Review of Liu et al. 2022 - *Human-Level Control Through Directly Trained Deep Spiking Q-Networks* for PENG9560

Michael Joseph Tarlton - MICHAELT@OSLOMET.NO

Abstract

We review the recent paper Liu et al. 2022 - *Human-Level Control Through Directly Trained Deep Spiking Q-Networks* [1] and their implementation of a directly trained spiking implementation of a Deep Q-Network. We cover the previous literature leading into their design and touchstones of the methodology. We also add our commentary on how this intersects with novel models of neuro-inspired reinforcement learning models, and future directions for the field of Deep Spiking Reinforcement Learning.

Index Terms

DRL, SNN, DQN, DSQN

I. Introduction

THE approach Reinforcement Learning (RL) takes that of some "agent" in an environment where it must optimize its actions based on evaluations of the state of the surrounding environment in order to optimize for a reward. This has its basis in real-world biological models of learning, borrowing from studies of animal models where an animal is conditioned to actions in the presence of environmental inputs to elicit a reward such as food.

Current RL methods utilize tabular methods, where an association of *state* and *action* is mapped to *reward* as a state-action pair, or a *Q-policy*, denoted: $Q(S, A)$. These policies are optimized by some *Q-function*.

For a *Q-Function*, there is internally is a Q-table that contains all the *Q-policies*: mappings between the state of the environment and the subsequent most likely course of action to result in reward. This is represented in the table by a *Q-value*. Given a state and action, our Q-Function will search into its Q-table the corresponding value.

However, this approach suffers from the *curse of dimensionality*. In regimes of increasingly larger environmental input and action space, the policy space scales exponentially.

Additionally, this discrete policy method fails in more naturalistic regimes where spaces have continuous values, causing discrete mappings to fail as even small differences between values can result in wildly different outcomes.

The success of Deep Neural Networks (DNN) which are able to encode high-dimensional data in the latent space of relatively condense network structures and are capable of continuous mappings offers a potential solution to the inherent weaknesses of RL. However, until relatively recently, this combination of machine learning methods had no clear path forward. The key training method in traditional Artificial Neural Networks (ANN), back-propagation-through-time, can not be implemented in a Q-policy framework.

A precursor to the paper reviewed here, and a cornerstone of the Deep Reinforcement Learning (DRL) field Mnih et al 2015 [2], developed a novel DRL network, which implemented a new form of Q-Learning titled *experience replay*. Experience Replay, inspired by neuroscientific models of hippocampal memory replay [3] [4], randomizes over batches of data, removing correlations through time in the episode sequence. This provides the basis for their highly-successful implementation of RL in an DNN, dubbed a ***Deep Q-Network (DQN)***.

As all the methods so far have their basis in biological models of learning, there is a need to further develop models which mimic the capabilities and mechanisms of the human brain. Recent work has been done to develop Spiking Neural Networks (SNNs), networks that mimic the spiking functions of biological neurons at the neuronal level.

SNNs lie at an oblique juncture with traditional ANNs, since the mode of communication at the neuron level is fundamentally different. They eschew ANNs' static and continuous-valued activation for the binary and highly time-dependent information passing of spikes, where spike rates and timing are important [5]. The exact nature of spiking communication in the human brain is highly complex and not a solved problem by any means. Many plausible mechanisms have been successfully used in SNN models over the years [6].

The swath of potential benefits offered by SNNs is the main motivator behind their development. Neuromorphics, hardware designed specifically around SNNs to exploit their low-power power properties, alone offers an economic incentive to their development. SNNs naturally encode time information and utilize low-power, event-driven processing, which may further the possibilities in online and self-supervised RL models.

