Learning Algorithms for Spiking Neural Networks: Should One use Learning Algorithms from ANN/DL or Neurological Plausible Learning? - A Thought-Provoking Impulse

Martin Bogdan
Neuromorphic Information Processing
Leipzig University, Augustusplatz 10, 04109 Leipzig, Germany
bogdan@informatik.uni-leipzig.de

Abstract

Artificial Neural Networks (ANN) and Machine Learning (ML) currently also known as Deep Learning (DL) became more and more important in industrial applications during the last decade. This is due to new possibilities by strongly increased available computational power in connection with a renaissance of ANN in terms of so called Deep Learning (DL). As DL requires especially for Big Data extreme computational power, the question of resource preserving methods came recently into the focus. Also, the often propagated intelligence of DL resp. "Cognitive Computing" in terms of contextual information processing is more often discussed since it is effectively missed in DL solutions. One option to overcome both challenges might be the third generation of ANNs: Spiking Neural Networks (SNN). But since SNN training methods are slow compared to DL learning algorithms, the question of the way how to learn SNNs arose. We will discuss different aspects of learning algorithms for SNNs: Is it useful to adopt DL learning algorithms to SNN or not, especially if one will preserve the "cognitive" functions of SNNs?

Keywords: Spiking Neural Networks, Learning Algorithms, STDP, Hebbian Learning, Artificial Neural Networks, Deep Learning, Cognitive Computing

1 MOTIVATION AND PROBLEM STATEMENT

Currently, we can observe the 3rd heyday of Artificial Neural Networks (ANN) which leaded to the current hype of Artificial Intelligence, especially represented by the so called Deep Learning (DL). Following the first works from McCulloch and Pitts in 1943 [19] and the first decline initially due to the work of Perceptrons by Minsky and Papert in 1969 [20], several remarkable basic learning algorithms have been developed during the 70's and 80's (e.g. Hopfield nets [11], Self-Organised Feature Maps [15] or Neocognitron [3] to name just a few) before the second heyday started with the introduction of Backpropagation [24] end of the 80's and during the 90's. Remarkable for this period: Ivakhnenko developed the Group Handling of data Method (GMDH) [13] showing an option for the training of multilayer networks, nowadays known as Deep Learning. In this period falls as well the these days often used Long-Short-Term-Memory LSTM [9] for time-dependent data such as used in speech recognition - in my perspective one of the last really novel approaches within ANN.

Even though this period was very successful from a scientific point of view especially regarding the demonstration of different learning algorithms, the real break through towards industrial applications didn't occurred. The simple reason for that: the computational power was at that time not enough to achieve fast and reliable results in relative short training time. Thus the high promises already postulated in the 60's such as e.g. the singularity point (1961 one predicted, that in 10 or 15 years later, machines will be as intelligent as humans [23]) couldn't be fulfilled one more time. Despite several efforts to speed up computation for ANNs in the 90's like e.g. MA16 [25], KOBOLD [1] or CNAPS [4], the development of general purpose architecture overtook these specialised architectures simply by the number of human resources involved in the development. Thus, the computational speed for ANNs still depended on the speed of general purpose architecture resulting in another lean period for ANNs end of the 90's.

Starting during the middle of the first decade in the 21st century, a second renaissance of ANNs leading to DL was mainly driven by new computer architectures and high performance computing making it possible to learn in short and acceptable time frames even for deep structures with multiple layers and big data sets as well. Due to this technical advance, researcher remembered the roots of ANNs: the neurological system - and thus how the brain is built by neurons - as a paragon. The main idea is, that every problem can be solved if one uses enough neurons in order to mimic the abilities of at least parts of the brain to some ex-

https://doi.org/10.17979/spudc.9788497498418.0201

tent. Consequently, a huge number of artificial neurons in several hidden layers were introduced and combined with advanced learning algorithms and it's variations leading to DL in order to mimic such brain structures. As the spiral of advances in both areas - computer architecture and big data processing - are still ongoing and promoted by big companies, we are nowadays in the third heyday of ANNs resp. DL or more subsumed as Artificial Intelligence (AI).

The last years have shown, that DL based approaches are extremely successful for solving industrial problems especially for big data applications as long as the learning data show the problems well enough. This is due to the fact, that we use in general a parameter based training what leads at the end to a system, that fits it's weights on the statistical distributions of the presented data very well. Thus, this approach will fail, if abilities of brains such as e.g. contextual information processing, curiosity or cognition come into the play to solve problems. He et al. stated that classical ANNs and DL performing very well in several applications fields, such as e.g. image processing or speech recognition, but they fail in general when it is about contextual information [7]. Same can be observed as well in other domains like control or disease prediction. In my opinion, classical ANN/DL are unable to compete with human decision findings based on experience always in the moment we need kind of human cognition in its most literally sense.

