CCLF: A Contrastive-Curiosity-Driven Learning Framework for Sample-Efficient Reinforcement Learning

Chenyu Sun\textsuperscript{1,2,3}, Hangwei Qian\textsuperscript{2,4}\textsuperscript{*} and Chunyan Miao\textsuperscript{1,2,4}\textsuperscript{*}

\textsuperscript{1}Alibaba-NTU Singapore Joint Research Institute
\textsuperscript{2}School of Computer Science and Engineering, Nanyang Technological University
\textsuperscript{3}Alibaba Group
\textsuperscript{4}Joint NTU-UBC Research Centre of Excellence in Active Living for the Elderly (LILY)

\{chenyu002, qian0045\}@e.ntu.edu.sg, aszymiao@ntu.edu.sg

Abstract

In reinforcement learning (RL), it is challenging to learn directly from high-dimensional observations, where data augmentation has recently been shown to remedy this via encoding invariances from raw pixels. Nevertheless, we empirically find that not all samples are equally important and hence simply injecting more augmented inputs may instead cause instability in Q-learning. In this paper\textsuperscript{1}, we approach this problem systematically by developing a model-agnostic Contrastive-Curiosity-driven Learning Framework (CCLF), which can fully exploit sample importance and improve learning efficiency in a self-supervised manner. Facilitated by the proposed contrastive curiosity, CCLF is capable of prioritizing the experience replay, selecting the most informative augmented inputs, and more importantly regularizing the Q-function as well as the encoder to concentrate more on under-learned data. Moreover, it encourages the agent to explore with a curiosity-based reward. As a result, the agent can focus on more informative samples and learn representation invariances more efficiently, with significantly reduced augmented inputs. We apply CCLF to several base RL algorithms and evaluate on the DeepMind Control Suite, Atari, and MiniGrid benchmarks, where our approach demonstrates superior sample efficiency and learning performances compared with other state-of-the-art methods.

1 Introduction

Despite the success of reinforcement learning (RL), extensive data collection and environment interactions are still required to train the agents [Laskin et al., 2020b]. In contrast, human beings are capable of learning new skills quickly and generalizing well with limited practice. Therefore, bridging the gap of sample efficiency and learning capabilities between machine and human learning has become a main challenge in the RL community [Rakelly et al., 2019; Schwarzer et al., 2020; Yarats et al., 2021b; Malik et al., 2021; Sun et al., 2022]. This challenge is particularly vital in learning directly from raw pixels. More recently, data augmentation methods are leveraged to incorporate more invariances, promote data diversity, and thereby enhance representation learning [Laskin et al., 2020b; Yarats et al., 2020; Yarats et al., 2021a]. Ideally, injecting a larger number of augmented samples should lead to a better model with invariances. Nevertheless, a noticeable trade-off is the computational complexity introduced. What’s worse, simply increasing the number of augmented inputs may alter the semantics of samples, which has been empirically shown in our experimental results. Moreover, the samples used for data augmentation are uniformly drawn from the replay buffer [Laskin et al., 2020a; Laskin et al., 2020b; Yarats et al., 2020; Yarats et al., 2021a], which is inefficient as they are not equally important to learn. These assumptions deviate from human-like intelligence, where humans can learn efficiently by curiously focusing on novel knowledge and revisiting old knowledge less frequently. Therefore, replaying the most under-explored experiences and selecting the most informative augmented inputs are the keys to improving sample efficiency and learning capability.

To tackle these challenges, we propose a Contrastive-Curiosity-driven Learning Framework (CCLF) by introducing contrastive curiosity into four important components of RL including experience replay, training input selection, learning regularization, and task exploration without much computational overhead. Inspired by the psychological curiosity that can be externally stimulated, encompassing complexity, novelty, and surprise [Berlyne, 1960; Spielberger and Starr, 2012; Liquin and Lombrozo, 2020], we define the contrastive curiosity based on the surprise conceptualized by the agent’s internal belief towards the augmented inputs. The internal belief is modeled by reusing the contrastive loss term in CURL [Laskin et al., 2020a], which can quantitatively measure the curiosity level without introducing any additional network architecture. With the proposed contrastive curiosity, agents can sample more under-explored transitions from the replay buffer, and select the most informative augmented inputs to encode invariances. This process can significantly reduce the amount of data used in RL without sacrificing the invariances. Thereafter, CCLF further utilizes the contrastive curiosity to regularize both Q-function and encoder by concentrating more on the surprising inputs, and intrinsically rewards agents for exploring under-learned observations.

\textsuperscript{*}Co-corresponding authors
\textsuperscript{1}Accepted by IJCAI 2022
Our contribution can be highlighted as follows. 1) We empirically demonstrate that not all samples nor their augmentations are equally important in RL. Thus, agents should learn curiously from the most important ones in a self-supervised manner. 2) A surprise-aroused type of curiosity, namely, contrastive curiosity, is proposed by reusing the representation learning module without increasing the network complexity. 3) The proposed CCLF is capable of improving the sample efficiency and adapting the learning process directly from raw pixels, where the contrastive curiosity is fully exploited in different RL components in a self-navigated and coherent way. 4) CCLF is model-agnostic and can be applied to model-free off-policy and on-policy RL algorithms. 5) Compared to other approaches, CCLF obtains state-of-the-art performance on the DeepMind Control (DMC) suite [Tunyasuvunakool et al., 2020], Atari Games [Bellemare et al., 2013], and MiniGrid [Chevalier-Boisvert et al., 2018] benchmarks.

2 Related Works

Data Augmentation in Sample-Efficient RL. Data augmentation has been widely applied in computer vision but is only recently introduced in RL to incorporate invariances for representation learning [Laskin et al., 2020b; Yarats et al., 2020; Laskin et al., 2020a]. To further improve the sample efficiency, one approach is to automatically apply the most effective augmentation method on any given task, through a multi-armed bandit or meta-learning the hyper-parameters to adapt [Raileanu et al., 2020]. However, the underlying RL algorithm can become non-stationary and it costs more time to converge. Another approach is to regularize the learning process with observations from different training environments [Wang et al., 2020] or different steps [Yarats et al., 2021a]. By injecting greater perturbations from other tasks and steps, the encoded features and learned policies can become more robust to task invariances. Different from these works, the proposed CCLF primarily focuses on perturbations generated in a single task and step, and selects the most under-learned transition tuples and their augmented inputs. As not all samples nor their augmented inputs are equally important, our work exploits the sample importance to adapt the learning process by concentrating more on under-explored samples. Most importantly, the amount of augmentations can be greatly reduced and the sample efficiency is improved without introducing complicated architectures.

Curiosity-Driven RL. In curiosity-driven RL, agents are intrinsically motivated to explore the environment and perform complex control tasks by incorporating curiosity [Aubret et al., 2019; Sun et al., 2022]. In particular, curiosity is mainly used as a sophisticated intrinsic reward based on state novelty [Bellemare et al., 2016], state prediction errors [Pathak et al., 2017], and uncertainty about outcomes [Li et al., 2021] or environment dynamics [Seo et al., 2021]. Meanwhile, it can also be employed to prioritize the experience replay towards under-explored states [Schaul et al., 2016; Zhao and Tresp, 2019]. However, additional networks are required to model curiosity, which can be computationally inefficient and unstable for high-dimensional inputs with continuous controls. Moreover, none of these works have yet attempted to improve the sample efficiency and resolve the instability caused by data augmentation. CCFDM [Nguyen et al., 2021] is a concurrent work that incorporates CURL with action embedding and forward dynamics to formulate an intrinsic reward. Different from CCFDM, our framework does not require any additional architecture but only reuses the contrastive term in CURL that predicts more stably. More importantly, the proposed CCLF seamlessly integrates the curiosity mechanism into experience replay, training input selection, learning regularization, and environment exploration to concentrate more on under-learned samples and improve the sample efficiency stably.

3 Background

In this paper, we consider a Markov Decision Process (MDP) setting with the state $s_t \in S$, the action $a_t \in A$, the transition probability $P$ mapping from the current state $s_t$ and action $a_t$ to the next state $s_t'$, and the (extrinsic) reward $r_t \in R$. More details are provided in Appendix A.

