

MASTERARBEIT | MASTER'S THESIS

Titel | Title

Understanding Policy Churn: Susceptibility, Exploration and
Plasticity Loss

verfasst von | submitted by
Thomas Geier BSc

angestrebter akademischer Grad | in partial fulfilment of the requirements for the degree of
Master of Science (MSc)

Wien | Vienna, 2026

Studienkennzahl lt. Studienblatt | Degree
programme code as it appears on the
student record sheet:

UA 066 645

Studienrichtung lt. Studienblatt | Degree
programme as it appears on the student
record sheet:

Masterstudium Data Science

Betreut von | Supervisor:

Assoz. Prof. Dipl.-Ing. Dr.techn. Sebastian
Tschatschek BSc

Acknowledgements

This work was made possible thanks to the continuous support, guidance, and encouragement of many people. First, I would like to express my sincere gratitude to my mentor, Timo Klein, for his constant support, helpful advice, and motivation whenever I needed it. I am also deeply grateful to Sebastian Tschatschek for supervising this thesis and for his valuable guidance throughout the process. Special thanks go to my fellow students and friends Botond, Predrag, Felix, and Thalís, as well as to all the other friends who supported me during my master's programme. Their encouragement and companionship made this journey much more meaningful, and I would not have been able to complete my degree without them. Finally, I am especially grateful to my family for their unwavering love, encouragement, and financial support throughout my studies.

Abstract

Value-based deep reinforcement learning agents, such as DQN and Double DQN, learn a neural estimate of action values and typically act greedily with respect to this estimate. Since even small changes in Q-values can alter the selected greedy action, training can induce *policy churn*: frequent changes in the greedy policy over time. This phenomenon is relevant for exploration, because changing greedy actions can expose the agent to new trajectories even without additional random actions. It may also be related to network plasticity, since a network’s ability to respond to new learning signals can influence how its policy changes during training.

In this thesis, we conduct a systematic empirical study of policy churn in value-based reinforcement learning using Double DQN across MiniGrid and Atari environments. We analyse policy churn in relation to exploration, network capacity, plasticity loss, and learning stability.

Our results show that policy churn is commonly observed and can support exploration in simple MiniGrid environments, but this effect is brittle and seed-dependent. In more structured environments such as MiniGrid FourRooms, churn does not reliably produce the temporally coherent behaviour required for deep exploration. In Atari, particularly in Qbert and Phoenix, churn is shaped by learning dynamics and plasticity-related properties of the network. Plasticity-preserving interventions such as Shrink & Perturb, weight resets, and Layer Normalization reshape churn dynamics and can improve learning stability or long-term performance.

Overall, the experiments indicate that policy churn alone is not a reliable substitute for structured deep exploration. Rather, churn is best interpreted as a diagnostic signal of the underlying learning dynamics, especially network plasticity and training stability.

Kurzfassung

Wertbasierte Deep-Reinforcement-Learning-Agenten wie DQN und Double DQN lernen eine neuronale Schätzung von Aktionswerten und wählen Aktionen typischerweise "greedy" auf Grundlage dieser Schätzung. Da bereits kleine Veränderungen in den Q-Werten die ausgewählte greedy Action verändern können, kann das Training *Policy Churn* hervorrufen: häufige Veränderungen der greedy Policy über den Trainingsverlauf hinweg. Dieses Phänomen ist für Exploration relevant, da wechselnde greedy Actions den Agenten auch ohne zusätzliche zufällige Aktionen neuen Trajektorien aussetzen können. Zudem kann Policy Churn mit der Plastizität des Netzwerks zusammenhängen, da die Fähigkeit eines Netzwerks, auf neue Lernsignale zu reagieren, beeinflussen kann, wie sich seine Policy während des Trainings verändert.

In dieser Arbeit führen wir eine systematische empirische Untersuchung von Policy Churn im wertbasierten Reinforcement Learning durch, unter Verwendung von Double DQN in MiniGrid- und Atari-Umgebungen. Wir analysieren Policy Churn im Zusammenhang mit Exploration, Netzwerkkapazität, Plasticity Loss und Lernstabilität.

Unsere Ergebnisse zeigen, dass Policy Churn häufig beobachtet wird und Exploration in einfachen MiniGrid-Umgebungen unterstützen kann, dieser Effekt jedoch fragil und seed-abhängig ist. In stärker strukturierten Umgebungen wie MiniGrid FourRooms erzeugt Policy Churn nicht zuverlässig das zeitlich kohärente Verhalten, das für Deep Exploration erforderlich ist. In Atari, insbesondere in Qbert und Phoenix, wird Policy Churn durch die Lerndynamik und plastizitätsbezogene Eigenschaften des Netzwerks geprägt. Interventionen zur Erhaltung der Plastizität wie Shrink & Perturb, Weight Resets und Layer Normalization verändern die Churn-Dynamik und können die Lernstabilität oder die langfristige Performance verbessern.

Insgesamt deuten die Experimente darauf hin, dass Policy Churn allein kein verlässlicher Ersatz für strukturierte Deep Exploration ist. Vielmehr lässt sich Policy Churn am besten als diagnostisches Signal der zugrunde liegenden Lerndynamik interpretieren, insbesondere im Hinblick auf Netzwerkplastizität und Trainingsstabilität.

Contents

Acknowledgements	i
Abstract	iii
Kurzfassung	v
1. Introduction	1
2. Background	3
2.1. Reinforcement Learning Basics	3
2.2. Policy Churn	3
2.3. Double DQN and Bootstrapped DQN	3
2.4. Measures	5
2.4.1. Policy Churn	5
2.4.2. Plasticity Measures	5
2.4.3. Exploration	7
2.5. Plasticity Methods	7
3. Related Work	9
4. Methodology	11
4.1. Overview	11
4.2. Policy Churn	11
4.3. Environments	12
4.4. Algorithms and Baselines	14
4.5. Plasticity Interventions	16
4.6. Metrics	17
4.7. Experimental Setup	19
5. Experiments	21
5.1. Exploration in Simple Environments	21
5.2. Does Policy Churn Drive Deep Exploration?	24
5.3. Policy Churn and Network Capacity	26
5.4. Policy Churn and Plasticity Loss	29
5.5. Post-hoc Analysis	32
6. Discussion	37
6.1. Policy Churn and Exploration	37

Contents

6.2. Churn as a Symptom of Learning Dynamics	38
6.3. Network Capacity and Churn	39
6.4. Plasticity Loss: The Link Between Churn and Performance	39
6.5. Interpreting the Plasticity Metrics	40
6.6. Policy Churn as a Diagnostic Signal	41
7. Conclusion	43
Bibliography	45
A. Appendix	49
A.1. Hyperparameters	49
A.2. BDQN as Deep Exploration Baseline	50
A.3. Exploration in Atari	51
A.4. Gradient Rank and Target Fitting Capacity	51
A.5. Plasticity interventions in MiniGrid	51
A.6. Declaration of Generative AI and AI-Assisted-Technologies in the Manuscript Preparation Process	56

1. Introduction

Reinforcement Learning (RL) [Sutton and Barto, 2018] has redefined what machines can achieve: It has demonstrated groundbreaking successes across numerous fields, such as mastering complex games like Go [Silver et al., 2016], automating navigation in autonomous vehicles [Inamdar et al., 2024], optimizing strategies in healthcare [Srinivasagopalan, 2024] and financial applications [Bai et al., 2024].

Despite these successes, efficient exploration of the state and action spaces remains a fundamental challenge in RL, especially in environments characterized by sparse or delayed rewards [Osband et al., 2016]. Classical exploration methods, such as ϵ -greedy, often lead to superficial or redundant exploration, resulting in slow learning and suboptimal performance in more complex or high-dimensional tasks.

A recent discovery in value-based RL is the phenomenon of *policy churn*, which describes rapid and often undirected changes in an agent’s greedy policy during training [Schaul et al., 2022]. Policy churn has garnered attention due to its surprisingly widespread occurrence and potential role as an implicit exploration mechanism. Notably, agents trained with Deep Q-Networks (DQN) [Mnih et al., 2013] change their greedy action in approximately 10% of states after just a single gradient update [Schaul et al., 2022]. This rate is far beyond practitioner expectations about how quickly learned policies change. Crucially, policy churn persists even after policy performance has plateaued, indicating that churn is not merely a transient artifact of early-stage learning but a consistent and integral aspect of the training dynamics [Schaul et al., 2022].

The interaction of policy churn with another significant phenomenon, *plasticity loss*, is particularly intriguing and problematic. Plasticity loss describes the diminished capability of neural networks to learn or adapt effectively to new data after extensive training [Klein et al., 2024, Lyle et al., 2022]. The relationship between policy churn and plasticity loss remains poorly understood. Frequent policy changes may reflect ongoing adaptation during training, while reduced plasticity may conversely suppress policy variability by limiting a network’s ability to respond to new learning signals. Understanding how these phenomena interact is therefore important for explaining the learning dynamics of deep RL agents.

To systematically address these issues, we pose and investigate the following research questions:

1. What factors shape the magnitude and temporal behaviour of policy churn during deep RL?
2. To what extent can policy churn act as an implicit exploration mechanism, and how does it compare to explicit deep exploration methods such as Bootstrapped DQN?

1. Introduction

3. What is the interplay between policy churn, plasticity loss, and exploration, and how do they jointly affect learning efficiency?
4. How do plasticity-preserving methods, such as Shrink & Perturb and Layer Normalization, influence policy churn?

We investigate these questions through a comprehensive empirical study using Double DQN [van Hasselt et al., 2015] across MiniGrid and Atari environments. Our analysis combines direct exploration metrics with post-hoc diagnostics of plasticity. We find that policy churn is an ubiquitous phenomenon in value-based RL and can contribute to exploratory behaviour in simple environments. Despite this, churn alone does not reliably substitute for structured deep exploration in more challenging tasks, and we do not observe a consistent advantage in dedicated deep-exploration benchmarks. We show that policy churn is closely linked to plasticity-related properties of the network: interventions that preserve plasticity substantially reshape churn dynamics and can improve learning stability and, in some cases, long-term performance. Taken together, our results indicate that policy churn is better interpreted as a signal of the underlying learning dynamics rather than as an independent exploration mechanism.

2. Background

2.1. Reinforcement Learning Basics

Reinforcement Learning (RL) is commonly formalized as a Markov Decision Process (MDP), defined as a tuple $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ denote the sets of states and actions, P the state transition probabilities, R the reward function, and γ a discount factor determining the present value of future rewards. In this framework, agents learn policies $\pi : \mathcal{S} \rightarrow \mathcal{A}$ that maximize cumulative rewards [Sutton and Barto, 2018].

Over time, several approaches have been developed to train RL agents. Value-based methods, such as DDQN [van Hasselt et al., 2015], estimate state-action values $Q(s, a)$ and have proven particularly effective for discrete action spaces. A key challenge across all paradigms is the *exploration-exploitation tradeoff*: agents must balance gathering new information (exploration) with leveraging known rewards (exploitation).

2.2. Policy Churn

Policy churn quantifies rapid changes in an agent’s greedy policy $\pi_t(s) = \arg \max_a Q_t(s, a)$ during training. Formally, per-state churn in a discrete setting is:

$$W(\pi_t, \pi_{t-T}|s) = \mathbb{I}[\arg \max_a Q_t(s, a) \neq \arg \max_a Q_{t-T}(s, a)], \quad (2.1)$$

with average churn $\overline{W} = \mathbb{E}_s[W(s)]$ [Schaul et al., 2022]. Here, T denotes the lag in parameter updates between two policy snapshots. Deep RL agents exhibit $\overline{W} \approx 0.1$ per update—in large-scale training regimes such as Atari, where agents may perform on the order of 10^7 parameter updates, this implies roughly 10^6 greedy-action changes per state over the course of training [Schaul et al., 2022]. Crucially, churn persists even after performance plateaus, suggesting it is not merely a byproduct of learning.

2.3. Double DQN and Bootstrapped DQN

Deep Q-Networks (DQN) [Mnih et al., 2013] approximate the action-value function

$$Q^\pi(s, a) = \mathbb{E} \left[\sum_{t=0}^{\infty} \gamma^t r_t \mid s_0 = s, a_0 = a, \pi \right] \quad (2.2)$$

using a neural network trained via temporal-difference learning. The original DQN algorithm suffers from overestimation bias: the same network is used both to select and

2. Background

evaluate actions when computing the target values. This causes systematic overestimation of Q-values, which can degrade stability and lead to suboptimal convergence.

Double DQN (DDQN) [van Hasselt et al., 2015] addresses this issue by decoupling action selection from action evaluation. Specifically, DDQN maintains an online network Q_θ and a periodically updated target network Q_{θ^-} .

Instead of using the target network both to select and evaluate an action, DDQN computes the target as:

$$y = r + \gamma Q_{\theta^-} \left(s', \arg \max_{a'} Q_\theta(s', a') \right) \quad (2.3)$$

where the online network Q_θ selects the maximizing action and the target network Q_{θ^-} evaluates it. This decoupling significantly reduces the overestimation bias present in standard DQN.

Because policy churn reflects changes in the greedy policy $\pi(s)$, its magnitude is highly sensitive to noise in the Q-value landscape. DDQN therefore provides a cleaner and more controlled training signal than standard DQN, helping to isolate churn arising from the agent’s learning dynamics rather than artifacts of Q-value overestimation.

Bootstrapped DQN [Osband et al., 2016] extends the standard DQN framework by maintaining an ensemble of value functions instead of a single Q-network. Each ensemble member, or head, represents a different estimate of the action-value function.

Let Q_{θ_k} denote the k -th head with parameters θ_k , for $k = 1, \dots, K$. The network consists of a shared feature backbone and multiple independent output layers (Figure 4.2c). At the beginning of each episode, Bootstrapped DQN selects a single head k uniformly at random and follows the greedy policy

$$\pi_k(s) = \arg \max_a Q_{\theta_k}(s, a). \quad (2.4)$$

According to Osband et al. [2016], this induces *deep exploration*. Unlike dithering strategies such as ϵ -greedy, which perturb behaviour through independent random actions at each step, deep exploration relies on coherent exploration over multiple time steps. This allows the agent to commit to exploratory trajectories, such as following an unexplored branch of a maze, even when the intermediate actions are neither immediately rewarding nor immediately informative. Deep exploration is particularly important in environments with sparse or delayed rewards, where effective learning may require sequences of coordinated actions before any useful feedback is observed. By committing to a single head for an entire episode, Bootstrapped DQN produces this form of exploration [Osband et al., 2016].

During training, transitions are sampled with bootstrap masks, allowing each head to be trained on a slightly different subset of data. For a transition (s, a, r, s') and discount factor γ , the target for head k is

$$y_k = r + \gamma \max_{a'} Q_{\theta_k^-}(s', a'), \quad (2.5)$$

where θ_k^- denotes the target parameters of head k .

2.4. Measures

2.4.1. Policy Churn

Policy churn measures how frequently an agent’s greedy action changes during training. In this thesis, churn is quantified using the definition introduced in Section 2.2. To obtain a scalar metric, the per-state churn indicator is averaged over a batch of states.

The average churn over a batch of states $\mathcal{B}$ is:

$$\overline{W} = \frac{1}{|\mathcal{B}|} \sum_{s \in \mathcal{B}} W(\pi_t, \pi_{t-T} | s) \quad (2.6)$$

2.4.2. Plasticity Measures

Plasticity refers to a neural network’s ability to incorporate new learning signals during training. Measuring this property directly is difficult, and no universally accepted metric exists. Instead, empirical proxy measures are commonly used to estimate how easily a network can adapt its parameters and internal representations [Lyle et al., 2022]. In this work, we consider three complementary proxies: Target Fitting Capacity, gradient rank, and the fraction of dormant neurons.

Target Fitting Capacity

Target Fitting Capacity (TFC) has emerged as an informative empirical proxy for plasticity in deep RL, originally proposed by Lyle et al. [2022]. The measure quantifies how effectively a trained network can adapt its parameters to fit a randomly generated target function within a limited optimization budget.

Formally, let Q_θ denote the current Q-network and $Q_{\tilde{\theta}}$ a randomly initialized target network. Given a dataset of states $\mathcal{D} = \{s_i\}_{i=1}^N$, the target-fitting objective is

$$\mathcal{L}_{\text{TFC}}(\theta; \tilde{\theta}, \mathcal{D}) = \frac{1}{N} \sum_{i=1}^N \|Q_\theta(s_i) - Q_{\tilde{\theta}}(s_i)\|^2. \quad (2.7)$$

In practice, this loss is minimized for a fixed number of gradient steps starting from the trained network, and the final error is used as the TFC measure.

Low final error indicates that the network can easily reshape its representations to fit the new target function, implying retained plasticity. Conversely, a high residual error suggests that training has reduced the network’s ability to adapt, indicating plasticity loss.

While TFC is not a perfect measure of plasticity, it aligns well with intuitive and theoretical understandings of representational flexibility. It can reveal plasticity limitations that are otherwise difficult to observe directly, such as over-stabilization of weights or a loss of feature diversity in deep layers [Lyle et al., 2022].

2. Background

Gradient Rank

Many theoretically motivated measures, such as the effective rank of the gradient covariance matrix [Lyle et al., 2024, 2022, Kumar et al., 2021], quantify how many meaningful update directions remain available to the network.

In principle, this quantity can be derived from the covariance matrix of per-sample gradients. In practice, computing the effective rank of the gradient covariance matrix can be numerically unstable in large neural networks. To obtain a more stable metric while still capturing the diversity of update directions, we therefore compute the rank of the gradient matrix directly. Let $\ell(\theta; \tau_i)$ denote the temporal-difference loss of the i -th transition $\tau_i = (s_i, a_i, r_i, s'_i)$.