In order to tackle that challenge, the so called third generation of artificial neural networks are an auspicious option: Spiking Neural Networks (SNN) [17]. But, compared to DL, SNN have a severe drawback: Neurological plausible learning is in comparison extremely slow, even though they are comparable or even outperform DL at least once they are trained correctly. In the following sections, after a short introduction to SNN, the pros and cons of SNNs are discussed also in the light of its originally application of simulating the neural system to gain new insights versus its now upcoming industrials applications especially from the point of view of learning algorithms. The paper will close with a discussion in the sense to give an thought-provoking impulse towards the philosophy using training algorithms for SNN.

2 SPIKING NEURAL NETWORKS

In contrast to ANNs, which are based on mathematical abstract and relative simple models of biological neurons, Spiking Neural Networks were originally developed in order to better understand brain mechanisms and simulate neural tissue resp. populations of neurons in the brain. A very good overview of the basic related papers and ideas also discussing the first and second generations of ANNs is given in [17]. In order to simulate the neurological conditions and circumstances, different approaches from highly precise and complex ones towards more simplified ones have been developed. The challenge for these models are to find the balance between precise models to understand the mechanism down to the ionic currents in cells and computational easier models still incorporating detailed functions while not loosing the originary properties of neuromorphic information processing.

Probably one of the most known models for a precise reproduction down to ionic currents in the cell is the Hodgkin-Huxley-model [10]. The idea is to divide the cell membrane into compartments simulating the properties of ion currents via an electrical equivalent circuit whereas the behaviour of the current for each ion are described via resistors and capacitors individually for each compartment. The equivalent circuit can be described by differential equations. Consequently, by combing the different compartments a complete cell can be reconstructed and thus simulated. Obviously, this approach is a very complex one and consumes a lot of computational power.

In order to overcome the problem of complexity and required computational power Izhekevich developed another commonly used approach [12]. He simplified the Hodgkin-Huxley-model towards two differential equations representing one neuron. Herein, he introduced 4 variables which can be tuned in the sense of simulating every type of biological neurons.

Apart from the above described approaches, the probably most prominent model used for SNNs in a more common sense apart from pure simulation of the brain tissues is the Leaky-Integrate-and-Fire neuron (LIaF) [6], which will be introduced in the following subsection.

2.1 BASICS OF LEAKY-INTEGRATE-AND-FIRE NEURONS FOR SNN

The LIaF is depicted as well by an electrical equivalent circuit representing the ensemble of a synapse and the soma of a neuron as is shown in figure 1. Via the synapse(s) input current(s) representing an action potential from different preceding neurons is incorporated tot he model. As the neurological information processing systems don't know numbers, the corresponding action potentials depending on their firing points in time

 https://doi.org/10.17979/spudc.9788497498418.0201

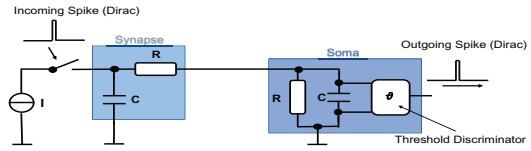

Figure 1: Electrical equivalent circuit of an Leaky-Integrate-and-Fire Neuron. Adopted from [6]

are represented by Diracs. Indeed, the information in biological neural networks is coded via the occurrence of action potentials in the course of time - what completely differ from the method we use in current computer architecture. Thus, SNNs incorporate time dependence intrinsic.

In figure 1 on the left hand side, the equivalent circuit of a synapse is shown. It represents a simple low pass filter in order to transform the Dirac into a time-smoothed signal more similar to an incoming action potential. The soma on the right hand side of figure 1 is depicted by a parallel circuit of a resistance and a capacitor. Every time a spike arrives via the synapse, the capacitor is charged - the integrative part of the model. The voltage of the capacitor is analogue to the membrane potential of a neuron. Once the voltage exceed a certain threshold, the discriminator emits a spike and the voltage of the capacitor is reset to zero. The charge of the capacitor is discharged all the time via the resistance in parallel - the leaky part of the model. The leaky part is the analogue for the sodium-potassium pump responsible to discharge the biological neuron towards its resting membrane potential. This simple circuit can be easily described with differential equations, and by changing the values of R and C individual different types of neurons can be - roughly spoken - simulated. For its simplicity the LIaF is probably the most prominent model for SNNs at the moment. In addition, the model can easily be extended e.g. for more realistic synaptic models like the Modified Stochastic Synaptic Model (MSSM) [2].

