MSAT: Multi-stage adaptive threshold for Deep Spiking Neural Networks

David S. Hippocampus*
Department of Computer Science
Cranberry-Lemon University
Pittsburgh, PA 15213
hippo@cs.cranberry-lemon.edu

Abstract

Spiking Neural Networks(SNNs) can do inference with low-power consumption natively because of its spike sparsity. Compared with the other two traning method: STDP and BP, Conversion from Artifical Neural Networks(ANNs), is a more easlier way to acheive deep SNNs and commly have the approximate performance compared with ANN. However, Conversion SNNs suffer from a accuracy degration and more latency at inference time. Lots of studys have tryed to make a trade-off between improving accuracy and reduce the latency using varied method including adjust ANN topology when mapping ANN to SNN, using a more efficient fring mechanism et.al Here we analyze conversion loss layer-to-layer and point it out that membrane potential matters in both SNN accuracy and inference latency. subsequently, we give a new perspective that most of current conversion method is optimization membrane potential to achieve higher accuracy and short latency. Meanwhile, Different from current converson schemes which use the same and invariant threshold with inference time in a layer, we propose a multi-stage adaptive threshold for deep spiking neural Networks. We examine the performance on CIFAR-100 and ImageNet for classfication. Futuremore, we show the propose method also behave well in objection detection on VOC and COCO. All above provide support on biological interpretability.

1 Introduction

At present, Artificial Neural Network (ANN) is widely used in speech recognition, image processing and other fields. However, with the complexity of neural networks increasing progressively, running such deep networks often requires large amounts of computational resources, such as memory and power. In addition, current ANN's work mechanism differs from out brain. Actually, Neurons in the brain communicate by transmitting sequences (i.e. spike) generated by action potentials. Spiking Neuron network (SNN) works in a similar way. It also transmits the spike sequence to the downstream neurons. These spikes often carry a high amount of information, and the spike distribution is sparse, so it has the characteristics of low power consumption.

SNNs potentially offer an efficient way of doing inference when it conbines with neuroncomputing hardware, futuremore, SNN inherently shows efficiency on processing temporal and spatial data. Its diverse coding mechanisms, and events-driven characteristics are also promising. However, because the internal state variables of neurons do not satisfy the continuously differentiable requirement, it is difficult to be trained. To solve this problem, some algorithms based on the rules of gradient descent and spike-time dependent plasticity (STDP) were proposed, which had partly solved the problem

*Use footnote for providing further information about author (webpage, alternative address)—*not* for acknowledging funding agencies.

Preprint. Under review.

of training SNNs. Frustratingly, It is still difficult to train deeper SNNs with complex network structures, and results in a remaining of huge gap of performance between SNNs and CNNs in complex recognition or detection tasks.

To narrow the performance gap between SNNs and CNNs, methods of converting CNNs to SNNs had been proposed. In these methods, a CNN is firstly trained using the standard stochastic gradient descent and back propagation algorithm, and then the trained weights are mapped to an SNN with the same structure as the CNN. Inference is performed on the converted SNNs. The main idea is that the firing rates of spiking neurons can approximate the activations of their counterparts (RELU) in ANNs with sufficient time steps. This finding has become the fundamental principle underlying the conversion scheme. Converted SNNs often suffer from a accu- racy degration and more latency at inference time. Lots of studys have tryed to make a trade-off between improving accuracy and reduce the latency using var- ied method including adjust ANN topology when mapping ANN to SNN, using a more efficient fring mechanism et.al. Here we forms conversion loss formula and shows that residual membrane potential in each IF neuron increase the latency which mean firing rates approximate to activation value. We also find that most of current converson schemes, they use threshold invariant with inference time and are same and in a layer. This mechanism is inconsistent with a phenomenon which has been widely observed in the central nervous system, e.g. visual cortex , auditory midbrain, hippocampus, somatosensory cortex. It has been proposed that threshold variability reflects an adaptation of the spike threshold to the membrane potential. Inspired by this, we propose a multi-stage adaptive threshold for deep spiking neural Networks. For each neuron, its threshold vaires with its own membrane potential.We both do experimental on object recognition and detection in non-trival datasets to prove proposed method is as well as efficiency with the current mainstream schemes when doing visual tasks.

