9th International IEEE EMBS Conference on Neural Engineering
San Francisco, CA, USA, March 20 - 23 , 2019

Reservoir Transfer on Analog Neuromorphic Hardware

Xu He, Tianlin Liu, Fatemeh Hadaeghi, Herbert Jaeger

***Abstract*— Analog, unclocked, spiking neuromorphic microchips open new perspectives for implantable or wearable biosensors and biocontrollers, due to their low energy consumption and heat dissipation. However, the challenges from a computational point of view are formidable. Here we outline our solutions to realize the reservoir computing paradigm on such hardware and address the combined problems of low bit resolution, device mismatch, approximate neuron models, and timescale mismatch. The main contribution is a computational scheme, called Reservoir Transfer, which enables us to transfer the dynamical properties of a well-performing neural network which has been optimized on a digital computer, onto neuromorphic hardware that displays the abovementioned problematic properties. Here we present a case study of implementing an ECG heartbeat abnormality detector to showcase the proposed method.**

I. Introduction

Implantable or wearable biosensors, signal generators, controllers or bidirectional brain-machine interfacing modules are widely investigated. For such devices, a very small energy consumption is always desirable and sometimes mandatory – for instance, in brain implants that must not dissipate noticeable heat [1]. A promising route toward such devices is to develop analog, unclocked, spiking neuromorphic microchips implementing artificial neural networks (ANNs). These ANNs obtain their functionality by being *trained* rather than being *programmed*, using methods from machine learning. With regards to the latter we perceive two main trends. First, adapt *deep learning* methods [2] to spiking neural networks (for instance [3]). Second, transfer computational mechanisms known from biological brains to neuromorphic hardware [4]. Here we follow the second route and employ principles of *reservoir computing* to realize a heartbeat abnormality detector on an analog spiking neuromorphic device.

Reservoir computing (RC) is a learning paradigm in recurrent neural networks (RNNs) for processing temporal input signals. The core RC principles have been discovered several times independently, both in AI/machine learning and in computational/cognitive neuroscience (brief history in [5]), indicative of the nature of RC as bridging between machine learning and neuroscience. Within computational neuroscience, RC is best known as *liquid state machines* (LSMs) [6], whereas the approach is known as *echo state networks* (ESNs) [7] in machine learning. Reflecting the different objectives in these fields, LSM models are typically built around more or less detailed, spiking neuron models with biologically plausible parametrizations, while ESNs mostly use highly abstracted rate models for its neurons.

An RC architecture is composed of three major parts: the *input layer* feeds the input signal into a random, large, fixed recurrent neural network that constitutes the *reservoir*, from which the neurons in the *output layer* read out a desired output signal. In traditional (and "deep") RNN training methods, all synaptic weights in a neural learning architecture are optimized by descent on the output error gradient. In contrast, the input-to-reservoir and the recurrent reservoir-to-reservoir weights in an RC system are left unchanged after a random initialization, and only the reservoir-to-output weights are optimized during training – typically by minimizing the mean square error between the teacher signal and the network's output signal through linear regression.

In the work reported here, we sought to create a functional spiking reservoir on an analog, unclocked, spiking neuromorphic microprocessor board. The system's task is to detect and classify heartbeat abnormalities from electrocardiographic (ECG) input signals. The system should function in an online processing mode in real-time. Our hardware is the Dynap-se board [8]. This work constituted a work package in the European Horizon 2020 project NeuRAM3 (`http://neuram3.vsos.ethz.ch/`), which is concerned with the design and fabrication of memristor-based, spiking neuromorphic microchips.

II. The Challenges

Our project encountered interesting difficulties that arose from its dual nature of being half engineering/machine learning, half bio-/neuroscience:

Machine learning: We want to achieve numerically accurate and robust information processing performance, goals which are characteristic for machine learning rather than for computational neuroscience. This would suggest using the machine learning version of RC, namely ESNs. Indeed we found that the addressed processing task could be solved by the standard ESN methods with high performance. However, standard ESN methods rely on floating-point precision and reproducible arithmetics, and ESN neuron models are non-spiking and thus entirely inappropriate to capture Dynap-se neurons.