In this paper we review Liu et al. 2022 - *"Human-Level Control Through Directly Trained Deep Spiking Q-Networks"*, which provides a complete and effective implementation of RL, DNN, and SNNs; By expanding on the original DQN network

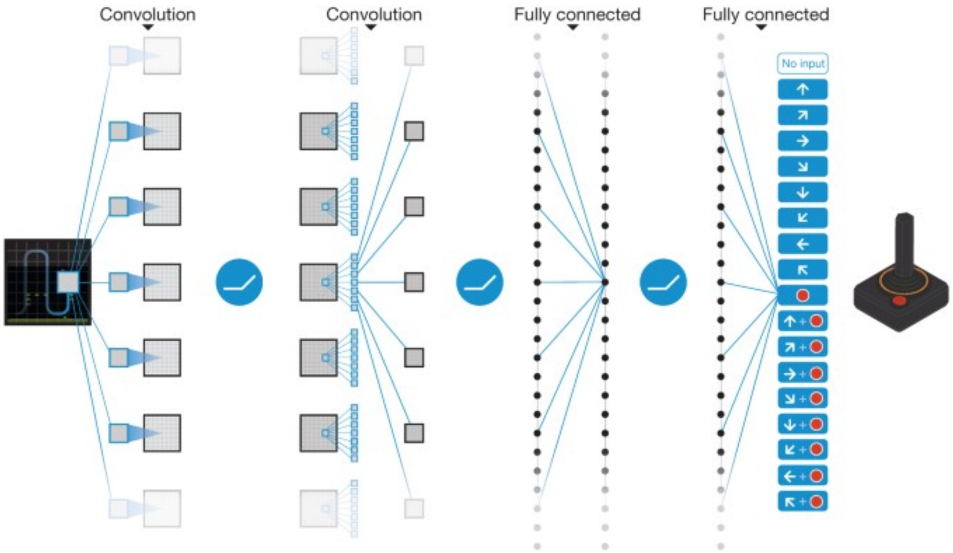


Fig. 1. Architecture of a Deep Q-network, adapted from [2].

[2] with a full SNN implementation. Liu et al. 2022 is the first of its kind to implement a full spiking network, whereas any previous SNN-DQN models relied on a traditional ANN at a point in the process. This review will cover the methods of Liu et al. 2022 and how they built on models post-Mnih et al. 2015.

II. Methods

A. *Mnih et al. 2015 - Deep Q-network*

The *Deep Q-network (DQN)* architecture upon which these models are based comes from Minh et al. 2015 [2]. In broad strokes, this model implements RL in a deep learning regime by mapping the Q-policies into the state of the network by training the hyperparameters to the target network with a Q-learning algorithm. An image is used as the observed environmental input, the resultant Q-values being interpreted as the values on the output layer, with each output node corresponding to some action.

The implementation of RL into a DNN model does not come easily. RL disallows for use of back propagation through time, where in Q-learning, policies can be dependent on the trajectory of actions taken over multiple time steps.

The authors of Mnih et al. 2015 provide a solution to this in what they introduce as *experience replay*. This method is inspired by biological modes of learning where episodes of action in memory are replayed during sleep. Likewise, experience replay that randomizes over the data, to remove correlations in the observation sequence and smoothing over changes in the data distribution. This is done by storing episodes of experience at each time step $e_t = (s_t, a_t, r_t, s_{t+1})$ in a buffer, then for some time window, a training step is performed, where a sub-sample of experiences are randomly selected and trained on with Q-learning. To stay a tractable problem, the training step is performed in regular interval windows throughout the duration of the experiment, e.g. every 100,000 timesteps.

As a benchmark the DQN was tested with 49 games from the Atari 2600, with a different network being trained for each game. For each games, the image of the current screen, action state, and score (corresponding to the reward) is available to the model.

1) DQN - Architecture: The DQN is an ANN composed of three initial convolutional layers followed by two fully-connected layers. A schematic of the architecture is illustrated in figure 1

The input layer receives an 84 x 84 x 4 input of game frames which have been preprocessed into 84 x 84 pixel images, with the 4 most recent frames at the experience time step t in the subsample being selected.

The first hidden layer convolves 32 filters of 8 x 8 with stride. The second hidden layer convolves 64 filters of 4 x 4 with stride 2. The third and final convolution layer convolves 64 filters of 3 x 3 with stride 1. Each convolution layer is rectified by ReLUs before passing to the next layer.

The next two layers are fully-connected layers, the first consists of 512 rectifier units, and the output layer is a fully connected layer with a number of outputs corresponding to the number of valid actions in the Atari games tested. This varied between 4 and 18 actions in the games tested.

