compute overall loss $\mathcal{L}^j_{Total}$.
    Back propagation:
    Update the $g'(\cdot)$ via ESG:
    $g'(x) = \frac{1}{2}K(i)(1 - tanh(K(i)(x - V_{th}))^2)$
    Calculate the gradients w.r.t. $\mathbf{W}$:
    $\frac{\partial \mathcal{L}^j_{Total}}{\partial \mathbf{W}} = \sum_{t=1}^{T} \frac{\partial \mathcal{L}^j_{Total}}{\partial y^t} g'(u^t) \frac{\partial u^t}{\partial \mathbf{W}}$, where y^t and u^t denote the target output and membrance potential at t-th timestep.
    Parameters Update
    Update $\mathbf{W} : \mathbf{W} = \mathbf{W} - \eta \frac{\partial \mathcal{L}^j_{Total}}{\partial \mathbf{W}}$, where η is learning rate.
  end for
end for
return the trained SNN.