Our major contribution can be summarized as:

- sufficient experimental on object recognition and detection in non-trival datasets, shows that our proposed method is both efficiency and biological interpretability
- a formula on layer-by-layer conversion error, a new perspective diving existing method into membrane potential optimization
- a multi-stage adaptive threshold mechanism, which is widely existing in the center nervous system and thus more biological plausible. We use it for deep spiking neural Networks.

2 PRELIMINARIES

Our conversion pipeline exploits the threshold balancing mechanism (Diehl et al., 2015; Sengupta et al., 2018) between ANN and SNN with modified ReLU function on the source ANN to reduce the consequential conversion error. Through in this mechanism, we give a two-layer MLP conversion Framework Diagram as fig1 to Convenient for our discussion.

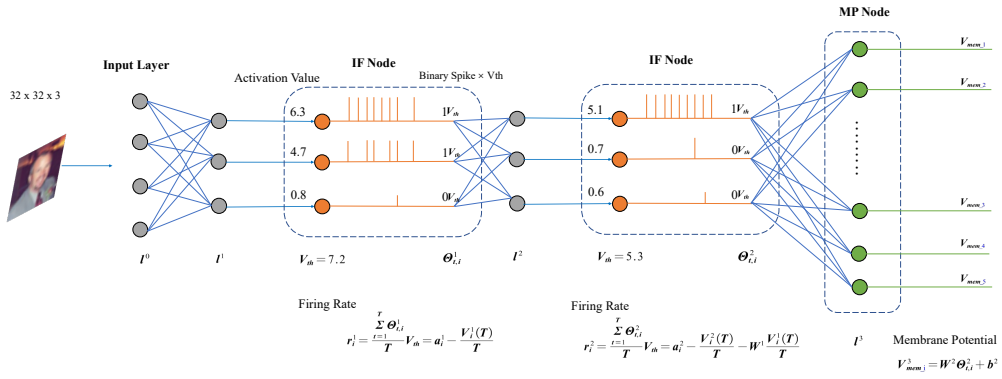

Figure 1: A two-layer MLP conversion Framework for demonstration.

We here introduce common notations used in the current paper.

3 Diving existing method into membrane potential optimization

All headings should be lower case (except for first word and proper nouns), flush left, and bold.

First-level headings should be in 12-point type.

3.1 Headings: second level

Second-level headings should be in 10-point type.

3.1.1 Headings: third level

Third-level headings should be in 10-point type.

Paragraphs There is also a `\paragraph` command available, which sets the heading in bold, flush left, and inline with the text, with the heading followed by 1 em of space.

4 Multi-stage adaptive threshold

Let's take a look at existing methods, no matter weight normalization or threshold balancing, they aim at zipping the gap between ANN and SNN. Though, we should be aware that the real advantage of SNN is its sparse spike which simultaneously low-power and brain-Inspired. Current method, however, set threshold voltage as the same in the same layer and these threshold will remain unchanged despite inference time increasing. It ignores a fact that neurons in different regions of brain represent distinct dynamics and process information differently from other regions. The threshold voltage of neurons is also known to have a broad range rather than a single value. Some neuroscience literature indicates that threshold value is variable in the same neuron and threshold variability is a genuine feature of neurons, which reflects adaptation to the membrane potential at a short timescale.

In vivo, the spiking threshold, varies with firing history and input properties. This phenomenon has been widely observed in the central nervous system, e.g. visual cortex [1,2], auditory midbrain [3], hippocampus [4], somatosensory cortex [5]. It has been proposed that threshold variability measured in vivo reflects an adaptation of the spike threshold to the membrane potential. To our best knowledge, threshold varies in conversion only be used in [x], but they only use two-stage threshold and still cannot represent the homoeostasis well.

Inspired by this, we propose a adaptive threshold, which is multi-stage and vaires with inference time. The method can be briefly sumed up as: **spike threshold adapts tracking the membrane potential at a short timescale**. Also, it is consistent with some other threshold adaptation models: the threshold increases after each spike and decreases if there is no spike. The relationship between threshold and membrane potential is described as

$$\frac{dV_{th}}{dt} = \gamma(R - T) \tag{1}$$

where V_{th} is variable threshold value, and T is target activity defined for the neurons (i.e., a number of times an output neuron should spike over an extended period of time). here T is equivalent to V_{mem}. R is mean firing rate for a neuron, γ is a multiplicative postive constant.