Bio-/neuroscience: The physical neurons on the Dynap-se board share core properties with biological neurons. They are spiking, unclocked, individually different from each other ("device mismatch"), (slightly) noisy, and must be (approximately) modeled by rather intricate, multi-parameter ODEs

The work reported in this article was funded through the European Horizon 2020 project NeuRAM3 (grant 687299).
X. He, T. Liu, F. Hadaeghi, and H. Jaeger are with the Department of Computer Science and Electrical Engineering, Jacobs University Bremen, 28759, Bremen, Germany (email: [x.he, t.liu, f.hadaeghi, h.jaeger]@jacobs-university.de).

with several time constants. Furthermore, the input ECG signal is of biological origin and must be processed with the appropriate biological timescales. All of this would indicate to use LSM models. The LSM literature offers numerous successful accounts of modeling biological neural systems. However, these reports did not provide useful guidance for us because first, they rest on approximate copies of biological systems whose neuron models and parameters (especially time constants) are incommensurate with our physical device, and second, LSM literature reports are typically proof of concept studies whose numerical accuracy or robustness would not match our needs.

Additional challenges result from the circumstance that many physical variables (currents and potentials) determining the behavior of Dynap-se neurons are not measurable – a condition familiar to neuroscientists but alien to machine learning programmers.

In summary, we were suspended between two well-proven computational paradigms of which we would want to use the first for accuracy and robustness, the other for matching our hardware, but none of the two could easily be accommodated.

The solution came from an unexpected side. In our initial investigations, we found ourselves confronted with yet another problem which we did not anticipate: a mismatch of timescales. Our Dynap-se hardware was far too fast for processing our slow ECG signals in a time-coupled online fashion. Classifying a heartbeat as normal vs. abnormal requires to integrate information over a timespan in the order of 1 second. Even when the few programmable time constants on the Dynap-se board were set to the slowest possible values, the Dynap-se dynamics was still inherently too fast to realize the requisite real-time memory spans.

At this point, we recall some facts about short-term memory in RNNs. There are two complementary mechanisms by which information is preserved through time in an RNN (or, in fact, in any input-driven dynamical system modeled by ODEs):

- The model's timescale can be set by the time constants of the governing equations. With large (slow) time constants, the effects of an input signal $\mathbf{u}$ will only slowly "wash out" over time. However, large time constants amount to smoothing the effects of the input signal on the current system state. In our concrete case, if we would induce slowness solely by slow time constants, any characteristic of the input signal $\mathbf{u}(t_0)$ which is relevant for the classification at time t_0+1 (sec) would become overshadowed by almost equally preserved but distracting information from earlier input times.
- The second mechanism is inherent in the nonlinear geometry of dynamical systems. In its core, it is captured by extensions of Takens' theorem [9], [10]. Roughly speaking, in any dynamical system (continuous or discrete time) some portion of the preceding input and internal state history is nonlinearly encoded in the current system state, such that parts of the previous input history can be recovered from the current state. This effect has been investigated in depth in the RNN literature in general and the ESN literature in particular (for instance [11], [12]). This dynamical short-term memory capacity is unfortunately quite sensitive to system noise.

After setting the accessible hardware time constants on the Dynap-se board to their slowest values, we capitalized on the second mechanism to bridge the remaining memory gap into the required 1 second time range. To this end we first optimized the Takens-type dynamic memory characteristics of a standard ESN with leaky integrator neurons using on a digital computer, employing routine methods from the ESN field. The optimized dynamics of this reservoir was then "mirrored" in a hardware reservoir on the Dynap-se board with the same number of neurons, in a novel, local, neuron-by-neuron training scheme that relied on linear regression. In this hardware training process, the reservoir-to-reservoir synaptic weights on the Dynap-se board were determined in a way that made the hardware reservoir inherits the (slow, input-history preserving) dynamical characteristics of the optimized source ESN. By using a regularized version of linear regression, the noise robustness of the hardware reservoir was optimized as a side-effect.