For each transition τ_i , we compute the normalized gradient

$$\tilde{g}_i = \frac{\nabla_{\theta} \ell(\theta; \tau_i)}{\|\nabla_{\theta} \ell(\theta; \tau_i)\|_2}, \quad G = \begin{bmatrix} \tilde{g}_1^{\top} \\ \vdots \\ \tilde{g}_B^{\top} \end{bmatrix} \in \mathbb{R}^{B \times P}. \quad (2.8)$$

Given a batch of B transitions, the normalized gradients are stacked row-wise into the matrix G , where P denotes the number of trainable parameters of the network.

The *gradient rank* is then estimated using the singular values $\sigma_j(G)$ obtained from a singular value decomposition of G . Specifically, we compute the smallest k such that the cumulative energy of the top k singular values explains at least $(1 - \epsilon_{\text{rank}})$ of the total energy:

$$\text{GradRank}(G) = \min \left\{ k : \frac{\sum_{j=1}^k \sigma_j^2}{\sum_{j=1}^B \sigma_j^2} \geq 1 - \epsilon_{\text{rank}} \right\}, \quad (2.9)$$

where ϵ_{rank} is a small threshold parameter.

Intuitively, this metric measures the effective dimensionality of gradient updates within a batch. While small changes in this gradient rank may not have a clear theoretical interpretation, large-scale trends remain informative. A high gradient rank indicates that gradients vary across samples and span many directions in parameter space, which is consistent with higher plasticity. Conversely, a low gradient rank indicates that gradients are aligned or confined to a low-dimensional subspace, suggesting reduced learning flexibility and potential plasticity loss.

Dormant Neurons

A third perspective on plasticity stems from analysing *dormant neurons*, i.e., units whose activations remain consistently low across the training distribution. Dormant neurons were introduced by Sokar et al. [2023] as a measure of reduced network expressivity and were shown to correlate with performance plateaus and instability in DQN agents on Atari benchmarks.

Following Sokar et al. [2023], we compute a normalized activation score for each neuron i in each non-output layer l :

$$s_i^l = \frac{\mathbb{E}_{x \sim D} [|h_i^l(x)|]}{\frac{1}{H_l} \sum_{k=1}^{H_l} \mathbb{E}_{x \sim D} [|h_k^l(x)|]}, \quad (2.10)$$

where D is the replay-buffer distribution, $h_i^l(x)$ denotes the post-nonlinearity activation of neuron i , and H_l is the number of units in layer l . A neuron is declared τ -dormant if $s_i^l \leq \tau$. Letting H_τ^l denote the number of τ -dormant neurons in layer l , the overall fraction of dormant neurons across all hidden layers is defined as

$$\beta_\tau = \frac{\sum_l H_\tau^l}{\sum_l H_l}. \quad (2.11)$$

A *high* value of β_τ indicates that many neurons contribute little to the forward pass, signalling reduced representational flexibility and plasticity loss. Conversely, a *low* value of β_τ suggests that the network maintains a broad set of active units capable of adapting to new learning signals.

2.4.3. Exploration

State Visitation Entropy

State visitation entropy (*Shannon entropy*) is a common measure of exploration in discrete state spaces. It requires the state space to be enumerable and is therefore suitable for gridworld environments but not for high-dimensional domains such as Atari. Let $p(s)$ denote the empirical probability of visiting state s during training. The entropy of the visitation distribution is defined as

$$H(S) = - \sum_{s \in \mathcal{S}} p(s) \log p(s). \quad (2.12)$$

Entropy is maximized when the agent visits all grid cells uniformly and minimized when its behaviour concentrates on a small subset of states.

2.5. Plasticity Methods

Plasticity loss in deep RL can be mitigated through architectural and algorithmic interventions [Klein et al., 2024]. In this work, we focus on three commonly used approaches: soft and hard non-targeted weight resets, and Layer Normalization.

Shrink & Perturb

Shrink and Perturb (S & P), also referred to as *soft resets*, provide a simple mechanism to restore plasticity without discarding all accumulated knowledge [Ash and Adams, 2020].

2. Background

Instead of fully reinitializing a network, S & P blends the current parameters θ_{old} with freshly initialized weights ψ drawn from the model’s original initializer:

$$\theta_{\text{new}} = \alpha * \theta_{\text{old}} + (1 - \alpha) * \psi, \quad (2.13)$$

where $\alpha \in (0, 1)$ controls how strongly the network is pulled back toward its initial state. A larger α preserves more of the existing representation, while smaller values induce a stronger reset.

S & P reintroduces a small amount of fresh initialization noise into the network parameters. This can reactivate neurons that have become inactive or overly specialized, restoring the ability of the network to respond to new gradient signals while preserving much of the previously learned representation [Ash and Adams, 2020].

Weight Resets

Periodic weight resets, often referred to as *hard resets*, provide a direct mechanism for counteracting plasticity loss in deep RL [Nikishin et al., 2022]. The core idea is simple: at fixed intervals, selected layers of the network are reinitialized using the original weight distribution. This strategy is motivated by the observation that plasticity loss is most pronounced in the later layers of deep RL agents, where representations become overly specialized and gradients diminish [Berariu et al., 2023]. Resetting these layers periodically allows the network to incorporate new information that would otherwise be ignored.

Layer Normalization

Layer Normalization (LayerNorm) [Ba et al., 2016] is applied before the nonlinearities of each hidden layer and has been identified as a reliable tool for stabilizing gradient flow in deep RL. Recent analyses show that LayerNorm is one of the most effective architectural modifications for mitigating plasticity loss [Lyle et al., 2022]. By normalizing the per-layer activation statistics, LayerNorm keeps pre-activations within a moderate range, reducing the risk of large parameter updates.

Formally, given a pre-activation vector a^l from layer l , LayerNorm normalizes it using its mean and variance:

$$a^l \leftarrow \frac{a^l - \mathbb{E}[a^l]}{\sqrt{\text{Var}[a^l] + \epsilon}} \cdot \gamma + \beta, \quad (2.14)$$

where γ and β are optional learnable scale and bias terms, and $\epsilon > 0$ ensures numerical stability. In this thesis, Layer Normalization is applied without learnable affine parameters, so only the normalization operation itself is used (see Section 4.5). Normalizing before the activation function stabilizes the distribution entering the nonlinearity, preventing the drift into saturated or low-gradient regimes that often lead to plasticity loss.

3. Related Work

Deep Exploration in Reinforcement Learning

Efficient exploration is a central challenge in reinforcement learning, especially in environments with sparse or delayed rewards [Sutton and Barto, 2018, Ladosz et al., 2022]. In such settings, simple dithering strategies such as ϵ -greedy often fail to produce temporally coherent exploration and can lead to slow learning [Osband et al., 2016]. This has motivated research on *deep exploration*, where agents commit to structured exploratory behaviour over multiple steps.

A classical approach to principled exploration is Thompson sampling [Thompson, 1933]. While it is well understood in bandit problems, applying it directly in deep RL is typically impractical. Bootstrapped DQN [Osband et al., 2016] provides a scalable approximation by training an ensemble of value-function heads. Sampling a head per episode induces consistent exploration policies over extended horizons. Empirically, Bootstrapped DQN can substantially outperform ϵ -greedy on exploration-heavy problems [Osband et al., 2016, Mou et al., 2020]. In this thesis, we use Bootstrapped DQN as a reference method for deep exploration.

Policy Churn and Implicit Exploration

Schaul et al. [2022] identify *policy churn* as a persistent phenomenon in value-based deep RL. They show that greedy policies can change rapidly during training, and that these changes can persist even after performance has saturated. Importantly, policy churn is not simply a consequence of adding explicit exploration noise, since it remains present even under reduced or disabled ϵ -greedy exploration.

Policy churn has been observed across different value-based algorithms and settings. Schaul et al. [2022] report substantial churn in Double DQN and in distributed agents such as R2D2 [Kapturowski et al., 2019], suggesting that it is not tied to a specific implementation detail. They further argue that persistent greedy-policy changes could act as an implicit driver of exploration by changing the agent’s behaviour over time [Schaul et al., 2022]. However, existing work does not provide a controlled comparison between churn-driven exploration and explicit deep exploration methods such as Bootstrapped DQN. This leaves open to what extent churn can substitute for directed exploration, particularly in environments that require temporally extended commitments.

Beyond exploration, policy churn has also been connected to training instability. Tang and Berseth [2024] describe a mechanism where fluctuations in Q-values induce policy changes that further perturb learning, potentially amplifying errors. These findings suggest that churn may have both beneficial and harmful effects, depending on the training

3. Related Work

regime. This motivates analysing churn jointly with learning dynamics rather than treating it as purely an exploration signal.

Plasticity Loss in Deep Reinforcement Learning

A separate line of work studies *plasticity loss*, i.e., the tendency of neural networks in deep RL to lose the ability to adapt during long training runs [Lyle et al., 2022, Berariu et al., 2023]. Plasticity loss can manifest as increasingly ineffective updates, reduced diversity in learning signals, and representational collapse, which can limit continued improvement even when training continues.

Lyle et al. [2022] analyse how update effectiveness can degrade over time, and Berariu et al. [2023] investigate mechanisms that can reduce learning flexibility in neural networks. Klein et al. [2024] provide a broader overview of plasticity loss in deep RL and discuss how low-plasticity regimes may reduce an agent’s ability to adjust its policy. Notably, the survey points to a potential link between reduced plasticity and reduced policy variability, but does not empirically test this connection.

To mitigate plasticity loss, several interventions have been proposed. These include perturbation-based approaches such as Shrink & Perturb [Ash and Adams, 2020], periodic weight resets [Nikishin et al., 2022], and normalization methods such as Layer Normalization [Ba et al., 2016]. Prior work shows that these techniques can improve stability or prolong effective learning, but their impact on policy churn has not been systematically studied.

Positioning and Research Gap

In summary, existing research has largely studied exploration mechanisms and plasticity loss separately. Systematic analyses of policy churn within standard value-based algorithms, particularly under varying plasticity conditions, remain scarce. As a result, it is still unclear to what extent churn itself can function as an effective exploration mechanism when compared to established approaches designed specifically for deep exploration.

Furthermore, previous work has not provided a controlled comparison between churn-driven exploration and Bootstrapped DQN, nor has it systematically examined how policy churn relates to network capacity and overall learning performance. The interaction between policy churn and interventions intended to preserve plasticity has also received little attention. This thesis addresses these gaps by jointly analysing policy churn, exploration behaviour, and plasticity loss through a series of controlled experiments conducted in both MiniGrid and Atari environments.

4. Methodology

4.1. Overview

The overarching goal of this thesis is to understand how policy churn, plasticity loss, and exploration interact and how these dynamics affect the learning efficiency of deep reinforcement learning agents. All experiments presented in this chapter examine these relationships across different environments, algorithmic variants, and plasticity interventions. The methodology therefore follows a research-question driven structure. We first describe the environments chosen for studying exploration and plasticity. We then present the experimental frameworks used to investigate whether policy churn enables deep exploration and how plasticity methods affect learning dynamics. Finally, we outline the analysis procedures used to interpret performance, exploration behaviour, and plasticity.

A central quantity underlying all subsequent experiments and analysis is **policy churn**. Before introducing the specific experimental setups and interventions, we therefore first describe how churn is defined and computed in our framework.

4.2. Policy Churn

Policy churn compares the greedy policies induced by two sets of network parameters when evaluated on the same set of states (see Formula 2.1). In our experiments, we use the periodically updated target network as a stable reference policy. Because the target network remains fixed between updates, it provides a parameter snapshot θ_{t-T} against which changes in the online network can be measured.

After each target update, we store the target parameters and treat this snapshot as the reference policy π^- . All churn metrics compare the greedy policy of the current online network to this same reference. We deliberately employ hard target network updates rather than soft updates, as this ensures that the reference policy remains constant over the entire measurement interval and therefore yields consistent churn estimates.

Based on this reference snapshot, churn is evaluated using the following procedure, which balances computational efficiency with measurement accuracy. At each churn evaluation step,

1. a fixed set of states (512) is sampled from the replay buffer,
2. both the online and target networks are switched to evaluation mode to disable stochastic effects such as dropout or batch normalization,
3. greedy actions are obtained by applying *arg max* to the predicted Q-values, and

4. Methodology

4. the fraction of sampled states for which the selected actions differ is recorded as the churn measure.

To capture changes at different temporal scales, we evaluate the churn measure for different lag values T :

- W_1 : the fraction of states for which the greedy action of the online network differs from that of the target snapshot after 1 update
- W_{10} : the same comparison, but after 10 updates of the online network
- W_{100} : the same comparison, but after 100 updates of the online network

Thus, all three measures use the target network Q_{θ^-} as a stable reference; only the online network evolves between W_1 , W_{10} , and W_{100} . This allows us to quantify how rapidly the agent’s greedy policy diverges across short, medium, and longer time horizons within each target update cycle. This measurement approach captures the *empirical* policy churn on the current training distribution, as states are drawn from the replay buffer rather than from a fixed evaluation set.

4.3. Environments

This section addresses the question of *what* environments were used and *why* they were selected. We chose two complementary environment classes to examine the interplay between exploration, churn, and plasticity: MiniGrid, which provides a controlled test-bed for analysing exploration behaviour directly, and Atari, which offers complex, high-dimensional benchmarks where plasticity loss is known to emerge.

MiniGrid Environments

MiniGrid provides lightweight gridworld environments designed for reinforcement learning research. The agent operates in a discrete two-dimensional grid and must complete navigation tasks such as reaching a goal location. These environments are particularly useful for studying exploration because the dynamics are simple, fully observable, and allow direct inspection of agent behaviour [Chevalier-Boisvert et al., 2023].

Formally, each environment defines a Markov Decision Process (S, A, P, R, γ) . The state space S consists of the agent’s grid observation, which encodes nearby tiles as discrete features describing object type, color, and state. The action space A is discrete with seven possible actions: `turn-left`, `turn-right`, `move-forward`, `pickup`, `drop`, `toggle`, and `done`. In the navigation tasks considered here, behaviour relies only on the first three actions. The reward function R is sparse: the agent receives a positive reward when reaching the green goal square, scaled by the remaining episode length, and zero reward otherwise. Episodes terminate when the goal is reached or a step limit is exceeded [Chevalier-Boisvert et al., 2023].

We used two MiniGrid environments that differ in exploration difficulty. The first environment, **MiniGrid-Empty-16x16-v0** (Figure 4.1a), consists of a large empty grid in which the agent must navigate to a fixed goal location. The agent always starts in the corner opposite the goal, allowing learning dynamics to be studied in the absence of structural exploration challenges. Larger variants of this environment are commonly used to analyse exploration behaviour under sparse rewards.

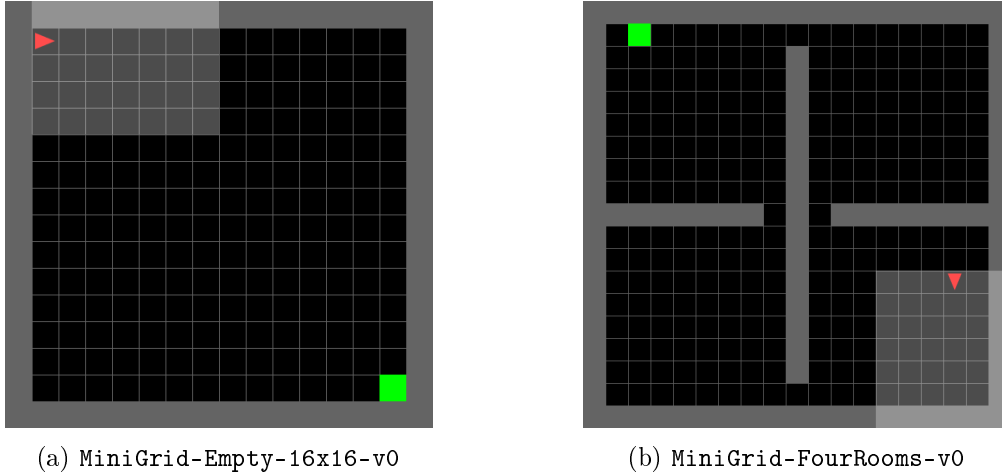


Figure 4.1.: MiniGrid environments used in this study. The agent (red triangle) must navigate to the goal (green square). **Empty-16x16** provides an open navigation task with deterministic spawn locations, while **FourRooms** introduces structural bottlenecks through four interconnected rooms.

To study more challenging exploration, we additionally used **MiniGrid-FourRooms-v0** (Figure 4.1b). This environment divides the grid into four rooms connected by narrow passages. Both the agent and the goal are randomly placed in different rooms at the start of each episode, requiring the agent to discover doorways and navigate across multiple rooms before reaching the goal. Such bottleneck structures make exploration significantly more challenging and require sustained exploratory behaviour across longer trajectories. We refer to environments with such layout and long-horizon navigation requirements as hard-exploration environments.

Together, these environments provide a controlled setting for analysing how policy churn interacts with exploration behaviour under different structural exploration challenges.

Atari Environments

The second class of environments consists of Atari games from the Arcade Learning Environment (ALE), a widely used benchmark suite for evaluating deep reinforcement learning agents. Atari environments provide high-dimensional visual observations and long-horizon decision-making tasks, making them suitable for studying optimization dy-

4. Methodology

namics and plasticity during extended training.

Each Atari game defines a Markov Decision Process (S, A, P, R, γ) . The state space S consists of image observations of the game screen. Following the standard DQN preprocessing pipeline, observations are converted to grayscale, resized to 84×84 pixels, and stacked across four consecutive frames to capture short-term motion. The action space A is discrete and corresponds to joystick movements and button presses defined by the ALE interface. The reward function R is derived from the game score and provides scalar rewards whenever the agent performs score-increasing actions, such as defeating enemies or collecting points.