The resulting model was able to reach and surpass human level performance, in 29 of the 49 games tested.

B. The addition of an SNN

The goal of our Liu et al. 2022 is to take the well established framework of [2] and create a fully spiking implementation of it. The motivation for the addition of spiking neurons is expanded in [7] [8] [9], the main advantages being low energy and low computational costs. In RL they potentially offer a powerful tool in designing fully-online and self-supervising models.

The design of spiking neural networks is still relatively unexplored, and our first steps are to implement these in established methods. Likewise, as a biologically inspired model, there are potential benefits from adaptation of neurological mechanisms in neural networks. Neurological models, which are necessarily deep and reinforcement trained, can possibly have their properties exploited for better DRL models.

However, their unique and non-conventional communication technique means implementation in traditional models requires some method of encoding and decoding the spike activity.

Previous work which introduced SNNs to the DQN framework did so by pre-training the ANN network before conversion into an SNN. This was done by converting the ReLU units into *Leaky Integrate and Fire **(LIF)*** spiking units [10] and treating the spike frequency over the training window as the output value. These networks proved to perform similarly to the original DQN, and even proved to be more robust to input perturbations than the DQN.

Tan et al. 2020 further improved the conversion of ANNs to SNNs, through use of *Integrate and Fire **(IF)*** neurons, which approximates the ReLU output in its firing rate over time (explained below). This improved the accuracy of the converted SNNs, and stands as the example to which the authors of Liu et al. 2022 compare their model against. However, this model still requires conversion from an ANN as well as a larger simulation time window (100 time steps vs 64 in Liu et al. 2022) making for a less tractable problem.

However, in order to train a full Deep SNN, there is an additional caveat to mitigated. The spike function of the neuron is non-differentiable, where the output is defined by the Heaviside step function, and equal to 1 at the time of a spike, and equal to 0 at all other times. This is mitigated by surrogate gradient descent, where the Heaviside step function is replaced with a surrogate arctan function.

C. Liu et al. 2022 - The Deep Spiking Q-Network (DSQN)

The Liu et al. 2022 model, which they title the *Deep Spiking Q-Network **(DSQN)***, replicates the Mnih et al. 2015 DQN architecture, however with the full replacement of artificial units with spiking LIF neurons. Thus, circumventing the need for a pretrained ANN and obviating any issues arising during conversion between architectures.

They describe the neuronal dynamics of LIF neurons as follows, for layer $l \in \{1, \ldots, L-1\}$ at simulation time t:

$$U^{l,t} = V^{l,t-1} + \frac{1}{\tau_{\mathrm{m}}}\left(W^{l} S^{l-1,t} - V^{l,t-1} + V_{\mathrm{r}}\right)$$

Equation (1) describes the pre-spike, sub-threshold membrane potential of neurons. V_{th} describes the threshold potential, the limit at which if exceeded, the neuron will emit a spike and reset the membrane voltage. τ_{m} denotes the membrane time constant, essentially the decay rate at which membrane potential will "leak." W^{l} denotes the learnable weights of the neurons in layer l, and V_{r} denotes the initial membrane potential.

They describe two varieties of "reset" when the threshold membrane voltage is exceeded, a "hard reset" which mimics the typical dynamics of biological neurons, fully resetting the membrane potential back to some baseline, typically $V = 0$; and a "soft reset", a less common but still biologically plausible model of neuron dynamics, where after exceeding the threshold, the threshold membrane potential is subtracted from the current membrane potential. The membrane potential of neurons when reached V_{th} is described by:

$$V^{l,t} = \begin{cases} U^{l,t}\left(1 - S^{l,t}\right) + V_{\mathrm{r}} S^{l,t}, & \text{hard reset} \\ U^{l,t} - V_{\mathrm{th}} S^{l,t}, & \text{soft reset.} \end{cases}$$

Though for their experiments they only use the hard reset in LIF, they exhibit this distinction to compare with previous methods which rely on non-leaky IF neurons for the conversion of ANNs to SNNs. Because the IF neuron is reset by a soft reset as an unbiased estimator of the ReLU activation function over time (see figures 2 and 3 for comparison).