Soft Actor-Critic (SAC). SAC [Haarnoja et al., 2018] is an off-policy model-free algorithm that learns a stochastic policy $\pi_\theta$ (actor) with state-action value functions $Q_{\phi_1}, Q_{\phi_2}$ (critics), and a temperature $\alpha$ by encouraging exploration through a $\gamma$-discounted maximum-policy-entropy term. However, agents are often required to learn directly from high-dimensional observations $o_t \in O$ rather than states $s_t$ in practice. In this paper, we demonstrate our framework mainly using SAC with raw pixel inputs as the base algorithm.

Contrasative Unsupervised RL (CURL). CURL [Laskin et al., 2020a] utilizes data augmentation and contrastive learning to train an image encoder $f_\delta(o)$ in a self-supervised way, imposing an instance discrimination between similar ($+$) and dissimilar ($-$) encoded states. Given a batch of visual observations $o$, each is augmented twice and encoded into a query $q = f_\delta(o_k)$ and a key $k = f_\delta(o_k)$. The key encoder $f_\delta$ is a momentum-based moving average of the query encoder $f_\delta$ to ensure consistency and stability, and $f_\delta$ is learned by enforcing $q$ to match with $k^+$ while keeping far apart from $k^-$. Let $g(o)$ represent the random image crop augmentation on observations $o$. It should ideally preserve the Q-values s.t. $Q(o, a) = Q(g_i(o), a), \forall o \in O, a \in A, i = 1, 2, 3, \ldots$. DrQ then applies data augmentation to each transition tuple $\tau_t = (\alpha_t, a_t, r_t, d_t, o_{t+1})$ that is uniformly sampled from the replay buffer $B$, where $d_t$ is the done signal. With $K$ augmented next observations $g_k(o_{t+1})$ and $M$ augmented current observations $g_m(o_t)$, the critic $Q_\phi$ can be regularized by averaging over $M$ augmented inputs from $o_t$,

$$
L_\phi(\phi) = \mathbb{E}_{\tau \sim B} \left[ \frac{1}{M} \sum_{m=1}^{M} \left( Q_\phi(g_m(o_t), a_t) - \alpha \log \pi_\theta(a_t \mid g_k(o_{t+1})) \right)^2 \right]
$$

where $\bar{T}_i$ is the soft target value and it can also be regularized by averaging over $K$ augmented inputs from $o_k$.

$$
\bar{T}_i = \frac{1}{K} \sum_{k=1}^{K} \left[ \min_{m=1,2} Q_\phi(g_k(o_t), a_t') - \alpha \log \pi_\theta(a_t' \mid g_k(o_t')) \right].
$$
4 The Proposed CCLF

Contrastive-Curiosity-driven Learning Framework (CCLF) extends the model-free RL to further improve sample efficiency when learning directly from the raw pixels. In particular, it fully exploits sample importance for agents to efficiently learn from the most informative data. Firstly, we re-purpose the contrastive term in CURL [Laskin et al., 2020a] without additional architectures to quantify the contrastive curiosity (Section 4.1). Subsequently, this contrastive curiosity is coherently integrated into four components to navigate RL with minimum modification: augmented input selection (Section 4.2), experience replay (Section 4.3), Q-function and encoder regularization (Section 4.4), and environment exploration (Section 4.5), as illustrated in Figure 1. Without loss of generality, we apply CCLF on the state-of-the-art off-policy RL algorithm, SAC [Haarnoja et al., 2018] as summarized in Algorithm 1. Extensions on other base algorithms are carried out in Section 5.2 and Appendix B.2 and B.3.

4.1 Contrastive Curiosity

Curiosity can be aroused by an unexpected stimulus that behaves differently from the agent’s internal belief. To quantify such surprise-aroused curiosity, we define the agent’s curiosity $c_{ij}$ by the prediction error of whether any two augmented observations $g_i(o)$, $g_j(o)$ are from the same observation $o$,

$$c_{ij} = 1 - \text{IB}(g_i(o), g_j(o)) \in [0, 1]$$

(3)

where IB represents agent’s internal belief of whether $g_i(o), g_j(o)$ are augmented (e.g., randomly cropped) from the same $o$ with similar representations. Since the contrastive loss can be viewed as the log-loss of a softmax-based classifier to match a query $q$ with the key $k$ from the same observation in a batch, it becomes a natural choice for measuring agent’s internal belief $\text{IB}(g_i(o), g_k(o)) = \frac{\exp(q^T W_k\hat{k})}{\exp(q^T W_k\hat{k}) + \sum_{l=1}^{B} \exp(q^T W_k\hat{l})}$, where $B$ is the batch size and $q$ is the query encoder $q = f_{\theta}(g_i(o))$. Moreover, we denote the key encoder $k = f_{\theta}(g_j(o))$ as $k^+$ if its input is the same as that in the query encoder $q$; otherwise, we denote it as $k^-$. An immediate merit is that the contrastive curiosity does not require any additional architecture or auxiliary loss because IB is updated directly through representation learning in a self-supervised way.

A higher contrastive curiosity value indicates that the agent does not believe $q$ is similar to $k^+$ or the agent mistakenly matches $q$ with some $k^-$ instead, which ultimately results in a surprise in a self-supervised way. It further implies that the sampled transition tuple contains novel information that has yet been learned by the agent, and the encoder $f_{\theta}$ is not optimal to extract a meaningful state representation from raw pixels. With the proposed contrastive curiosity in-place, we can integrate different curiosity-driven mechanisms in the proposed CCLF to achieve sample-efficient RL, which is discussed in the following sub-sections.

4.2 Curiosity-Based Augmented Input Selection

Although DrQ [Yarats et al., 2020] has shown that increasing $[K, M]$, i.e., the amounts of augmented inputs on next and current observations respectively, can potentially improve agent’s performances through regularized Q-learning, a crucial trade-off is the introduced higher computational complexity. In addition, more augmented data does not necessarily lead to better performance, as data transformations might alter the semantics and result in the counterproductive performance. To tackle these challenges, we aim to select the most informative inputs for the subsequent learning. Without loss of generality, we assume two inputs are selected from $M$ augmented current observations $o_i$ and similarly two are selected from $K$ augmented next observations $o'_j$ in this paper.

It should be noted that there are various ways to select the most informative augmented inputs, where one straightforward way is to select by least overlap in pixels,

$$i^*, j^* = \arg\min_i \text{Overlap}(g_i(o), g_j(o)) \forall i, j, i \neq j.$$  (4)

However, a more human-like way is to select the most representative inputs based on the curious level conceptualized by the internal belief rather than simple visual overlaps. Therefore, we propose to select the augmented inputs that cause highest contrastive curiosity as defined in Eq. (3),

$$i^*, j^* = \arg\max_i c_{ij} \forall i, j, i \neq j.$$  (5)

In this way, the augmented inputs that are most challenging for matching can be curiously identified since they potentially
contain novel knowledge that has yet been learned; meanwhile, this selection mechanism can help to encode more representative state information from the selected inputs, while the agent’s internal belief can be jointly updated. As a result, an improved encoder that is robust to different views of observations can be trained with fewer inputs, potentially yielding an improvement for the sample efficiency.

4.3 Curiosity-Based Experience Replaying

In the conventional off-policy RL, agents uniformly sample transitions \( \tau \) from the replay buffer to learn policies. Although they can eventually perform a complex task by repeatedly practicing in a trial-and-error fashion, we hypothesize that a more sample-efficient and generalizable way is to revisit the transitions that are relatively new or different more frequently. Therefore, we prioritize the experience replay by assigning different prioritization weights \( w \in [0, 1] \) to all transitions stored in the replay buffer \( \mathcal{B} \). In particular, the prioritization weight is initialized to \( w_0 = 1 \) for any newly added transition tuple. Thereafter, we propose to update the weights of transitions with the overall contrastive curiosity at each training step \( s \),

\[
\begin{align*}
    w_s &= \beta w_{s-1} + \frac{1}{2} (1 - \beta) (c_{i^s,j^s} + c_{i^s,j^s'}) \\
    \text{where } \beta &\in [0, 1] \text{ is a momentum coefficient, and } c_{i^s,j^s}, c_{i^s,j^s'} \text{ are the contrastive curiosity about } o^s \text{ and } o'^s \text{ respectively.}
\end{align*}
\]

The intuition of the momentum update is to maintain a stable update such that the transitions arousing low curiosity will be gradually de-prioritized for learning. Mathematically, the probability of \( \tau_i \) to be replayed is \( p(\tau_i) = \frac{w_i}{\sum_{n=1}^{N} w_n} \) and it becomes small only when \( \tau_i \) has been sampled many times. Hence, more recent and surprising transitions arousing high curiosity can be sampled more frequently to learn.

