WEIGHT REGULARIZATION IN SPIKING NEURAL NETWORKS[1]

D.I. Antonov[1,2], S.V. Sukhov[2]

[1]Ulyanovsk State Technical University

[2]Ulyanovsk Branch of Kotel’nikov Institute of Radio Engineering and Electronics of Russian Academy of Sciences, ***d.antonov@ulireran.ru***

The separation of useful information from noise is an essential process in many machine-learning phenomena. The process of useful feature extraction and their separation from noisy input can be found in many applications of artificial neural networks (ANNs).

The desired result of any learning is not the mere memorization of training data but the ability to generalize the knowledge for solving new problems [1]. During the training of the ANN, the error on the training set (loss function) is calculated and minimized. However mindless minimization of the loss function may result in *overfitting* of the model [2]. Overfitting is a result of training taking into account both essential and insignificant features, noise. *Regularization* methods are intended to minimize the influence of random noise and to identify regular features during the model training process. Thus, regularization helps in preventing ANNs from overfitting.

Segregation and different treatment of familiar and unfamiliar data and noise might be necessary during the predictive processing of information in ANNs [3]. Unfamiliar information should propagate further along the neural network for processing, the familiar (expected) result should not result in additional processing, and the noise should not propagate along the network at all [3].

Another serious problem for practical applications of artificial neural networks is their weak protection against various types of adversarial attacks. Noise-like changes to the images that are insignificant for human perception may cause errors in their classification by ANNs. This, in turn, may cause major failures in the operation of ANN-based devices.

To prevent the overfitting in ANNs or make them more robust against adversar-

[1] The reported study was funded by the Russian Science Foundation (project number 24-21-00470).

ial attacks, various regularization methods are used. Many regularization methods are related to the minimization or pruning of some redundant weights. In the practice of ANN training, three methods are most often used: dropout, L1-, and L2-regularizations.

The L2-regularization (also called Ridge regression) uses the square of weights as a penalty term to the loss function:

$$F_{loss} = E(Y - \hat{Y}) + \lambda \sum_{i=0}^{n} w_i^2, \quad (1)$$

where λ is the regularization parameter, E is some error function measuring the difference between ground truth Y and the prediction $\hat{Y}$. The L2-regularization encourages a more balanced distribution of weights across features and prevents them from acquiring large values.

In the case of L1-regularization (also called Lasso regression), in the loss function F_{loss}, a regularization term with absolute values of weights is added:

$$F_{loss} = E(Y - \hat{Y}) + \lambda \sum_{i=0}^{n} |w_i|. \quad (2)$$

While L2-regularization minimizes weights, L1-regularization makes redundant weights equal exactly to zero creating a sparse model.

Similar to L1-regularization, dropout involves randomly selecting weights or entire neurons to be removed from the learning process. The effect of dropout is that the network becomes less sensitive to the specific weights of neurons.

Spiking neural networks (SNNs) are the next generation of neural networks, with a promise of more energy-efficient and more biologically plausible computations [4]. The neurons in SNNs transmit information by short pulses (spikes), and training of SNNs is performed by local rules. Some prior research found that SNNs are more robust against exposure to noise and more robust against adversarial attacks [5-7]. However, other research found SNNs still susceptible to noise [8]. Here, we propose a regularization method for SNNs that significantly reduces the neural activity caused by noise input.

Instead of error backpropagation, local training methods are often used in SNNs. Spike-timing-dependent plasticity (STDP) is the most frequently used learning rule for unsupervised learning in SNNs. The property of STDP to cluster data directly corresponds to the needs of unsupervised learning. During such training according to the STDP rule, the synaptic weights change according to the law:

$$\Delta w(\Delta t) = \begin{cases} A_{pre} \cdot \exp(-\Delta t/\tau_{pre}), & \Delta t > 0 \\ A_{post} \cdot \exp(\Delta t/\tau_{post}), & \Delta t < 0 \end{cases} \tag{3}$$

where Δt is the time interval between the occurrence of a spike in the postsynaptic neuron and the occurrence of a spike in the presynaptic neuron, the coefficients $A_{pre} > 0$, $A_{post} < 0$, and the time constants $\tau_{pre} > 0$ and $\tau_{post} > 0$.