We focused on two games with complementary learning dynamics: *Phoenix* and *Qbert*. *Phoenix* is known to exhibit strong plasticity loss in deep reinforcement learning agents, where training performance often plateaus despite continued updates [Nikishin et al., 2023]. Its learning dynamics therefore provide a natural testbed for studying how policy churn interacts with plasticity degradation over long training horizons. *Qbert*, in contrast, is commonly used as a verification environment where value-based agents can achieve competitive performance. It is classified as a hard exploration environment with dense rewards [Bellemare et al., 2016], making it suitable for analysing exploration behaviour under different reward structures.

4.4. Algorithms and Baselines

As a baseline, we trained a *standard DDQN* agent with ϵ -greedy exploration and a small final exploration rate of $\epsilon = 0.01$ on all environments. This agent represents the conventional form of random-action exploration through dithering. By evaluating churn, exploration, and plasticity metrics for this widely used baseline, we obtained a reference point against which all other variants can be meaningfully compared. To study the relationship between policy churn and exploration, we implemented a *reduced-churn* agent following Schaul et al. [2022]. Instead of selecting actions with the online network, this agent acts solely with the target network. Since the target network is updated less frequently, its greedy policy changes more slowly, which should result in lower policy churn. This variant allowed us to test whether reducing churn suppresses exploration and how such suppression affects learning performance in both MiniGrid and Atari. To isolate churn as the dominant source of exploration, we implemented an $\epsilon = 0$ DDQN agent. This agent follows a standard ϵ -greedy exploration schedule during the initial phase of training but anneals ϵ to zero, such that no random actions are taken in the later stages of learning. Once ϵ has reached zero, the agent always selects the greedy action with respect to its current Q-network, forcing exploration to arise exclusively from endogenous policy changes. Prior work showed that such agents can still achieve non-trivial performance in several Atari games [Schaul et al., 2022], suggesting that policy churn induced by stochastic optimization and function approximation can itself drive exploration. In this thesis, the $\epsilon = 0$ agent serves as the canonical *churn-only* exploration baseline. It was evaluated on MiniGrid environments in the main experiments, while additional results on Atari are reported in Appendix A.3 to assess whether churn alone can support deep

Table 4.1.: Overview of agent variants used in the experiments.

Agent	Exploration	Action Policy	Architecture
DDQN (baseline)	$\epsilon \rightarrow 0.01$	Online network	Single-head
Reduced-churn DDQN	$\epsilon \rightarrow 0.01$	Target network	Single-head
$\epsilon = 0$ DDQN	$\epsilon \rightarrow 0$	Online network	Single-head
Bootstrapped DQN	Bootstrapped	Random head per episode	Multi-head ($k = 10$)

or long-horizon exploration.

Deep Exploration Baseline

As a benchmark against a method explicitly designed for deep exploration, we included *Bootstrapped DQN (BDQN)* [Osband et al., 2016]. BDQN maintains $k = 10$ heads that share lower-level features but differ in their final layers and in the data each head trains on (Figure 4.2c). At the beginning of every episode, one head is chosen and followed throughout that episode, which induces temporally extended and structured exploration. BDQN is known to achieve significantly faster exploration than ϵ -greedy methods in several environments and improves performance across many Atari games. In this work, BDQN serves as a *deep-exploration* reference method against which the behaviour of churn-driven agents is compared. Because BDQN maintains multiple heads, churn could not be measured directly in the same way as for single-head networks. To ensure comparability, we implemented a milestone-based tracking system that records how often each head is updated. Churn metrics such as W_1 , W_{10} , and W_{100} were then computed by aligning comparisons with these per-head milestones so that each head’s policy changes were measured over approximately consistent numbers of parameter updates. This procedure ensured that churn measurements for BDQN were meaningfully aligned with single-head architectures, such as the DDQN variants. A summary of all agent variants is provided in Table 4.1.

Architectural Variations

We tested whether policy churn is driven by network capacity using a controlled architectural manipulation in which the overall network structure was kept fixed, while only the width of the later layers was varied. To avoid architectural bottlenecks and to isolate the effect of capacity, we used symmetric width scaling. Table 4.2 summarizes the conducted experiments. In MiniGrid, the original hidden layer width of 120 units (Figure 4.2a) was replaced by symmetric widths of 64, 128, 256, and 512 units for both hidden layers of the two-layer MLP. In Atari, we varied the width of the single fully connected hidden layer in the Q-head around the original size of 512 units (Figure 4.2b), using widths of 128, 256, 512, and 1024 units, while leaving the convolutional feature extractor unchanged. This design allowed us to study how network capacity influences churn dynamics without introducing confounding changes to optimization or representation structure.

4. Methodology

Table 4.2.: Network width variations used in the capacity experiments.

Environment	Component	Layers Affected	Baseline Width	Tested Widths
MiniGrid	MLP hidden layers	2	120	64, 128, 256, 512
Atari	Fully connected Q-head	1	512	128, 256, 512, 1024

4.5. Plasticity Interventions

Plasticity Interventions

To investigate how plasticity-preserving methods affect policy churn, we implemented three distinct interventions. We evaluated these interventions in both MiniGrid and Atari environments to observe how maintaining plasticity influenced exploration and policy churn. All interventions were applied without modifying the exploration mechanism. Their experimental instantiation, including the chosen hyperparameters, is described here, while Section 2.5 provides the conceptual background and motivation.

Shrink and Perturb is applied every 100,000 steps in MiniGrid and every 500,000 steps in Atari using a blending coefficient of $\alpha = 0.8$. Each parameter is replaced by a convex combination of its current value and a newly sampled value drawn from the network’s original initialization distribution. After this perturbation, the target network is synchronized with the perturbed online network, meaning that all parameters of the online network are copied to the target network via a hard update, to maintain consistency in the TD targets.

Hard weight resets are applied every 100,000 steps in MiniGrid and every 2,000,000 steps in Atari. The online network is fully reinitialized using the original weight distribution, and a new optimizer is created to ensure that all internal states, such as momentum buffers, match the fresh initialization. The target network is then synchronized to maintain consistent behaviour during learning. While MiniGrid environments do not exhibit pronounced plasticity loss and would therefore achieve their best performance with one or no resets, we retain the 100,000-step reset interval to explicitly study the effect of hard resets on policy churn. To avoid perturbing the final evaluation phase, the last scheduled reset in each run is omitted. This omission is applied only to hard resets, since a final full reinitialization would directly perturb the final evaluation phase.

Layer Normalization is applied without learnable affine parameters, following recent work that studies the effect of normalization itself in reinforcement learning [Lyle et al., 2024]. Therefore, only the normalization operation itself is performed and no additional scale or shift parameters are introduced (see Equation 2.14). This reduces parameter complexity and stabilizes activations. In the architectures shown in Figure 4.2, LayerNorm is inserted after the hidden linear layers and before the ReLU activation. For the Atari networks, normalization is applied after the fully connected layer in the Q-value head, whereas for the MiniGrid networks it is applied after each hidden fully connected layer (Figure 4.2a).

A complete overview of all intervention hyperparameters is given in Table A.1.

4.6. Metrics

The experiments are evaluated using a set of metrics covering task performance, plasticity-related properties and churn of the network. Together, these measures provide the basis for comparing agent variants and analysing their learning dynamics.

Performance

Episodic return denotes the total reward accumulated within an episode and serves as the main performance metric in all experiments.

Plasticity

Target Fitting Capacity (TFC) [Lyle et al., 2022] quantifies how easily a network can adapt to a new target function. It serves as an empirical proxy for plasticity by measuring how well a trained network can adapt to a new target function within a fixed optimization budget. At fixed intervals during training, we evaluated TFC using the following four steps. First, we construct a dataset from the first 100,000 states from the replay buffer. Second, we instantiate a random target network by reinitializing all parameters. Third, for each random target network, we initialize a copy of the current Q-network and optimize it for 10,000 minibatch updates, with batch size 32, to minimize $\mathcal{L}_{\text{TFC}}$ defined in Equation 2.7. Fourth, we repeat the process for ten random seeds and report the average final MSE. Since the target network is randomly initialized, this procedure measures pure adaptation capacity, independent of task structure.

Gradient rank measures the diversity of gradient directions induced by different transitions within a batch. When gradients span many directions in parameter space, the network can update its parameters flexibly; when they become aligned, updates collapse into a low-dimensional subspace. For each batch, we construct a gradient matrix $G \in \mathbb{R}^{B \times P}$, where B is the batch size and P the number of parameters. Each row contains the normalized gradient of the TD loss for a single transition. We estimate the numerical rank using an SVD-based method, which is more stable than eigenvalue-based criteria. In our experiments, the gradient matrix is constructed from a batch of 64 transitions ($B = 64$), so the maximum possible rank is 64.

Dormant fraction captures how much of the network becomes effectively inactive during training. For each layer, we compute normalized mean activations across the batch. A neuron is classified as dormant if its normalized activation falls below $\tau = 0.1$, following Sokar et al. [2023]. The dormant fraction is then averaged across all hidden layers.

These metrics jointly capture the network’s adaptability, the diversity of gradient information, and the extent of representational collapse. For more details, see Section 2.4.2. In this thesis, we used dormant fraction and gradient rank as the primary plasticity diagnostics, as they are stable and comparable across environments and runs. Target fitting capacity (TFC) was used as an auxiliary measure in selected time-series plots to illustrate target adaptation behaviour under interventions. However, TFC was not used as

4. Methodology

a primary variable in the cross-run correlation analysis. Measurement frequencies and related analysis settings are summarized in Table A.1.

Diagnostic Analysis

This section presents a post-hoc diagnostic analysis based on data collected from the plasticity intervention experiments. The goal was to assess redundancy and interdependence between policy churn and standard learning and plasticity signals.

As a first step, we visualize pairwise relationships between exploration, plasticity, and performance metrics using scatter plots. These plots provide an initial assessment of the direction and form of co-variation and allow us to verify whether linear relationships are a reasonable assumption.

For each scatter plot, we additionally overlay linear regression lines obtained by ordinary least squares. Unless stated otherwise, these fits are estimated separately for each experiment group using the corresponding run-level observations, and are shown without confidence intervals. The fitted lines are used only as descriptive summaries of linear trends and not for inferential testing.

Following the visual inspection of pairwise relationships, we quantify statistical associations between metrics by computing correlation matrices based on run-level observations. For each pair of metrics, we estimate the Pearson correlation coefficient r , which measures the strength and direction of their linear association across runs.

Pearson correlation is used because the metrics are numerical run-level summaries, and the goal is to measure the direction and strength of linear co-variation between them. Some metrics are bounded proportions or discrete-valued summaries rather than strictly continuous variables, but they are still measured on ordered numerical scales and are used here descriptively. Correlations are computed on the raw metric values, since Pearson correlation is invariant to separate linear rescalings of the variables.

Before computing correlations, we formulate the following hypotheses about the expected relationships between the metrics. First, we hypothesize that higher plasticity is associated with better performance, because a network that remains adaptable should be better able to incorporate new learning signals throughout training. Accordingly, gradient rank is expected to correlate positively with episodic return, whereas dormant fraction is expected to correlate negatively with return, since a high dormant fraction indicates that a larger proportion of the network contributes little to learning.

We further hypothesize that the plasticity-related metrics should relate to each other in a consistent way: gradient rank and dormant fraction are expected to be negatively correlated, because more diverse gradient directions should coincide with a larger set of active and adaptable neurons.

Finally, we hypothesize that high policy churn reflects instability rather than productive exploration. Although Schaul et al. [2022] show that churn can drive exploration even in the absence of explicit exploration logic, they also observe that substantial churn persists after return has converged. We therefore do not expect high churn as evidence of beneficial exploration per se. Instead, we expect it to reflect instability in the greedy policy, with exploratory effects arising as a byproduct. Under this interpretation, policy

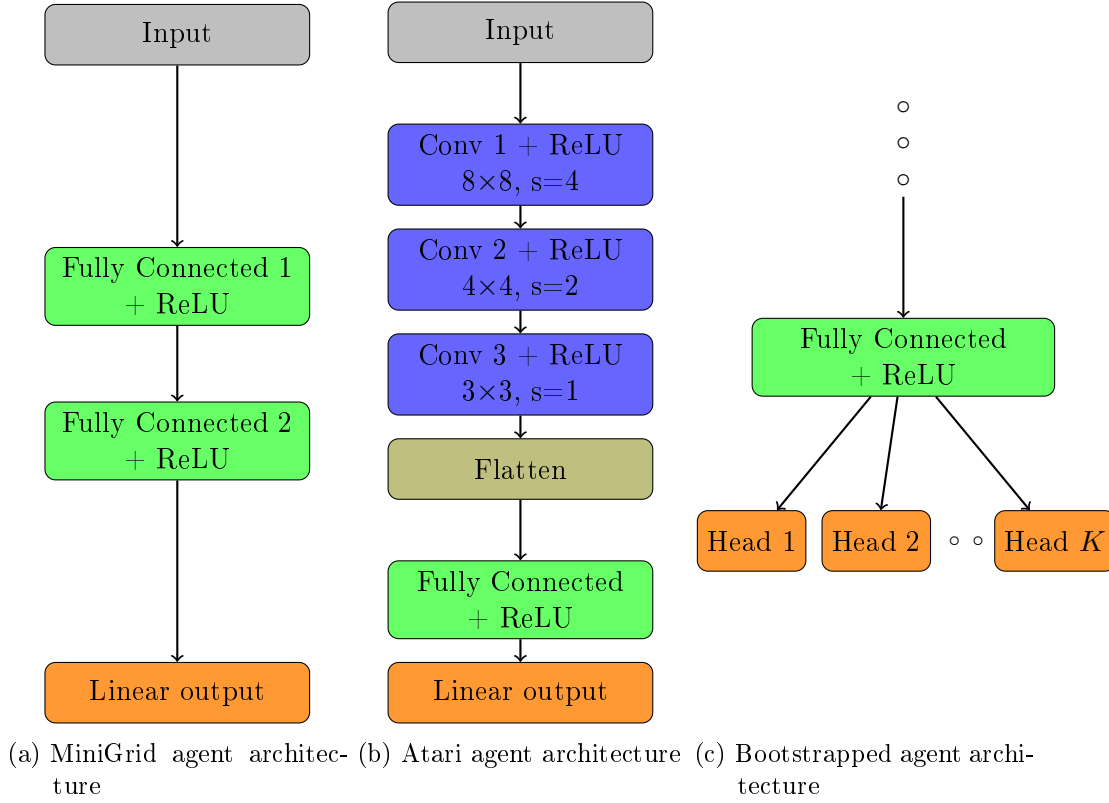


Figure 4.2.: Main network architectures used in the experiments. The MiniGrid agent uses a two-layer MLP, while the Atari agent uses a convolutional feature extractor followed by a fully connected Q-head. For convolutional layers, kernel size and stride are indicated. The Bootstrapped DQN architecture differs only in the output layer, which consists of $k = 10$ separate heads.

churn is expected to correlate negatively with episodic return and with gradient rank, and positively with dormant fraction.

4.7. Experimental Setup

Network Architecture

In our implementation, the network architecture depended on the environment and task. The main DDQN architectures can be found in Figures 4.2a and 4.2b. For MiniGrid, we used a fully connected network operating on flattened observations with two hidden layers and a linear output layer. For Atari, we used the standard DQN convolutional architecture consisting of three convolutional layers followed by a fully connected hidden layer and a linear output layer. The online and target networks share an identical architecture but are updated on different schedules.

4. Methodology

Experimental Protocol

For each environment and each plasticity intervention or agent variant, 10 independent training runs were conducted. The same set of 10 random seeds was reused across methods within an environment, which reduced variance and enabled paired comparisons between algorithms. All Atari experiments were trained for a total of 10,000,000 environment steps, following common practice in the literature. MiniGrid experiments were trained for a total of 500,000 training steps, which was sufficient for all considered agents to either converge or clearly plateau. Detailed hyperparameter settings for all MiniGrid and Atari experiments are provided in Table A.1 in the Appendix.

5. Experiments

This section presents the empirical results of our investigation into policy churn. We first examine whether policy churn can act as an implicit exploration mechanism in simple environments and whether this effect scales to structured, long-term exploration tasks. We then analyse factors influencing the magnitude and behaviour of policy churn, including network capacity and learning dynamics. Finally, we study how plasticity-preserving interventions reshape policy churn and learning behaviour across Atari environments.

5.1. Exploration in Simple Environments

As discussed earlier and shown by Schaul et al. [2022], policy churn can act as an implicit exploration mechanism. We replicate this phenomenon by evaluating several variants of DDQN, described in Section 4.4, in the MiniGrid Empty environment.

Figure 5.1 presents the episodic return, state visitation entropy, and policy churn (W_{10}) for the baseline DDQN, the $\epsilon = 0$ variant, Bootstrapped DQN (BDQN), and the reduced-churn agent. Figure 5.1a indicates that the $\epsilon = 0$ agent occasionally exhibits exploratory behaviour driven solely by policy churn. However, the agent fails to solve the environment consistently, which is reflected in the large variance of its episodic returns. In contrast, the baseline DDQN and BDQN solve the environment relatively quickly, while the reduced-churn variant improves more slowly but approaches the maximum reward towards the end of training.

This seed dependence is further illustrated in Figure 5.2, which depicts the distribution of final episodic returns across seeds for all agent variants. While the $\epsilon = 0$ agent is occasionally able to achieve near-optimal performance, its outcomes are highly variable, ranging from complete failure to maximum reward. In contrast, the baseline DDQN and BDQN exhibit consistently high final returns across seeds, indicating robust learning behaviour. The reduced-churn agent occupies an intermediate regime, with most runs converging to high performance but with occasional lower-performing seeds. This result highlights that exploration driven purely by policy churn is brittle and highly sensitive to random initialization.