In summary, solving the slow timescale problem resulted in a *Reservoir Transfer Method* which effectively built a bridge from the engineering/machine learning world of ESNs to the bio/neuroscience world of LSM-like systems. We gave a preliminary account of this method in [20]. We proceed by providing the technical details.

III. Reservoir Transfer Method

Given an ESN with leaky integrator neurons driven by some m-dimensional input signal $\mathbf{u}(t)$

$$\dot{\mathbf{x}}(t) = -\lambda_x \mathbf{x}(t) + \tanh(W\mathbf{x}(t) + W^{\text{in}}\mathbf{u}(t)), \qquad (1)$$

where $\mathbf{x}(t)$ is the n-dimensional state vector of the network at time t, λ_x the leaking rate, W^{in} and W are input and recurrent weights. $\mathbf{x}(t)$ can be considered as high-dimensional temporal features of $\mathbf{u}(t)$, which we would like to transfer to a reservoir of leaky integrate-and-fire (LIF) neurons, whose dynamics is approximately modeled by:

$$\dot{\mathbf{v}}(t) = -\lambda_v \mathbf{v}(t) - \theta \mathbf{s}(t) + \hat{W}\mathbf{r}(t) + W^{\text{in}}\mathbf{u}(t) + I, \qquad (2)$$

$$\dot{\mathbf{r}}(t) = -\lambda_s \mathbf{r}(t) + \mathbf{s}(t), \qquad (3)$$

where $\mathbf{v}, \mathbf{s}, \mathbf{r}$ are all n-dimensional vectors whose i-th entries are denoted as v_i, s_i and r_i, respectively. v_i is the membrane potential of the i-th neuron, and $s_i(t) = \sum_{t_s} \delta(t - t_s)$ is its output spike train with spike times t_s and Dirac delta function δ, θ is the difference between spiking threshold and reset potential; r_i is the exponentially decaying synaptic currents triggered by s_i. I is a constant current set near or at the rheobase (threshold to spiking) value, as used in [26]. To transfer the temporal features $\mathbf{x}(t)$, we inject the ESN feature signals $x(t)$ element-wise into the corresponding LIF neurons, replacing the recurrent inputs $\hat{W}r(t)$. This results

in the dynamics:

$$\dot{\mathbf{v}}_x(t) = -\lambda_v \mathbf{v}_x(t) - \theta \mathbf{s}_x(t) + \mathbf{x}(t) + W^{\text{in}}\mathbf{u}(t) + I \quad (4)$$

$$\dot{\mathbf{r}}_x(t) = -\lambda_s \mathbf{r}_x(t) + \mathbf{s}_x(t) \quad (5)$$

Ideally, we would like the same dynamics $\mathbf{v}_x(t)$ to be sustained by the recurrent input $\hat{W}\mathbf{r}(t)$ instead of the teacher signal $\mathbf{x}(t)$. In other words, if we consider $\mathbf{y}(t) := \hat{W}\mathbf{r}_x(t)$ as the state vector of the spiking reservoir, we aim to learn the weight matrix $\hat{W}$ such that $\mathbf{y}(t) = \mathbf{x}(t)$. In order to make the learned weights generic and input-independent, we choose $\mathbf{u}(t)$ to be a white noise signal for training, and compute the weights by linear regression to minimize $\sum_{t_k} ||\hat{W}\mathbf{r}_x(t_k) - \mathbf{x}(t_k)||_2^2$, where t_k are discrete time samples. In this way, we have turned the weight initialization problem into a linear regression problem.