They find that the LIF neuron with a hard reset to be more robust during the training stage, with a wider range for different thresholds. "LIF neurons have more potential to obtain optimal results in our DSQN and could be directly trained without relying on the normalization technique in conversion methods" [1].

The output of the LIF neurons in layer $l \in \{1, \ldots, L-1\}$ at simulation time t is described by the equation:

$$S^{l,t} = \Theta\left(U^{l,t} - V_{\mathrm{th}}\right)$$

$$\Theta(x) = \begin{cases} 1, & x \geq 0 \\ 0, & \text{otherwise} \end{cases}$$

where $\Theta(x)$ is the spiking function of the neurons. As this is a non-differentiable function, a surrogate function must be used during training. In this paper, the authors use the arc-tangent function, as it provides less complexity in comparison to the sigmoid function, which is also commonly used.

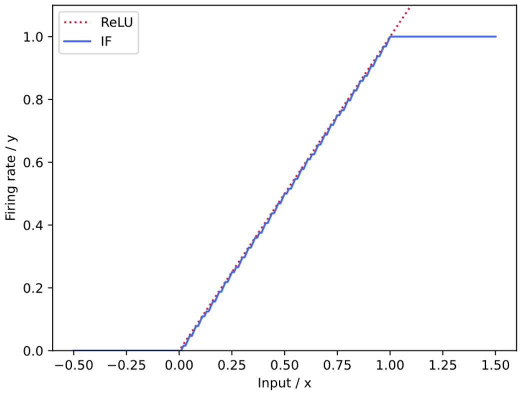


Fig. 2. Firing rate of an IF neuron with soft reset compared with the output of a ReLU unit. Adapted from [1]

To read out the Q-values of the final layer, they take they sum of weighted spike input from the final hidden layer over the time simulation window. Their output is read as:

$$O^L = W^L \frac{1}{t} \sum_{t'=1}^{t} S^{L-1,t'}$$

where O^L denotes the output Q-values of DSQN, W^L is the learnable weights of the neurons in the final layer, and t is the simulation time window.

The DSQN is then trained according to the deep Q-learning algorithm [2], using the following loss function:

$$\mathcal{L}(W) = \mathbb{E}_{(s,a,r,s') \sim U(D)} \left[\left(y_{(r,s')} - Q(s,a;W) \right)^2 \right]$$

with

$$y_{(r,s')} = r + \gamma \max_{a'} Q\left(s', a'; W^-\right)$$

where $Q(s,a;W)$ denotes the approximate Q-value function parameterized by DSQN. W and W^-denote the weights of DSQN at the current and previous steps, respectively. $(s,a,r,s') \sim U(D)$ denotes the minibatches drawn uniformly at random from the experience replay memory D, and γ is the reward discount factor [1].

III. Model Evaluation

The qualities measured were performance, stability, learning capability, and energy efficiency. They compared their results with their own reproductions of the original DQN model in Mnih et al. 2015 as well as the conversion-based SNN in Tan et al. 2020 [5], using the 17 best performing Atari games that were also used in Tan et al. 2020, these are also the games for which the original DQN performed highest.

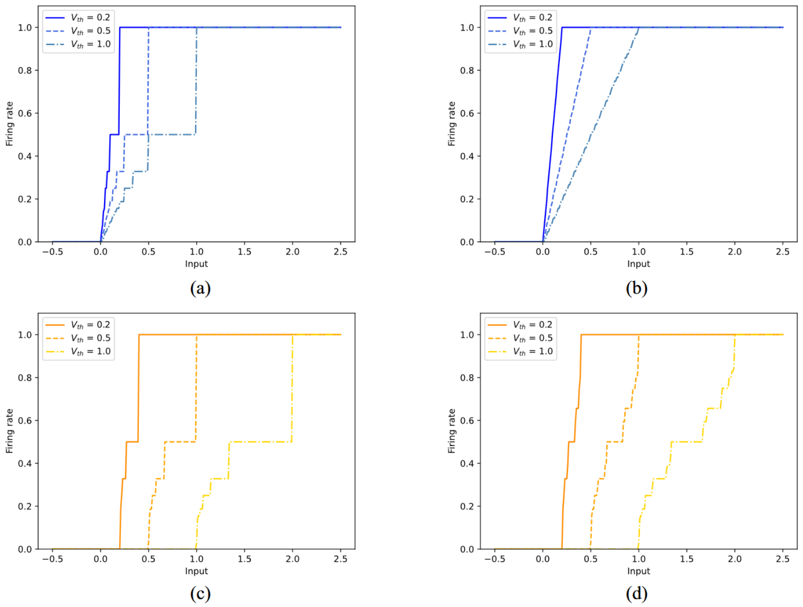
The in game frames are preprocessed to render them in grayscale and down-resolutioned into 84 x 84 pixel images. The m most recent frames (where $m = 4$) are stacked as input. The real pixel values are fed as input to the network without neural encoding.