Examining the state visitation entropy in Figure 5.1b reveals that all agents display very similar entropy levels. The $\epsilon = 0$ agent has the lowest state visitation entropy, which is expected since it does not employ any explicit exploration mechanism.

The policy churn curves in Figure 5.1c clarify this pattern. The $\epsilon = 0$ agent maintains consistently high churn, highlighting its reliance on endogenous policy changes as a source of exploration. This is especially visible around 250k steps, where the exploration parameter ϵ reaches zero, leading to an increase in churn. In contrast, the other agents

5. Experiments

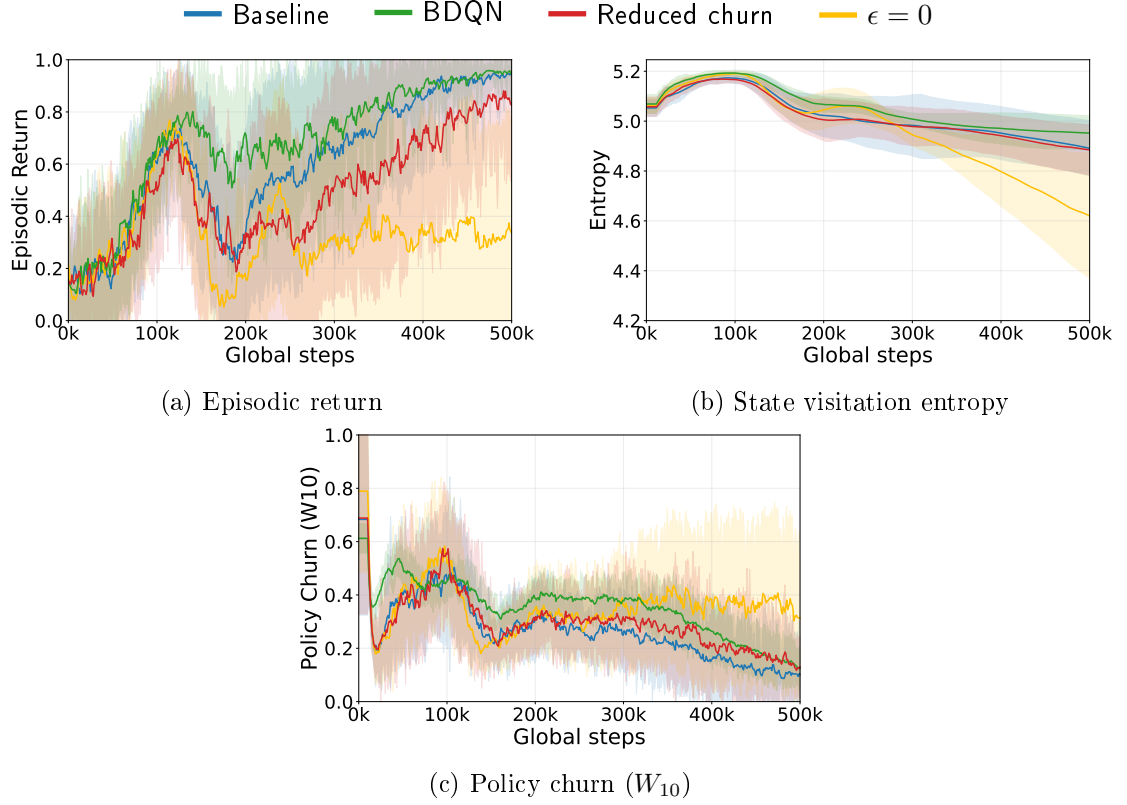


Figure 5.1.: Episodic return, state visitation entropy, and policy churn (W_{10}) in the MiniGrid Empty environment. Baseline DDQN (blue), Bootstrapped DQN (green), reduced churn (red), and $\epsilon = 0$ (yellow).

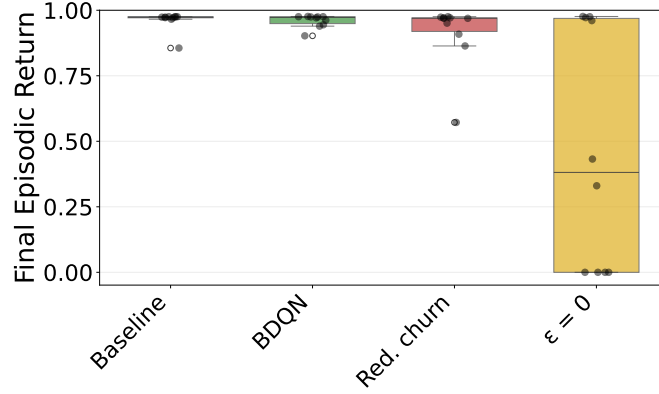


Figure 5.2.: Distribution of final episodic returns across seeds in the MiniGrid Empty environment. Baseline DDQN (blue), Bootstrapped DQN (green), reduced churn (red), and $\epsilon = 0$ (yellow).

5.1. Exploration in Simple Environments

display a decrease in churn over time, stabilizing their policies as learning progresses. By the end of training, the remaining variants converge to similar churn levels. Interestingly, the reduced-churn variant tends to exhibit slightly higher churn than the baseline agent throughout training, although the difference remains small and the error bands overlap.

Overall, these results suggest that policy churn can induce exploratory behaviour in simple environments, but it does not provide a reliable exploration mechanism. In the absence of directed exploration mechanisms, learning remains inconsistent and strongly dependent on random factors such as initialization and seed.

Following the aggregate analysis, we further investigate the variability observed for the $\epsilon = 0$ agent in Figure 5.2. To this end, we group individual runs based on their final episodic return into three categories: *good performance* (final episodic return > 0.75), *medium performance* ($0.25 < \text{final episodic return} \leq 0.75$), and *poor performance* (final episodic return ≤ 0.25).

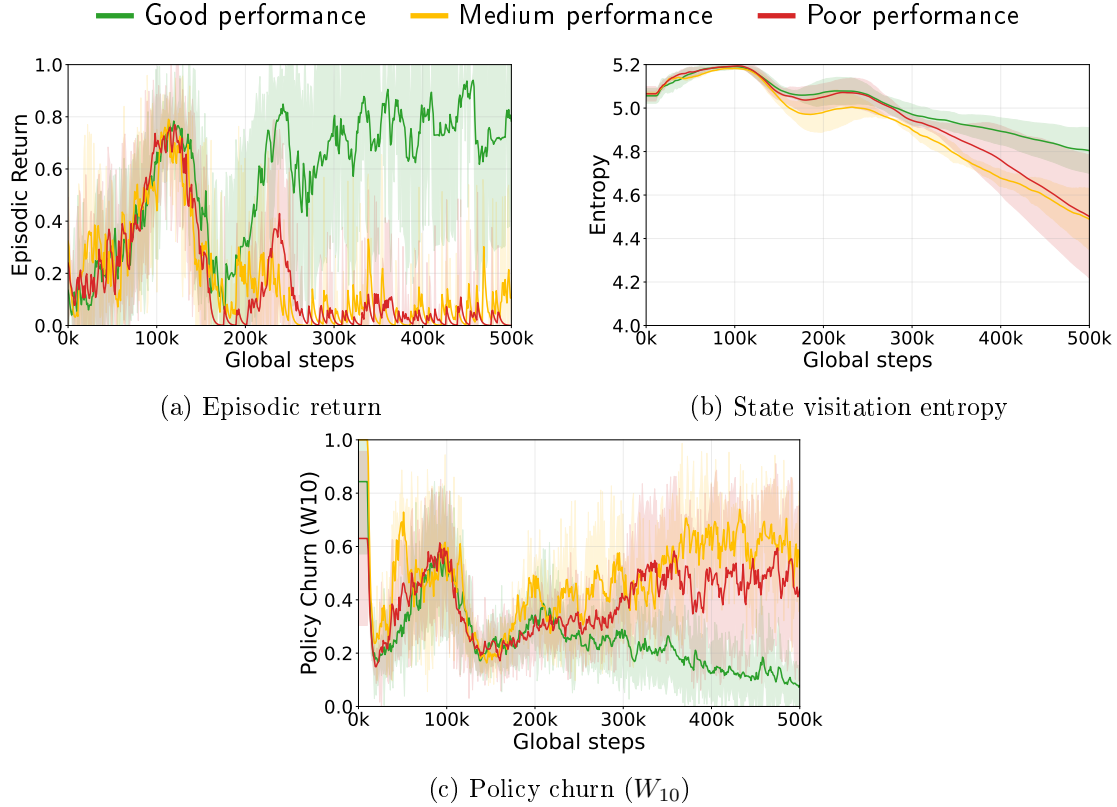


Figure 5.3.: Episodic return, state visitation entropy, and policy churn (W_{10}) of $\epsilon = 0$ agents grouped by final episodic return in the MiniGrid Empty environment. Good performance: final episodic return > 0.75 (green); Medium performance: $0.25 < \text{final episodic return} \leq 0.75$ (yellow); Poor performance: final episodic return ≤ 0.25 (red).

5. Experiments

Figure 5.3 shows the same set of metrics for these three groups. Across all metrics, the groups exhibit very similar behaviour during the early phase of training. Up to approximately 190k steps, no clear separation between the groups is observable. Beyond this point, the trajectories begin to diverge clearly: the high-performing group separates from the other runs, while the medium- and low-performing groups continue to behave similarly.

Two key patterns emerge from this analysis. First, the high-performing group exhibits substantially broader exploration towards the end of training, as reflected by higher state visitation entropy in Figure 5.3b. This state visitation entropy is still lower than that of the other variants in Figure 5.1b. Second, these successful runs are associated with significantly lower policy churn compared to the other groups (Figure 5.3c). In fact, their churn levels are comparable to successful agents in Figure 5.1c.

Having demonstrated that policy churn can induce exploratory behaviour in a simple open-grid environment, we now investigate whether this mechanism also supports exploration in more challenging settings. In particular, we ask whether churn-based exploration remains effective when the environment requires structured exploration through bottlenecks and sustained commitment to longer trajectories.

5.2. Does Policy Churn Drive Deep Exploration?

We study this question in the MiniGrid FourRooms environment, which explicitly tests an agent’s ability to explore through bottlenecks and commit to long-term trajectories. We evaluate the same set of DDQN variants as before, focusing on their exploration behaviour and learning performance.

Figure 5.4 summarizes the results. From the episodic return plot in Figure 5.4a, it is clear that none of the agents, including BDQN, are able to consistently solve the environment within the training budget. It is therefore difficult to identify any evidence of stable learning progress for any of the agents.

Despite none of the agents solving the task reliably, the exploration metrics still reveal clear behavioural differences. The door-crossing metric in Figure 5.4b, which counts agent transitions between rooms, provides a first indication of structured exploration. The baseline DDQN crosses doors more frequently than the other agents, whereas the $\epsilon = 0$ agent exhibits the lowest number of crossings. This suggests that increased exploration activity alone does not ensure that agents discover and traverse bottlenecks.

A similar pattern emerges when considering state visitation entropy (Figure 5.4c). All agents maintain relatively high state visitation entropy levels, with BDQN and the $\epsilon = 0$ agent showing the most diverse state visitation patterns. However, this increased diversity does not translate into higher door-crossing behaviour. Taken together, these results indicate that neither policy churn nor state visitation entropy reliably translates into meaningful exploration of the environment structure.

The policy churn dynamics in Figure 5.4d further clarify this behaviour. In contrast to the Empty environment (Figure 5.1), churn does not decrease over time but remains consistently high or even increases slightly for all agents. This indicates that, in the ab-

5.2. Does Policy Churn Drive Deep Exploration?

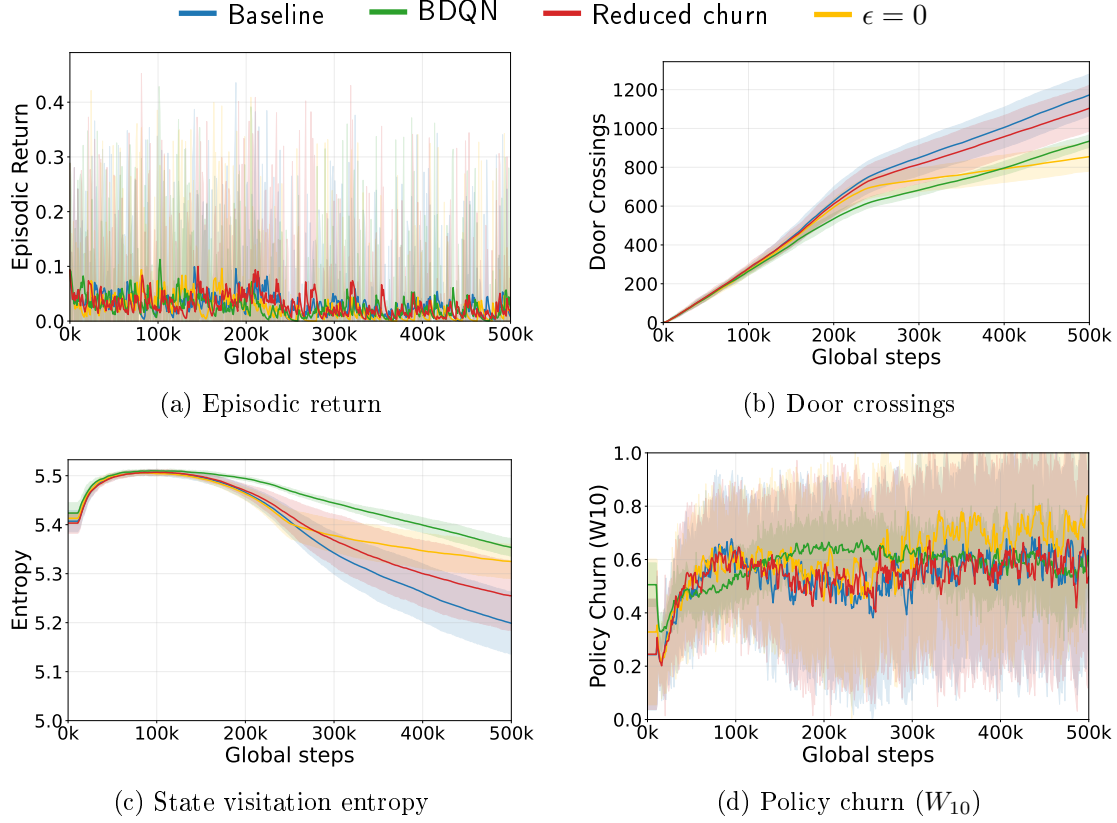


Figure 5.4.: Episodic return, door crossings, state visitation entropy, and policy churn (W_{10}) in the MiniGrid FourRooms environment. Baseline DDQN (blue), Bootstrapped DQN (green), reduced churn (red), and $\epsilon = 0$ (yellow).

5. Experiments

sence of learning progress, policy churn reflects persistent instability rather than effective adaptation.

To gain further insight into these exploration patterns, Figure 5.5 visualizes the state visitation distributions of all agents.

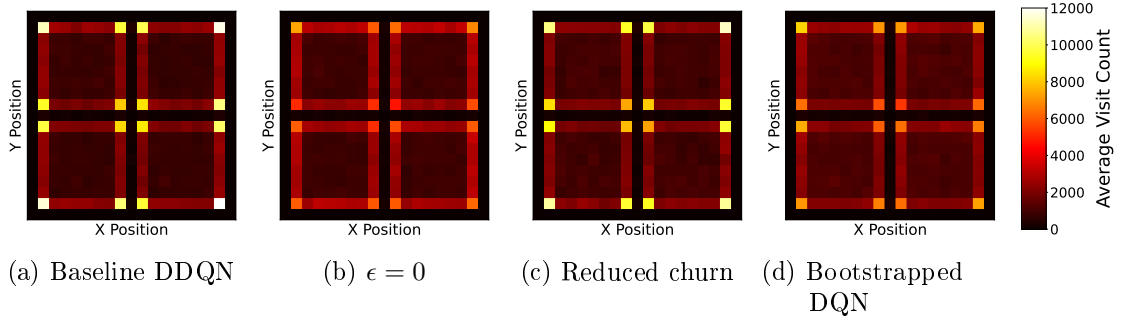


Figure 5.5.: Average state visitation heatmaps in the MiniGrid FourRooms environment by agent type.

Figure 5.5 illustrates heatmaps of the average state visitation frequencies after training. All agents concentrate their visits primarily in the corners of the four rooms, followed by states along the edges. This concentration is more pronounced for the baseline DDQN and the reduced-churn agent, whereas the $\epsilon = 0$ agent and BDQN exhibit slightly more evenly distributed visitation patterns.

These differences remain limited. Across all agents, the overall visitation structure is similar, and none consistently explore transitions between rooms. While BDQN and the $\epsilon = 0$ agent distribute visits more evenly, consistent with their higher state visitation entropy, the agents still fail to discover and exploit bottlenecks.

Collectively, these results show that policy churn can induce exploration in simple environments, but this effect does not extend reliably to settings that require structured, long-term exploration. While some exploration patterns resemble those of BDQN, policy churn alone does not appear sufficient to produce effective bottleneck-crossing behaviour in FourRooms. Its exploratory impact is therefore fragile and strongly dependent on the environment.

Importantly, these experiments also demonstrate that substantial policy churn can persist even when agents fail to learn meaningful behaviour. This observation motivates a closer examination of the factors that shape the magnitude and character of policy churn.

5.3. Policy Churn and Network Capacity

To better understand what influences the magnitude of policy churn, we next examine the role of network capacity. Specifically, we vary the hidden layer size of the Q-network in the MiniGrid environment while keeping all other components fixed. This allows us

5.3. Policy Churn and Network Capacity

to test whether differences in policy churn arise from architectural capacity or from the interaction between learning dynamics and the environment.

