Extended Noisy Softplus and training method that enable layered up SNNs to be trained as ANNs

Anonymous Author(s)
Affiliation
Address
email

Abstract

We extended the work of proposed activation function, Noisy Softplus, to fit into training of layered up spiking neural networks (SNNs). Thus, any ANN employing Noisy Softplus neurons, even of deep architecture, can be trained simply by the traditional algorithm, for example Back Propagation (BP), and the trained weights can be directly used in the spiking version of the same network without any conversion. Furthermore, the training method can be generalised to other activation units, for instance Rectified Linear Units (ReLU), to train deep SNNs off-line. This research is crucial to provide an effective approach for SNN training, and to increase the classification accuracy of SNNs with biological characteristics and to close the gap between the performance of SNNs and ANNs.

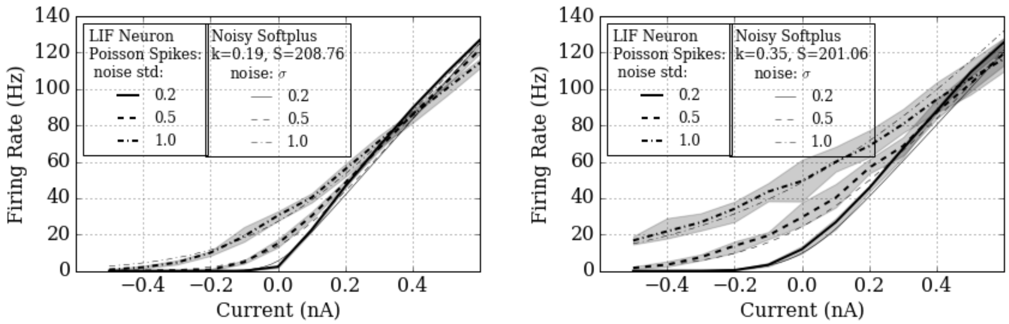
1 Introduction

DNNs are the most promising research field in computer vision, even exceeding human-level performance on image classification tasks [1], and spiking DNNs offer the prospect of neuromorphic systems that combine remarkable performance with energy-efficient training and operation.Theoretical studies have shown that biologically-plausible learning, e.g. Spike-Timing-Dependent Plasticity (STDP), could approximate a stochastic version of powerful machine learning algorithms such as Contrastive Divergence [2], Markov Chain Monte Carlo [3] and Gradient Descent [4].Yet, in practice, SNNs have not achieved the recognition/classification performance of their non-spiking competitor, and it remains an unsolved problem to develop SNNs with equivalent performance.

On the other hand, the offline training of an ANN, which is then mapped to an SNN, has shown near loss-less conversion and state-of-the-art classification accuracy. Jug et al[5] first proposed the use of the Siegert function to replace the sigmoid activation function in training Restricted Boltzmann Machine (RBM). The Siegert units map incoming currents driven by Poisson spike trains to the response firing rate of a Leaky Integrate-and-Fire (LIF) neuron. The ratio of the spiking rate to its maximum is equivalent to the output of a sigmoid neuron. A spiking Deep Belief Network (DBN) [6] of four layers of RBMs was implemented on neuromorphic hardware, SpiNNaker [7], to recognise hand written digits in real time.

Based on the fact that cortical neurons seldom saturate their firing rate as sigmoid neurons,ReLU [8] were proposed to replace sigmoid neurons and surpassed the performance of other popular activation units . Recent developments on ANN-trained SNN models has focused on using ReLU units and converting trained weights to fit in SNNs. Better performance [9, 10] than Siegert-trained RBM has been demonstrated in Spiking ConvNets, but this employed simple integrate and fire (IF) neurons without leakage.The training used only ReLUs and zero bias to avoid negative outputs, and applied a deep learning technique, dropout[11] , to increase the classification accuracy. Normalising the trained weights for use on an SNN employing IF neurons only was relatively straightforward and maintained

Submitted to 31st Conference on Neural Information Processing Systems (NIPS 2017). Do not distribute.