4.4 Curiosity-Based Regularization

Although agents can benefit from learning the selected complex inputs, it imposes challenges for agents as well, which may cause unstable and poor performances. Hence, it is crucial to adapt the learning process by concentrating more on under-learned knowledge. To achieve this, we propose an adaptive regularization for both Q-function and the observation encoder, guided by the contrastive curiosity in order to learn more from the selected inputs arousing high curiosity. In particular, we modify Eq. (2) and Eq. (1) as

\[
\begin{align*}
    \mathcal{L}_Q(\phi) &= \mathbb{E}_{s \sim \mathcal{B}} \left[ (1 - c_{i^s,j^s}) \mathcal{E}^2_s + c_{i^s,j^s} \mathcal{E}^2_s \right], \\
    \mathcal{T}_i &= (1 - c_{i^s,j^s'}) \mathcal{T}_{i^s} + c_{i^s,j^s'} \mathcal{T}_{i^s'}, \\
    \text{where } &\mathcal{E}_m = Q_\phi(g_m(a_i), a_t) - (r_t^* + \gamma (1 - d_t) \mathcal{T}_t), m = i^*, j^*, \\
    \text{and } &\mathcal{T}_i^k = \min_{l=1,2} Q_{\phi_l}(g_l(a'_k), a'_k) - \alpha \log p_\psi(a'_k | g_l(a'_k)), k = i^*, j^*.
\end{align*}
\]

It is worth noting that this regularized Q-function is rather general to recover other state-of-the-art works as special cases. When all augmented inputs arouse exactly moderate level of curiosity \( c_{i^s,j^s} = c_{i^s,j^s'} = \frac{1}{2} \), the proposed regularization is equivalent to DrQ with \( [K, M] = [2, 2] \). Moreover, when the agent can perfectly match the two augmented inputs with no contrastive curiosity \( c_{i^s,j^s} = c_{i^s,j^s'} = 0 \), it is sufficient to update the Q-function with only one input; when the agent fails to encode any similarity and becomes extremely curious \( c_{i^s,j^s} = c_{i^s,j^s'} = 1 \), it should focus completely on the novel input instead. Both cases can reproduce the work of RAD [Laskin et al., 2020b]. Most importantly, our proposed Q-function regularization enables the agent to adapt the learning process in a self-supervised way that it is fully controlled by the conceptualized contrastive curiosity to exploit sample importance and stabilize the learning process.

Similarly, we also regularize the representation learning in a curious manner, inspired by the solution to the supervised class imbalance problem. To deal with the training set containing under-represented classes, a practical approach is to inversely weight the loss of each class according to their sizes. We follow this motivation to incorporate the contrastive curiosity \( c_{i^s,j^s} \), about the current observations as the weight for the log-loss class \( b \) to update the encoder \( f_\theta \),

\[
\mathcal{L}_f(\theta) = - \sum_{b=1}^{B} \frac{\log \exp(q_f^b(Wk_x^b) + \sum_{l=1}^{N} \log \exp(q_f^l(Wb_i^l))}{\exp(q_f^b(Wk_x^b)) + \sum_{l=1}^{N} \exp(q_f^l(Wb_i^l))},
\]

where samples arousing high contrastive curiosity will be considered as under-represented classes and therefore agents need to adaptively pay more attention during the representation learning by optimizing \( f_\theta \). Meanwhile, agents also jointly re-calibrate a proper internal belief by updating \( W \).

4.5 Curiosity-Based Exploration

Intrinsic rewards can motivate agents to explore actively [Sun et al., 2022], improving the sample efficiency in the conventional RL. While SAC alone can be viewed as the entropy

**Algorithm 1** An Implementation of CCLF on SAC

**Input:** MDP \( \tau_t = (o_t, a_t, r_t^+, d_t, o_{t+1}^t) \), numbers of augmented inputs \([K, M]\), replay buffer \( \mathcal{B} \), training step \( T \), batch size \( B \)

**Parameter:** Observation encoder network \( \theta \), actor network \( \psi \), critics networks \( \phi_j \), temperature coefficient \( \alpha \), and bilinear product weight \( W \)

**Output:** Optimal policy \( \pi^*_t \)

for \( t = 1 \) to \( T \) do

- \( a_t \sim \pi^*_t : (\cdot | g(o_t)) \)
- \( \mathcal{B} \cup (o_t, a_t, r_t^+, d_t, o_{t+1}) \rightarrow \mathcal{B} \) with \( w_t = 1 \)
- Sample a minibatch \( \{(o_t, a_t, r_t^+, d_t, o_{t+1})\}_i \) \( \mathcal{B} \) based on the prioritization weight \( w_t \)

for each sample \( \tau_i \) in the minibatch do

- Augment \( o_t \) and \( o_i \) via \( g(\cdot) \) to obtain \( M \) and \( K \) inputs
- Evaluate the contrastive curiosity \( c_{ij} \) by Eq. (3)
- Select \( g_{ij}(o_t), g_{ij}(o_i) \) from \( M \) augmented \( o_t \) and select \( g_{ij}(o_t'), g_{ij}(o_i') \) from \( K \) augmented \( o_i' \) by Eq. (5)
- \( r_t = r_t^+ + r_i' \) with Eq. (9)
- Update \( w_t \) according to Eq. (6)

end for

Update critics \( Q_{\phi} \) by Eq. (7)

Update the actor \( \pi \) and temperature coefficient \( \alpha \)

Update encoder \( f_\theta \) and \( W \) by Eq. (8)

end for
maximization of agent’s actions intrinsically, in the proposed CCLF, we explicitly define an intrinsic reward proportional to the average contrastive curiosity about $o_t$ and $o_t'$:

$$r_t^i = \lambda \exp(-\eta t) \frac{r_{t, \text{max}}^e q_{t, *}' + c_{t, *}' r_{t, \text{max}}^i}{2}$$

(9)

where $\lambda$ is a temperature coefficient, $\eta$ is a decay weight, $t$ is the environment step, $r_{t, \text{max}}^e$ and $r_{t, \text{max}}^i$ are respectively the maximum extrinsic and intrinsic rewards over step $t$.

With the proposed $r_t^i$ to supplement $r_t^e$ in Eq. (7), agents can be encouraged to explore the surprising states that arouse high contrastive curiosity substantially. In particular, higher $r_t^i$ rewards agents for exploration when different views of the same observations produce inconsistent representations. Meanwhile, $r_t^i$ is decayed with respect to the environment step $t$ to ensure the convergence of policies. As the extrinsic reward $r_t^e$ differs across different tasks, the normalization is performed to balance $r_t^e$ and $r_t^i$. This formulation is similar to the intrinsic reward in CCFDM [Nguyen et al., 2021], but the proposed CCLF does not require a forward dynamic model or action embedding that increases the model complexity.

5 Experiments and Results

5.1 Experimental Setup

We empirically evaluate the proposed CCLF in terms of sample efficiency and ultimate performance, on 6 continuous control tasks from the DMC suite [Tunyasuvunakool et al., 2020], 26 discrete control tasks from the Atari Games [Bellemare et al., 2013] and 3 navigation tasks with sparse extrinsic rewards from the MiniGrid [Chevalier-Boisvert et al., 2018]. In this section, we mainly present the experimental results in the DMC suite with SAC being the base algorithm while detailed settings and results in the Atari Games and the MiniGrid are included in Appendix B.2, B.3, C.2, and C.3. For a comprehensive evaluation in the DMC suite, we include the following baselines to compare against:

- Pixel-based SAC (SAC-Pixel) [Haarnoja et al., 2018]
- CURL [Laskin et al., 2020a]
- DrQ [Yarats et al., 2020] with $[K, M] = [2, 2]$ and a modified augmentation method for consistency.

- Hybrids of CURL and DrQ: CURL+ and CURL++, where contrastive representation learning is integrated to DrQ for $[K, M] = [2, 2]$ and $[5, 5]$ respectively.

- Augmented input selection models: 2 out of 5 inputs for each sample are selected by pixel overlap (Select) via Eq. (4) and contrastive curiosity (Select+) via Eq. (5) without the other curiosity-based components.

The detailed setting of hyper-parameters is provided in Appendix B.1. For our proposed CCLF, we initialize it with $[K, M] = [5, 5]$ to generate a sufficiently large amount of augmented inputs. For simplicity, we fix $\eta$ randomly and only select $j^*$ via Eq. (5) for the augmented input selection.