Supervised learning is commonly used for classification and regression problems. In [9, 10], we modified STDP to be applicable for supervised learning. A law similar to eq. (3) describes another bio-plausible rule, the all-long-term-depression rule (all-LTD). The change in synaptic connection in the case of all-LTD is determined by the expression:

$$\Delta w(\Delta t) = \begin{cases} -A_{pre} \cdot \exp(-\Delta t/\tau_{pre}), & \Delta t > 0 \\ A_{post} \cdot \exp(\Delta t/\tau_{post}), & \Delta t < 0 \end{cases} \tag{4}$$

The STDP and all-LTD rules were used for supervised learning of the SNN network [10].

Both STDP and all-LTD learning rules do not assume any global loss. Thus, usual regularization methods (1), (2) are not suitable for SNNs. Thus, redundant weights are not eliminated during the training process. Indeed, looking at eqs. (3), (4) one can notice that weights change only in the case of presynaptic neuron firing. In the case of silent presynaptic neurons, their outgoing weights remain the same as during initialization. These redundant weights can be later misused for the propagation of noise or unnecessary/adverse information.

In this paper, we propose the weight regularization method for SNNs based on the biological 'use it or lose it' concept [11]: if a synaptic connection is not used in a brain, it is pruned. To ensure the pruning of unnecessary weight in SNNs, all the weights become time-dependent:

$$w(t) = w(t_s) \exp\left(-k_p \left(\frac{t-t_s}{\tau_{\text{prun}}}\right)^2\right), \tag{5}$$

where t_s is the time of the last spike, τ_{prun} is the characteristic time over which pruning occurs, k_p is the pruning coefficient. For frequently spiking neurons, the exponent in eq. (5) is almost unity and no weight decay occurs. However, the weights of a neuron silent over $\tau_{\mathbf{prun}}$ time become pruned.

To test our regularization concept, we perform a series of experiments with SNNs. In experiments, the SNN is trained with the aim of classification of MNIST images of handwritten digits. The purpose of the experiments was to compare the SNN output activity with or without weight regularization. The SNN in all the experiments uses the leaky integrate-and-fire neurons with an adaptive threshold for excitatory neurons. The network consists of two layers (Fig. 1) and has an architecture similar to the one described in the paper [12]. The first input layer of the SNN (Fig. 1) contains 784 Poisson neurons, and the second layer consists of an equal number of excitatory and inhibitory neurons (10 neurons per data class). Each neuron in the Poisson layer corresponds to one pixel of input data. Each Poisson neuron generates a spike train with an average frequency equal to the intensity of a given pixel (rate coding). The signal passes from Poisson neurons to excitatory neurons of the second layer through synapses of the S_1 group in a one-to-all manner. Through the synapses of the S_2 group, the spikes pass to the inhibitory neurons of the second layer in a one-to-one manner. And through synapses of the S_3 group, the spikes generated by inhibitory neurons return to excitatory neurons in a one-to-all-except-initiator manner.

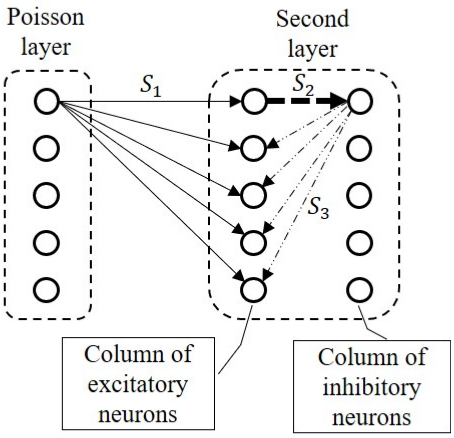

Fig. 1. SNN architecture: the first layer contains 784 Poisson neurons, the second layer contains 30 excitatory and 30 inhibitory neurons; S_1, S_2, and S_3 are the synaptic connections.