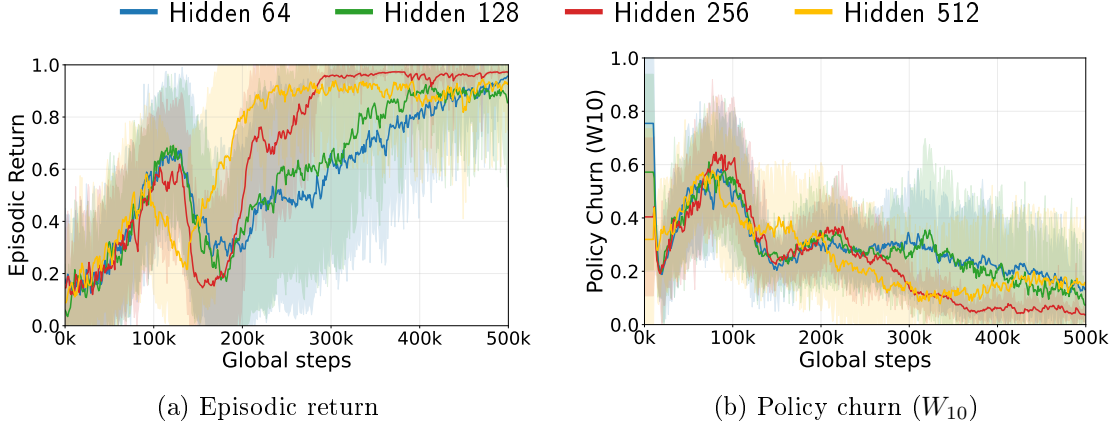


Figure 5.6.: Episodic return and policy churn (W_{10}) in the MiniGrid Empty environment for different hidden layer sizes: 64 (blue), 128 (green), 256 (red), 512 (yellow).

Figure 5.6 shows the episodic return and policy churn (W_{10}) for DDQN agents with hidden layer sizes of 64, 128, 256, and 512 units in the MiniGrid Empty environment, with all other hyperparameters and training settings held fixed. The episodic return curves in Figure 5.6a suggest a tendency for larger networks to reach high returns earlier than smaller ones, although the trajectories are noisy across seeds. All agents eventually converge to similar performance levels by the end of training.

The policy churn curves in Figure 5.6b exhibit a consistent qualitative pattern across all network sizes. Churn is high during early training and decreases once agents approach near-maximal performance. The timing of this decrease depends on how quickly each agent learns: larger networks stabilize earlier, while smaller networks exhibit elevated churn for longer periods. Importantly, the final level of churn does not follow a monotonic relationship with hidden size. While the network with 256 hidden units exhibits the lowest churn at the end of training, both smaller and larger architectures retain slightly higher residual churn. Given the high standard deviations, these differences are relatively minor.

Taken together, these results suggest that, in the MiniGrid Empty environment, we do not observe a clear or monotonic relationship between network capacity and policy churn. Instead, the observed differences in churn appear more closely tied to learning dynamics and convergence speed in this simple setting. We revisit the role of network capacity in the more complex, long-horizon tasks below.

Figure 5.7 shows the episodic return and policy churn (W_{10}) for DDQN agents with hidden layer sizes of 128, 256, 512, and 1024 units in the Atari Qbert and Phoenix environments. We first focus on the results for Qbert. From Figure 5.7a, we observe that all architectural variants steadily improve their episodic return throughout training, indicating that the task can be learned reliably across network sizes. The corresponding policy churn curves in Figure 5.7b demonstrate that churn stabilizes at approximately

5. Experiments

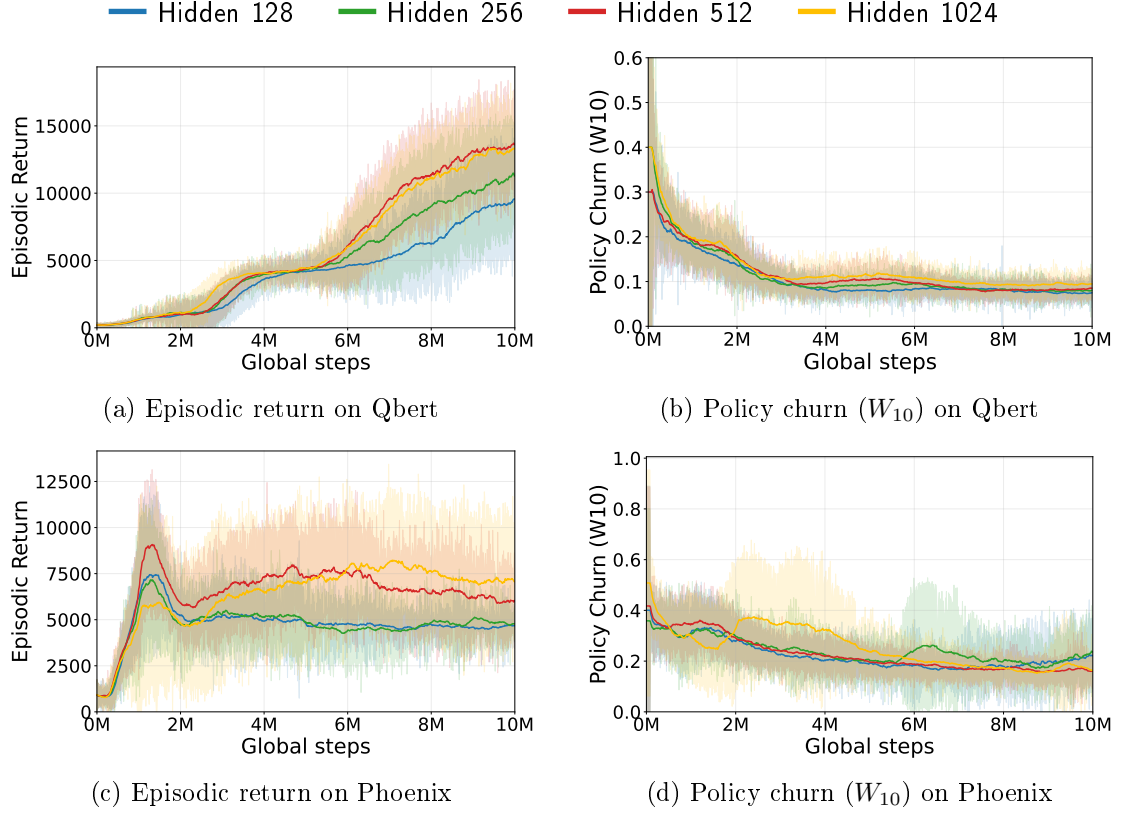


Figure 5.7.: Episodic return and policy churn (W_{10}) in the Atari Qbert and Phoenix environments for different hidden layer sizes: 128 (blue), 256 (green), 512 (red), and 1024 (yellow).

10% for all agents after around 2.5 million environment steps and remains at this level until the end of training. In this setting, there is a slight tendency for larger networks to exhibit higher final churn than smaller ones, but the curves are not clearly separated relative to their standard deviations.

The results for Phoenix suggest a different learning regime. From the episodic return curves in Figure 5.7c, we see that agents reach peak performance after roughly one million steps and subsequently stagnate. The policy churn dynamics in Figure 5.7d are less stable over training and converge to a higher final level than in Qbert, reaching approximately 20% by the end of training.

In contrast to the MiniGrid results, Phoenix provides some indication that network capacity may influence the magnitude of churn. Larger networks with 512 and 1024 hidden units tend to exhibit slightly higher final churn than smaller networks with 128 and 256 units. However, the error bands also overlap substantially in this environment, suggesting that these differences are small and not statistically conclusive.

In light of these results, policy churn does not appear to be a pure architectural artifact. While network capacity may affect the magnitude of churn in complex Atari environments, this influence depends strongly on the task and its learning dynamics. In particular, higher churn persists even when performance stagnates, suggesting that capacity alone is insufficient to explain the observed behaviour. This observation further motivates an investigation into how learning dynamics and plasticity loss interact with policy churn in such settings.

5.4. Policy Churn and Plasticity Loss

In the following, we examine whether plasticity loss can help explain why policy churn remains high even when performance stagnates. We analyse how plasticity-preserving interventions, which modify an agent’s ability to adapt without introducing additional exploration, reshape policy churn and learning behaviour. Since plasticity loss is most pronounced in complex environments, we focus here on the Atari results. The corresponding MiniGrid results are provided in Appendix A.5. We first consider the Qbert environment.

Figure 5.8 summarizes the results. The episodic return curves in Figure 5.8a show that Layer Normalization improves both learning speed and final performance relative to the baseline DDQN. The soft and hard weight reset agents follow slightly different training dynamics, but both reach a final performance comparable to the baseline.

The plasticity-related metrics show a consistent pattern for both reset-based interventions. The hard weight reset agent achieves the lowest TFC mean squared error in Figure 5.8b, followed by the S&P agent. Similarly, both interventions maintain a lower dormant neuron fraction than the baseline DDQN throughout training, as shown in Figure 5.8c. Together, these results suggest that reset-based interventions are effective at preserving plasticity.

Despite their similar effects on plasticity metrics, the interventions exhibit different policy churn dynamics, as shown in Figures 5.8d, 5.8e, and 5.8f. The S&P agent exhibits

5. Experiments

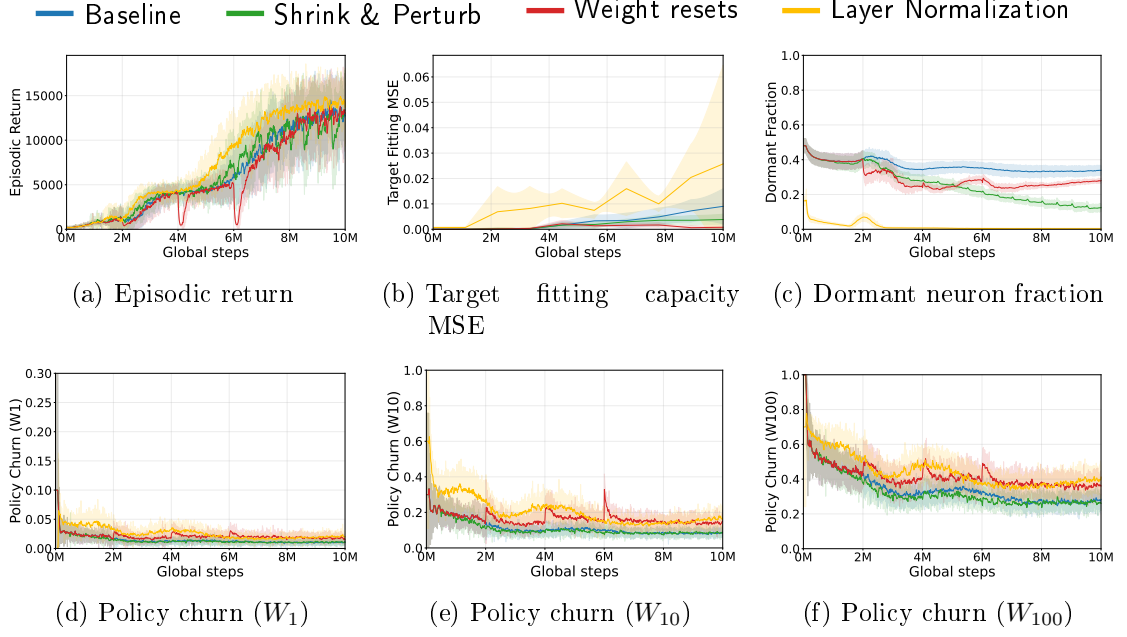


Figure 5.8.: Evaluation of plasticity interventions in the Atari Qbert environment. Baseline DDQN (blue), Shrink & Perturb (green), weight resets (red), and Layer Normalization (yellow).

policy churn dynamics similar to those of the baseline DDQN. In contrast, the hard weight reset agent displays substantially higher policy churn throughout training, which is expected given the disruptive nature of full network reinitializations.

The Layer Normalization agent shows a distinct pattern. Despite achieving the highest episodic return, it also exhibits the highest TFC MSE among all agents (Figure 5.8b), suggesting reduced ability to fit new targets. At the same time, it maintains an exceptionally low dormant neuron fraction, remaining close to zero for most of training (Figure 5.8c). This is consistent with the expected effect of Layer Normalization on pre-activations, which can reduce the emergence of persistently inactive units. While Layer Normalization keeps neurons active, it does not necessarily prevent limitations in target adaptation. The policy churn metrics reveal that the Layer Normalization agent consistently exhibits higher churn than the baseline DDQN, even exceeding the hard weight reset agent.

Taken together, the Qbert results already show that the interventions produce distinct plasticity and churn profiles. Compared to Qbert, which largely involves structured navigation in a relatively stable environment, Phoenix requires continuous reactive control under rapidly changing dynamics. We therefore next examine the Phoenix environment to assess how these interventions behave under conditions where plasticity loss is expected to be more pronounced (Figure 5.9).

Figure 5.9a presents the episodic return for all agents in the Atari Phoenix environment.

5.4. Policy Churn and Plasticity Loss

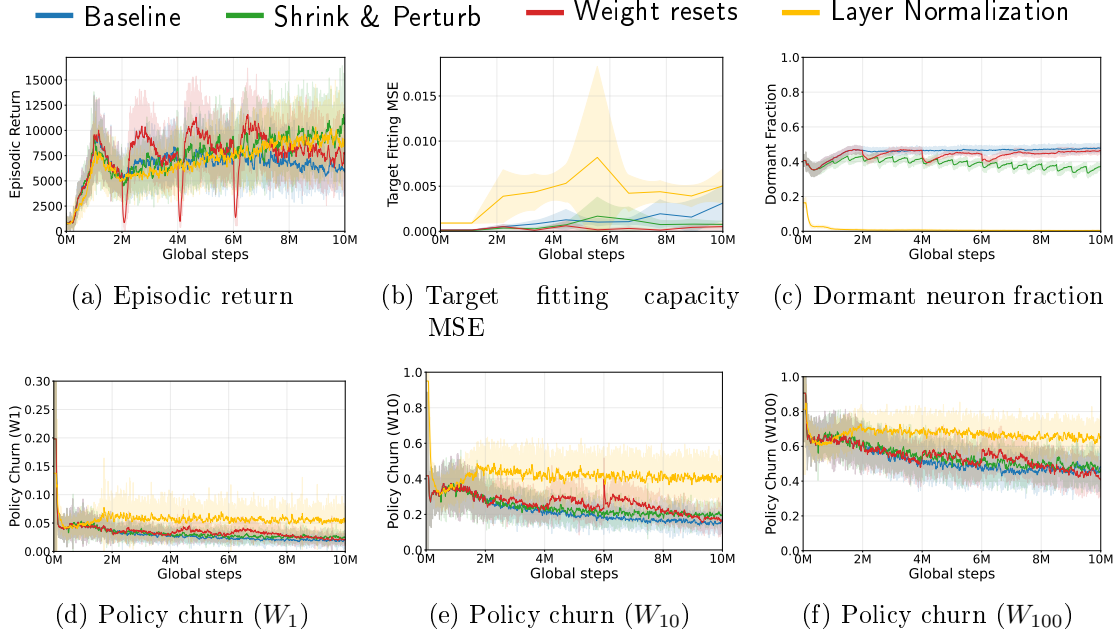


Figure 5.9.: Evaluation of plasticity interventions in the Atari Phoenix environment. Baseline DDQN (blue), Shrink & Perturb (green), weight resets (red), and Layer Normalization (yellow).

The baseline DDQN shows the typical learning behaviour observed in this task. After an initial improvement phase, performance peaks at around one million environment steps and then varies around a similar level without further improvement. All plasticity-preserving interventions modify this learning dynamic in different ways.

The Layer Normalization agent continues to improve after its initial performance peak and reaches a higher final return than the baseline. The S&P agent follows a similar trajectory and attains the highest final return among all variants. The hard weight reset agent displays the most distinct learning behaviour. After an initial performance peak comparable to the other agents, subsequent weight resets enable further improvements. Performance decreases slightly towards the end of training.

The plasticity metrics in Figures 5.9b and 5.9c are broadly consistent with the patterns observed in Qbert. Both reset-based interventions reduce the TFC MSE relative to the baseline DDQN, whereas the Layer Normalization agent exhibits the highest TFC MSE throughout training. The dormant neuron results show a different ordering: Layer Normalization again maintains the lowest dormant fraction, followed by the S&P agent, while the hard weight reset agent reduces dormancy only slightly compared to the baseline.

The policy churn dynamics in Phoenix differ from those observed in Qbert. Figures 5.9d, 5.9e, and 5.9f show that the Layer Normalization agent exhibits by far the highest churn across all window sizes W_i . For the hard weight reset agent, transient spikes in churn occur following each reset, as expected. The S&P agent tends to show

5. Experiments

slightly higher churn than the baseline DDQN throughout training, although this difference remains small relative to the variability across runs. By the end of training, no clear difference in churn remains between the baseline, the hard weight reset agent, and the S&P agent. This is particularly visible for the largest window size, W_{100} (Figure 5.9f), where the corresponding curves remain close to one another.

Overall, these results demonstrate that plasticity-preserving interventions can substantially alter learning dynamics and plasticity metrics in complex Atari environments, while their effects on policy churn are more method-dependent. Layer Normalization stands out as the only intervention that consistently produces markedly higher churn. In contrast, the hard weight reset and S&P agents differ more in their transient behaviour than in their final churn levels. Taken together, these results suggest that changes in plasticity do not map onto policy churn in a simple one-to-one manner.

While the preceding results illustrate how plasticity-preserving interventions affect performance, plasticity metrics, and policy churn individually, they do not reveal how these quantities relate to one another. To better understand these relationships, we next examine scatter plots and correlation heatmaps that capture the co-variation between policy churn, plasticity measures, and learning performance across agents and environments.