(a) τ_{syn}=1 ms (b) τ_{syn}=10 ms

Figure 1: Noisy Softplus fits to the response firing rates of LIF neurons.

classification accuracy. This work was extended to a Recursive Neural Network (RNN) [12] and run on the TrueNorth[13] neuromorphic hardware platform.

Except for the popular, simplified version of ReLU, $max(0, \sum wx)$, the other implementation of $\log(1 + e^x)$, "Softplus", is more biologically realistic. Recent work [14] proposed the Soft LIF response function for training SNNs, which is equivalent to Softplus activation of ANNs. Furthermore, neuroscientific study has showed that the spike train of individual neurons is far from being periodic, which thus brings noisy to the input signal of spiking neurons [15]. Therefore, in the previous work of Qian Liu et al. [16], the difference between analytical estimation and practical simulations of spiking neurons were compared, and a new activation function named Noisy Softplus was proposed to match the response function of LIF neurons. In order to close the gap between the performance of SNNs and ANNs, and to further improve the performance of SNNs, we extended Noisy Softplus with a scale factor and proposed a complete layered up SNN training method by using artificial neurons of combined activation.

2 ANN-Trained SNNs

2.1 Extended Noisy Softplus

Suppose λ_{out} represents the firing rate of a LIF neuron driven by a noisy current x, we extended Noisy Softplus[16] with a scale factor, S, then

$$\begin{aligned} \lambda_{out} &\simeq f_{ns}(x, \sigma) \times S \\ &= k\sigma \log[1 + \exp(\frac{x}{k\sigma})] \times S. \end{aligned} \tag{1}$$

Noisy Softplus fits well to the practical response firing rate of the LIF neuron with suitable calibration of k and S, see Figure 1. The parameter pair of (k, S) is curve-fitted with the triple data points of (λ, x, σ). The fitted parameter was set to $(k, S) = (0.19, 208.76)$ for the practical response firing rate driven by synaptic noisy current with $\tau_{syn} = 1$ ms and was set to $(k, S) = (0.35, 201.06)$ when $\tau_{syn} = 10$ ms. The calibration currently is conducted by linear least squares regression; numerical analysis is considered however for future work to express the factors with biological parameters of a LIF neuron. From now on, we can model the response firing activity of a LIF neuron with a unified activation function, extended Noisy Softplus.

In order to illustrate how we can use the extended Noisy Softplus to train layered up SNNs, we will demonstrate the mapping between the physical activity and numerical ANN calculations in the following subsections.

2.2 Equivalent Input and Output

Neurons in ANNs take inputs from their previous layer, and feed the weighted sum of their input, $net_j = \sum_i w_{ij} x_i$, to the activation function. The transformed signal then forms the output of an

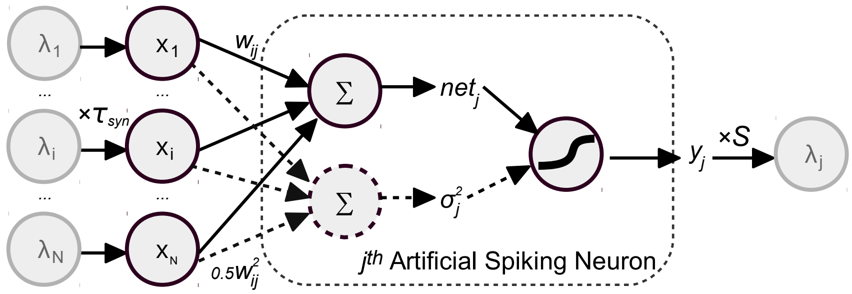


Figure 2: Artificial spiking neuron takes scaled firing rates as input, then transforms weighted sum in some activation unit to its output which can be scaled-up to the firing rate of an output spike train.