The game is then simulated with a training time step occurring with a window of 64 time steps for the DSQN and 100 time steps for the conversion-based SNN.

In performance, of the 17 Atari games tested, the DSQN outperformed (in terms of points) the DQN by an average of 106%. The DSQN was able to outperform the DQN on four games, scored equally in nine games, and scored worst (maximum 20% difference) in four of the games. The DSQN proved to be more stable than the DQN, obtaining lower standard deviations in score in six games, equal to the DQN in two games, and inferior in nine games.

In learning capability, a total proof is not given. In Figure 4 the learning curve for DQN and DSQN is compared for two games. From these plots, it is apparent that the DSQN reaches a higher score equilibrium faster than the DQN, as well as in the average max Q-value. Though the authors claim this is representative in all cases, no metric is provided. In the future an area under curve metric may be a useful measure compare.

As with the DQN, we can see instability in the learning curves which is native problem with the DQN. In DQNs this is mitigated through implementation of a Double-DQN [11] , where a *target network* is simultaneously trained to calculate the target Q-values in the next state. The authors actually implement this DQN model as well as the more recent CDQN [12]. The authors claim their implementations of these variations claim a better performance to their ANN counterparts by 151.4% and 141.8% to the Double-DQN and CDQN respectively. Unfortunately, no standard deviation measurements are provided, but the comparison of Figure 4 and Figure 5 appears to show less deviation.



(a)

(b)

(c)

(d)

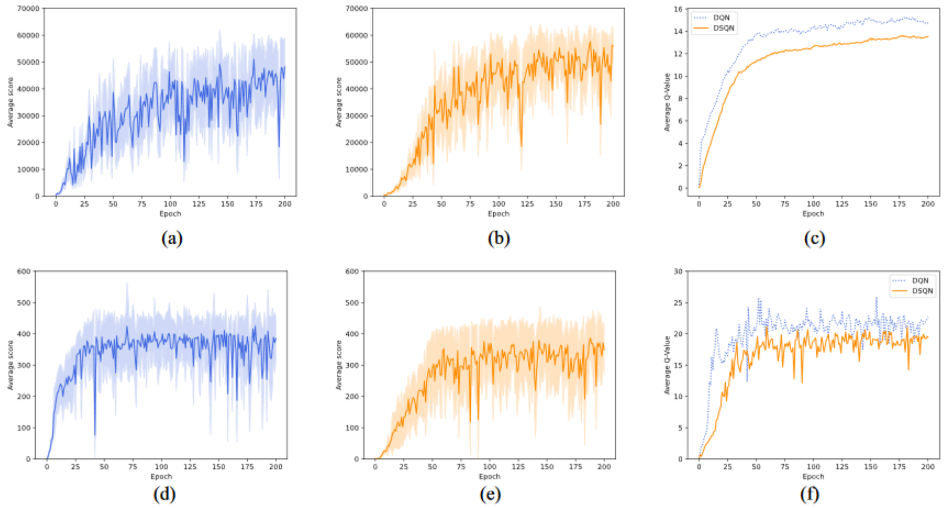
Fig. 3. Comparisons between firing rate and inputs of the IF and LIF neurons in both hard and soft resets.

$$V_{th}$$

is set to 0.2, 0.5, and 1, respectively.