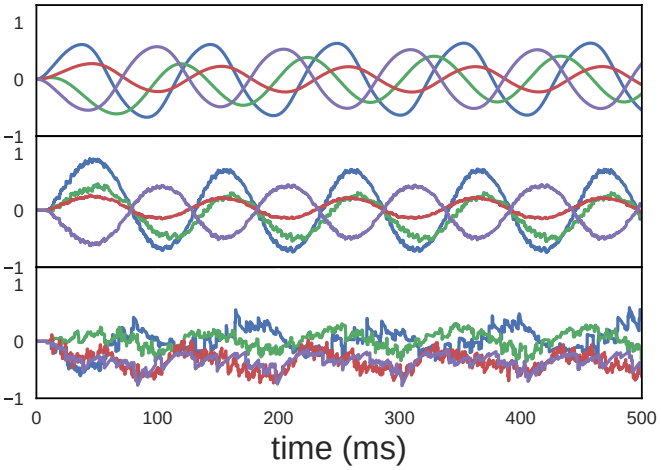

Fig. 1. Dynamics of different reservoirs in response to a sinusoidal input. For visual simplicity, only the first four dimensions of the state vectors are displayed. **Top**: $\mathbf{x}(t)$ in a leaky ESN teacher reservoir. **Middle**: corresponding $\mathbf{y}(t)$ in the target LIF reservoir created using the reservoir transfer method. **Bottom**: corresponding $\tilde{\mathbf{y}}(t)$ in a reservoir whose recurrent weights are from a sparse matrix with randomly distributed values (for explanation, see text).

Figure 1 shows the dynamics $\mathbf{y}(t) = \hat{W}\mathbf{r}(t)$ of a reservoir of 100 LIF neurons created in Brian simulator [27] using this method and the dynamics $\mathbf{x}(t)$ of its teacher leaky ESN reservoir of the same size when they are driven by a sinusoidal input. Only the first four dimensions of $\mathbf{x}(t)$ and $\mathbf{y}(t)$ are displayed for visual simplicity. One can see that although the reservoir was trained using white noise input, its similarity to the teacher reservoir can generalize to other types of input signals (a sinusoidal wave in this case). For comparison, we also created a LIF reservoir whose recurrent weights $\tilde{W}$ are from a sparse matrix with values first randomly sampled from a standard normal distribution, then scaled by a constant so that the recurrent dynamics do not evoke constant bursts of spikes. The first four dimensions of its response $\tilde{\mathbf{y}}(t) = \tilde{W}\mathbf{r}(t)$ to the same sinusoidal input is shown in the bottom panel of Figure 1.

IV. Experiments

We empirically evaluate the transferred reservoirs by experiments with both software simulation and hardware experiments.

A. Short-term Memory of Transferred Reservoir

In order to validate that the reservoir generated from the transfer learning method has short-term memory despite the connections are of low bit-precision and there is no synaptic short-term plasticity, we created such a reservoir in Brian simulator [27]. Instead of using the standard ridge regression during transfer learning, we used ternarized linear regression [13] so that the resulting weights are of ternary precision: $-a, 0, a$. To test its short-term memory, a sequence of pulses with very short pulse widths (10 ms) separated by long (200 ms) periods of silence was used as input to drive the ternary reservoir. A linear full precision readout was then trained to map the filtered spikes to a reverse-chirp signal. The time constant of the exponential decay kernel used to filter spike trains is only 15 ms. Since the output depends on the past input values, it can only be possible if the reservoir preserves some information about the input history. Figure 2 shows the step signal, the output of the reservoir and its target during the testing phase.

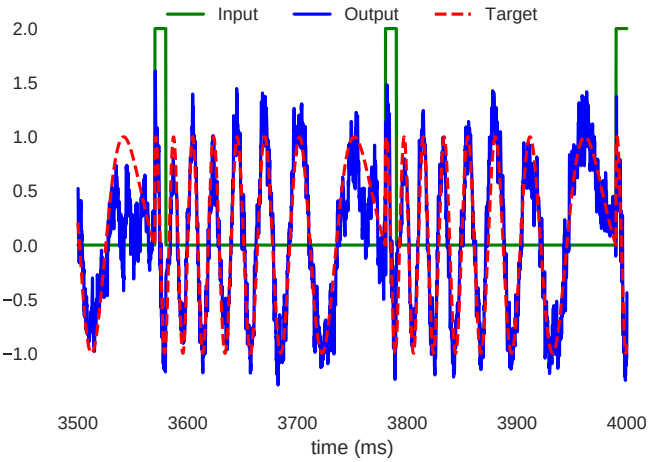

Fig. 2. A reverse-chirp signal can be generated by a reservoir of LIF neurons with ternary weights when it is driven by a step signal with very short (10 ms) high signal followed by a long (200 ms) silence.