5.2 Results and Discussion

Not all Samples are Equally Important. In CURL+, data augmentation is applied twice for each sampled transition while 5 times in CURL++. Since CURL++ injects $2.5 \times$ larger amount of inputs than CURL+, its computational complexity increases dramatically. Table 1 shows that CURL++ performs worse than CURL+ in 4 tasks at 100K steps and slightly outperforms CURL+ in only 2 tasks at 500K steps. In Figure 2 and Appendix Figure 5, the learning curve of CURL++ is clearly below CURL+ at first and gradually approaches to the same level as CURL+. Since more augmented inputs may not guarantee the consistency of semantics, additional training is often required for convergence. Therefore, we can empirically validate the hypothesis that not all augmented inputs are equally important and simply increasing the number of augmentations is instead inefficient. A similar result can be found DrQ Appendix F [Yarats et al., 2020].

Main Results on the DMC Suite

The average sample efficiency and asymptotic performance are shown in Table 1 at 100K and 500K environment steps, respectively. Meanwhile, Figure 2 demonstrates the agent’s learning capability over 500K steps. Compared SAC-Pixel to the other models in Figure 2, its performance is not improved in all 6 tasks even until 500K steps while the other models can asymptotically perform well. Thus, it is challenging for the conventional SAC to learn directly from raw pixels and a sample-efficient RL method is needed to aid that.

| 100K Step Scores | SAC-Pixel | CURL | DrQ | CURL+ | CURL++ | Select | Select+ | CCLF |
|------------------|-----------|------|-----|-------|--------|--------|--------|------|
| Finger, Spin     | 230±194   | 686±113 | 784±173 | 780±96 | 735±120 | 699±138 | 696±138 | 768±90 | 944±42 |
| Cartpole, Swingup| 237±149   | 524±179 | 675±174 | 694±87 | 665±122 | 624±182 | 561±181 | 799±61 |
| Reacher, Easy    | 239±183   | 566±226 | 682±86 | 541±190 | 479±216 | 646±171 | 616±284 | 738±99 |
| Cheetah, Run     | 118±13    | 286±65 | 332±36 | 302±50 | 264±53 | 251±26 | 265±95 | 317±38 |
| Walker Walk      | 95±19     | 482±237 | 492±267 | 484±61 | 504±142 | 453±91 | 408±170 | 648±110 |
| Ball in Cup, Catch| 85±130   | 667±197 | 828±131 | 687±260 | 728±143 | 732±223 | 739±132 | 914±20 |

| 500K Step Scores | SAC-Pixel | CURL | DrQ | CURL+ | CURL++ | Select | Select+ | CCLF |
|------------------|-----------|------|-----|-------|--------|--------|--------|------|
| Finger, Spin     | 346±195   | 783±192 | 803±198 | 855±164 | 838±164 | 803±167 | 879±153 | 974±6 |
| Cartpole, Swingup| 330±173   | 847±287 | 858±19 | 853±22 | 852±17 | 855±26 | 837±38 | 869±9 |
| Reacher, Easy    | 307±65    | 956±40 | 939±44 | 933±62 | 937±40 | 939±78 | 906±80 | 941±48 |
| Cheetah, Run     | 85±51     | 440±144 | 536±115 | 518±24 | 495±97 | 417±59 | 470±78 | 588±22 |
| Walker Walk      | 71±52     | 926±26 | 887±126 | 916±27 | 914±24 | 921±27 | 850±64 | 936±23 |
| Ball in Cup, Catch| 162±122  | 956±14 | 956±14 | 951±19 | 956±8 | 949±21 | 949±24 | 961±9 |

Table 1: Performance scores (mean & standard deviation) on DMC evaluated at 100K and 500K environment steps. CCLF outperforms other approaches on 5 out of 6 tasks in both sample efficiency (100K) and asymptotic performance (500K) regimes, across 6 random seeds.
According to Table 1, Select performs better than Select+, on 3 tasks at 100K and 4 tasks at 500K, with more stable learning curves as shown in Figure 2. Indeed, the inputs in Select may contain some invariances to improve sample efficiency and learning capability. However, more under-learned inputs with even richer invariances are present in Select+, and agents cannot adapt the learning process in this model, causing the instability issue in Select+.

To tackle this issue, the proposed CCLF collaboratively adapts the learning process with the selected inputs and contrastive curiosity, so the learning curves become more smooth than others in Figure 2. In particular, CCLF outperforms all baselines in 5 tasks at both 100K and 500K regimes in Table 1. Moreover, it converges much faster than Select+ according to the results at 100K steps. In fact, the proposed CCLF only requires about 50% environment steps to converge to desirable performances on 3 tasks (Ball in Cup, Walker, and Cartpole) as the other baselines. In addition, it even benchmarks on Cheetah-Run and Finger-Spin tasks at 500K steps. Therefore, we can conclude that the proposed CCLF can improve the sample efficiency and learning capabilities of RL agents, with fewer environment interactions and 60% reduced augmented inputs. We also analyze the computational complexity on the Cartpole task by model sizes and training time in Appendix C.1, where CCLF can avoid increasing the training cost dramatically.

**Additional Experiments in Atari Games.** In addition to continuous control tasks, CCLF can also be incorporated into Rainbow DQN [Hessel et al., 2018] to perform discrete control tasks. As shown in Appendix C.2, the proposed CCLF attains state-of-the-art performances in 8 out of 26 Atari Games at 100K steps. In particular, CCLF is superior to CURL in 11 games and DrQ in 18 games, which favorably indicates the effectiveness of improving sample efficiency.

**Further Investigation on MiniGrid.** Apart from off-policy algorithms, we also investigate the compatibility on the on-policy algorithm. More specifically, we apply the proposed CCLF to A2C [Mnih et al., 2016] and RE3 [Seo et al., 2021] to perform navigation tasks with sparse rewards in MiniGrid. We first adapt CCLF to the on-policy algorithm by removing the experience replay component. Note that the input from MiniGrid is already a compact and efficient \(7 \times 7 \times 3\) embedding of partially-observable \(7 \times 7\) grids, so even slight augmentation will induce highly inconsistent learned features. Thus, we directly duplicate the embedding without random augmentations to obtain contrastive curiosity for regularization and intrinsic reward. Figure 3 shows that CCLF exhibits superior sample efficiency and learning capabilities in all three tasks, even model-agnostic with the state-of-the-art curiosity-driven method RE3. In the Empty-16×16 task, our CCLF can reach the optimal level in about 50% and 55% of the training steps of RE3 and A2C, respectively. By comparing the ultimate performance scores, the proposed CCLF obtains 1.63× higher average performance than RE3 in DoorKey-6×6 and 1.3× in DoorKey-8×8.

**Effectiveness of the Proposed RL Components.** One might wonder if the proposed CCLF benefits mainly from one or several curiosity-based components in practice. Hence, we empirically examine the effectiveness of all possible combinations of the four curiosity-driven components on the Cartpole task from the DMC suite. The results are included in Appendix C.4, where it can be concluded that all four components are necessary and important to attain state-of-the-art performances. Our proposed CCLF can navigate all four RL components together to improve the sample efficiency and resolve instability, which demonstrates effective collaboration.

### 6 Conclusion

In this paper, we present CCLF, a contrastive-curiosity-driven learning framework for RL with visual observations, which can significantly improve the sample efficiency and learning capabilities of agents. As we empirically find that not all samples nor their augmented inputs are equally important for RL, CCLF encourages agents to learn in a curious way, exploiting sample complexity and importance systemically.
Acknowledgments
This research is supported in part by the National Research Foundation, Prime Minister’s Office, Singapore under its NRF Investigatorship Programme (NRF Award No. NRF-NRFI05-2019-0002). Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore. This research is supported in part by the Alibaba-NTU Singapore Joint Research Institute, Nanyang Technological University, and in part by the Singapore Ministry of Health under its National Innovation Challenge on Active and Confident Ageing (NCFP Project No. MOH/NIC/COG04/2017) and (NIC Project No. MOH/NIC/HAIG03/2017). H.Qian thanks the support from the Wallenberg-NTU Presidential Postdoctoral Fellowship.

References
[Aubret et al., 2019] Arthur Aubret, Laetiitia Matignon, and Salima Hassas. A survey on intrinsic motivation in reinforcement learning. arXiv:1908.06976, 2019.

[Bellemare et al., 2013] Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. JAIR, 47, 2013.

[Bellemare et al., 2016] Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. NeurIPS, 29:1471–1479, 2016.