$$V_r = 0$$

constantly. (a) IF neuron reset by hard reset. (b) IF neuron reset by soft reset. (c) LIF neuron reset by hard reset. (d) LIF neuron reset by soft reset. Adapted from [1]



(a)

(b)

(c)

(d)

(e)

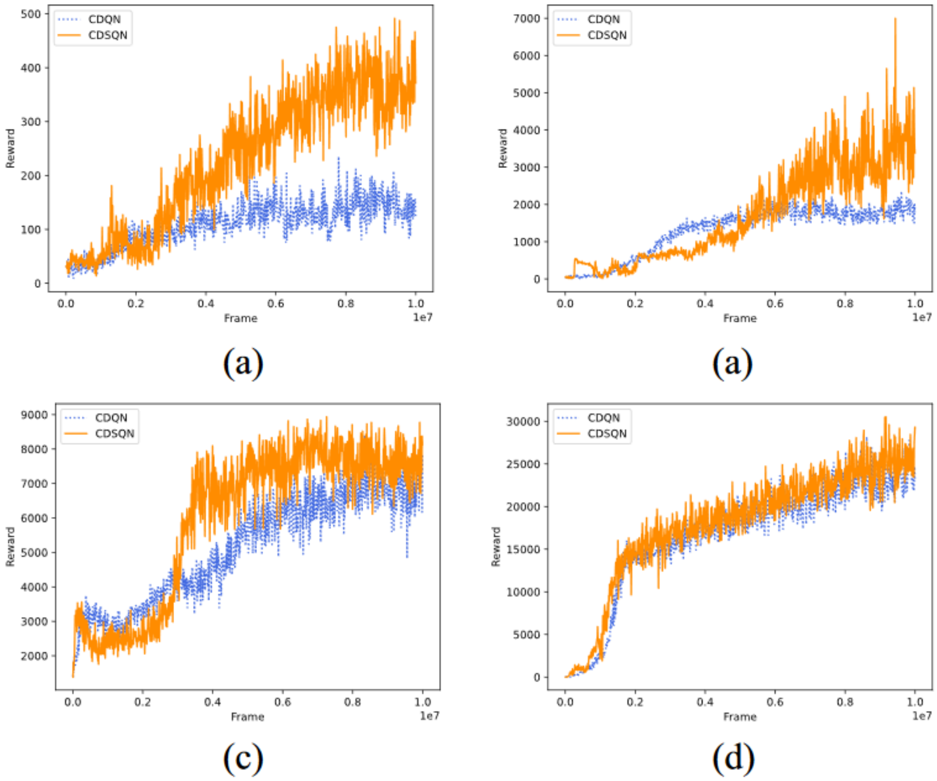
(f)

Fig. 4. Learning curves of the DSQN and DQN in games Star Gunner and Breakout. Each episode is run for 30 rounds.(a), (b), (c), (d), and (e) plot the average score per episode. (c) and (f) plot the average max Q-value achieved per episode (epoch = 250,000 time steps). Adapted from [1].

Lastly, they score the model in terms of energy-efficiency. This is a useful metric for prospective uses in neuromorphic hardware, where the energy savings from spiking can be best obtained. However, there is no direct method to compare the DSQN with the DQN in this manner. A rough estimate can be made by comparing the number of calculations needed per inference. In terms of energy transfer, the SNN must be evaluated in average spike count. In this case, it was better to compare it to the conversion-based SNN of [5]. The results show the DSQN used around 85% of the energy in comparison to the conversion-based SNN. However, this comparison was only shown for two games.

IV. Review

I find Liu et al. 2022 to be a very thorough and complete work which takes the next logical step in deep reinforcement learning networks with spiking neurons. This paper appears to be the first to showcase DRL in a fully SNN model, as other



(a) (a)

(c) (d)

Fig. 5. Learning curves of CDQN and CDSQN in four Atari games. Adapted from [1].

models require conversion of pre-trained ANNs, or mediatory ANN side-networks [13]. The following proof of increased performance and robustness with a full SNN model, sets the stage for future models designed around total SNN architecture.