artificial neuron, which can be denoted as $y_j = f(net_j)$. However, Noisy Softplus takes physical quantities of current, and firing rate as input/output, thus an extra step is still needed to map the firing rate to numerical values in ANNs. According to Equation [16], the mean of the current feeding into a spiking neuron is equivalent to net of artificial neurons, where

$$\begin{aligned} m_{Ij} &= \sum_i w_{ij}(\lambda_i \tau_{syn}) \text{ , then} \\ net_j &= \sum_i w_{ij} x_i \text{ , and } x_i = \lambda_i \tau_{syn} \text{ .} \end{aligned} \tag{2}$$

The noise level of Noisy Softplus, σ^2, is the variance of the current, which also can be seen as a weighted sum of the same input x but with different weights:

$$\begin{aligned} s^2_{Ij} &= \sum_i (\frac{1}{2} w^2_{ij})(\lambda_i \tau_{syn}) \text{ , then} \\ \sigma^2_j &= \sum_i (\frac{1}{2} w^2_{ij}) x_i \text{ .} \end{aligned} \tag{3}$$

Noisy Softplus transforms the noisy current with parameters of (net_j, σ_j) to the equivalent ANN output y_j , where it can be scaled up by the factor S to the firing rate of SNNs. Note that the calculation of noise level is not necessary for activation functions other than Noisy Softplus, for example, it can be set to a constant for Softplus or 0 for ReLU. We name the neuron model 'artificial spiking neurons' which takes firing rates of spike trains as input and output. The entire artificial spiking neuron model is then generalised to any ReLU/Softplus-like activation functions, See Figure 2.

2.3 Layered-up Network

Referred to Figure 2, if we move the left end process of $\times \tau_{syn}$ to the right end after λ_j, Figure 2 forms the same neuron model and structure as multilayer perceptron: neurons take x as input and outputs y, and this conversion is illustrated in Figure 3. The process within such an artificial neuron is divided into weighted summation and activation, which also applies to SNN modelling by combining the scaling factor S and the synaptic time constant τ_{syn} to activation functions. Thus the combined activation function for modelling SNNs should be:

$$y = f(x) \times S \times \tau_{syn} \text{ ,} \tag{4}$$

and its derivative function which is used when back propagates is:

$$\frac{\partial y}{\partial x} = f'(x) \times S \times \tau_{syn} \text{ .} \tag{5}$$

Thus, using this method of ANN-trained SNNs, the activation functions are of lower complexity than the Siegert formula, and their corresponding derivative functions can be directly used for back propagation. Furthermore, the method enables ReLU-like activation functions for SNN training, thus improving the recognition accuracy while keeping a relative lower firing rate compared to sigmoid neurons. Most significantly, the ANN-trained weights are ready for use in SNNs without any transformation, and the output firing rate of a spiking neuron can be estimated in the ANN simulation.

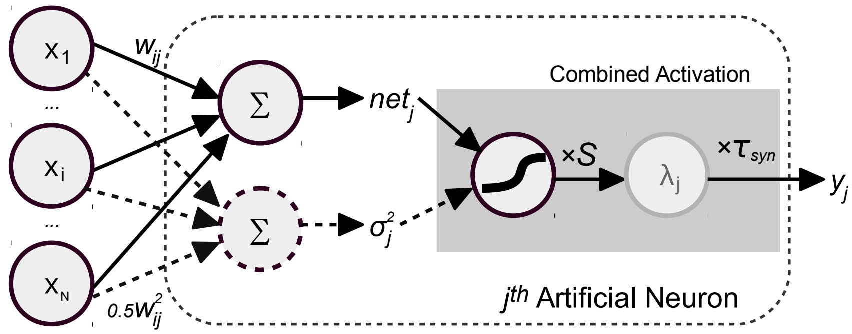


Figure 3: Transforming artificial spiking neurons to artificial neurons for SNN modelling. The combined activation links the firing activity of a spiking neuron to the numerical value of ANNs.

2.4 Fine Tuning

There are two aspects to the fine tuning which makes the ANN closer to SNNs: Firstly, using Noisy Softplus activation functions in a whole trained network operates every single neuron running in a similar noise level as in SNNs, thus the weights trained by other activation functions will be tuned to fit closer to SNNs. Secondly, the output firing rate of any LIF neuron is greater than zero as long as noise exists in their synaptic input. Thus adding up a small offset on the labels directs the model to approximate to practical SNNs.