To implement the combined Hebbian learning rule *'STDP + all-LTD'* we divided the excitatory neurons of the second layer into subsets, the number of subsets is equal to the number of data classes, and the sizes of these subsets are the same. When the input data of a certain class are given, synapses of the S_1 group associated with a subset of neurons recognizing the target class are trained according to the STDP rule. If an excitatory neuron of a wrong class becomes active, its incoming weights are modified according to the all-LTD rule.

During the test stage, the recognized class of data is determined by the highest activity of the neuron populations assigned to each class.

We used just 15000 images for training and 1500 images for testing from the MNIST dataset. In the experiments, we used only 3 classes of images out of 10 possible.

The SNN was implemented in the Brian 2.0 package, an open-source framework for SNN modeling. We used a computer with an Intel Core i9 processor (3,1 GHz), 32 GB RAM, PyTorch 1.8.0, and Ubuntu 20.0 to run the code.

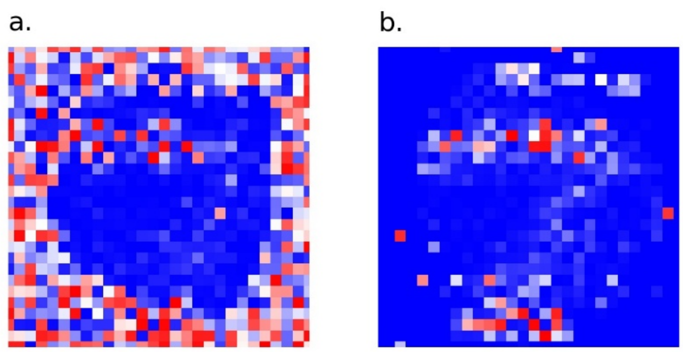

Fig. 2. 784 weights of the synaptic group S_1, associated with one of the excitatory neurons: a. in the model without regularization technique; b. the same weights in the model with regularization.

The result of training with and without weight regularization is illustrated in Fig.2. Fig. 2 shows the 784 weights of the synaptic group S_1 that are responsible for the recognition of one of the features of the digit 7. 784 synapses connect all Poisson neurons to one of the excitatory neurons. Fig.2a shows the weights without regularization technique, while Fig.2b shows weights with regularization. The effect of regularization is clearly visible: the nonzero weights around the border of Fig.2a corresponding to the silent neurons are pruned by regularization as can be seen in Fig. 2b.

To study the effects of regularization, the trained networks were tested on four different datasets with 1500 images each:

1. 28×28 pixel images with delta-correlated noise with an average intensity equal to the average intensity of the images in the training dataset;
2. 28×28 pixel noise images processed with a Gaussian filter with a standard deviation of 2 pixels (finite-correlation noise images);
3. images of the digits corresponding to the same classes the network was trained on;
4. images of the digits different from the classes the network was trained on.

The elimination of the insignificant synaptic weights that are not involved in recognition is expressed in a decrease in the overall neural activity of the SNN when it is tested on images of both ‘known’ (image classes coincide with those used in training) and ‘unknown’ digits (see Table).

Table

Neuronal output activity (total number of spikes) for different datasets

No	Test set	Model without regularization	Model with regularization
1	Delta-correlated noise	3162	1
2	Gaussian blurred images	3265	1
3	‘Known’ digits	416	344
4	‘Unknown’ digits	346	240

In the model without regularization, 'unknown' data generated 17% fewer spikes than 'known' data; in the model with regularization, the difference in spike number increased up to 30%. The trained network also shows a significant difference in neuronal spiking activity for noise images with and without regularization (see Table). According to the data in the Table, it is clearly seen that the SNN with regularization is almost completely unresponsive to the noise signal.