The paper is a very exhaustive with fine detail of network architecture, neuron design, training methods, as well as thorough comparison with a breadth of other models in the experiments. The provided methods and code repositories makes readily the reproduction of their model. Though as of writing, the original github repository is unavailable.

As with Mnih et al. 2015, this paper will likely serve as a benchmark for future DSRL models. The success of a fully spiking model in this example means development of further models which may utilize SNN's efficiency to address problems with larger state and action spaces.

Additionally, there are mechanisms here that future deep SNN learning methods may be expected to duplicate. Namely, experience-replay, which is again based on biological processes of memory formation and learning. [2] directly based the concept on hippocampal replay wherein the during sleep, the brain replays time-condensed activation of neural circuits associated with an experienced physical activity [3][4]. In SNNs, where spike timing larger oscillatory time cycles of neuron activity known as phases are critical to network communication, perhaps a designed neuronal circuit based around replicating the buffer like memory and replaying the buffer in phases of time could be an aim of future experiments.

Because the varieties of spiking neurons have no canonical set, further experiments may be done with variations in neuron types, as well as unique training methods specific to their mechanics. Testing the performance of spiking neuron varieties in a DSQN will provide a valuable point of comparison.

However, the reliance on gradient training may prove to be a computational bottleneck. It may be prudent to then test the capacity and computational costs of DSQNs. The unique properties of SNN communication may lead to the development of alternative training techniques better suited to accelerate SNN performance.

V. Future Directions

This paper marks an important milestone in the development of deep spiking reward based models, but as an in-place model its efficacy in a naturalistic and dynamic environment, ones that RL and automata algorithms are typically designed for, is extremely limited. The experience-replay method effectively leverages a Q-learning framework to work in a DNN, but this limits the DSQN model to an offline training algorithm for an offline, single-task-specific network.

While not the totality of RL motivations, a reward based framework can allow for a dynamically adapting model able to train in an online and self-supervised environment. With the goal in mind of designing autonomous and reactive hardware, there is a need for models which are able to learn and adapt to unpredictable and isolated environments. The DSQN can inform us of the capabilities of an in-place SNN trained with an RL framework. The next step is to experiment with reward mechanisms which will allow for dynamic learning in an online setting. To this end, it is necessary to replicate the abilities of SNNs similar to those found in biological agents.

This problem is studied from the field of biologically inspired circuits is the *spatial credit assignment* problem. Similarly to RL, this problem of how to accurately assign reward, to neuronal circuits, which by nature are highly entangled and recurrent structures. and Local plasticity is a well-known mechanism of adaptation at the neuron and neuronal assembly level. A well studied mechanism that is being studied for use in SNNs is that of Spike-Timing Dependent Plasticity (STDP) as well as its extension *three-factor dependent plasticity* which performs training updates on the basis of a *third-factor* which may come in the form of a reward-signal. Additionally, plausible (though without biological basis) methods of back-propagation in SNNs have lately been proposed [14] [15] further adding alternative plausible methods of credit-assignment in Deep-SNNs.

This however is a fairly nascent field, although emerging over two decades ago in 1996 [16], it has gained heavy traction in the past few years [8] [9] alongside the acceleration of computational neuroscience and application-specific integrated circuits in the machine learning space; and so there is a need to develop and prove the efficacy of proposed methods in this space.

At this point, we are approaching a model of reinforcement learning from an entirely new direction. This can be considered as a gap between artificially driven methods and biologically-inspired ones, and there is recent motivation to bridge the field of neuroscience back to deep learning [7] , by designing entirely new mechanisms of reward-based learning that are effective in SNNs, as opposed to attempting to apply the methods designed for ANNs, then somewhat awkwardly fitting SNNs to them. This is not to say the gains made from the ANN-to-SNN approach are non-contributing, but instead we can use these proven models such as the DSQN to bridge the gap from one direction and build towards it from the other. Any future Deep SNN implementations will need to match the benchmark set here by Liu et al. 2022.

References

[1] Guisong Liu et al. “Human-Level Control Through Directly Trained Deep Spiking Q-Networks”. In: *IEEE Transactions on Cybernetics* (2022), pp. 1–12. ISSN: 2168-2267, 2168-2275. DOI: 10.1109/TCYB.2022.3198259.

