Proceedings of the 4th International Conference on Machine Learning and Automation
DOI: 10.54254/2755-2721/2026.38740

Reward Design in Self-Play Reinforcement Learning for Dou Shou Qi: Effects on Learning Efficiency and Behavioral Characteristics

Junheng Niu

College of Science, Beijing Information Science and Technology University, Beijing, China
niujunheng2004@outlook.com

Abstract. Reward Design helps to motivate the learning process and modify the behavior of agents in reinforcement learning through intermediate feedback. However, the impact of reward-design parameters on the learning efficiency and behavior of competitive self-play board games has not been fully explored. Research on Reward Shaping and Reward-Parameter Strength in Self-Play Reinforcement Learning for Dou Shou Qi. Controlled reward-ablation experiments were conducted using Proximal Policy Optimization with a shared 64-channel, four-block residual network on a 7×9 board. Three independent seeds were used to train the agents for 1,000,000 environment steps in total. Compare the experimental results of reward shaping and no shaping, vary the shaping coefficient from 0.25 to 1.0, and change the repetition-penalty coefficient between 0.002 and 0.05. Learning efficiency is the number of environment steps needed to reach a stable performance of 0.95, and behavior indicators include episode length, repetition rate, policy entropy, and the final evaluation win rate. Under the reward-shaping condition, the sustained high-performance group needed 34.7% fewer environment steps in IQM than the non-shaping group. Increased shaping strength did not lead to further improvements in learning efficiency, and behavior was altered by changing reward parameters. Therefore, the design of rewards should be judged based on learning efficiency and behavioral changes.

Keywords: Dou Shou Qi, self-play reinforcement learning, reward design, reward shaping, learning efficiency

1. Introduction

Reinforcement learning (RL) is a way for agents to learn behavior by interacting with an environment and receiving reward signals. In competitive board games, self-play is a way for agents to generate training experience by repeatedly interacting with themselves or other learning policies, and it has become a popular method for learning complex game strategies. The success of AlphaGo, AlphaGo Zero, and AlphaZero has shown that self-play learning can be applied to Go, chess, and Shogi [1-3]. Although there have been some progress, the final results of games often require a long series of decisions before feedback is provided to the agent. Reward Design is thus also necessary;

© 2026 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2755-2721/2026.38740

otherwise, the added intermediate signals will not provide effective direction during learning and will fail to achieve the original goal of the task.

Reward shaping is added to the task's reward function to reduce sparsity and low information content in the feedback. Ng and his colleagues provided the theoretical foundation for the change in this way, showing that under some circumstances, policy optimization can be achieved [4]. Later studies have extended this idea to multi-agent environments and shown that reward shaping can reduce the learning period, as well as influence the policies and group-level results of interaction among learning agents [5]. Research in game environments has also shown that the effect of reward shaping varies depending on the specific learning context and may differ from that of other types of learning support [6]. More recently, adaptive and automatically learned shaping mechanisms have been introduced to improve the learning efficiency and convergence stability of the work [7, 8]. In short, all of the above studies indicate that reward shaping can modify the immediate reward signal and also alter the learning process and behavior during training [4-8]. However, relatively little research has systematically explored the role of shaping strength in competitive self-play board games, especially in combination with learning efficiency and behavioral traits. Results from deep reinforcement learning are also subject to the random seed and other methods; therefore, several experiments must be repeated and statistically analyzed to obtain reliable results [9, 10].

Given the above deficiencies, this study will investigate the effect of reward-design parameters on learning efficiency and behavior in self-play reinforcement learning for Animal Chess (Dou Shou Qi). First, train agents with and without potential-based reward shaping in the same environment and learning algorithm, and use other main training parameters. Second, with a fixed reward structure, vary the shaping coefficient to determine whether an increase in the strength of shaping will lead to continuous improvement in learning efficiency. Third, a repetition-penalty coefficient is altered to ascertain whether the modification of reward design is correlated with differences in episode length, repetition behavior, and policy entropy.