2.2 ADVANTAGES AND DRAWBACKS

In comparison with ANNs there are several advantages, but also drawbacks for SNNs. Before disclosing these arguments, please remark an important information: Regarding the performance resp. accuracy SNNs are not inferior to ANNs but often even better. In [26] SNNs have shown comparable results to ANN and ML for the classification of the Iris data set. Same has been shown for the MNIST data set in comparison with a CNN [16], whereas the letters have been recorded with a retinomorphic data set [22] for the SNN. Thus, from this point of view, both approaches are on an equal footing.

Nevertheless, several advantages of SNNs can be listed. First of all, due to the intrinsic information representation of SNNs, the communication between the neurons are easier to realise in hardware: SNNs are requiring only one bit for the communication instead of 64 or higher bit numbers for current data representations via modern data bus systems. Thus, in principle, the connection structure that can be formed might be much more complex for SNNs than for ANNs. As well, since natural systems are built for energy efficiency, power consumption of SNN is intrinsic lower. Consequently, on the equal chip surface more and complexer networks might be implemented compared to ANNs. Most important, but still not finally proofed is the hypothesis, that SNNs will perform better in cases where data contains contextual information or if cognitive abilities are needed to solve a problem.

Despite all the named advantages, SNNs have some drawback as well: Training algorithms for SNNs are less effective and less elaborated in comparison to ANNs. In addition, real neuromorphic hardware to speed up training algorithms for SNNs taking into account the named advantages are still rare [21]. Moreover, since the coding of data in industrial applications is due to missing neuromorphic sensors not appropriate for the information processing coded via spike trains, the recorded data have to be converted first into an according spike train representation for further processing with SNNs - and the outcome as well back to a numerous representation.

3 LEARNING FOR SPIKING NEURAL NETWORKS

Although the problem of hardware to speed up SNN is an important issue also related to training [21] in general, this paper focuses on the issue of training algorithms for SNNs and not their hardware implementation. For this, current learning strategies will be shown in the next subsections. In general, two trends can be observed: Adopting well known training algorithms from ANNs to SNNs or the use of neurological inspired learning algorithms. For both options, the problem of transforming the data into a SNN-conform data representation will be neglected at this point and assumed as being solved in an applicable way.

3.1 NEUROLOGICAL PLAUSIBLE LEARNING ALGORITHMS

A classical training method for ANNs was originally derived from Neurology and thus is also used

https://doi.org/10.17979/spudc.9788497498418.0201

for SNNs: the Hebbian Learning rule [8], based on the idea of correlated learning by James [14] in 1890, also known as *what fire together, wire together*. That means, when the pre-synaptic neuron is firing in correlation with the connected postsynaptic neuron, the connection between the two neurons is strengthen. But, following the Hebbian Learning rule would mean, that connections will only be strengthen, what is contradictory to nature and to learning in general. Learning means also the ability to weaken connections. Therefore, the rate-based Hebbian Learning rule has been developed [28, 6]. Herein the weakening of the synaptic connection is not only implemented in the moment of learning, but also while not using a synapse over a longer time period. By applying this approach not only SNNs but also ANNs can be trained - but not as efficient in terms of training time as known for training algorithms from ANN, since the convergence towards the optimal efficient state of the synapses (the paragon in ANNs would be the weights) evolves in relation to ANN training algorithms quite slow.

Looking on Hebbian Learning, the correlation in time of Diracs from pre- to post-synaptic neurons is not really included neither in the sense of strengthen or weakening the connections between both neurons. Thus, to introduced this neurological effect as well, Spike-Timing-Dependent Plasticity (STDP) [18, 6] has been introduced. Herein the change of the transmission efficiency between the neurons via the connecting synapse is modulated in form of an exponential functions depending on the time difference of the occurrence of spike in the pre- and postsynaptic neuron as shown in figure 2. More specific, the shorter the

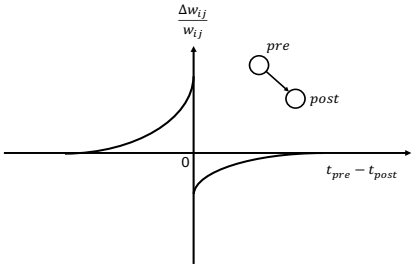

Figure 2: Spike-Timing-Dependent Plasticity STDP. The transmission efficiency between two neurons depends on the correlated spike occurrence of connected neurons pre and post.

time slot between pre- and postsynaptic spike, the stronger is the change of transmission efficiency in terms of strengthen it until the exact point of synchronicity is reached resulting in a maximum positive change. Exactly after that maximum point of strengthening, the effect is inverted completely. Once, the postsynaptic spiking is before the presynaptic spiking, efficiency of transmission is weakened, starting from a maximum weakening toward zero, again via an exponential function.