[2] Volodymyr Mnih et al. “Human-Level Control through Deep Reinforcement Learning”. In: *Nature* 518.7540 (Feb. 2015), pp. 529–533. ISSN: 1476-4687. DOI: 10.1038/nature14236.

[3] Daniel Bendor and Matthew A. Wilson. “Biasing the Content of Hippocampal Replay during Sleep”. In: *Nature neuroscience* 15.10 (Oct. 2012), pp. 1439–1444. ISSN: 1097-6256. DOI: 10.1038/nn.3203.

[4] Joseph O’Neill et al. “Play It Again: Reactivation of Waking Experience and Memory”. In: *Trends in Neurosciences* 33.5 (May 2010), pp. 220–229. ISSN: 0166-2236. DOI: 10.1016/j.tins.2010.01.006.

[5] Weihao Tan, Devdhar Patel, and Robert Kozma. *Strategy and Benchmark for Converting Deep Q-Networks to Event-Driven Spiking Neural Networks*. Dec. 2020. DOI: 10.48550/arXiv.2009.14456. arXiv: 2009.14456 [cs].

[6] Alex Vigneron and Jean Martinet. “A Critical Survey of STDP in Spiking Neural Networks for Pattern Recognition”. In: *2020 International Joint Conference on Neural Networks (IJCNN)*. July 2020, pp. 1–9. DOI: 10.1109/IJCNN48605.2020.9207239.

[7] Friedemann Zenke et al. “Visualizing a Joint Future of Neuroscience and Neuromorphic Engineering”. In: *Neuron* 109.4 (Feb. 2021), pp. 571–575. ISSN: 0896-6273. DOI: 10.1016/j.neuron.2021.01.009.

[8] Luke Y. Prince et al. “Current State and Future Directions for Learning in Biological Recurrent Neural Networks: A Perspective Piece”. In: *arXiv:2105.05382 [cs, q-bio]* (Jan. 2022). arXiv: 2105.05382 [cs, q-bio].

[9] A. Mehonic and A. J. Kenyon. “Brain-Inspired Computing Needs a Master Plan”. In: *Nature* 604.7905 (Apr. 2022), pp. 255–260. ISSN: 1476-4687. DOI: 10.1038/s41586-021-04362-w.

[10] Devdhar Patel et al. “Improved Robustness of Reinforcement Learning Policies upon Conversion to Spiking Neuronal Network Platforms Applied to Atari Breakout Game”. In: *Neural Networks* 120 (Dec. 2019), pp. 108–115. ISSN: 08936080. DOI: 10.1016/j.neunet.2019.08.009.

[11] Jiayi Weng et al. *Tianshou: A Highly Modularized Deep Reinforcement Learning Library*. Aug. 2022. DOI: 10.48550/arXiv.2107.14171. arXiv: 2107.14171 [cs].

[12] Zhikang T. Wang and Masahito Ueda. *Convergent and Efficient Deep Q Network Algorithm*. May 2022. DOI: 10.48550/arXiv.2106.15419. arXiv: 2106.15419 [cs].

[13] Ding Chen et al. “Deep Reinforcement Learning with Spiking Q-learning”. In: *arXiv:2201.09754 [cs]* (Jan. 2022). arXiv: 2201.09754 [cs].

[14] Alexandre Payeur et al. “Burst-Dependent Synaptic Plasticity Can Coordinate Learning in Hierarchical Circuits”. In: *Nature Neuroscience* 24.7 (July 2021), pp. 1010–1019. ISSN: 1546-1726. DOI: 10.1038/s41593-021-00857-x.

[15] Ezekiel Williams et al. “Neural Burst Codes Disguised as Rate Codes”. In: *Scientific Reports* 11.1 (Aug. 2021), p. 15910. ISSN: 2045-2322. DOI: 10.1038/s41598-021-95037-z.

[16] Wolfgang Maass. “Networks of Spiking Neurons: The Third Generation of Neural Network Models”. In: *Neural Networks* 10.9 (Dec. 1997), pp. 1659–1671. ISSN: 08936080. DOI: 10.1016/S0893-6080(97)00011-7.