2. Method

2.1. Dou Shou Qi setting

As shown in Figure 1, Dou Shou Qi (Animal Chess) is a two-player, turn-based strategy board game, and each player has eight animal pieces of different ranks. A 7x9 board is selected to construct rivers, traps and dens for all the players in a game. The order from top to bottom of the pieces is: Elephant (8), Lion (7), Tiger (6), Leopard (5), Wolf (4), Dog (3), Cat (2) and Rat (1). According to the rule for capture, only enemy pieces of the same or lower rank can be captured. The exception is the Rat-Elephant rule; the Rat beats the Elephant, but the Elephant does not beat the Rat. Most pieces move one square horizontally or vertically. Only the Rat is allowed to enter the river; otherwise, the Lions and Tigers cannot jump over the river unless the path is clear of a Rat. A piece in the opponent's trap loses the advantage of rank and can be taken by any enemy piece. A player wins by entering the opponent's den or taking all the opponent's pieces, and loses when there are no legal moves. A game will be generated after 300 moves have been played. In short, the above asymmetrical rules form a competitive system with strong strategic interaction.

DOI: 10.54254/2755-2721/2026.38740



Figure 1. Visual interface of the Dou Shou Qi environment, showing the board layout, rivers, traps, dens, and animal pieces (photo/picture credit: original)

A 7x9 board with 63 cells and eight pieces on each side is used in the official experiment. The total number of 504 discrete actions is the number of board cells times 8 movement directions. The first four are for the regular one-step movements, and the last four are for the explicit river-jump operations. Illegal behavior is hidden prior to action sampling or greedy selection. The observed data are 19×9×7 tensors of observations. The first eight planes are for the current player's pieces, the next eight planes are for the opponent's pieces, and planes 16-18 are the river, traps and dens, respectively. The 48 parallel environments for training are used, and the maximum length of an episode is 300 moves.

2.2. Proximal policy optimization and model architecture

Proximal Policy Optimization (PPO) will be selected as the reinforcement learning algorithm [11]. PPO is an on-policy policy gradient method that improves the stability of training by limiting how far the new policy can deviate from the policy used to generate the rollout. A Policy Change is Represented by a Probability Ratio

DOI: 10.54254/2755-2721/2026.38740

$$r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \tag{1}$$

Here, $r_t(\theta)$ denotes the probability ratio between the current policy and the previous policy for the action selected at time step t. The parameter θ represents the parameters of the current policy, while θ_{old} denotes the parameters of the previous policy used to collect the rollout data. The term s_t is the state observed at time step t, and a_t is the action taken in that state. The notation $\pi_\theta(a_t s_t)$ represents the probability assigned to action a_t by the current policy given state s_t, whereas $\pi_{\theta_{\text{old}}}(a_t s_t)$ is the corresponding probability under the previous policy. This ratio measures how much the current policy changes the probability of the sampled action relative to the policy that generated the training data.

The capped objective reduces the motivation for large-scale policy changes, as shown in equation (2).

$$L^{CLIP}(\theta) = E_t[min(r_t(\theta)A_t, clip(r_t(\theta), 1-\epsilon, 1+\epsilon)A_t)] \tag{2}$$

In this expression, $L^{\text{CLIP}}(\theta)$ denotes the clipped PPO policy objective, and $E_t[\cdot]$ represents the expectation over the sampled training time steps. The probability ratio $r_t(\theta)$ is defined above, and A_t denotes the advantage estimate at time step t, which measures how much better or worse the selected action performs compared with the expected value of the current state. Generalized Advantage Estimation (GAE) is used to obtain the advantage estimates [12]. The clip$(\cdot)$ operator restricts the probability ratio to the interval $[1-\epsilon, 1+\epsilon]$, where ϵ is the clipping parameter that controls the maximum allowed policy change in a single update. The $\min(\cdot,\cdot)$ operator selects the more conservative of the unclipped and clipped objectives, thereby preventing excessively large policy updates while maintaining the direction of the policy improvement.