[Berlyne, 1960] Daniel E Berlyne. Conflict, arousal, and curiosity. 1960.

[Burda et al., 2018] Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. arXiv preprint arXiv:1810.12894, 2018.

[Chevalier-Boisvert et al., 2018] Maxime Chevalier-Boisvert, Lucas Willems, and Suman Pal. Minimalistic gridworld environment for openai gym. https://github.com/maximecb/gym-minigrid, 2018.

[Cobb et al., 2019] Karl Cobb, Oleg Klimov, Chris Hesse, Tae-hoon Kim, and John Schulman. Quantifying generalization in reinforcement learning. In ICML, pages 1282–1289. PMLR, 2019.

[Fortunato et al., 2017] Meire Fortunato, Mohammad Gheshlaghi Azar, Bilal Piot, Jacob Menick, Ian Osband, Alex Graves, Vlad Mnih, Remi Munos, Demis Hassabis, Olivier Pietquin, et al. Noisy networks for exploration. arXiv preprint arXiv:1706.10295, 2017.

[Haarnoja et al., 2018] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. arXiv preprint arXiv:1812.05905, 2018.

[Hafner et al., 2019a] Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In International Conference on Learning Representations, 2019.

[Hafner et al., 2019b] Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In ICML, pages 2555–2565. PMLR, 2019.

[He et al., 2020] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 9729–9738, 2020.

[Hessel et al., 2018] Matteo Hessel, Joseph Modayil, Hado Van Hasselt, Tom Schaul, Georg Ostrovski, Will Dabney, Dan Horgan, Bilal Piot, Mohammad Azar, and David Silver. Rainbow: Combining improvements in deep reinforcement learning. In AAAI, 2018.

[Kaiser et al., 2019] Lukasz Kaiser, Mohammad Babaeizadeh, Pieter A, Blazej Osisnki, Roy H Campbell, Konrad Czechowski, Dumitru Erhan, Chelsea Finn, Piotr Kozakowski, Sergey Levine, et al. Model-based reinforcement learning for atari. arXiv preprint arXiv:1903.00374, 2019.

[Kieflak, 2019] Kacper Piotr Kieflak. Do recent advancements in model-based deep reinforcement learning really improve data efficiency? 2019.

[Kingma and Ba, 2014] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

[Laskin et al., 2020a] Michael Laskin, Aravind Srinivas, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. In ICML, pages 5639–5650. PMLR, 2020.

[Laskin et al., 2020b] Misha Laskin, K. Lee, A. Stooke, L. Pinto, Pieter Abbeel, and Aravind Srinivas. Reinforcement learning with augmented data. NeurIPS, 33, 2020.

[Lee et al., 2019] Alex X Lee, Anusha Nagabandi, Pieter Abbeel, and Sergey Levine. Stochastic latent actor-critic: Deep reinforcement learning with a latent variable model. arXiv preprint arXiv:1907.00953, 2019.

[LEE et al., 2020] KIMIN LEE, Kibok Lee, Jinwoo Shin, and Honglak Lee. Network randomization: A simple technique for generalization in deep reinforcement learning. In Eighth International Conference on Learning Representations, ICLR 2020. International Conference on Learning Representations, 2020.

[Lei Ba et al., 2016] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. ArXiv e-prints, pages arXiv–1607, 2016.

[Li et al., 2021] Kevin Li, Abhishek Gupta, Ashwin Reddy, Vitchy H Pong, Aurick Zhou, Justin Yu, and Sergey Levine. Murai: Meta-learning uncertainty-aware rewards for outcome-driven reinforcement learning. In ICML. PMLR, 2021.

[Lin et al., 2019] Yijiong Lin, Jiancong Huang, Matthieu Zimmer, Juan Rojas, and Paul Wang. Towards more sample efficiency in reinforcement learning with data augmentation. arXiv preprint arXiv:1910.09959, 2019.

[Liquin and Lombrozo, 2020] Emily G Liquin and Tania Lombrozo. Explanation-seeking curiosity in childhood. Current Opinion in Behavioral Sciences, 35:14–20, 2020.

[Malik et al., 2021] Dhruv Malik, Aldo Pacchiano, Vishwak Srinivasan, and Yuanzhi Li. Sample efficient reinforcement learning in continuous state spaces: A perspective beyond linearity. In ICML. PMLR, 2021.

[Mnih et al., 2013] 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.

[Mnih et al., 2015] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex
Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7549):529–533, 2015.

[Mnih et al., 2016] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *ICML*. PMLR, 2016.

[Nguyen et al., 2021] Thanh Nguyen, Tung M Luu, Thang Vu, and Chang D Yoo. Sample-efficient reinforcement learning representation learning with curiosity contrastive forward dynamics model. *arXiv preprint arXiv:2103.08255*, 2021.

[Oord et al., 2018] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[Paszke et al., 2019] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32:8026–8037, 2019.

[Pathak et al., 2017] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *ICML*. pages 2778–2787. PMLR, 2017.

[Raileanu et al., 2020] Roberta Raileanu, Max Goldstein, Denis Yarats, Ilya Kostrikov, and Rob Fergus. Automatic data augmentation for generalization in deep reinforcement learning. *arXiv preprint arXiv:2006.12862*, 2020.

[Rakelly et al., 2019] Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *ICML*. pages 5331–5340. PMLR, 2019.

[Saxe et al., 2013] Andrew M Saxe, James L McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*, 2013.

[Schaal et al., 2015] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. *arXiv preprint arXiv:1511.05952*, 2015.

[Schaal et al., 2016] Tom Schaul, John Quan, Ioannis Antonoglou, and David Silver. Prioritized experience replay. In *ICLR*, 2016.

[Schwarzer et al., 2020] Max Schwarzer, Ankesh Anand, Rishab Goel, R Devon Hjelm, Aaron Courville, and Philip Bachman. Data-efficient reinforcement learning with self-predictive representations. In *ICLR*, 2020.

[Seo et al., 2021] Younggyo Seo, Lili Chen, Jinwoo Shin, Honglak Lee, Pieter Abbeel, and Kimin Lee. State entropy maximization with random encoders for efficient exploration. *arXiv preprint arXiv:2102.09430*, 2021.

[Spielberger and Starr, 2012] Charles D Spielberger and Laura M Starr. Curiosity and exploratory behavior. In *Motivation: Theory and research*, pages 231–254, 2012.

[Sun et al., 2022] Chenyu Sun, Hangwei Qian, and Chunyan Miao. From psychological curiosity to artificial curiosity: Curiosity-driven learning in artificial intelligence tasks. *arXiv preprint arXiv:2201.08300*, 2022.

[Sutton and Barto, 1998] Richard S Sutton and Andrew G Barto. Reinforcement learning: an introduction mit press. *Cambridge, MA*, 22447, 1998.

[Tieleman et al., 2012] Tijmen Tieleman, Geoffrey Hinton, et al. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.

[Todorov et al., 2012] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.

[Tunyasuvunakool et al., 2020] Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020.

[Van Hasselt et al., 2016] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

[van Hasselt et al., 2019] Hado P van Hasselt, Matteo Hessel, and John Aslanides. When to use parametric models in reinforcement learning? *NeurIPS*, 32:14322–14333, 2019.

[Wang et al., 2016] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Hasselt, Marc Lanctot, and Nando Freitas. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*, pages 1995–2003. PMLR, 2016.

[Wang et al., 2020] Kaixin Wang, Bingyi K, Jie S, and Jiashi F. Improving generalization in reinforcement learning with mixture regularization. In *NeurIPS*, 2020.

[Yarats et al., 2020] Denis Yarats, Ilya Kostrikov, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *ICLR*, 2020.

[Yarats et al., 2021a] Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. *arXiv preprint arXiv:2107.09645*, 2021.

[Yarats et al., 2021b] Denis Yarats, Amy Zhang, Ilya Kostrikov, Brandon Amos, Joelle Pineau, and Rob Fergus. Improving sample efficiency in model-free reinforcement learning from images. In *AAAI*, 2021.