5.5. Post-hoc Analysis

In this chapter, we analyse the relationships between learning performance, policy churn, and plasticity-related metrics using post-hoc scatter plots and correlation heatmaps computed across completed runs. The goal is to assess whether policy churn captures information beyond standard learning and plasticity signals, and to identify consistent patterns across environments and interventions. In the following analyses, we use dormant fraction and gradient rank as plasticity measures, since they provide stable cross-run comparisons. TFC is omitted because its associations are weak and inconsistent in our experiments (see Appendix A.4).

Figure 5.10 shows pairwise scatter plots of episodic return, policy churn (W_{10}), gradient rank, and dormant fraction across all plasticity intervention runs in the Atari Phoenix and Qbert environments. Each point corresponds to a single run and is color-coded by intervention type. Although the overall range of gradient rank values is relatively narrow in both environments, several consistent relationships between the variables can still be observed.

Across both environments, gradient rank shows a positive association with episodic return. Runs with higher gradient rank tend to achieve higher returns, consistent with the interpretation of gradient rank as a measure of effective learning dimensionality. Gradient rank is also negatively related to policy churn, indicating that runs with more diverse gradient updates tend to display more stable policies. As expected, it is likewise negatively associated with dormant fraction, suggesting that higher update diversity is linked to fewer inactive neurons.

The dormant fraction reveals additional structure in the data. As already observed in Figures 5.8c and 5.9c, the Layer Normalization agent consistently maintains a very low

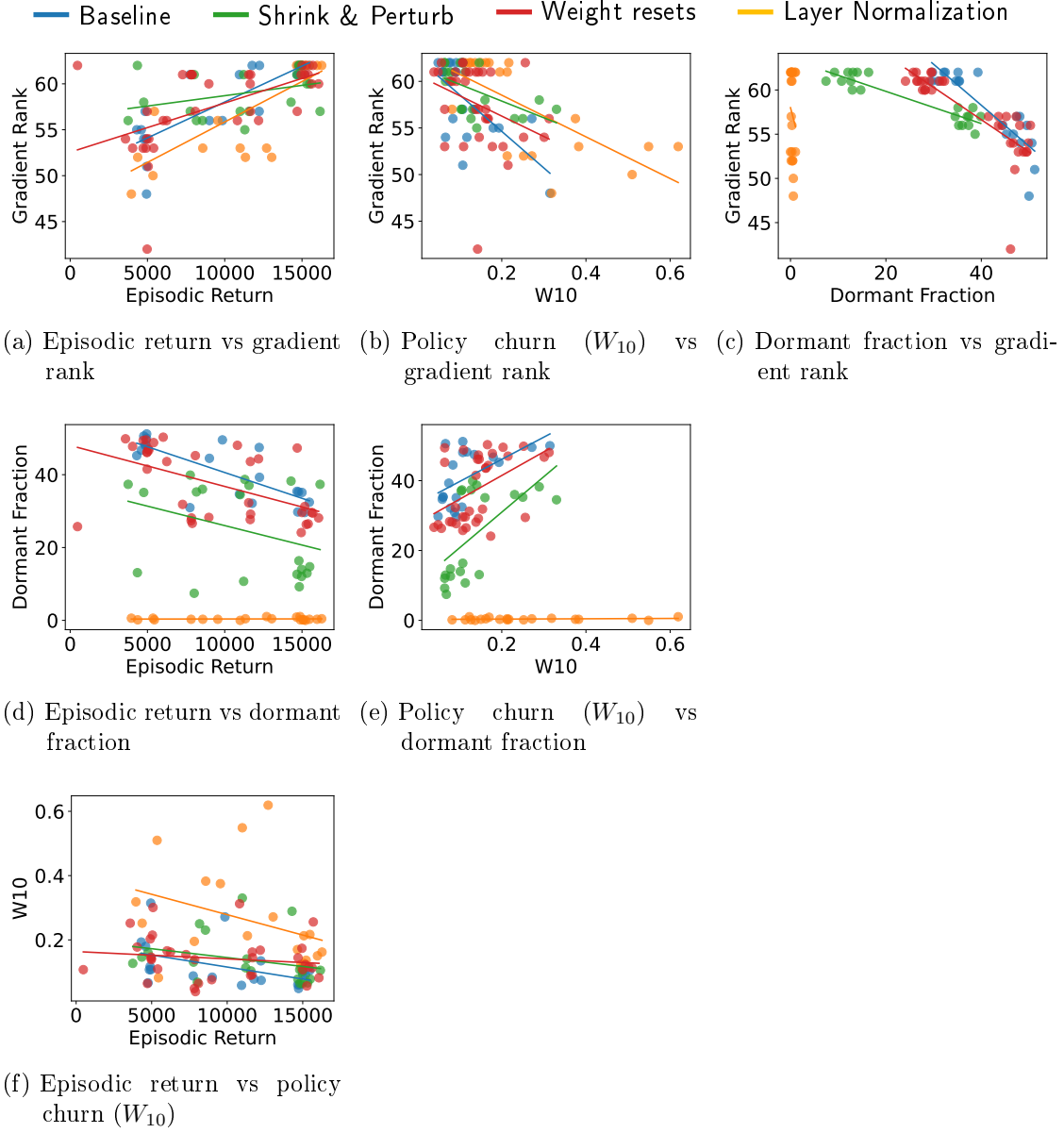


Figure 5.10.: Pairwise scatter plots of episodic return, policy churn (W_{10}), gradient rank, and dormant fraction across all plasticity intervention runs in Atari Phoenix and Qbert environments. Each point represents a single run. Baseline DDQN (blue), Shrink & Perturb (green), weight resets (red), Layer Normalization (yellow).

5. Experiments

dormant fraction, clearly separating it from the other interventions. This separation is particularly visible in Figures 5.10c, 5.10d, and 5.10e. Excluding this distinct cluster, the remaining interventions exhibit a weak negative relationship between dormant fraction and episodic return, as well as a weak positive relationship between dormant fraction and policy churn.

Finally, a slight negative association between episodic return and policy churn can be observed in Figure 5.10f. Although this relationship is not strong, it is consistent with the view that higher churn in these settings more often reflects instability rather than productive learning.

Taken together, these scatter plots suggest that plasticity-related measures, particularly gradient rank and dormant fraction, capture meaningful aspects of learning dynamics and policy stability across runs. To quantify these relationships more systematically, we next examine a correlation heatmap computed across environments and intervention types.

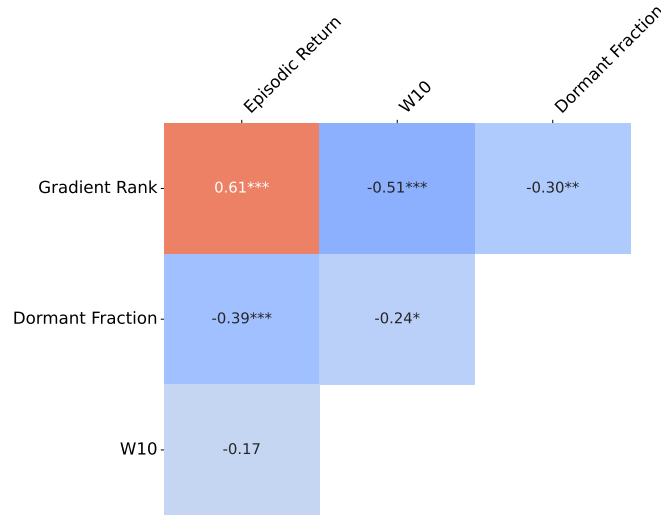


Figure 5.11.: Pearson correlation heatmap of plasticity-related measures, policy churn (W_{10}) and episodic return across all runs in the Atari Phoenix and Qbert environments. Significance levels are denoted by asterisks: * ($p < 0.05$), ** ($p < 0.01$), and *** ($p < 0.001$)

Figure 5.11 presents the Pearson correlation heatmap computed in the same setting as the scatter plots. The heatmap quantifies pairwise relationships between episodic return, policy churn (W_{10}), gradient rank, and dormant fraction across all runs in the Atari Phoenix and Qbert environments.

Consistent with the scatter plot observations, gradient rank is strongly positively correlated with episodic return ($r = 0.61$, $p < 0.0001$), indicating that runs with higher update dimensionality tend to achieve better performance. Gradient rank is also significantly negatively correlated with both policy churn ($r = -0.51$, $p < 0.0001$) and dormant

fraction ($r = -0.30$, $p < 0.001$), suggesting that richer gradient updates are associated with more stable policies and fewer inactive neurons.

Dormant fraction is likewise negatively correlated with episodic return ($r = -0.39$, $p < 0.001$). Its correlation with policy churn is weaker ($r = -0.24$, $p < 0.05$) and should be interpreted together with the scatter plots, where the Layer Normalization runs form a distinct cluster with very low dormancy. Finally, episodic return and policy churn show only a weak negative correlation ($r = -0.17$) that does not reach statistical significance, reinforcing the observation that churn magnitude alone is not a reliable indicator of learning success.

As a whole, the post-hoc analyses show that plasticity-related measures capture aspects of learning dynamics that are not reflected by policy churn alone. Gradient rank and dormant fraction maintain more consistent relationships with performance than policy churn does, while the association between churn and return remains weak. This supports the view that policy churn must be interpreted in the context of underlying learning dynamics and plasticity rather than as a standalone indicator of exploration or performance.

6. Discussion

This thesis investigated the generality of policy churn, its role in exploration, and its interaction with plasticity loss. Across MiniGrid and Atari environments, policy churn was observed consistently in all value-based agents studied. In simple environments, churn alone produced non-trivial exploration but did not yield reliable performance. In environments with bottlenecks and long-horizon navigation requirements, neither churn-driven DDQN nor BDQN consistently exhibited deep exploration behaviour. Network capacity did not show a systematic relationship with churn. We observed only weak indications of higher late-stage churn in regimes of learning stagnation. Plasticity-preserving interventions altered churn dynamics in distinct ways, and correlation analysis revealed consistent relationships between churn, gradient rank, dormant fraction, and episodic return. We now discuss these findings in relation to the research questions introduced in Chapter 1.

6.1. Policy Churn and Exploration

Research Question 2 asked whether policy churn could act as an implicit exploration mechanism and how it compared to explicit methods such as Bootstrapped DQN (BDQN). This question follows directly from Schaul et al. [2022], who interpret rapidly changing greedy policies as a potential source of implicit exploration because they can broaden the data distribution even when explicit ϵ -greedy noise is reduced. This provides the main churn-based hypothesis tested here. It contrasts with the motivation behind deep-exploration methods such as BDQN, where exploration is expected to arise from coherent policies maintained over multiple time steps rather than from local action switches [Osband et al., 2016, Thompson, 1933].

In simple MiniGrid environments, we reproduced the qualitative finding of Schaul et al. [2022] that explicit exploration is not always necessary for successful learning. In MiniGrid Empty, the $\epsilon = 0$ agent was able to solve the environment in a subset of runs despite the absence of explicit exploration in later training (Figures 5.1 and 5.2). This showed that policy churn could, in some cases, be sufficient to support exploration (Figure 5.3). At the same time, this effect was brittle: performance was strongly seed-dependent (Figure 5.2), and even successful $\epsilon = 0$ runs remained below DDQN and BDQN in overall state visitation entropy (Figure 5.3b and 5.1b).

This effect did not extend to hard-exploration environments that required coordinated multi-step exploration. In MiniGrid FourRooms, both the $\epsilon = 0$ agent and BDQN exhibited relatively high state visitation entropy (Figure 5.4c), yet heatmaps showed that this state visitation entropy reflects broadly distributed but undirected movement. Agents did not reliably traverse doorways to reach distant rewards (Figure 5.4b). Increased

6. Discussion

state visitation entropy therefore does not imply meaningful exploration of task-relevant structure.

We further evaluated BDQN across multiple environments. In Empty and FourRooms, BDQN did not consistently outperform DDQN (Figures 5.1 and 5.4). To isolate deep exploration behaviour, we conducted experiments in the SimpleCrossing environments and placed the goal behind one, two, or three bottlenecks. BDQN did not reliably discover the distant reward. Even in the Chain environment, explicitly highlighted in the BDQN paper, we did not observe a consistent advantage over DDQN (see Appendix A.2).

One important difference is that the original BDQN work compared against standard DQN, whereas our experiments used DDQN. Since DDQN already stabilizes value estimates, part of BDQN’s previously reported gains may have stemmed from improved value stability rather than temporally extended exploration alone.

Overall, policy churn increased variability in greedy action selection but did not reliably produce sustained behavioural consistency across time steps. In our experiments, neither churn-driven DDQN nor BDQN consistently exhibited deep exploration in hard-exploration environments.

6.2. Churn as a Symptom of Learning Dynamics

Research Question 1 asked what factors shaped the magnitude and behaviour of policy churn during training. Consistent with Schaul et al. [2022], churn appeared across all value-based agents and environments studied. Rather than being specific to a particular environment layout or exploration difficulty, churn was more closely tied to how learning evolved over time.

During phases of steady learning progress, that is, when performance improved steadily, churn declined. This was often observed shortly before episodic return reached a plateau or maximum reward (e.g., Figure 5.6). In contrast, churn remained high when learning stagnated. This pattern was evident in FourRooms (Figures 5.4a and 5.4d) and especially in Phoenix (Figures 5.7c and 5.7d), where performance plateaued while churn persisted at elevated levels. Larger networks in Phoenix exhibited particularly high late-stage churn despite no performance gains.

Churn was also elevated in regimes associated with plasticity degradation. Compared to Qbert (Figures 5.8e and 5.8f), Phoenix showed stronger plasticity loss and higher sustained churn (Figure 5.9e and 5.9f), suggesting that elevated churn reflected unstable value estimates rather than effective exploration. A similar relationship was observed within the $\epsilon = 0$ agent itself: runs that achieved high performance exhibited substantially lower churn, comparable to the baseline agents, whereas poorly performing runs maintained elevated churn levels (Figures 5.3c and 5.1c). This further supported the interpretation that high churn was primarily a symptom of unstable learning dynamics.

The reduced-churn agent reinforced this interpretation. Although it acted with the target network to slow policy changes, we did not observe a clear reduction in measured churn relative to baseline DDQN (Figure 5.1c). One possible explanation is that acting on delayed value estimates weakened learning progress and prevented stable Q-value

separation. Slower intentional switching therefore did not guarantee lower churn if value estimates remained unstable.

Taken together, churn is neither inherently beneficial nor detrimental. It reflects the sensitivity of greedy action selection to parameter updates and depends more on learning stability and plasticity than on exploration mechanisms alone.

6.3. Network Capacity and Churn

We further investigated whether policy churn is systematically related to network capacity. If churn were primarily an architectural phenomenon, one would expect a relationship between hidden layer size and churn magnitude. Across environments, we did not observe a clear relationship between hidden layer size and churn.

In MiniGrid Empty, larger networks converged faster, but final churn levels were similar once performance stabilized (Figure 5.6). In Qbert, churn again converged to comparable levels across architectures (Figure 5.7b). In Phoenix, there were weak indications that larger networks exhibited slightly higher late-stage churn despite plateauing performance, although these differences were small and not statistically conclusive (Figures 5.7c and 5.7d).

These results indicate that capacity alone does not determine churn. Its effect depends on the learning regime and on the environment. During phases of steady performance improvement, policy churn remained similar across network sizes. When performance plateaus or plasticity degraded, additional parameters may contribute to higher churn without improving return.

6.4. Plasticity Loss: The Link Between Churn and Performance

Research Questions 3 and 4 addressed how policy churn interacts with plasticity loss and how plasticity-preserving methods influence churn. Plasticity-preserving interventions reshaped learning dynamics, plasticity metrics, and policy churn, with environment-dependent effects (Figure 5.8 and Figure 5.9).

In baseline DDQN, performance increased early and then plateaued. In Phoenix, this plateau coincided with worsening target fitting capacity, rising dormant fraction, and sustained high churn. Churn persisted without performance gains.

Weight resets and Shrink & Perturb altered this pattern. Both reduced dormant fraction and improved target fitting capacity relative to baseline. Hard resets induced transient spikes in churn after reinitialization. In contrast, Shrink & Perturb blends the current parameters with a fresh initialization and largely preserves the relative ordering of Q-values. As a result, it restores plasticity without substantially destabilizing greedy action selection, leading to only moderate increases in churn. Performance improvements depended on the environment.

6. Discussion

Layer Normalization showed a distinct profile. As expected, it maintained a near-zero dormant fraction, indicating that almost all hidden units continued to participate in the forward pass. This is consistent with the role of Layer Normalization in keeping neuron activations well-scaled during training. By stabilizing the inputs to each activation function, Layer Normalization prevents units from becoming persistently inactive. Recent work further suggests that Layer Normalization can even help inactive ReLU units become active again, which provides an additional explanation for why it preserves plasticity [Lyle et al., 2024, Xu et al., 2019]. Furthermore, Layer Normalization exhibited consistently higher churn than the baseline. In Phoenix, this coincided with improved performance; in Qbert, it achieved the highest returns despite high target fitting MSE. High churn can therefore coexist with strong performance when the network maintains a broadly active set of hidden units, as indicated here by the near-zero dormant fraction. One possible explanation is that, when a larger proportion of neurons participates in the forward pass, greedy action selection becomes more sensitive to parameter updates, thereby increasing observed churn without necessarily indicating harmful instability.

To summarise, churn interacts closely with plasticity but does not track it in a one-to-one manner. When plasticity is preserved, churn can reflect ongoing adaptation. When plasticity degrades, similar churn levels reflect instability. The interpretation of churn thus depends on the representational state of the network.