Since the implementation reduces the loss, the policy-loss term in the optimization is the negative of the clipped surrogate objective defined above. The actual training goal is as follows: Policy Loss + Value Loss + Entropy Regularization.

$$L = L_{policy} + 0.5L_{value} - 0.01H \tag{3}$$

Here, L denotes the total training loss optimized by the PPO learner. L_{policy} is the PPO clipped policy loss, L_{value} is the clipped value-regression loss, and H is the policy entropy. The policy and value functions share a convolutional residual backbone. The input layer receives the 19×9×7 observation and maps it to 64 feature channels using a 3×3 convolution, followed by GroupNorm and ReLU. This is followed by four residual blocks [13]. Each residual block contains two 3×3 convolutional layers with GroupNorm and ReLU activation, together with a residual skip connection.

The shared representation is then processed by separate policy and value heads. The policy head uses a 1×1 convolution from 64 to 2 channels, followed by GroupNorm, ReLU, flattening, and a linear layer that outputs 504 action logits. The value head uses a 1×1 convolution from 64 to 1 channel, followed by GroupNorm, ReLU, flattening, a linear layer from 63 to 64 units, ReLU, and a final linear layer with one output followed by a Tanh activation. The complete network contains 375,375 trainable parameters. Before the categorical policy distribution is constructed, illegal actions are assigned a logit of -10^9, ensuring that only legal actions can be sampled.

DOI: 10.54254/2755-2721/2026.38740

2.3. Experimental hyperparameters and evaluation metrics

All formal reward-design experiments use the same environment, network architecture, PPO configuration, and training budget so that differences between reward conditions can be attributed primarily to reward design. The nominal training budget is 1,000,000 environment steps, with 48 parallel environments and 128 rollout steps per update. Evaluation is performed every 5,000 environment steps. The initial learning rate is 3×10^{-4} and is linearly annealed during training according to the remaining training progress. The Adam optimizer is used with $\epsilon= 10^{-5}$. Other PPO settings are γ=0.99, GAE λ=0.95, policy clipping coefficient 0.2, value clipping coefficient 0.2, entropy coefficient 0.01, value coefficient 0.5, maximum gradient norm 0.5, four optimization epochs, and a minibatch size of 256. Advantage normalization and automatic mixed precision are enabled. Three independent seeds are used for each reward-ablation configuration.

Evaluation will be based on learning efficiency and behavior. The first learning efficiency index is the number of environment steps needed to reach and keep a performance of 0.95. A run is considered to have reached the sustained 0.95 when its evaluation performance first reaches 0.95 or higher and remains at or above 0.95 for all subsequent evaluation points. Therefore, instead of a simple first-threshold crossing, the criterion of RL learning curves may be used. There are 200 games per evaluation area, the starting team is reversed, drawn games are given a score of 0.5, and a fixed random opponent is selected. To compare the results of independent runs, the study reports the interquartile mean (IQM) with a 95% bootstrap confidence interval and does not select an advantageous seed after training. Besides learning efficiency and winning rate, other indicators for whether different strengths of reward shaping lead to significant changes in the learned behavior are episode length, repetition behavior and policy entropy.

3. Results and discussion

3.1. Effect of reward shaping on learning efficiency

Table 1 shows the learning-efficiency results, which are measured by the number of environment steps needed to reach sustained 0.95 performance against a random opponent. The E3 condition is a combination of reward shaping and repetition handling; the other reward components remain unchanged, and after 65,584 steps in the IQM environment (95% bootstrap confidence interval: 45,718-78,589), it reached a stable value of 0.95; the E4 no-shaping condition required 100,400 steps (95% bootstrap confidence interval: 92,874-114,787). Therefore, IQM has a difference of 34,816 fewer environment steps and is about 34.7% lower than E4. The direction of the difference was the same for all three seeds, and E3 reached the stable threshold earlier than E4 in all the experiments.

Figure 2 is the seed-level learning-efficiency results and IQM and its 95% bootstrap confidence intervals.