B. ECG Monitoring using Dynap-se

In this experiment, we exploited our transferred spiking reservoir in a patient-customized electrocardiogram (ECG) beat classification task realized on Dynap-se, an unclocked analog spiking neural network hardware [8]. On a Dynap-se board, each of the four neuro-chips contains four cores with 256 neurons each. The behavior of one neuron is characterized by parameters such as injection current level, refractory period length, time constants, and synaptic efficacy. These parameters can be set globally for each of the cores but not on the individual neuron level. Due to device mismatch, effective values of these parameters vary across a neuron population. As a result, although we can specify the same parameter bias for all neurons on Dynap-se, every individual neuron exhibits very different behavior, as reported in the

Dynap-se user guide[1]. Moreover, the connections between neurons are restricted to only ternary values $+a, -a, 0$ (the scaling factor a can be set per core), corresponding to excitatory, inhibitory or no connections. Finally, since the chip is an analog device, state variables such as membrane potentials and post-synaptic currents are only observable through an oscilloscope. For information processing and learning, only spike trains can be recorded and used.

To circumvent these constraints imposed by hardware, we implemented the above-mentioned Reservoir Transfer method to create a reservoir using 3 cores on Dynap-se. A leaky ESN of equal size was created in simulation as a teacher reservoir, and its response to white noise input signal $\mathbf{u}(t)$ was collected and converted into spike trains to be sent to the target reservoir in Dynap-se. After the output spike trains from the hardware neurons are recorded, we smooth both the input and output spike trains by an exponential decay kernel to get $\mathbf{x}(t)$ and $\mathbf{r}(t)$, respectively. Finally, ternarized linear regression [13] was applied to compute the ternary weight matrix W_{ternary} such that $\mathbf{x}(t) \approx W_{\text{ternary}}\mathbf{r}(t)$. In this way, there is no need to know the exact values of the membrane potentials and other parameters, which are unobservable and varying across individual neurons. The neurons do not have to share the same parameter value as long as their collective response to the input current $\mathbf{x}(t)$ contains enough information to linearly decode $\mathbf{x}(t)$. Moreover, learning is needed only once using a white noise signal, afterwards the connection weights can stay fixed. Hence no online adaptation on hardware is needed.

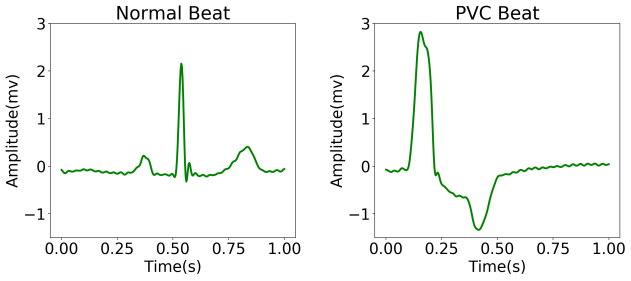

Fig. 3. Left panel: a normal heartbeat. Right panel: a PVC heart beat.

To verify that the transfer learning method yields a functional physical spiking reservoir, we conducted an ECG signal classification experiment using the learned reservoir on Dynap-se. The experiment aims to detect Premature Ventricular Contractions (PVCs), which are abnormal heartbeats initiated by the heart ventricles. Figure 3 shows a normal heartbeat and a PVC. More concretely, we formulate the PVC detection task as a supervised temporal classification problem which demands a binary output signal $z(n)$ for each heartbeat indexed by n:

$$z(n) = \begin{cases} 1 & \text{if the } n\text{-th heartbeat is a PVC,} \\ 0 & \text{otherwise.} \end{cases} \quad (6)$$