6.5. Interpreting the Plasticity Metrics

The correlation analysis (Figure 5.11) highlighted consistent relationships between gradient rank, dormant fraction, episodic return, and policy churn. These relationships should be interpreted in light of how churn is measured in this thesis. Our churn metric compares the greedy actions of the current online network to those of a fixed target-network snapshot on replay-buffer states. It therefore measures short-horizon sensitivity of greedy action selection to recent parameter updates, rather than exploration quality or plasticity directly.

This interpretation links the correlation analysis back to our original hypotheses. Higher gradient rank was associated with higher episodic return, while higher dormant fraction was associated with lower return. This supports the view that agents with richer update directions and fewer inactive neurons tend to learn more effectively, consistent with prior work associating rank collapse and dormant neurons with plasticity loss and reduced learning ability in deep RL [Kumar et al., 2021, Lyle et al., 2022, Sokar et al., 2023]. At the same time, policy churn showed only a weak relationship with episodic return, suggesting that churn magnitude alone does not explain performance differences across runs.

Policy churn, however, did not map onto any single plasticity metric in a one-to-one manner. High churn appeared both in regimes marked by plasticity loss, such as the Phoenix baseline in Figure 5.9, and in regimes with broadly active representations, such as Layer Normalization in Figure 5.8. This suggests that the same measured churn value can have different meanings depending on the representational state of the network.

Gradient rank captures the diversity of update directions, dormant fraction captures representational availability, and churn captures the sensitivity of the greedy argmax policy to recent updates. Churn therefore gains meaning only when interpreted jointly with plasticity diagnostics and episodic return. In regimes marked by plasticity loss, high churn is more naturally interpreted as unstable value estimation. In regimes with preserved plasticity, it may instead reflect continued adaptation.

6.6. Policy Churn as a Diagnostic Signal

Taken together, the results of this thesis suggest that policy churn should neither be maximized nor suppressed in isolation. Since churn captures policy sensitivity rather than the usefulness of policy change, its interpretation depends on the learning state in which it occurs.

This distinction is important when relating churn to performance. In an idealized converged agent with stable value estimates and large action gaps, we would expect low churn. Small parameter updates should no longer change the greedy action in task-relevant states. In this sense, late-stage churn can indicate unstable value estimates when learning has plateaued and plasticity metrics deteriorate. This pattern was observed for the Phoenix baseline in Figures 5.9a and 5.9e, and for the poor-performance group in Figure 5.3c.

At the same time, high performance does not require churn to vanish completely. Schaul et al. [2022] show that substantial policy churn can persist even after performance has stabilized, and argue that policy changes need not affect performance when they occur within policy null spaces. In such cases, different greedy actions may lead to equivalent outcomes, or action switches may occur in states where the selected action has little consequence. This explains why a well-performing agent may still exhibit nonzero churn.

Recent work on value and policy churn provides a complementary interpretation. Tang and Berseth [2024] describe a chain effect in which churn in value estimation and policy improvement can compound and bias learning dynamics over successive updates. In value-based methods, changes in value estimates can alter the greedy action, and these altered policies can affect the data and targets used for subsequent learning. This supports the view that churn should not be interpreted only as an exploration mechanism. Instead, the same measured policy change may reflect different underlying processes depending on the state of learning.

Our findings are consistent with this more cautious interpretation. Policy churn was not a reliable predictor of episodic return on its own. Instead, its meaning depended on the surrounding learning dynamics. When late-stage churn coincided with low gradient rank, high dormant fraction, or stagnant return, it suggested unstable value estimation. When churn occurred alongside active representations and strong performance, as under Layer Normalization, it was more consistent with continued adaptation. Policy churn is therefore best understood as a diagnostic signal of learning dynamics: monitored alongside plasticity metrics, it can help distinguish stable adaptation from degraded

6. *Discussion*

learning or other forms of policy sensitivity.

7. Conclusion

This thesis investigated the generality of policy churn, its role in exploration, and its interaction with plasticity loss in deep reinforcement learning. Across MiniGrid and Atari environments, policy churn was consistently observed in value-based agents. In simple environments, churn alone produced non-trivial exploration, but it did not lead to reliable performance and remained strongly seed-dependent. In hard-exploration environments requiring long-term commitment, exploration driven by policy churn alone did not reliably outperform ϵ -greedy DDQN or provide effective deep exploration.

Our results further show that policy churn is closely tied to learning dynamics and plasticity-related properties of the network. Importantly, policy churn does not admit a single interpretation: high churn can arise from ongoing adaptation when plasticity is preserved, but it can also reflect instability when plasticity degrades. Higher gradient rank and lower dormant fraction are associated with better performance, indicating that stable and diverse updates contribute to improved learning. Policy churn shows a weaker and more context-dependent relationship with these plasticity metrics, suggesting that churn must be interpreted jointly with the representational state of the network. Plasticity-preserving interventions reshape churn dynamics in different ways. Weight resets and Shrink & Perturb restore adaptability, while Layer Normalization maintains broad activation but often increases churn. Overall, policy churn is not a standalone exploration mechanism, but a signal reflecting the interaction between optimization, plasticity, and learning stability.

Limitations

This work has several limitations. The most important concerns the measurement of policy churn itself. Policy churn is not a measurement-free quantity; it must be defined relative to two policy snapshots, a lag window, and an evaluation state distribution. Following the general formulation of Schaul et al. [2022], this thesis measures churn as the fraction of replay-buffer states for which the greedy action differs between the current online network and a fixed target-network reference. The reported values therefore reflect churn under this specific operationalization, on the agent’s empirical training distribution and at the chosen update lags. Although Schaul et al. [2022] provide the formal definition of policy churn, they do not specify all implementation details required to reproduce a particular empirical estimator exactly. As a result, absolute churn magnitudes and comparisons across agents, environments, or studies should be interpreted with caution, even though the within-study comparative trends remain informative.

A second limitation is that the diagnostic analysis is post-hoc and does not establish directional relationships between variables. The observed correlations between churn,

7. Conclusion

plasticity metrics, and episodic return should therefore be interpreted as descriptive associations rather than evidence of causal mechanisms.

The empirical scope of the thesis is also limited. We focus exclusively on value-based methods, which limits generalization to other algorithm classes. In addition, we were not able to demonstrate consistent success in truly deep-exploration benchmarks, and some MiniGrid environments did not behave as expected. This limited their suitability for comparing exploration and plasticity metrics.

Finally, the plasticity diagnostics themselves are imperfect. In particular, the gradient rank metric is a practical approximation and remains heuristic in nature. It is useful for comparing runs within the same experimental setup, but should not be interpreted as a complete measure of plasticity.

Future Work

Several directions remain for future work. First, the analysis should be extended beyond value-based learning to actor-critic methods and continuous-control settings. This would help determine whether similar churn dynamics also emerge when policies are represented and updated more directly.

Second, future studies should examine how different churn estimators affect the observed results. In this thesis, churn is measured on replay-buffer states, and therefore on the agent’s empirical training distribution. Measuring churn on fixed evaluation states, or on states stratified by action gap, could help distinguish behaviourally relevant policy changes from action switches that occur in policy-null-space regions and have little effect on return.

Third, the relationship between plasticity and churn remains causally unresolved. Plasticity-preserving interventions reshape churn dynamics, but the present experiments do not determine whether churn contributes to preserving plasticity or mainly reflects underlying changes in the network. This is especially relevant for Layer Normalization, which produced high churn despite maintaining active representations. A possible explanation is that broadly active networks are more sensitive to parameter updates, increasing greedy-action changes without necessarily indicating unstable learning.

Bibliography

- Jordan T. Ash and Ryan P. Adams. On warm-starting neural network training, 2020. URL <https://arxiv.org/abs/1910.08475>.
- Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016. URL <https://arxiv.org/abs/1607.06450>.
- Yahui Bai, Yuhe Gao, Runzhe Wan, Sheng Zhang, and Rui Song. A review of reinforcement learning in financial applications. *arXiv preprint arXiv:2411.12746*, 2024. URL <https://arxiv.org/abs/2411.12746>.
- Marc G. Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation, 2016. URL <https://arxiv.org/abs/1606.01868>.
- Tudor Berariu, Wojciech Czarnecki, Soham De, Jorg Bornschein, Samuel Smith, Razvan Pascanu, and Claudia Clopath. A study on the plasticity of neural networks, 2023. URL <https://arxiv.org/abs/2106.00042>.
- Maxime Chevalier-Boisvert, Bolun Dai, Mark Towers, Rodrigo de Lazcano, Lucas Willems, Salem Lahlou, Suman Pal, Pablo Samuel Castro, and Jordan Terry. Minigrid & miniworld: Modular & customizable reinforcement learning environments for goal-oriented tasks, 2023. URL <https://arxiv.org/abs/2306.13831>.
- Rohan Inamdar, S. Kavin Sundarr, Deepen Khandelwal, Varun Dev Sahu, and Nitish Katal. A comprehensive review on safe reinforcement learning for autonomous vehicle control in dynamic environments. *e-Prime - Advances in Electrical Engineering, Electronics and Energy*, 10:100810, 2024. ISSN 2772-6711. doi: <https://doi.org/10.1016/j.prime.2024.100810>. URL <https://www.sciencedirect.com/science/article/pii/S2772671124003905>.
- Steven Kapturowski, Georg Ostrovski, Will Dabney, John Quan, and Remi Munos. Recurrent experience replay in distributed reinforcement learning. In *International Conference on Learning Representations*, 2019. URL <https://openreview.net/forum?id=r1lyTjAqYX>.
- Timo Klein, Lukas Miklautz, Kevin Sidak, Claudia Plant, and Sebastian Tschatschek. Plasticity loss in deep reinforcement learning: A survey, 2024. URL <https://arxiv.org/abs/2411.04832>.

Bibliography

- Aviral Kumar, Rishabh Agarwal, Dibya Ghosh, and Sergey Levine. Implicit underparameterization inhibits data-efficient deep reinforcement learning. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=09bnihsFfXU>.
- Pawel Ladosz, Lilian Weng, Minwoo Kim, and Hyondong Oh. Exploration in deep reinforcement learning: A survey. *Information Fusion*, 85:1–22, September 2022. ISSN 1566-2535. doi: 10.1016/j.inffus.2022.03.003. URL <http://dx.doi.org/10.1016/j.inffus.2022.03.003>.
- Clare Lyle, Mark Rowland, and Will Dabney. Understanding and preventing capacity loss in reinforcement learning. In *International Conference on Learning Representations*, 2022. URL <https://openreview.net/forum?id=ZkC8wKoLbQ7>.
- Clare Lyle, Zeyu Zheng, Khimya Khetarpal, James Martens, Hado van Hasselt, Razvan Pascanu, and Will Dabney. Normalization and effective learning rates in reinforcement learning, 2024. URL <https://arxiv.org/abs/2407.01800>.
- Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.
- Wenlong Mou, Zheng Wen, and Xi Chen. On the sample complexity of reinforcement learning with policy space generalization, 2020. URL <https://arxiv.org/abs/2008.07353>.
- Evgenii Nikishin, Max Schwarzer, Pierluca D’Oro, Pierre-Luc Bacon, and Aaron Courville. The primacy bias in deep reinforcement learning, 2022. URL <https://arxiv.org/abs/2205.07802>.
- Evgenii Nikishin, Junhyuk Oh, Georg Ostrovski, Clare Lyle, Razvan Pascanu, Will Dabney, and André Barreto. Deep reinforcement learning with plasticity injection, 2023. URL <https://arxiv.org/abs/2305.15555>.
- Ian Osband, Charles Blundell, Alexander Pritzel, and Benjamin Van Roy. Deep exploration via bootstrapped dqn, 2016. URL <https://arxiv.org/abs/1602.04621>.
- Tom Schaul, André Barreto, John Quan, and Georg Ostrovski. The phenomenon of policy churn, 2022. URL <https://arxiv.org/abs/2206.00730>.
- David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016. doi: 10.1038/nature16961. URL <https://www.nature.com/articles/nature16961>.

- Ghada Sokar, Rishabh Agarwal, Pablo Samuel Castro, and Utku Evci. The dormant neuron phenomenon in deep reinforcement learning, 2023. URL <https://arxiv.org/abs/2302.12902>.
- Lakshmi Narasimhan Srinivasagopalan. Applying reinforcement learning to optimize healthcare insurance premium pricing. *Frontiers in Health Informatics*, 13(2):11282, 2024. URL https://www.researchgate.net/publication/388817233_Applying_Reinforcement_Learning_to_Optimize_Healthcare_Insurance_Premium_Pricing.
- Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.
- Hongyao Tang and Glen Berseth. Improving deep reinforcement learning by reducing the chain effect of value and policy churn, 2024. URL <https://arxiv.org/abs/2409.04792>.
- William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3-4):285–294, 12 1933. ISSN 0006-3444. doi: 10.1093/biomet/25.3-4.285. URL <https://doi.org/10.1093/biomet/25.3-4.285>.
- Hado van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning, 2015. URL <https://arxiv.org/abs/1509.06461>.
- Jingjing Xu, Xu Sun, Zhiyuan Zhang, Guangxiang Zhao, and Junyang Lin. Understanding and improving layer normalization. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

A. Appendix

A.1. Hyperparameters

For completeness and reproducibility, Table A.1 lists all hyperparameters used in the experiments.

Table A.1.: Hyperparameters for MiniGrid and Atari Experiments

Parameter	MiniGrid	Atari
DQN Algorithm Parameters		
Total Timesteps	500,000	10,000,000
Learning Rate (α)	2.5×10^{-4}	1.0×10^{-4}
Replay Buffer Size	100,000	1,000,000
Discount Factor (γ)	0.99	0.99
Target Network Update Rate (τ)	1.0	1.0
Target Network Freq. (steps)	500	1,000
Batch Size	128	32
Training Frequency (steps)	10	4
Learning Starts (steps)	10,000	80,000
Exploration Parameters (ϵ-greedy)		
Starting ϵ (ϵ_{start})	1.0	1.0
Ending ϵ (ϵ_{end})	0.05	0.01
Exploration Fraction	0.5	0.10
Bootstrapped DQN Parameters		
Number of Heads (k)	10	10
Plasticity Intervention Parameters		
Shrink & Perturb Coefficient (α)	0.8	0.8
Shrink & Perturb Frequency (steps)	100,000	500,000
Weight Reset Frequency (steps)	100,000	2,000,000
Measurement & Analysis Parameters		
Policy Churn Sample Size (states)	512	512
Dormant Fraction Threshold (τ)	0.1	0.1
Target Fitting Freq. (steps)	100,000	1,000,000
Gradient Covariance Freq. (steps)	20,000	100,000

A.2. BDQN as Deep Exploration Baseline

Bootstrapped DQN (BDQN) is commonly used as a benchmark for deep exploration. In our MiniGrid experiments we were unable to consistently reproduce its reported exploration advantage over DDQN. Figure A.1 shows the results for the Simple Crossing environments with $n = 1, 2, 3$ crossings. In these environments, BDQN does not consistently outperform DDQN in terms of episodic return and does not show a clear advantage.

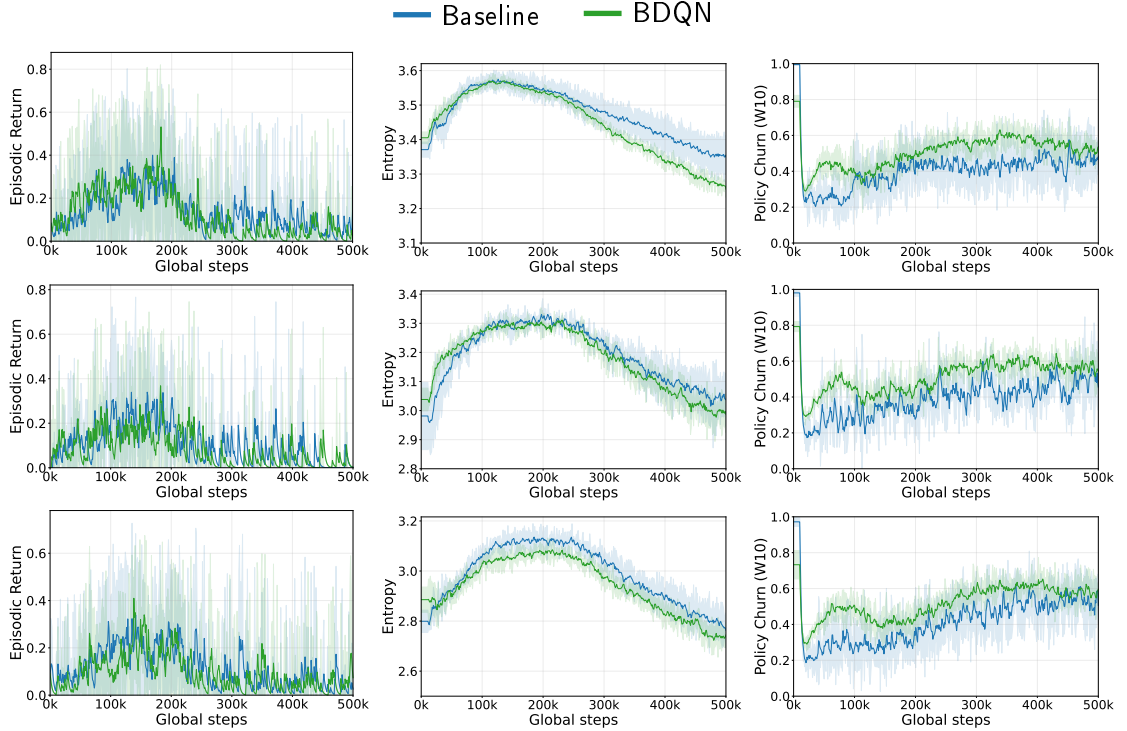


Figure A.1.: Episodic Return (left), Exploration Entropy (middle) and Policy Churn W10 (right) in the Simple Crossing environments (with $n = 1, 2, 3$ crossings). Baseline DDQN (blue) and Bootstrapped DQN (green).