DOI: 10.54254/2755-2721/2026.38740

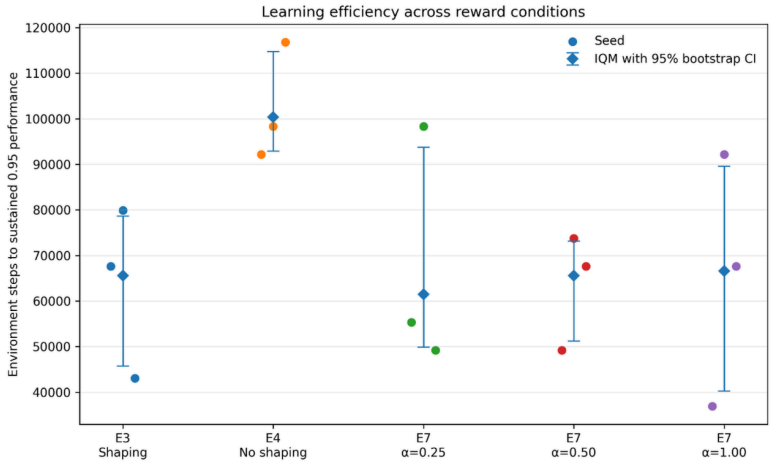


Figure 2. Learning efficiency under different reward conditions, showing seed-level results and IQM with 95% bootstrap confidence intervals (photo/picture credit: original)

Based on the above results, the addition of a shaping module accelerated the attainment of stable high performance under the tested conditions. Based on the above analysis, reward shaping can be employed to provide learners with intermediate reward signals and enhance learning efficiency when end-of-task feedback is absent. Only three independent seeds are used for each configuration in the ablation study; therefore, the results can only be taken as a reference from the tested runs, and it cannot be generally concluded that reward shaping improves sample efficiency.

Table 1. Learning efficiency under reward-shaping conditions

Condition	Seeds	IQM sustained-0.95 steps	95% bootstrap confidence interval
E3: shaping	3	65,584	45,718–78,589
E4: no shaping	3	100,400	92,874–114,787
E7: α=0.25	3	61,488	49,866–93,693
E7: α=0.50	3	65,584	51,197–73,110
E7: α=1.00	3	66,608	40,240–89,546

3.2. Effect of shaping strength

The E7 experiment was conducted to determine whether increasing the strength of reward shaping leads to a general improvement in learning efficiency. The IQM for sustained-0.95 performance was 61,488 environment steps at α=0.25, 65,584 at α=0.5, and 66,608 at α=1.0. The corresponding 95% bootstrap confidence intervals were 49,866–93,693, 51,197–73,110, and 40,240–89,546. Rather than decreasing with an increase in α, the number of environment steps needed to achieve stable performance increased in all three environments. The seed-level results are also significantly different at α=0.25 and α=1.0. Therefore, it cannot be concluded that stronger shaping is more effective for improving learning, based on these changes. Therefore, in the range investigated here, an increase in α did not lead to improved learning efficiency. The results show that, instead of a monotonic trend, the effect of shaping strength varies according to the training dynamics that arise in different reward environments.

DOI: 10.54254/2755-2721/2026.38740

3.3. Behavioral effects of reward parameters

Behavioral statistics provide additional information that cannot be obtained from the final win rate. In the final logged evaluation, all tested configurations achieved a 100% win rate against a random opponent, so there was no further improvement for this benchmark. The Behavioral Indicators are not the same under different Reward Settings. The mean length of the E3 shaping condition (54.50) was shorter and its repetition rate lower (0.97%) than that of E4 (73.96 and 1.86%, respectively); however, the evaluation win rates were the same. Therefore, reward design can be applied to modify the behavior of trajectory following even when task-level performance has reached a plateau against a random opponent.

The E6 repetition-penalty sweep provides a more direct test of behavioral sensitivity to reward parameters. Mean repetition rate was 1.15%, 1.86%, and 1.96% for λ_{rep} =0.002, 0.01, and 0.05, respectively. Thus, increasing the penalty coefficient did not produce the expected monotonic reduction in repetition. Episode length was similarly non-monotonic, with means of 101.97, 73.96, and 95.87 steps. The end-of-training mean episode return became progressively more negative as the penalty increased, from −0.0035 to −0.0300 and −0.1143. Because increasing the penalty coefficient directly changes the magnitude of the repetition-related reward component, this return measure is treated as a behavioral diagnostic rather than a direct performance comparison.

Figure 3 shows the repetition rate of the three repetition-penalty strengths in E6.