The labels of data are always converted to binary values for ANN training. This enlarges the disparities between the correct recognition label and the rest to train the network for better classification capability. Consequently, we can train the network with any activation function and then fine-tune it with Noisy Softplus to take account of both accuracy and practical network activities of SNNs. However, we add a small number, for example 0.01, to all the binary values of the data labels. Doing so helps the training to loosen the strict objective function to predict exact labels with binary values. Instead, it allows a small offset to the objective. An alternative method is to use Softmax function at the top layer, which aims to map real vectors to the range of $(0, 1)$ that add up to 1. However, without a limit on the input of Softmax, it will be easy to reach or even exceed the highest firing rate of a spiking neuron. The result of fine tuning on a Convnet will be demonstrated in subsection 3.2.

3 Results

A convolutional network model was trained on MNIST, a popular database in neuromorphic vision, using the ANN-trained SNN method stated above. The architecture contains 28×28 input units, followed by two convolutional layers 6c5-2s-12c5-2s, and 10 output neurons fully connected to the last pooling layer to represent the classified digit.

The training only employed Noisy Softplus units that all the convolution, average sampling, and the fully-connected neurons use Noisy Softplus function with no bias. The parameters of the activation function were calibrated as, $(k = 0.30, S = 201)$, for LIF neurons ($C_m = 0.25$nF, $\tau_m = 20.0$ms, $\tau_{refrac} = 1.0$ms, $v_{reset} = -65.0$mV, $v_{rest} = -65.0$mV, $v_{thresh} = -50.0$mV, $i_{offset} = 0.1$nA, $\tau_{syn} = 5$ ms). The input images were scaled by 100 Hz to present the firing rates of input spikes. The weights were updated using a decaying learning rate, 50 images per batch and 20 epochs. The ANN-trained weights were then directly applied in the corresponding convolutional SNN without any conversion for recognition tasks.

3.1 Neural Activity

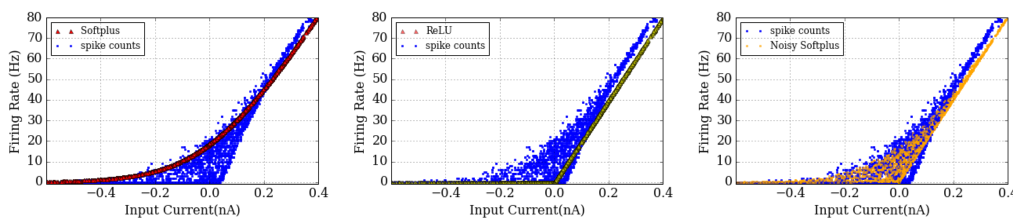
To validate how well the Noisy Softplus activation fits to the response firing rate of LIF neurons in a real application, we simulated the model on NEST using the Poisson MNIST dataset [17] and the neurons of a convolutional map were observed.

A small test of ten MNIST digits presented in Poisson spike trains for 1 s each. A trained 5×5 kernel was convolved with these input digits, and the convolved output of the feature map, the output firing rate was recorded during a real-time SNN simulation on NEST, and compared to the modelled activations of Equation 4 in ANNs.

The input x of the network was calculated as Equation 2: $x_i = \lambda_i \tau_{syn}$, and so as the weighted sum of the synaptic current (see Equation 2), net_j and its variance (see Equation 3), σ_j^2. With three combined activation functions as Equation 4:

$$
\begin{aligned}
&(1)\ \text{Noisy Softplus:}\ \ y_j = k\sigma_j \log[1+\exp(\frac{net_j}{k\sigma_j})] \times S \times \tau_{syn}\ \ , \\
&(2)\ \text{ReLU:}\ \ \ y_j = max(0, net_j) \times S \times \tau_{syn}\ \ , \\
&(3)\ \text{Softplus:}\ \ \ y_j = k\sigma \log[1+\exp(\frac{net_j}{k\sigma})] \times S \times \tau_{syn}\ \ ,\ \ \ \sigma = 0.45,
\end{aligned}
\tag{6}
$$