Osband et al. [2016] evaluate deep exploration using a deterministic chain environment in which the agent must repeatedly choose the same action to reach a distant reward at the end of a length- N chain. Since intermediate states yield no reward, successful learning requires temporally extended, consistent exploration. In the original paper, BDQN scales better than DQN as the chain length increases. In our experiments with chain length 100, however, we do not observe a clear advantage of BDQN over DDQN (Figure A.2).

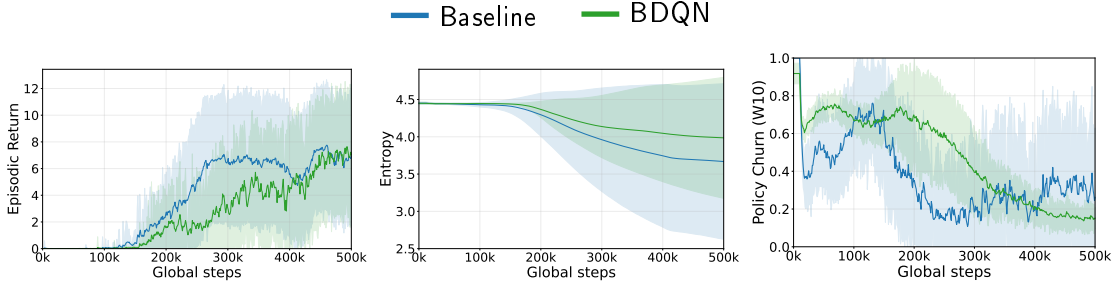


Figure A.2.: Episodic Return (left), Exploration Entropy (middle) and Policy Churn W10 (right) in the Chain environment with chain length $N = 100$. Baseline DDQN (blue) and Bootstrapped DQN (green).

A.3. Exploration in Atari

As a supplementary analysis, we evaluate the exploration variants in the Atari environments Phoenix and Qbert. These results (Figure A.3) complement the MiniGrid experiments and provide additional context for the exploration behaviour observed in the main text.

A.4. Gradient Rank and Target Fitting Capacity

Although gradient rank is used as a plasticity measure in the post-hoc diagnostic analysis, the corresponding plots for Qbert and Phoenix are not shown in the main results section. Instead, we present the target fitting capacity, which we found to be more informative in this context. For completeness, the gradient rank plots for both environments are provided in Figures A.4a and A.4b. The post-hoc diagnostic analysis of the TFC MSE is shown in Figure A.5.

A.5. Plasticity interventions in MiniGrid

The environments in MiniGrid are relatively simple and agents can often solve them quickly. As a result, plasticity loss may not manifest strongly within the training duration. Nonetheless we conduct experiments with plasticity interventions in the `MiniGrid-Empty-16x16-v0` and `MiniGrid-Empty-Random-6x6-v0` environments to evaluate the effects of these interventions on learning dynamics and policy churn in simpler settings. The results are shown in Figures A.6 and A.7.

A. Appendix

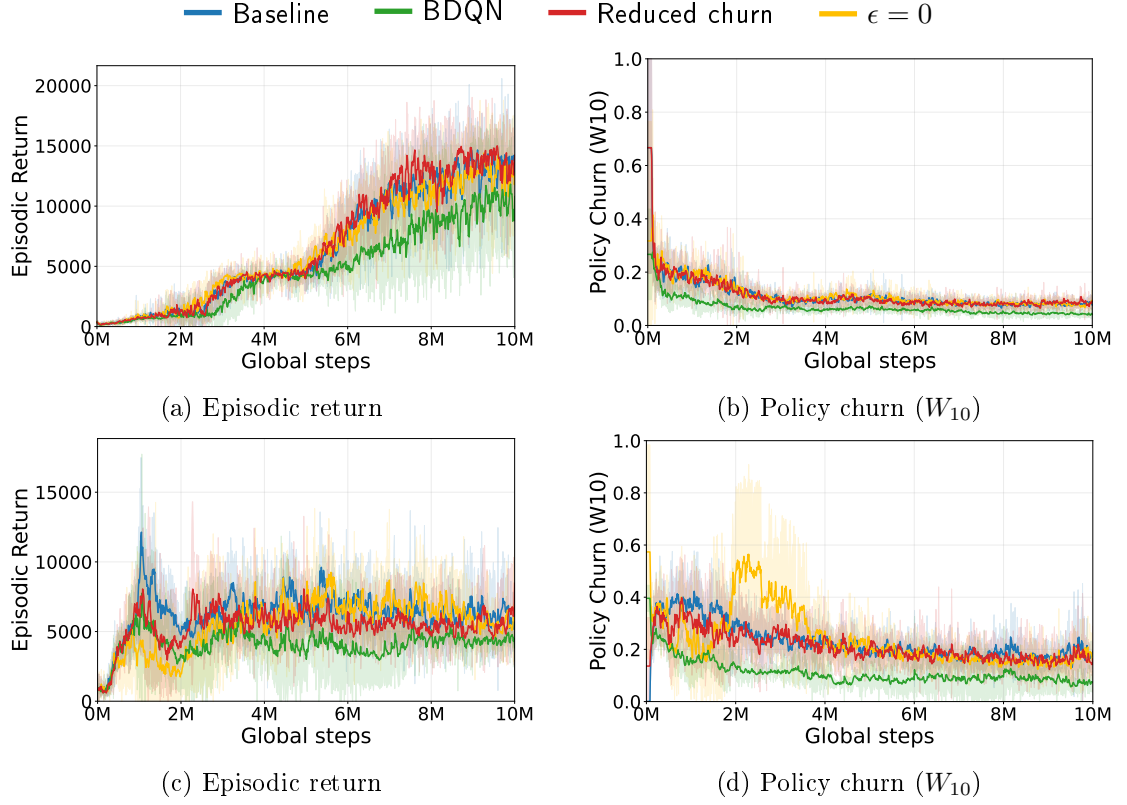


Figure A.3.: Episodic return, exploration entropy, and policy churn (W_{10}) in the Atari Qbert and Phoenix environments. Baseline DDQN (blue), $\epsilon = 0$ (yellow), Bootstrapped DQN (green), reduced churn (red).

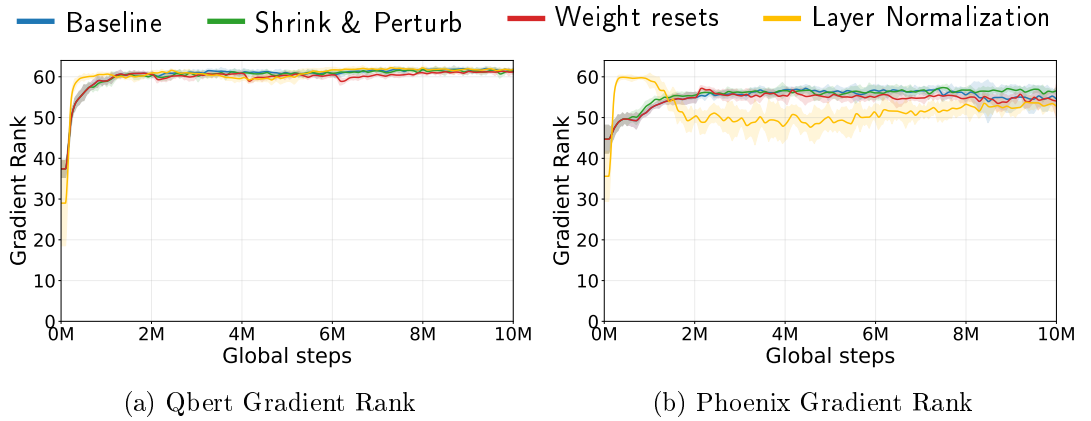


Figure A.4.: Gradient rank of plasticity interventions in the Atari Qbert and Phoenix environments. Baseline DDQN (blue), Shrink & Perturb (green), weight resets (red), and Layer Normalization (yellow).

A.5. Plasticity interventions in MiniGrid

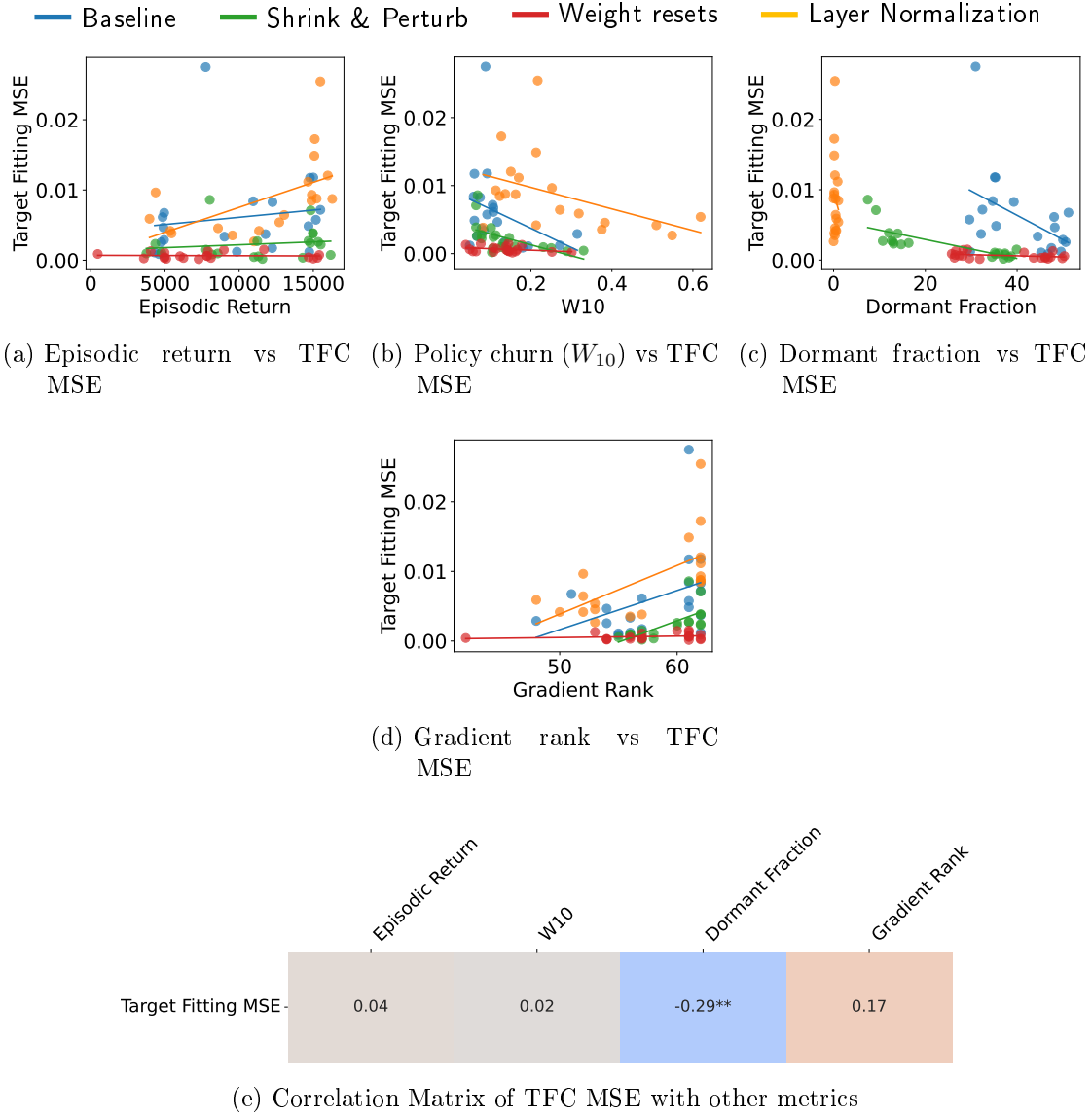


Figure A.5.: Post-hoc diagnostic analysis of Target Fitting Capacity (TFC) MSE.

A. Appendix

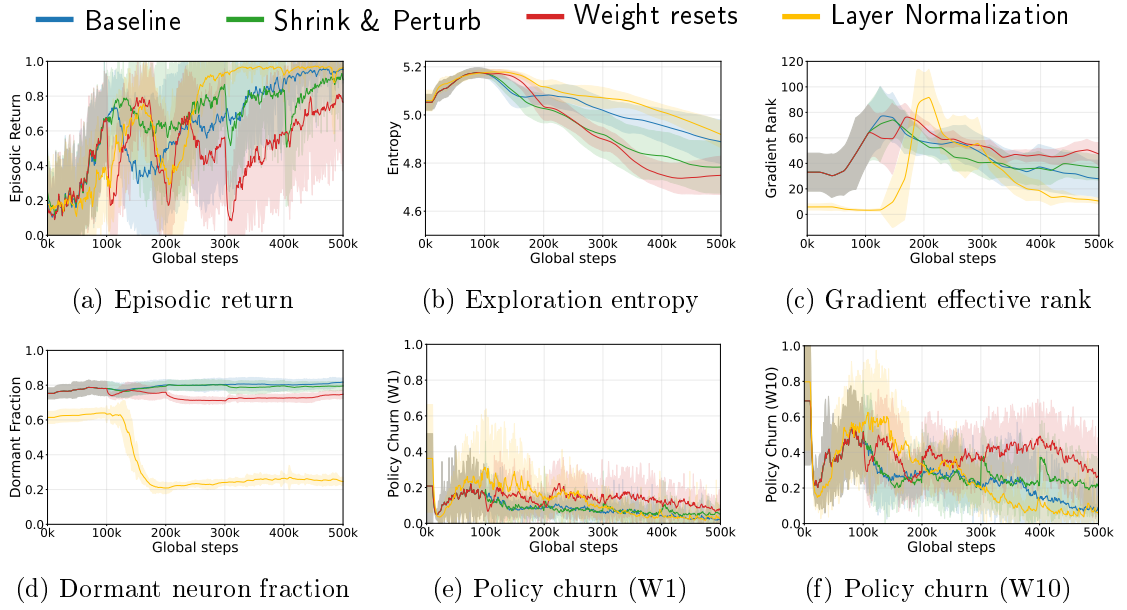


Figure A.6.: Evaluation of plasticity interventions in the MiniGrid-Empty-16x16-v0 environment. Shown metrics are (a) episodic return, (b) exploration entropy, (c) gradient effective rank, (d) dormant neuron fraction, and policy churn measured at (e) W1 and (f) W10. Baseline DDQN (blue), Shrink & Perturb (green), weight resets (red), and Layer Normalization (yellow).

A.5. Plasticity interventions in MiniGrid

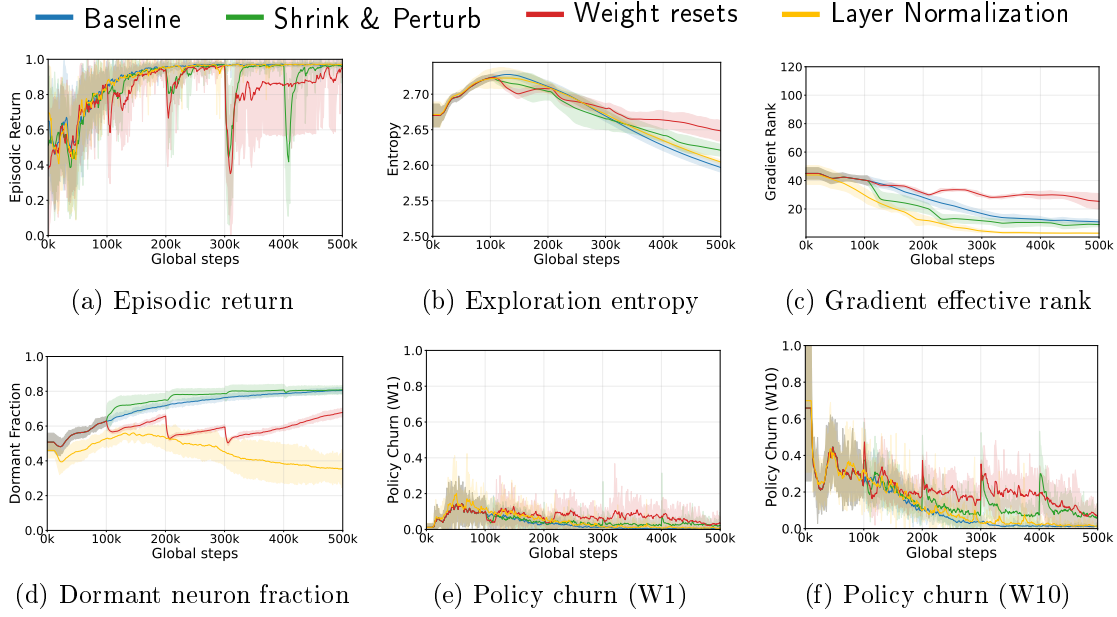


Figure A.7.: Evaluation of plasticity interventions in the MiniGrid-Empty-Random-6x6-v0 environment. Shown metrics are (a) episodic return, (b) exploration entropy, (c) gradient effective rank, (d) dormant neuron fraction, and policy churn measured at (e) W1 and (f) W10. Baseline DDQN (blue), Shrink & Perturb (green), weight resets (red), and Layer Normalization (yellow).

A.6. Declaration of Generative AI and AI-Assisted-Technologies in the Manuscript Preparation Process

During the preparation of this thesis, the authors used ChatGPT by OpenAI to support the manuscript preparation process. The tool was used for language refinement, improving clarity and structure, as well as assisting with code development. The authors reviewed and edited all AI-assisted output and takes full responsibility for the content of this thesis.

No generative AI or AI-assisted technology was used to generate experimental results or conduct the empirical analysis.