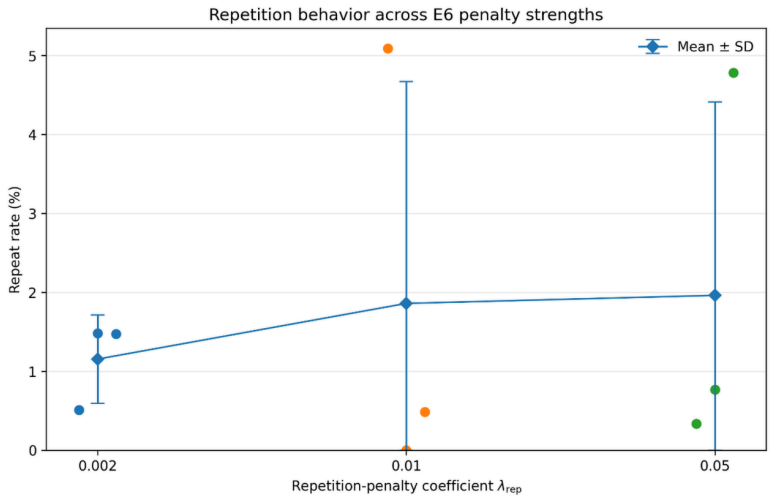


Figure 3. Repetition rate at different strengths of repetition penalty in E6, showing the mean ± standard deviation for the three seeds (photo/picture credit: original)

A typical one appears in E7. As α increased from 0.25 to 1.0, the mean episode length increased from 59.01 to 98.05, the repetition rate rose from 1.10% to 2.50%, and the policy entropy grew from 0.947 to 1.270. On the other hand, the point estimates of learning efficiency were relatively close in the three environments. Together, these results indicate that reward-shaping strength can significantly change the behavior induced by self-play training without causing a corresponding monotonic change in the main learning-efficiency index.

Figure 4 is the change in mean end-of-training episode length for the three E7 shaping coefficients.

DOI: 10.54254/2755-2721/2026.38740

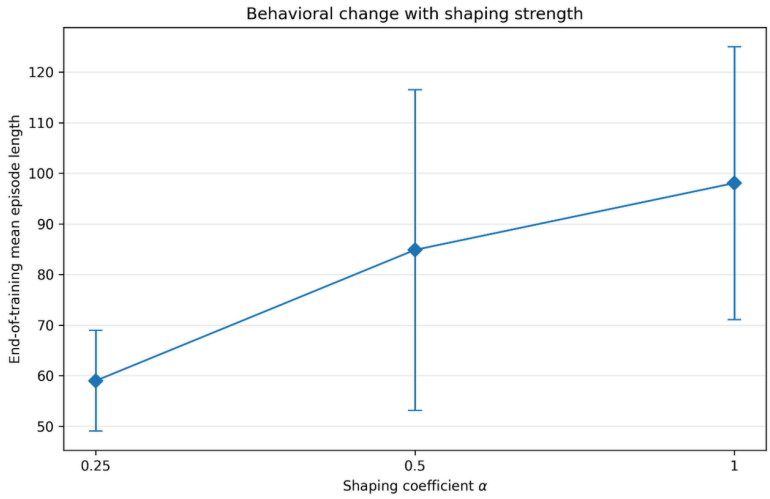


Figure 4. End-of-training mean episode length for different shaping coefficients in E7, with the mean and standard deviation of three seeds shown (photo/picture credit: original)

Table 2 shows the general behavior data under the various reward conditions.

Table 2. Behavioral characteristics at the end of training under different reward conditions, with means and standard deviations reported for the three seeds

Condition	Episode length	Repeat rate	Policy entropy
E3	54.50 ± 41.79	0.97% ± 1.65%	0.754 ± 0.501
E4	73.96 ± 56.10	1.86% ± 2.81%	1.023 ± 0.258
E6, λ=0.002	101.97 ± 30.04	1.15% ± 0.56%	1.260 ± 0.190
E6, λ=0.01	73.96 ± 56.10	1.86% ± 2.81%	1.023 ± 0.258
E6, λ=0.05	95.87 ± 15.00	1.96% ± 2.45%	1.209 ± 0.141
E7, α=0.25	59.01 ± 9.94	1.10% ± 1.10%	0.947 ± 0.275
E7, α=0.50	84.86 ± 31.66	1.92% ± 1.81%	1.158 ± 0.219
E7, α=1.00	98.05 ± 26.94	2.50% ± 1.53%	1.270 ± 0.079