In conclusion, the use of weight regularization in SNNs eliminates the weights that are not involved in the recognition process. As a result, this allows:

- to significantly reduce the noise impact on the recognition process and thereby significantly reduce the risk of adversarial attacks on the SNN,
- to reduce neural activity in response to ‘unknown’ data (in experiments by 1,8 times), which increases the accuracy of the SNN and improves its ability to generalize,
- to modulate the propagation of known and unknown information that is vital for the implementation of predictive computations in SNNs.

Список литературы

1. Urolagin S. Generalization capability of artificial neural network incorporated with pruning method / S.Urolagin, K.V.Prema, N.V.S.Reddy // Advanced computing, networking and security. Lecture notes in computer science. Eds. P.S.Thilagam, A.R.Pais, K.Chandrasekaran, N.Balakrishnan. — Berlin, Heidelberg: Springer. — 2012. — V.7135. — P.171–178.
2. Montesinos López O.A. Overfitting, model tuning, and evaluation of prediction performance / O.A.Montesinos López, A.Montesinos López, J.Crossa // Multivariate statistical machine learning methods for genomic prediction. — Springer Cham. — 2022.
3. N'dri A.W. Predictive coding light: learning compact visual codes by combining excitatory and inhibitory spike timing-dependent plasticity / A.W.N'dri, T.Barbier, C.Teulière, J.Triesch // Proceedings of the IEEE/CVF Conference on computer vision and pattern recognition. – 2023. – P.3997-4006.
4. Maass W. Networks of spiking neurons: The third generation of neural network models / W.Maass // Neural Networks. – 1997. – V.10. – No.9. – P.1659–1671.
5. Sharmin S. A comprehensive analysis on adversarial robustness of spiking neural networks / S.Sharmin, P.Panda, S.S.Sarwar, C.Lee, W.Ponghiran, K.Roy // 2019 International joint conference on neural networks (IJCNN). Budapest. Hungary. — 2019. — P.1-8.
6. Tan Y.X.M. Exploring the back alleys: analysing the robustness of alternative neu-

ral network architectures against adversarial attacks / Y.X.M.Tan, Y.Elovici, A.Binder // E-print: https://www.arxiv.org/abs/1912.03609.

7. Marchisio A. Is spiking secure? A comparative study on the security vulnerabilities of spiking and deep neural networks / A.Marchisio, G.Nanfa, F.Khalid, M.A.Hanif, M.Martina, M.Shafique, // 2020 International joint conference on neural networks (IJCNN). – 2020. – P.1-8.
8. Leontev M. Robustness of spiking neural networks against adversarial attacks / M.Leontev, D.Antonov, S.Sukhov // 2021 International Conference on Information Technology and Nanotechnology (ITNT). – 2021. – P.1-6.
9. Antonov D. Spiking neural networks training with combined Hebbian rules / D.Antonov, B.Batuev, S.Sukhov // X International conference on information technology and nanotechnology (ITNT). — Samara. Russian Federation. — 2024. — P.1-5.
10. Antonov D.I. Training of spiking neural networks-classifiers with local rules / D.I.Antonov, S.V.Sukhov // Proceedings of XXV international conference on artificial neural networks «Neuroinformatics-2023». – 2023. – P.116-125. (in Russian)
11. Shors T.J. Use it or lose it: how neurogenesis keeps the brain fit for learning / T.J.Shors, M.L.Anderson, D.M.Curlik Ii, M.S.Nokia // Behavioural brain research. – 2012. – V.227(2). – P.450-458.
12. Diehl P. Unsupervised learning of digit recognition using spike-timing-dependent plasticity / P.Diehl, M.Cook // Frontiers in Computational Neuroscience. – 2015. – V.9. – P. 1-9.

Министерство науки и высшего образования РФ
Сибирское отделение РАН
Российская ассоциация нейроинформатики
Институт вычислительного моделирования СО РАН

НЕЙРОИНФОРМАТИКА, ЕЁ ПРИЛОЖЕНИЯ И АНАЛИЗ ДАННЫХ

МАТЕРИАЛЫ
ТРИДЦАТЬ ВТОРОГО ВСЕРОССИЙСКОГО СЕМИНАРА
27 сентября 2024 года

Красноярск 2024