Both approaches, rate-based Hebbian Learning and Spike-Timing-Dependent Plasticity are often combined for the training of SNNs, since both approaches are directly derived from neurological paragons. For this, these training algorithms performing a learning method very close to its neurological paragon - but still lack efficiency from an engineering point of view, since learning is relatively slow.

3.2 ADOPTED LEARNING ALGORITHMS FROM ANN

Another trend to train SNNs is kind of two-folded: the use of well known training algorithms from ANN/DL transferred to SNNs. Herein, two options can be observed. The first option is quite obvious: since the sense of training networks is to find the optimal combination of weights for certain network architecture to solve the problem in question. Once, the architecture is found with an optimal weight combination using a classical ANN/DL learning algorithm, the same architecture and weights can be assigned to a SNN. Actually, that way, the SNN is not involved in the training at all.

The second and more complicated way is to transform the ANN/DL training algorithms directly to train SNNs. Hereby, two algorithms are currently in the focus: Backpropagation (BP) [24] and Reinforcement Learning (RL) [27]. Backpropagation including its variations is probably the most used training algorithm for ANNs/DL. But it uses a different paradigm for learning as nature don't use exact during target values and corresponding errors during learning. Since nature never knows the perfect target value in an engineering sense while training it is slowly approaching the learning target by experience, kind of try-and-error approach without teaching inputs.

More specific, regarding the transformation of Backpropagation towards training of SNNs, there are two severe problems to overcome. First of all, SNNs working with Diracs, a non steady differentiable function - what is needed for Backpropagation. To solve this problem, one can do a workaround as in [16], In order to obtain a steady differentiable function, the authors are using the virtual potential of the LIaf neurons. The problem of the reset of the virtual membrane potential to zero in the moment of spiking is avoided by filtering the virtual mebrane potential using a low pass filter, thus eliminating the drop due to the occu-

 https://doi.org/10.17979/spudc.9788497498418.0201

rance of a spike. This way, the virtual membrane potential becomes a steady differentiable function and can be used for Backpropagation.

The second problem with Backpropagation for SNN is a neurological one: Biological neural networks - and thus SNNs - do not possess a mean to propagate the error back through the neurons. To overcome this fact, adapted Backpropagation algorithms introduce a kind of virtual path to propagate the error back, thus the weights can be adopted accordingly the Backpropagation paradigm [16].

In contrast to Backpropagation the learning paradigm of Reinforcement Learning is quite close to the natural learning paradigm from the neurological point of view. RL learns due to rewards, thus trying to maximise the reward obtained by doing learning steps in regard of achieving a long term goal. That way, in a first glance, again, the weights must be adjusted in a natural way. To overcame an artificial adjustment of the weights, RL is often combined with the rate-based Hebbian rule in combination with STDP. In this combination, this approach is probably currently the closest training algorithm compared with the natural learning paradigm.

3.3 NETWORK ARCHITECTURES FOR ANN AND SNN

One point we didn't consider yet is the architecture of networks used in ANN/DL and SNN. Basically, two typically network architectures are used as shown in figure 3. Most ANN/DL architectures

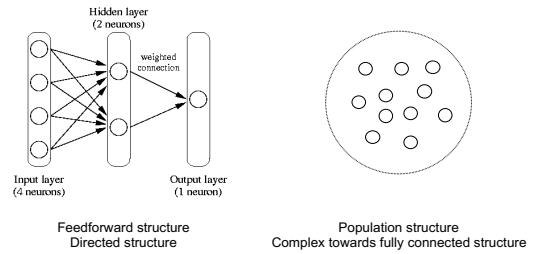

Figure 3: Connectivity structures. Left: typical feedforward structure as can be found in the peripheral nervous system or the optic nerve. Right: Population structure as can be found in the brain.

are based on feedforward structures including its variations such as e.g. feedback between layers. These structures can be observed in parts of the natural paragon as well, e.g. in the peripheral nervous system for the control of the musculoskeletal system or in the brain for the optic nerve from retina until the visual cortex.