3.4. Discussion

In short, the three main reasons are: Firstly, the tested shaping condition achieved a stable high-performance state in fewer environment steps compared to the no-shaping condition. Second, an increase in shaping strength did not boost the efficiency of learning, and behavior statistics under different reward conditions were significantly different. Both a relatively high win rate for a saturated random opponent and altered behavior indicators have been observed; thus, only random opponents cannot be used for evaluation. Based on the above results, future work should consider the difficulty and variety of opponents in the evaluation, and the Design of rewards may also need to take into account learning processes and behavior changes. The three-seed ablations also show that the effect of reward design needs several rounds. Collectively, the above results show that when evaluating the design of rewards, efficiency in learning and the type of behavior displayed by an agent should be considered in addition to achieving high performance.

DOI: 10.54254/2755-2721/2026.38740

4. Conclusion

Research on the Effects of Reward Design Parameters on Learning Efficiency and Behavior in Self-Play Reinforcement Learning for Dou Shou Qi. Using controlled reward-ablation experiments, reward shaping was compared with no shaping and various strengths of shaped and repeated penalties across three independent seeds. Reward shaping was related to a faster achievement of sustained high performance, but stronger shaping did not result in a monotonic improvement in learning efficiency. The Reward Parameters also resulted in an extension of the episode length, repetition rate and policy entropy. A deficiency is the small number of seeds and the use of a random opponent. More seeds and powerful, diverse opponents will be added in the future for robustness and generalization tests.

References

[1] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., et al. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature, 529*(7587), 484–489. https://doi.org/10.1038/nature16961

[2] Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., et al. (2017). Mastering the game of Go without human knowledge. *Nature, 550*(7676), 354–359. https://doi.org/10.1038/nature24270

[3] Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., et al. (2018). A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play. *Science, 362*(6419), 1140–1144. https://doi.org/10.1126/science.aar6404

[4] Ng, A. Y., Harada, D., & Russell, S. (1999). Policy invariance under reward transformations: Theory and application to reward shaping. In *Proceedings of the International Conference on Machine Learning (ICML)* (pp. 278–287).

[5] Devlin, S., Kudenko, D., & Grześ, M. (2011). An empirical study of potential-based reward shaping and advice in complex, multi-agent systems. *Advances in Complex Systems, 14*(2), 251–278. https://doi.org/10.1142/S0219525911003031

[6] Pocius, R., Isele, D., Roberts, M., & Aha, D. W. (2018). Comparing reward shaping, visual hints, and curriculum learning. In *Proceedings of the AAAI Conference on Artificial Intelligence, 32*(1).

[7] Mguni, D., Jafferjee, T., Wang, J., Perez-Nieves, N., Song, W., Tong, F., et al. (2023). Learning to shape rewards using a game of two partners. In *Proceedings of the AAAI Conference on Artificial Intelligence, 37*(10), 11604–11612.

[8] Ma, H., Sima, K., Vo, T. V., Fu, D., & Leong, T. Y. (2024). Reward shaping for reinforcement learning with an assistant reward agent. In *Proceedings of the Forty-First International Conference on Machine Learning (ICML)*.

[9] Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., & Meger, D. (2018). Deep reinforcement learning that matters. In *Proceedings of the AAAI Conference on Artificial Intelligence, 32*(1).

[10] Agarwal, R., Schwarzer, M., Castro, P. S., Courville, A. C., & Bellemare, M. G. (2021). Deep reinforcement learning at the edge of the statistical precipice. *Advances in Neural Information Processing Systems, 34*, 29304–29320.

[11] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347. https://arxiv.org/abs/1707.06347

[12] Schulman, J., Moritz, P., Levine, S., Jordan, M., & Abbeel, P. (2015). High-dimensional continuous control using generalized advantage estimation. arXiv preprint arXiv:1506.02438. https://arxiv.org/abs/1506.02438

[13] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (pp. 770–778). https://doi.org/10.1109/CVPR.2016.90