we compare the output to the recorded SNN simulations. ReLU assumes a non-noise current, and Softplus takes a static noise level thus σ_j is not used for either of them, meanwhile Noisy Softplus adapts to noise automatically with σ_j. The experiment took the sequence of 10 digits to the same kernel and the estimated spike counts using Noisy Softplus fit to the real recorded firing rate much more accurately than ReLU and Softplus, see 4. The Euclidean distance, $\sqrt{\sum_j (y_j/\tau_{syn} - \lambda_j)}$, between the spike counts and the predicted firing rates by Noisy Softplus, ReLU and Softplus was 184.57, 361.64 and 1102.76 respectively. We manually selected a static noise level of 0.45 for Softplus, whose estimated firing rates located roughly on the top slope of the real response activity. This resulted in longer Euclidean distance than using ReLU, since most of the input noisy currents were of relatively low noise level in this experiment. Hence, the firing rate driven by lower noise level is closer to ReLU curve than Softplus.



(a) Softplus (b) ReLU (c) Nosiy Softplus

Figure 4: Noisy Softplus fits to the neural response firing rate in an SNN simulation. The recorded firing rate of the same kernel convolved with 10 images in SNN simulation, comparing to the prediction of activations of Softplus, ReLU, and Noisy Softplus.

The SNN successfully classified the digits where the correct label neuron fired the most. We trained the network with binary labels on the output layer, thus the expected firing rate of correct classification was $1/\tau_{syn} = 200$ Hz according to Equation 3. The firing rates of the recognition test fell to the valid range around 0 to 200 Hz. This shows another advantage of the proposed ANN-trained method that we can constrain the expected firing rate of the top layer, thus preventing SNN from exceeding its maximum firing rate, for example 1000 Hz when time resolution of SNN simulation set to 1 ms.

3.2 Recognition Performance

Here we focus on the recognition performance of the proposed ANN-trained SNN method. Before looking into the recognition results, it is significant to see the learning capability of the proposed activation function, Noisy Softplus. We compared the training using ReLU, Softplus, and Noisy Softplus by their loss during training averaged over 3 trials, see Figure 5. ReLU learned fastest with the lowest loss, thanks to its steepest derivative. In comparison, Softplus accumulated spontaneous firing rates layer by layer and its derivative may experience vanishing gradients during back propagation, which result in a more difficult training. Noisy Softplus performance lay between these two in terms of loss and learning speed. However, the loss stabilised fastest, which means a possible shorter training time.

The recognition test took the whole testing dataset of MNIST which contains $10,000$ images. At first, all trained models were tested on the same artificial neurons as used for training in ANNs,

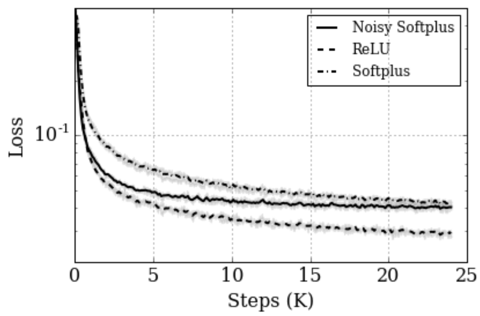


Figure 5: Comparisons of Loss during training using Noisy Softplus, ReLU and Softplus activation functions. Bold lines show the average of three training trials, and the grey colour illustrates the range between the minimum and the maximum values of the trials.