The experiment was conducted with the following routine. (1) *ECG pre-processing*: we removed the baseline drift from an ECG signal by applying a high-pass Butterworth filter and then normalized the signal into the numerical range [0,1]. (2) *Signal-to-spike conversion*: we placed a spike at a time index if the increase/decrease of the ECG signal relative to its value at the previous spike time surpassed a threshold of numerical value 0.1. (3) *Reservoir response harvesting*: we sent ECG-converted spike trains into Dynap-se to harvest the reservoir responses, which were in the form of spike trains. (4) *Spike-to-signal conversion*: on a digital computer, we smoothed the spike trains collected from the physical reservoir to continuous-valued time-series by an exponential decay kernel with a decay time constant 3.5. (5) *Classifier training*: the training of the readout mechanism amounted to solving a linear regression problem, where the input for linear regression was the smoothed reservoir responses and the target output was a $\{0, 1\}$-valued binary signal indicating the correct labels of heartbeats. (6) *Result evaluation*: with a testing ECG time-series, we repeated the above procedure to procure its smoothed reservoir responses and then readout the predicted labels with learned weights. We used the following standard metrics to evaluate the classification performance: accuracy (Ac), sensitivity (Se), precision (P), and F1-score.

We used MIT-BIH ECG arrhythmia database [23], [24] in this experiment. The database provides 48 half-hour excerpts of two-channel ambulatory ECG recording files, obtained from 47 different patients. The recordings were digitized with a sampling frequency of 360 Hz and acquired with 11-bit resolution over a 10mV range. Each record was annotated by two or more cardiologists independently, both in timing information and beat classification. In this work, we present the results of a case study where recordings from lead II of file #119 (female, age 51) and its corresponding annotation file were used to train the test classifier: we employed the first 10 minutes of the recording signal for training and the next 5 minutes for testing. A comparison of classification accuracy on testing data between the low-precision spiking reservoir and the digitally simulated, high-precision reservoir baseline is provided in Table I. Simulation on baseline standard ESN was performed with parameters set as leakage rate = 0.99, spectral radius= 0.9 and regression parameter = 1e-6.

TABLE I
PVC DETECTION RESULTS ON TESTING DATA

	Ac	Se	P	F1
Standard ESN	98.24 %	92.21 %	100 %	95.95 %
Dynap-se reservoir	97.56%	87.50 %	98.00 %	92.45 %

V. CONCLUSION

Implementing efficient algorithms on neuromorphic hardware with low energy consumption is a promising yet challenging path towards novel brain-machine interfacing

[1]See the legacy documentation in `https://ai-ctx.com/support`

neuro-technologies. In this paper, we reviewed the major difficulties we encountered along this path: low bit resolution, device mismatch, uncharacterized neural models, unavailable state variables, physical system noise and most importantly, timescale mismatch. As a solution, we proposed a computational scheme called Reservoir Transfer to circumvent these difficulties and created a functional spiking reservoir on the Dynap-se analog asynchronous neuromorphic microprocessor board. Empirical results from an ECG signal monitoring task showed that this reservoir with ternary weights was able to not only integrate information over a time span longer than the timescale of individual neurons but also functioned as an information processing medium with performance close to a standard, high precision, deterministic, non-spiking ESN.

For future work, it remains to be understood how a transferred weight matrix is related to the time constant and leakage rate of the teacher reservoir. In addition, applying this method to transferring trained recurrent neural networks instead of a randomly created ESN will also be an exciting direction for further research.

Acknowledgement We thank Roberto Cattaneo and Giacomo Indiveri for valuable help with Dynap-se.

REFERENCES

[1] K. Birmingham, et al., "Bioelectronic medicines: a research roadmap," *Nature Reviews — Drug Discovery*, 13, 399–400, 2014.

[2] I. Goodfellow, Y. Bengio and A. Courville, *Deep Learning*, MIT Press, `http://www.deeplearningbook.org`, 2016.

[3] W.Y. Tsai, et al., "Always-On speech recognition using TrueNorth,a reconfigurable, neurosynaptic processor," *IEEE Trans. on Computers*, 66(6):996–1007, 2017.

[4] G. Indiveri and , S.C. Liu, "Memory and information processing in neuromorphic systems," *Proceedings of the IEEE*, 103(8):1379–1397, 2015.

[5] H. Jaeger "Echo State Network" *Scholarpedia* p. 2330, 2007 `www.scholarpedia.org/article/Echo_State_Network`

[6] W. Maass, T. Natschläger, and H. Markram, "Real-time computing without stable states: a new framework for neural computation based on perturbations," *Neural computation*, 14(11):2531–2560, 2002.