Nevertheless, every time, the processing of contextual informations or even higher cognition comes into the play, population coding is much more likely and has been shown e.g. for the emergence of movement sequences in the brain [5]. The connections between the neurons of a population can be complex including recurrent up to fully connected structures. But, the more complex a network architecture becomes in the sense of feedback loops up to fully connected structures, the less efficient ANN/DL learning algorithms are.

4 DISCUSSION

After presenting the currently most used learning algorithms for SNNs with its main advantages and drawbacks, the fundamental question can be addressed: Should one use ANN/DL algorithms for training SNNs because of their higher efficiency or neurological plausible once despite their lack of training time?

In a first glance, from the engineering point of view, the answer might be easy: As long as we take the efficiency in terms of training time combined with the resulting accuracy into account, the transformation of well known and elaborated ANN/DL raining algorithms towards SNN training is favourable. But why one would do so, if the ANN/DL already used for industrial applications aren't inferior to SNNs? The only argument from my point of view would be the energy consumption argument, but up to now, we don't have appropriate hardware architecture to exploit this effect.

Moreover, the question already raised in the motivation section regarding contextual or even cognitive data processing is questionable for ANN/DL training algorithms due to its connective architectures of the used networks. Remember, that the brain network for cognition and emergence of will are most probably organised as populations. Even the whole brain is structured in a way, that you always will find a path from one neuron to another, even though the path will run over thousand or millions of other neurons. Thus, one should use SNNs to mimic this feature for applications. But in this moment, it's questionable, if ANN/ML learning algorithms are still able to train these extremely complex structures in an appropriate way. Unfortunately, up to now, we don't have the computational means to proof that hypotheses.

One should nit forget another important argument. SNNs are not designed from its roots to be applied in industrial applications, but to investigate the neuromorphic information processing in terms of a better understanding of the ongoing information processes in the brain and to simulate

them in order to proof their correctness. In this context, it is out of question, that we need to use a neurological plausible training algorithm even if this will require an enormous amount of computational power, otherwise the understanding of the neuromorphic information processing will falsified due to a non-appropriate learning method.

Finally, I would like to postulate the following to start a disputation: As long as there is no fast hardware adapted to the neurological model for training SNNs, I see no advantage of SNNs over ANNs/DL for industrial applications without contextual reference or cognition. However, if we look at applications from a research perspective in terms of understanding neuromorphic information processing in our neural networks and especially in the brain, which could also lead to understanding our cognition, neurologically plausible training methods for SNNs are indispensable. Subsequently, one could think about implementing cognitive decisions in non-industrial applications as well - if this is ethically justifiable. Let's start the disputation from this postulate!

Acknowledgement

This work was partly supported by the German Federal Ministry of Education and Research (BMBF, 01IS18026B) by funding the competence center for Big Data and AI "ScaDS.AI Dresden/Leipzig" and the graduate school "School of Embedded Composite Artificial Intelligence (SECAI)" (BMBF via DAAD, 57616814).

References

[1] Bogdan M, Speckmann H, Rosenstiel W (1994) Kobold: a neural coprocessor for backpropagation with online learning, *Proceedings of the Fourth International Conference on Microelectronics for Neural Networks and Fuzzy Systems*, DOI: 10.1109/ICMNN.1994.593222

[2] Ellaithy K, Bogdan M (2017) Enhancements on the Modified Stochastic Synaptic Model: The Functional Heterogeneity, *Int. Conf. on Artificial Neural Networks and Machine Learning ICANN*, Lecture Notes in Computer Science, Vol 10613

[3] Fukushma K, iyake S, Ito T (1983) Neocognitron: A neural network model for a mechanism of visual pattern recognition, *IEEE Transactions on Systems, Man, and Cybernetics*, Vol SMC-13:5, pp 826-834, DOI: 10.1109/TSMC.1983.6313076

[4] Granado B, Garda P (1997) Evaluation of the CNAPS neuro-computer for the simulation of MLPS with receptive fields, *Biological and Artificial Computation: From Neuroscience to Technology* IWANN 1997, Lecture Notes in Computer Science, vol 1240. Springer, doi.org/10.1007/BFb0032541

[5] Georgopoulos A P, Schwartz A B, Kettner R E (1986) Neuronal population coding of movement direction, *Science* 233(4771):1416-9, doi: 10.1126/science.3749885.