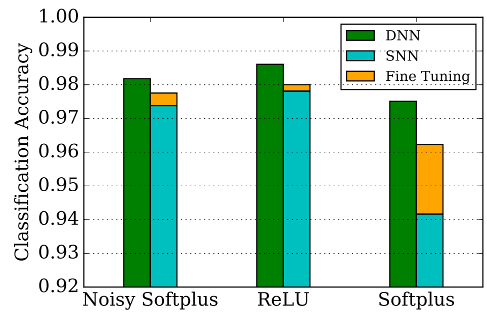


Figure 6: Classification accuracy compared among trained weights of Noisy Softplus, ReLU, Softplus on DNN, SNN and fine-tuned SNN.

and these experiments were called ‘DNN’ test since the network had a deep structure (5 layers). Subsequently, the trained weights were directly applied to SNN without any transformation, and these ‘SNN’ experiments tested their recognition performance on the NEST simulator. The LIF neurons had the same parameters as in training. The input images were converted to Poisson spike trains and presented for 1 s each. The output neuron which fired the most indicated the classification of an input image. Moreover, a ‘Fine tuning’ test took the trained model for fine tuning, and the tuned weights were tested on the same SNN environment. The tuning only ran for one epoch, 5% cost of the ANN training (20 epochs), using Noisy Softplus neurons with labels shifted for $+0.01$.

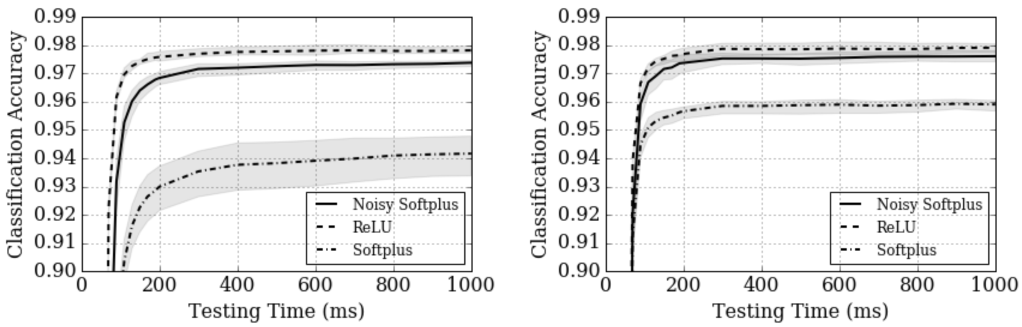
The classification errors for the tests are investigated in Table 1 and the averaged classification accuracy is shown in Figure 6. From DNN to SNN, the classification accuracy declines by 0.80%, 0.79% and 3.12% on average for Noisy softplus, ReLU and Softplus The accuracy loss was caused by the mismatch between the activations and the practical response firing rates, see example in Figure 4, and the strict binary labels for Noisy Softplus and Softplus activations. Fortunately, the problem is alleviated by fine tuning which increased the classification accuracy by 0.38%, 0.19% and 2.06%, and resulted in the total loss of 0.43%, 0.61%, and 1.06% respectively. The improvement of ReLU is not as great as the others, because there is no problem of strict labels during training. Softplus benefits the most from fine tuning, since not only the huge mismatch of response firing rate is greatly corrected, but also the offset on the labels helps the network to fit SNNs.

Table 1: Comparisons of classification accuracy (in %) of ANN-trained convolutional neural models on original DNN, NEST simulated SNN, and SNN with fine-tuned (FT) model.

Trial No.	1			2			3		
Model	DNN	SNN	FT	DNN	SNN	FT	DNN	SNN	FT
Noisy Sofplus	1.91	2.76	2.45	1.79	2.56	2.19	1.76	2.55	2.10
ReLU	1.36	2.03	1.88	1.46	2.28	2.00	1.36	2.25	2.12
Sofplus	2.30	5.66	3.91	2.75	5.22	3.55	2.42	6.62	3.87

The most efficient training in terms of both classification accuracy and algorithm complexity, takes ReLU for ANN training and Noisy Softplus for fine tuning. Softplus does not exhibit better classification capability and more importantly the manual selected static noise level hugely influences the mismatch between the predicted firing rates and the real data. Although Noisy Softplus shows the least classification drop from ANNs to SNNs, the training performance is still worse than ReLU.