[7] H. Jaeger and H. Haas "Harnessing Nonlinearity: Predicting Chaotic Systems and Saving Energy in Wireless Communication", *Science*, 304:78-80, 2004

[8] S. Moradi, N. Qiao, F. Stefanini and G. Indiveri, "A scalable multicore architecture with heterogeneous memory structures for dynamic neuromorphic asynchronous processors (dynaps)", *IEEE transactions on biomedical circuits and systems*, 12(1):106–122, 2018.

[9] J. Stark "Delay Embeddings for Forced Systems. I. Deterministic Forcing", *J. Nonlinear Sci.* 9:255332, 1999

[10] J. Stark et al. "Delay Embeddings for Forced Systems. II. Stochastic Forcing", *J. Nonlinear Sci.* 13:519577, 2003

[11] S. Ganguli., D. Huh, and H. Sompolinsky. "Memory traces in dynamical systems", *PNAS*, 105(48):18970 - 18975, 2008

[12] M. Hermans and B. Schrauwen "Memory in linear recurrent neural networks in continuous time", *Neural Networks*, 23(3):341-355, 2010

[13] C. Zhu, S. Han, H. Mao and W.J. Dally, "Trained ternary quantization",, *arXiv preprint arXiv:1612.01064*, 2016.

[14] Y.H. Hu, S. Palreddy and W.J. Tompkins, "A patient-adaptable ECG beat classifier using a mixture of experts approach", *IEEE transactions on biomedical engineering*, 44(9):891–900, 1997.

[15] M. Faezipour, et al., "A patient-adaptive profiling scheme for ECG beat classification", *IEEE Transactions on Information Technology in Biomedicine*, 14(5):1153–1165, 2010.

[16] O.T. Inan, L. Giovangrandi and G.-T.A. Kovacs, "Robust neural-network-based classification of premature ventricular contractions using wavelet transform and timing interval features", *IEEE transactions on Biomedical Engineering*, 53(12):2507–2515, 2006.

[17] D.A. Coast, R.M. Stern, G.G. Cano, S.A. Briller, "An approach to cardiac arrhythmia analysis using hidden Markov models contractions using wavelet transform and timing interval features", *IEEE Transactions on biomedical Engineering*, 37(9):826–836, 1990.

[18] J. Mateo, A.M. Torres, A. Aparicio and J.L. Santos, "An efficient method for ECG beat classification and correction of ectopic beats", *Computers & Electrical Engineering*, 53:219–229, 2016.

[19] I. Christov, et al., "Comparative study of morphological and time-frequency ECG descriptors for heartbeat classification beats", *Medical engineering & physics*, 28(9):876–887, 2006.

[20] X. He, "Transfer learning for reservoir computing in neuromorphic hardware with low bit precision", *Cognitive Computing: Merging Concepts with Hardware*, Hannover, Germany, December 18-20, 2018.

[21] I. Christov, et al., "ECG beat classification using particle swarm optimization and radial basis function neural network", *Expert systems with Applications*, 37(12):7563–7569, 2010.

[22] I. Christov, et al., "Real-time patient-specific ECG classification by 1-D convolutional neural networks", *IEEE Transactions on Biomedical Engineering*, 63(3):664–675, 2016.

[23] A.L. Goldberger, et al., "PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals", *Circulation*, 101(23):e215–e220, 2000.

[24] G.B. Moody and R.G. Mark, "The impact of the MIT-BIH arrhythmia database", *IEEE Engineering in Medicine and Biology Magazine*, 20(3):45–50, 2001.

[25] G. Bortolan, R. Degani, J.L. Willems, "ECG classification with neural networks and cluster analysis", *Computers in Cardiology 1991, Proceedings*, pp. 177–180, 1991.

[26] W. Nicola and C. Clopath, "Supervised learning in spiking neural networks with FORCE training", *Nature Communications*, 8(1):2208, 2017.

[27] D. Goodman and R. Brette, "The Brian simulator", *Frontiers in Neuroscience*, 3:26, 2009.