Thus, the equation[x] can be rewrited as the following form

$$V_i^l(t) = V_i^l(t-1) + \sum_j^{M^{l-1}} V_{th,j}^l(t)\left(W_{ij}^l o_{t,j}^{l-1} + b_i^l\right) - V_{th,i}^l(t) o_{t,i}^l \tag{2}$$

firing rate at timestep T is

$$R_i^l = \frac{\sum_{t'=1}^{T} V_{th,i}^l(t') o_{t',i}^l}{T} \tag{3}$$

the firing rate in higher layer still satisfy while neuron membrane potential can be faster close to zero, so the spike information could be more efficiency and thus shorten the conversion latency.

Above equation ensures that all the output neurons are used and adjust the neurons' thresholds to the stimuli for which they become specialized. What's more, our neuron threshold voltage need to give a initial value, in consideration of brian also uses prior knowledge when inference, so we choose half of max activation value as our initial membrane potential to better fit the model.

The pseudocodes for adaptive threshold algorithm are shown in Algorithm 1.

Algorithm 1 Conversion from ANN to SNN: Multi-stage adaptive threshold

Require: Pretrained ANN, training set, SNN's inference timestep T
Ensure: The converted SNN firing rate approximate ANN activation value with shorter latency

for s = 1 to # of samples **do**
  $a_l \leftarrow$ layer-wise activation value
  for l = 1 to L **do**
    $V_{th}^l \leftarrow \frac{1}{2} max[V_{th}^l, max(a_l)]$
    $SNN.layer[l].V_{th} \leftarrow V_{th}^l$
  end for
end for
for t = 1 to timestep T **do**
  for l = 1 to L **do**
    for j = 1 to neuron number of layer l **do**
      $dV_{th} \leftarrow \gamma(SNN.layer[l].R[j] - SNN.layer[l].V_{mem}[j])$
      $SNN.layer[l].V_{th}[j] \leftarrow SNN.layer[l].V_{th}[j] + dV_{th}$
    end for
  end for
end for

5 EXPERIMENTS

Do not change any aspects of the formatting parameters in the style files. In particular, do not modify the width or length of the rectangle the text should fit into, and do not change font sizes (except perhaps in the **References** section; see below). Please note that pages should be numbered.

Dataset	Method	Network	ANN	SNN Best	$T = 32$	$T = 64$	$T = 128$	$T = 256$
CIFAR100	**Li's Work (P-Norm, p=0.999 + LIPooling + Burst)**	VGG16	**78.49**	**78.52**	**67.33**	**76.35**	**78.08**	**78.32**
	This Work (MSAT, p=1 + LIPooling + Burst)	VGG16	**78.49**	**78.55**	**59.85**	**75.44**	**77.94**	**78.38**
	Li's Work (Channel-Norm, p=0.999 + LIPooling + Burst)	Resnet20	**80.69**	**81.19**	**78.33**	**80.35**	**80.97**	**81.09**
	This Work (MAST, p=1 + LIPooling + Burst)	Resnet20	**80.69**	**80.77**	**78.51**	**80.52**	**80.61**	**80.72**
ImageNet	Li's Work (P-Norm, p=0.999 + LIPooling + Burst)	VGG16	74.27	-	-	-	-	-
	This Work (MSAT, p=1 + LIPooling + Burst)	VGG16	74.27	-	-	-		-
	Li's Work (Channel-Norm, p=0.999 + LIPooling + Burst)	Resnet34	75.16	-	-	-	-	-
	This Work (MAST, p=1 + LIPooling + Burst)	Resnet34	75.16	-	-	-	-	-

Table x: Classification accuracy on CIFAR and ImageNet for our converted SNNs, and compared to other conversion methods and ANN.