[Zhao and Tresp, 2019] Rui Zhao and Volker Tresp. Curiosity-driven experience prioritization via density estimation. *arXiv preprint arXiv:1902.08039*, 2019.
Appendix A  Extended Background

A.1 Efficiency of Data Augmentation in RL

Data augmentation such as translation, crop, rotation, and cutout has been widely applied in computer vision but is only recently introduced in RL to incorporate invariances for representation learning [Cobbe et al., 2019; Lin et al., 2019; Raileanu et al., 2020; LEE et al., 2020; Laskin et al., 2020b; Yarats et al., 2020; Laskin et al., 2020a]. In particular, RAD [Laskin et al., 2020b] investigates several commonly used data augmentation methods, where random crop and translation are found most effective. Meanwhile, DRQ [Yarats et al., 2020] regularizes the Q-function to ensure that multiple augmentations from the same observation should have similar Q-values. CURL [Laskin et al., 2020a] leverages contrastive learning with data augmentation such that representation learning becomes more robust. When learning directly from raw pixels, these works have outperformed other methods, including pixel SAC [Haarnoja et al., 2018], PlaNet [Hafner et al., 2019b], Dreamer [Hafner et al., 2019a], SLAC [Lee et al., 2019] and SAC+AE [Yarats et al., 2021b]. However, sample importance has not been fully exploited in any data augmentation based RL.

A.2 Soft Actor-Critic (SAC)

We consider a Markov Decision Process (MDP) setting with the state $s_t \in S$, the action $a_t \in A$, the transition probability $P$ mapping from the current state $s_t$ and action $a_t$ to the next state $s'_t$, and the reward $r_t^\gamma \in R$. SAC [Haarnoja et al., 2018] is an off-policy model-free reinforcement learning (RL) algorithm that learns a stochastic policy $\pi_\psi(\cdot|s_t)$ with state-action value functions $Q_{\phi_1}, Q_{\phi_2}$ (critics), and a temperature coefficient $\alpha$ by encouraging exploration through a $\gamma$-discounted maximum-policy-entropy term. However, agents are often required to learn directly from high-dimensional visual observations $o_t \in O$ rather than states $s_t$ in practice.

In particular, the critics are trained by minimizing the squared Bellman error via uniformly sampling transitions $o_t = (s_t, a_t, r_t^\gamma, s_t', a_t')$ from a replay buffer $B$, $L_{Q_{\phi}}(\psi) = \mathbb{E}_{s_t \sim B} \left[ (Q_{\phi_1}(o_t, a_t) - (r_t^\gamma + \gamma(1 - d_t)T_t))^2 \right]$ where $d_t$ is the done signal and the soft target value $T_t$ is implicitly parameterized as $T_t = \min_{i=1,2} Q_{\phi_i}(o_t', a_t') - \alpha \log \pi_\psi(a_t'|o_t')$. $Q_{\phi_1}$ is the exponential moving average (EMA) of $Q_{\phi_1}$, which imposes training stability, and $a_t'$ is sampled stochastically by the actor network using the next observation $o_t'$.

The actor network $\pi_\psi(a|o)$ is a parametric tanh-Gaussian which stochastically samples $a = \text{tanh}(\mu_{\psi}(o) + \sigma_{\psi}(o)\epsilon)$ given the observation $o$, where $\epsilon \sim N(0, 1)$, and $\mu_\psi$ and $\sigma_\psi$ are the parametric mean and standard deviation, respectively. Meanwhile, the actor network is trained by minimizing the divergence from the exponential of the soft-Q function, equivalent to

$$L_\pi(\psi) = -\mathbb{E}_{o \sim \pi} \left[ \min_{i=1,2} Q_{\phi_i}(o_t, a) - \alpha \log \pi_\psi(a|o_t) \right].$$

Finally, $\alpha$ controls the priority of entropy maximization and is optimized against a target entropy as $L(\alpha) = \mathbb{E}_{o \sim \pi} \left[ -\alpha \log \pi_\psi(a|o_t) - \alpha \hat{H} \right]$ where $\hat{H} \in \mathbb{R}$ represents the target entropy with $\hat{H} = -|A|$.

A.3 Rainbow DQN

Rainbow DQN [Hessel et al., 2018] is an off-policy deep RL algorithm for discrete action space, extending the conventional DQN [Mnih et al., 2015] with multiple improvements of other DQN methods into a single learner. In particular, it utilizes the Double Q-Learning method [Van Hasselt et al., 2016] to resolve overestimation bias. As an off-policy algorithm, it employs the Prioritized Experience Replay technique [Schaul et al., 2015] to sample the novel transitions with high TD-errors more frequently. In addition, the dueling critic architecture network [Wang et al., 2016] is incorporated to learn valuable states, avoiding determining the effect of all actions and states. Instead of the expected return, it uses distributional reinforcement learning to output a distribution over possible value functions, with $n$-step learning [Sutton and Barto, 1998] and noisy linear layers [Fortunato et al., 2017] for better exploration. Combining all above-mentioned techniques and architectures, Rainbow DQN can obtain state-of-the-art sample efficiency on Atari Games [Bellemare et al., 2013] benchmarks. To further enhance the sample efficiency, an optimized configuration of Rainbow DQN hyperparameters is proposed to benchmark Atari Games at 100K training steps [Van Hasselt et al., 2019].

A.4 Advantage Actor-Critic (A2C)

In contrast to SAC and Rainbow DQN from off-policy RL algorithms, Advantage Actor-Critic (A2C) [Mnih et al., 2016] is an on-policy algorithm that combines the policy-based method (an actor) and value-based method (a critic), with an advantage function. While the value function measures how good the agent is at each state, the advantage function $A(s, a)$ captures how better an action $a$ against the others at a given state $s$, $A(s, a) = Q(s, a) - V(s)$.

Subsequently, the actor can leverage this information to optimize its policy. As the result, the high variance of policy gradient can be greatly reduced to stabilize the model.

Based on A2C, RE3 [Seo et al., 2021] proposed to incorporate the state entropy as an intrinsic reward to encourage efficient exploration, which can be estimated with a fixed random encoder. RE3 has been shown to be capable of improving the sample efficiency on navigation tasks from the MiniGrid [Chevalier-Boisvert et al., 2018] benchmark.

A.5 Contrastive Unsupervised RL (CURL)

To learn a meaningful representation from raw pixels, CURL [Laskin et al., 2020a] utilizes data augmentation and contrastive learning to train an image encoder $f_\theta(o)$ in a self-supervised way, imposing an instance discrimination between similar and dissimilar encoded states. Given a batch of visual observations $o$, each is augmented twice and encoded into a query $q = f_\theta(o_q)$ and a key $k = f_\theta(o_k)$, where the positive
keys \( k^+ \) indicate that both \( o_k \) and \( o_q \) are augmented from the same \( o \) and otherwise negative keys \( k^- \). The key encoder \( f_k \) is a momentum-based moving average [He et al., 2020] of the query encoder \( f_q \) to ensure consistency and stability, where \( f_q \) is learned by enforcing \( q \) to match with \( k^+ \) and keep far apart from \( k^- \). The contrastive loss term to optimize the query encoder is defined as

\[
\mathcal{L}_i(\theta) = -\frac{1}{B} \sum_{b=1}^{B} \log \frac{\sin(q_b, k^+)}{\sin(q_b, k^-) + \sum_{i=1}^{B-1} \sin(q_b, k_i^-)}
\]

where \( B \) is the batch size and \( \sin(q, k) = q^T W k \) is the bi-linear inner-product to measure the similarity between \( q \) and \( k \) [Oord et al., 2018].

### Appendix B Experimental Settings

We conduct experiments on four cloud servers and one physical server with the following configurations.

- **Operation System:** Ubuntu 18.04
- **Memory:** 32GiB / 32GiB / 32GiB / 32GiB / 128GiB
- **CPU:** Intel Core Processor (Skylake) / Intel Core Processor (Skylake) / Intel Core Processor (Skylake) / Intel Core Processor (Skylake) / Intel(R) Xeon(R) CPU E5-2620 v2 @ 2.10GHz
- **vCPU:** 8 / 8 / 16 / 16 / 24
- **GPU:** 2 \times NVIDIA Tesla P100 16GB / 2 \times NVIDIA Tesla P100 16GB / 1 \times NVIDIA Tesla V100S PCIE 32GB / 1 \times NVIDIA Tesla V100S PCIE 32GB / 2 \times NVIDIA GeForce RTX 3090 24GB

Our proposed CCLF is implemented using PyTorch [Paszke et al., 2019] based on CURL [Laskin et al., 2020a], where the contrastive learning module is incorporated to jointly improve the representation learning and Q-learning. As CCLF is model-agnostic, it can be applied to different base RL algorithms with some adaptations. In this paper, we mainly focus on SAC [Haarnoja et al., 2018] as the base algorithm while additional experiments with Rainbow DQN [Hessel et al., 2018] and A2C [Mnih et al., 2016] have also been carried out on a diverse range of exploration tasks from Atari Games [Bellemare et al., 2013] and MiniGrid [Chevalier-Boisvert et al., 2018]. The implementation details are as follows.