The best classification accuracy achieved by SNN was 98.85%, a 0.20% drop from ANN test (99.05%), which was trained with ReLU and fine-tuned by Noisy Softplus. The network structure was the same with the state-of-the-art model which reported the best classification accuracy of 99.1% [10] in ANN-trained SNNs: 12c5-2s-64c5-2s-10fc. Their nearly loss-less conversion from ANNs to SNNs was achieved by using IF neurons, while our network performs the best among SNNs consisted of LIF neurons to our knowledge.



(a) Before fine tuning

(b) After fine tuning.

Figure 7: The classification accuracy of 3 trials (averaged in bold lines, grey shading shows the range between minimum to maximum) over short response times, with (a) trained weights before fine tuning, and (b) after fine tuning.

As it is a major concern in neuromorphic vision, the recognition performance over short response times is also estimated in Figure 7. After fine tuning, Softplus significantly reduced the mismatch since the randomness among the three trials shrinks to a range similar to other experiments. More obviously, fine tuning improved its classification accuracy and the response latency. Notice that all of the networks trained by three different activation functions showed a very similar stabilisation curve against time, which means they all reached an accuracy close to their best by only taking 300 ms of test.

3.3 Power Consumption

Noisy Softplus can easily be used for energy cost estimation for SNNs. For a single neuron, the energy consumption of the synaptic events it triggers is:

$$\begin{aligned} E_j &= \lambda_j N_j T E_{syn} \\ &= \frac{y_j N_j T E_{syn}}{\tau_{syn}} \quad , \end{aligned} \tag{7}$$

where λ_j is the output firing rate, N_j is the number of post-synaptic neurons it connects to, T is the testing time, and E_{syn} is the energy cost for a synaptic event of some specific neuromorphic hardware, for example, about 8 nJ on SpiNNaker [18]. Thus to estimate the whole network, we can sum up all the synaptic events of all the neurons:

$$\sum_j E_j = \frac{T E_{syn}}{\tau_{syn}} \sum_j y_j N_j . \tag{8}$$

Thus, it may cost SpiNNaker 0.064 W, 192 J running for $3,000$ s with synaptic events of $8 \times 10^6/s$ to classify $10,000$ images (300 ms each) with an accuracy of 98.02%. The best performance reported using the larger network may cost SpiNNaker 0.43 W operating synaptic event rate at $5.34 \times 10^7/s$, consume 4271.6 J to classify all the images for 1 s each.

4 Discussions

Most significantly, we proposed the Noisy Softplus activation function which accurately models response firing rate of LIF neurons and overcomes the drawbacks of Siegert units.

- Noisy Softplus takes account of time correlation of the noisy synaptic current, e.g. τ_{syn}, which fits more to the actual response firing rate.

- Noisy Softplus can be applied easily to any training method, for example BP, thanks to its differentiability.
- the calculation on Noisy Softplus is no more than Softplus function, except for doubled computation on weighted sum of its input (net and σ in Equations 2 and 3), which is much more simplified than Siergert function.
- as one of the ReLU-liked activation function, the output firing rate seldom exceed the working range of a LIF neuron, for example the firing rates were around 0-200 Hz in the ConvNet model.
- the learning performance of Noisy Softplus is between Softplus and ReLU, which is supposed to outperform most of the other popular activation functions: for instance sigmoid.

Moreover, we proposed complete SNN modelling method by using artificial neurons of combined activation; this method can be generalised to activation units other than Noisy Softplus. The training of an SNN model is exactly the same as ANN training, and the trained weights can be directly used in SNN without any transformation. This method is simpler and even more straight-forward than the other ANN offline training methods which requires an extra step of converting ANN-trained weights to SNN's.

In terms of classification/recognition accuracy, the performance of ANN-trained SNNs is nearly equivalent as ANNs, and the performance loss can be partially solved by fine tuning. The best classification accuracy of 98.85% using LIF neurons in PyNN simulation outperforms state-of-the-art SNN models of LIF neurons which will be listed in Chapter 6, and is very close to the result using IF neurons [10].