PMID: 3749885 DOI: 10.1126/science.3749885

[6] Gerstner W, and Kistler W M (2002) Spiking Neuron Models, Cambridge University Press

[7] He H, Shang Y, Yang X, Di Y, Lin J,Zhu Y, Zheng W, Zhao J, Ji M, Dong L, Deng N, Lei Y and Chai Z (2019) Constructing an Associative Memory System Using Spiking Neural Network, *Front. Neurosci. 13:650*, doi: 10.3389/fnins.2019.00650

[8] Hebb D O (1949) The Organization of Behavior - A Neuropsychological Theory *Psychology Press New York*, Reprint from 2002, https://doi.org/10.4324/9781410612403

[9] Hochreiter S, Schmidhuber J (1997) Long Short-Term Memory, *Neural Computation* 9(8):1735-1780 DOI:10.1162/neco.1997.9.8.1735

[10] Hodgkin A L, Huxley A F (1952) A Quantitative Description of Membrane Current and Its Application to Conduction and Excitation in Nerve, *J Physiol.* 117(4):500-44, DOI: 10.1113/jphysiol.1952.sp004764

[11] Hopfield J J (1982) Neural networks and physical systems with emergent collective computational abilities, *Proc Natl Acad Sci USA*, 79(8):2554-8, doi: 10.1073/pnas.79.8.2554

[12] Izhikevich E M (2003) Simple Model of Spiking Neurons, *IEEE TRANSACTIONS ON NEURAL NETWORKS*, VOL. 14, NO. 6

[13] Ivakhnenko A G, Ivankhnenko G A (1995) The Review of Problems Solvable by Algorithms of the Group method of Data Handling (GMDH), *Pattern Recognition and Image Analysis*, Vol. 5, No. 4, pp 527-535

[14] James, W. (1890) *Psychology (Briefer Course)*, ch. 16. Holt, New York, 362, 393

[15] Kohonen T (1982) Self-organized formation of topologically correct feature maps, *Biological Cybernetics*, Vol. 43, pp 59-69

 https://doi.org/10.17979/spudc.9788497498418.0201

[16] Lee J H, Delbruck T, Pfeiffer M (2016) Training Deep Spiking Neural Networks Using Backpropagation, *Front. Neuroscience* 10:508, doi: 10.3389/fnins.2016.00508

[17] Maass W (1997) Networks of spiking neurons: The third generation of neural network models, *Neural Networks* 10 (9), pp 1659–1671

[18] Markram H, Lübke J, Frotscher M, Sakmann B (1997) Regulation of synaptic efficacy by coincidence of postsynaptic APs and EPSPs, *Science* 275 (5297): 213–5, doi:10.1126/science.275.5297.213.

[19] McCulloch W, Pitts W (1943) A Logical Calculus of the Ideas Immanent in Nervous Activity, *Bulletin of Mathematical Biology*, Vol 5, pp 115-133, cited from reprint Vol 52, No 1/2, pp 99-115, 1990

[20] Minsky M, Pappert S (1969) Perceptrons, ISBN: 9780262630221 258 pp., MIT Press

[21] Pfeiffer M, Pfeil T (2018) Deep Learning With Spiking Neurons: Opportunities and Challenges. *Front. Neuroscience* 12:774, doi: 10.3389/fnins.2018.00774

[22] Posch Ch, Serrano-Gotarredona T, Linares-Barranco B, and Delbruck T (2014) Retinomorphic Event-Based Vision Sensors: Bioinspired Cameras With Spiking Output, *Proceedings of the IEEE* Vol. 102, No. 10, pp. 1470-1484, doi: 10.1109/JPROC.2014.2346153

[23] MIT Centennial (1961) The Thinking Machine, film available on youtube

[24] Rumelhart D, Hinton G E, Williams R J (1986) Learning internal representations by error propagation, *Biology*, DOI:10.1016/B978-1-4832-1446-7.50035-2

[25] Ramacher U, Beichter J, Brüls N (1993) A general-purpose signal processor architecture for neurocomputing and preprocessing applications, *Journal of VLSI signal processing systems for signal, image and video technology* Vol. 6, pp 45–56

[26] Sboeva A, Vlasova D, Rybka R, Serenko A (2018) Spiking neural network reinforcement learning method based ontemporal coding and STDP, *Procedia Computer Science* Vol. 145, pp 458-463

[27] Sutton R S, Barto A G (2015) Reinforcement Learning: An Introduction, *The MIT Press*

[28] Taylor MM (1973) The Problem of Stimulus Structure in the Behavioural Theory of Perception, *South African Journal of Psychology*, Vol 3, pp 23–45