#### B.1 Implementation of CCLF on SAC for Continuous Control Tasks in DMC Suite

**Actor and Critic Networks**

Both the actor and critic networks employ the image encoder network as described above to encode pixel observations, and their convolution layers share the same weights. During batch learning, these weights can only be updated by the critic optimizer and the contrastive optimizer. In other words, we detach the actor encoder to stop the propagation of the shared convolution layers during the actor update.

For the actor network, the encoded state representation is fed into a 3-layer MLP with the ReLU activation to output the policy mean and covariance for the diagonal Gaussian. Similarly, the critic networks pass the encoded state representation into a 3-layer MLP with the ReLU activation after each layer except for the last one. For both actor and critic networks, we set the hidden dimension to 1024.

It should be noted that the clipped double Q-learning [Van Hasselt et al., 2016] is utilized for critic networks, a popular technique for stability in SAC-based methods. Meanwhile, the target critic network is momentum-based updated by the critic network to impose Q-learning stability. When we train the critics, only the parameters of the critic networks are updated by the optimizer.

**Contrastive Learning**

Following the same implementation of CURL [Laskin et al., 2020a], the contrastive learning module is built with the image encoder networks in the critics. In particular, the query is encoded by the encoder in the critic network and the keys are encoded by the encoder in the target network. The key encoder is also momentum-based updated by the query encoder to avoid instability [He et al., 2020]. To measure the similarity between query and keys, the bi-linear inner-product is utilized and its weight is initialized randomly.

**Training and Evaluation Setup**

During the model training, the 100 \( \times \) 100 visual observations are randomly cropped to 84 \( \times \) 84 pixels. We first collect 1000 transitions using a random policy and store them in the replay buffer. After that, the subsequent transition tuples are collected by sampling actions from the current policy. Meanwhile, the model update is performed by sampling a mini-batch of transition tuples from the replay buffer at each training step. For clarification, the environment step in this section refers to the multiplication of action repeat and training steps. For example, 50K training steps with the action repeat of 2 represents 100K environment steps. For a fair evaluation, all models are trained with 6 random seeds, during which agents are evaluated every 10K environment steps with 10 episodes. During the evaluation, the 100 \( \times \) 100 visual observations are center cropped to 84 \( \times \) 84 pixels and agents perform with the deterministic policy outputted by the actor network instead of sampling a stochastic action.

**Environments**

We evaluate the proposed CCLF from the perspectives of both sample efficiency and ultimate performance on six continuous control tasks in the DMC suite [Tunyasuvunakool et al., 2020]. The DMC suite provides a diverse collection of complex control using MuJoCo physics [Todorov et al., 2012].
with different settings such as sparse rewards and complex dynamics, which are challenging for agents to operate directly from visual observations. The six tasks are commonly used as benchmarks to evaluate the agent’s performance, where the evaluation metric is the performance score in the range of [0, 1000].

**Hyper-parameters**

Throughout all experiments, the network hyper-parameters are kept fixed and consistent unless otherwise stated. We employ the Adam optimizer [Kingma and Ba, 2014] and set the batch size to 512. To capture both spatial and temporal information, 3 consecutive frames are stacked into one observation. The detailed setting of hyper-parameters for baseline models is listed in Table 2, which follows the same settings of CURL [Laskin et al., 2020a] and DrQ [Yarats et al., 2020].

| Hyper-parameters | Value |
|------------------|-------|
| Observation rendering | (100, 100) |
| Observation downsampling | (84, 84) |
| Replay buffer size | 100K |
| Initial steps | 1K |
| Stacked frames | 3 |
| Action repeat | 2 Finger, Spin; Walker, walk |
| Hidden units (MLP) | 1024 |
| Evaluation episodes | 10 |
| Optimizer | Adam |
| Learning rate $(f_\theta, \pi_\psi, Q_\phi)$ | $(9, .999)$ |
| Learning rate $(f_\psi, \pi_\phi)$ | $2e - 4$ |
| Learning rate | $1e - 4$ |
| Batch Size | 512 |
| Initial temperature | 0.1 |
| Q function EMA rate | 0.01 |
| Critic target update frequency | 2 |
| Actor update frequency | 2 |
| Actor log stddev bounds | [-10, 2] |
| Convolutional layers | 4 |
| Number of filters | 32 |
| Activation | ReLU |
| Encoder EMA rate | 0.05 |
| Encoder feature dimension | 50 |
| Discount factor | 0.99 |

Table 2: Hyper-parameters used for CURL and DrQ based models with SAC being the base algorithm. Most hyper-parameters are fixed for all tasks while only action repeat and learning rate vary across different tasks.

For the proposed CCLF, we initialize it with $[K, M] = [5, 5]$ to generate a sufficiently large amount of augmented inputs. In addition to the hyper-parameters set in Table 2, we list the other hyper-parameters used in CCLF in Table 3, which are fixed when evaluating on the six continuous control tasks in the DMC suite.

| Hyper-parameters | Value |
|------------------|-------|
| Momentum coefficient for prioritization | 0.99 |
| Intrinsic reward decay weight | 2e-5 |
| Intrinsic reward temperature | 0.2 |

Table 3: Additional hyper-parameters used for the proposed CCLF on SAC, fixed across all evaluated tasks.

**B.2 Implementation of CCLF on Rainbow DQN for Discrete Control Tasks in Atari Games**

Following the similar setting and rationale in CURL, we are able to extend CCLF on discrete control tasks from Atari100K with minimal modifications. In particular, we select the Rainbow DQN with the data-efficient architecture (Efficient Rainbow) [van Hasselt et al., 2019] as the base algorithm for our proposed model-agnostic CCLF. For the encoder network, actor and critic networks, and contrastive learning module, we follow the exactly same network architectures as CURL Rainbow, from the publicly available repository https://github.com/aravindsrinivas/curl_rainbow. To incorporate the curiosity-driven experienced replay component, we modify the algorithm by replacing the TD-error-based priorization with our proposed contrastive-curiosity-based priorization, where the sampling weights are updated in a momentum manner.

**Training and Evaluation Setup**

During the model training, the $84 \times 84$ visual observations are randomly cropped to $80 \times 80$ pixels, padded by 4 pixels and then randomly cropped again to $84 \times 84$ pixels. We first collect 1600 transitions using a random policy and store them in the replay buffer. After that, the subsequent transition tuples are collected using the outputted actions from the current policy. Meanwhile, the model update is performed by sampling a mini-batch of transition tuples from the replay buffer at each training step. For a fair evaluation, our CCLF is trained with 4 random seeds, during which agents are evaluated every $10K$ training steps with 10 episodes. During the evaluation, the $84 \times 84$ rendered observations are directly inputted into the trained model without further augmentations. We evaluate CCLF from the perspective of sample efficiency at $100K$ training steps on 26 discrete control tasks in the Atari Games.

**Hyper-parameters**

Throughout all experiments, the network hyper-parameters are kept fixed and consistent unless otherwise stated. For a fair comparison and evaluation, the same hyper-parameters used in CURL on the Rainbow DQN are also utilized in our experiments. To further stabilize the model training, we set a coefficient for the contrastive loss term, which is consistent with the CURL method. We employ the Adam optimizer [Kingma and Ba, 2014] and set the batch size to 32. To capture both spatial and temporal information, 4 consecutive frames are stacked into one observation. The detailed setting of hyper-parameters is provided in Table 4 and Table 5.
B.3 Implementation of CCLF on A2C for Navigation Tasks in MiniGrid

In addition to the off-policy RL algorithms, we further demonstrate that CCLF can be applied to the on-policy RL algorithm A2C with minimal modifications. Note that A2C does not employ the replay buffer for experience replay, thus we remove the curiosity-based prioritization component. Moreover, we also apply our CCLF to RE3 that proposed another type of intrinsic reward to encourage entropy-based exploration in the base algorithm of A2C. In particular, for the encoder network as well as actor and critic networks, we utilize the publicly available implementation repository https://github.com/younggyoseo/RE3/tree/master/a2c_re3 with the default hyperparameters for the A2C implementation in MiniGrid benchmark. For the contrastive learning module, the query network is built with the encoder network shared by the actor and critic, while the key network is momentum-based updated from the query. To measure the similarity between query and keys, the bi-linear inner-product is utilized and its weight is initialized randomly.