Acknowledgments

To be added after reviewing.

References

[1] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1026–1034, 2015.

[2] Emre Neftci, Srinjoy Das, Bruno Pedroni, Kenneth Kreutz-Delgado, and Gert Cauwenberghs. Event-driven contrastive divergence for spiking neuromorphic systems. *Frontiers in neuroscience*, 7, 2013.

[3] Lars Buesing, Johannes Bill, Bernhard Nessler, and Wolfgang Maass. Neural dynamics as sampling: a model for stochastic computation in recurrent networks of spiking neurons. *PLoS Comput Biol*, 7(11):e1002211, 2011.

[4] Peter O'Connor and Max Welling. Deep spiking networks. *arXiv preprint*, 2016.

[5] F. Jug, J. Lengler, C. Krautz, and A. Steger. Spiking networks and their rate-based equivalents: does it make sense to use Siegert neurons? In *Swiss Soc. for Neuroscience*, 2012.

[6] Evangelos Stromatias, Daniel Neil, Francesco Galluppi, Michael Pfeiffer, Shih-Chii Liu, and Steve Furber. Scalable energy-efficient, low-latency implementations of trained spiking deep belief networks on SpiNNaker. In *Neural Networks (IJCNN), 2015 International Joint Conference on*, pages 1–8. IEEE, 2015.

[7] Steve B Furber, Francesco Galluppi, Sally Temple, Luis Plana, et al. The SpiNNaker project. *Proceedings of the IEEE*, 102(5):652–665, 2014.

[8] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *International Conference on Artificial Intelligence and Statistics*, pages 315–323, 2011.

[9] Yongqiang Cao, Yang Chen, and Deepak Khosla. Spiking deep convolutional neural networks for energy-efficient object recognition. *International Journal of Computer Vision*, 113(1):54–66, 2015.

[10] Peter U Diehl, Daniel Neil, Jonathan Binas, Matthew Cook, Shih-Chii Liu, and Michael Pfeiffer. Fast-classifying, high-accuracy spiking deep networks through weight and threshold balancing. In *Neural Networks (IJCNN), 2015 International Joint Conference on*, pages 1–8. IEEE, 2015.

[11] Nitish Srivastava, Geoffrey E Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[12] Peter U Diehl, Guido Zarrella, Andrew Cassidy, Bruno U Pedroni, and Emre Neftci. Conversion of artificial recurrent neural networks to spiking neural networks for low-power neuromorphic hardware. *arXiv preprint*, 2016.

[13] Paul A Merolla, John V Arthur, Rodrigo Alvarez-Icaza, Andrew S Cassidy, Jun Sawada, Filipp Akopyan, Bryan L Jackson, Nabil Imam, Chen Guo, Yutaka Nakamura, et al. A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science*, 345(6197):668–673, 2014.

[14] Eric Hunsberger and Chris Eliasmith. Spiking deep networks with lif neurons. *arXiv preprint*, 2015.

[15] Wulfram Gerstner and Werner Kistler. *Spiking Neuron Models: An Introduction*. Cambridge University Press, New York, NY, USA, 2002.

[16] Qian Liu and Steve Furber. Noisy softplus: A biology inspired activation function. In *Proceedings of the 23rd International Conference on Neural Information Processing, Part IV*, pages 405–412, 2016.

[17] Qian Liu, Evangelos Pineda-García Garibaldia nd Stromatias, Teresa Serrano-Gotarredona, and Steve Furber. Benchmarking spike-based visual recognition: A dataset and evaluation. *Frontiers in Neuroscience*, 10:496, 2016. http://journal.frontiersin.org/article/10.3389/fnins.2016.00496.

[18] Evangelos Stromatias, Francesco Galluppi, Cameron Patterson, and Steve Furber. Power analysis of large-scale, real-time neural networks on SpiNNaker. In *Neural Networks (IJCNN), The 2013 International Joint Conference on*, pages 1–8, 2013.