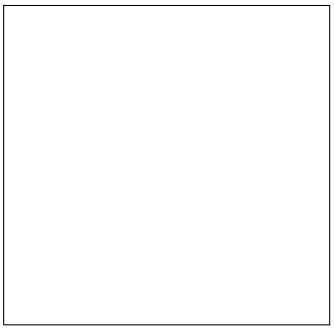

Figure 2: visual Vth varies with inference timestep.

Table x: detection mAP on VOC and COCO for our converted SNNs, and compared to other conversion methods and ANN.

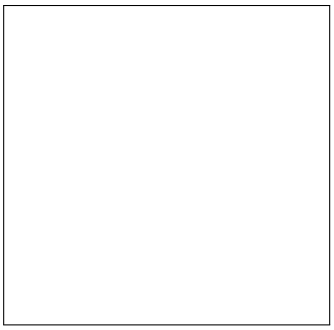

Figure 3: Spike count(efficiency) fig.

Dataset	Method	Network	ANN	SNN Best	$T = 32$	$T = 64$	$T = 128$	$T = 256$
VOC	**Kim's Work (channel-norm)**	Tiny-yolo	-	-	-	-	-	-
	This Work (MSAT)	YOLOv1	-	-	-	-	-	-
COCO	**Kim's Work (channel-norm)**	Tiny-yolo	-	-	-	-	-	-
	This Work (MSAT)	YOLOv1	-	-	-	-	-	-

6 discussion

some conclusion

Acknowledgments and Disclosure of Funding

Use unnumbered first level headings for the acknowledgments. All acknowledgments go at the end of the paper before the list of references. Moreover, you are required to declare funding (financial activities supporting the submitted work) and competing interests (related financial activities outside the submitted work). More information about this disclosure can be found at: `https://neurips.cc/Conferences/2022/PaperInformation/FundingDisclosure`.

Do **not** include this section in the anonymized submission, only in the final paper. You can use the `ack` environment provided in the style file to autmoatically hide this section in the anonymized submission.

References

References follow the acknowledgments. Use unnumbered first-level heading for the references. Any choice of citation style is acceptable as long as you are consistent. It is permissible to reduce the font size to `small` (9 point) when listing the references. Note that the Reference section does not count towards the page limit.

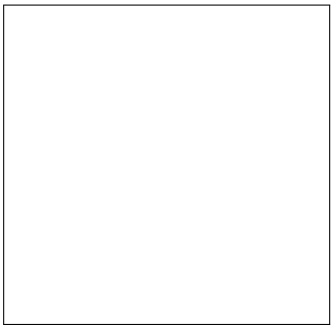

Figure 4: visual detection result and corresponding threshold with inference timestep.

[1] Alexander, J.A. & Mozer, M.C. (1995) Template-based algorithms for connectionist rule extraction. In G. Tesauro, D.S. Touretzky and T.K. Leen (eds.), *Advances in Neural Information Processing Systems 7*, pp. 609–616. Cambridge, MA: MIT Press.

[2] Bower, J.M. & Beeman, D. (1995) *The Book of GENESIS: Exploring Realistic Neural Models with the GEneral NEural SImulation System.* New York: TELOS/Springer–Verlag.

[3] Hasselmo, M.E., Schnell, E. & Barkai, E. (1995) Dynamics of learning and recall at excitatory recurrent synapses and cholinergic modulation in rat hippocampal region CA3. *Journal of Neuroscience* **15**(7):5249-5262.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? **[TODO]**
   (b) Did you describe the limitations of your work? **[TODO]**
   (c) Did you discuss any potential negative societal impacts of your work? **[TODO]**
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? **[TODO]**
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? **[TODO]**
   (b) Did you include complete proofs of all theoretical results? **[TODO]**
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? **[TODO]**
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? **[TODO]**
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? **[TODO]**
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? **[TODO]**
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? **[TODO]**
   (b) Did you mention the license of the assets? **[TODO]**
   (c) Did you include any new assets either in the supplemental material or as a URL? **[TODO]**
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? **[TODO]**
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? **[TODO]**

5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? **[TODO]**
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? **[TODO]**
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? **[TODO]**

A Appendix

Optionally include extra information (complete proofs, additional experiments and plots) in the appendix. This section will often be part of the supplemental material.