Training and Evaluation Setup

| Hyperparameter                  | Value                                      |
|---------------------------------|--------------------------------------------|
| Input Size                      | (7, 7, 3)                                  |
| Replay buffer size (for RE3 intrinsic reward) | 10K                                        |
| Stacked frames                  | 1                                          |
| Action repeat                   | 1                                          |
| Evaluation episodes             | 100                                        |
| Optimizer                       | RMSprop                                    |
| Number of processes             | 16                                         |
| Frames per process              | 8                                          |
| Discount                        | 0.99                                       |
| GAE λ                           | 0.95                                       |
| Entropy coefficient             | 0.001                                      |
| Value loss term coefficient     | 0.5                                        |
| Maximum norm of gradient        | 0.5                                        |
| RMSprop ϵ                       | 0.05                                       |
| Clipping ϵ                      | 0.2                                        |
| Recurrence                      | None                                       |
| Training Steps                  | 400K for Empty-16x16 and DoorKey-6x6; 2400K for DoorKey-8x8 |
| Evaluation frequency            | 3200 for Empty-16x16 and DoorKey-6x6; 19200 for DoorKey-8x8 |
| RE3: intrinsic reward coefficient | 0.1 in Empty-16x16; 0.005 in DoorKey-6x6; 0.001 in DoorKey-8x8 |
| RE3: k                          | 3                                          |

Table 6: Hyper-parameters used for baselines of A2C and RE3. Most hyper-parameters are fixed for all tasks while the training steps, evaluation frequency and RE3 intrinsic reward coefficient change across different tasks as specified in RE3 settings.

During the model training, the $7 \times 7 \times 3$ input from MiniGrid is directly passed to the base RL learner and contrastive
learning without augmentations or input selection. It is because any small augmentation on embedded values will cause highly inconsistent learned features. The on-policy agent first collects 8 frames of transitions per process with a total of 16 processes during one update. For a fair evaluation, our CCLF is trained with 5 random seeds and the mean scores are plotted to demonstrate the learning capabilities. We evaluate the proposed CCLF from the perspective of sample efficiency at 400K training steps on Empty-16×16 and DoorKey-6×6 while 2400K steps on DoorKey-8×8 in the MiniGrid, consistent with the baseline settings.

**Hyper-parameters**
In order to carefully examine the benefits of our CCLF without any other changes, we employ the exact same hyper-parameters specified in the RE3 paper appendix Section C [Seo et al., 2021]. Throughout all experiments, the network hyper-parameters are kept fixed and consistent unless otherwise stated. To further stabilize the model training, we set a coefficient for the contrastive loss term. We employ the RMSprop optimizer [Tieleman et al., 2012] and present a full list of hyperparameters that are used for baselines in Table 6 as well as CCLF in Table 7.

| Hyperparameter                                    | Value          |
|---------------------------------------------------|----------------|
| Contrastive: hidden units                         | 128            |
| Non-linearity                                     | ReLU           |
| Momentum (EMA for key encoder update)             | 0.001          |
| Coefficient for contrastive loss                  | 0.0001         |
| Intrinsic reward decay weight                     | 2e-5           |
| Intrinsic reward temperature                      | 2e-4           |

Table 7: Additional hyper-parameters used for the proposed CCLF on A2C and RE3, fixed across different tasks.

**Appendix C  Experimental Results**

**C.1  Additional Results on the DMC Suite**

**Overall Performance**
The full learning performances on all six continuous control tasks are shown in Figure 5, where the proposed CCLF enables the agent to outperform other baseline models in sample efficiency. To compare the overall performance of CCLF against the baselines, we average the performances over the 6 selected control tasks in the DMC suite. The results are shown in Figure 6, where CCLF significantly surpasses the other models at both 100K and 500K environment steps. At 100K environment steps, our CCLF achieves a $1.14 \times$ mean score of the second-best model (DrQ), indicating a significant improvement of the sample efficiency. At 500K environment steps, the average score of CCLF is $1.04 \times$ mean score of the second-best model (CURL+). In a nutshell, our proposed CCLF can obtain state-of-the-art performance from the perspectives of sample efficiency and learning capabilities.

**Comparison of Computational Complexity**

**Model Sizes.** Figure 7 shows the average model size across different models during training. As we employ five servers with different configurations, we take the mean size of the models running on different servers. It can be observed from Figure 7 that when increasing the number of augmentation inputs $[K, M]$ for both DrQ and CURL+DrQ, their model sizes increase substantially, which cause the increase of computational complexity as well. However, CCLF only requires relatively small model sizes and can efficiently reduce the prohibitive cost. In particular, CCLF only selects the two most informative augmented inputs from the five augmented inputs $[K, M] = [5, 5]$ respectively for both current observations and next observations, where the amount of inputs is reduced by 60%. On the one hand, our CCLF reduces 33% and 34.5% of the model sizes, respectively, compared to DrQ [5, 5] and CURL+DrQ [5, 5]. On the other hand, CCLF only slightly increases the model size by 10.1% and 10.6% compared to DrQ [2, 2] and CURL+DrQ [2, 2], and does not exceed DrQ [3, 3] nor CURL+DrQ [3, 3]. Therefore, we can conclude that the proposed CCLF is capable of learning efficiently by avoiding increasing the model complexity during agent learning.

**Training Time.** To compare the computational complexity by model training time, we average on the Cartpole-Swingup task across 6 runs. Figure 4 shows the average training time required by different models. It can be observed that increasing the number of augmented inputs per sample will result in a significant increase (around 33%) of training time by comparing CURL+ (2 inputs) and CURL++ (5 inputs). However, CCLF can reduce about 50% of the increased time caused by CURL++. Overall, it only costs a moderate level of training time increase compared to both DrQ and CURL+. Therefore, we may conclude that our proposed CCLF does not require too much additional running time and can efficiently reduce the unnecessary computational cost by avoiding injecting too many inputs.

![Figure 4: Average training hours on the Cartpole-Swingup task by different models.](image)

**C.2  Results on Atari Games**

We present the results on Arari Games at 100K training steps in Table 8. Overall, the proposed CCLF achieves the state-of-the-art performance on 8 out of 26 games. In particular, we include the following baselines to compare against:

- Random Agent and Human Performance (Human).
- SimPLe [Kaiser et al., 2019], a model-based algorithm.
- OverTrained Rainbow (OTRainbow) [Kielak, 2019].
Figure 5: Comparison of the sample efficiency and learning performances against baselines on the 6 continuous control tasks from the DMC Suite. Our proposed CCLF outperforms the other methods in terms of sample efficiency and converges much faster than the baselines.

Figure 6: Comparison of average scores at 100K and 500K environment scores for DMC Suite

Figure 7: Comparison of model sizes during training
Table 8: Average episode returns on each of 26 Atari games at 100K training steps, across 4 random runs. In each game, the highest score is bold, where the scores of baseline models are listed in both DrQ and CURL papers. The proposed CCLF demonstrates better overall performance on 8 out of 26 games.

| Atari100K Games | Human | Random | SimPLe | OTRainbow | Eff. Rainbow | Eff. DQN | CURL | DrQ | CCLF |
|-----------------|-------|--------|--------|-----------|--------------|---------|------|-----|------|
| ALIEN           | 712.7 | 227.8  | 616.9  | 824.7     | 739.9        | 558.1   | 558.2 | 771.2 | 920.0 |
| AMIDAR          | 1719.5| 5.8    | 88.0   | 82.8      | 188.6        | 63.7    | 142.1 | 102.8 | 154.7 |
| ASSAULT         | 742.0 | 222.4  | 527.2  | 351.9     | 431.2        | 589.5   | 600.6 | 452.4 | 612.4 |
| ASTERIX         | 850.3 | 210.0  | 1128.3 | 628.5     | 470.8        | 341.9   | 734.5 | 603.5 | 708.8 |
| BANK HEIST      | 753.1 | 14.2   | 34.2   | 182.1     | 51.0         | 74.0    | 131.6 | 168.9 | 36.0  |
| BATTLE ZONE     | 37187.5| 2360.0| 5184.4 | 4060.6    | 10124.6      | 4760.8  | 14870.0| 12954.0| 5775.0|
| BOXING          | 12.1  | 0.1    | 9.1    | 2.5       | 0.2          | -1.8    | 1.2   | 6.0   | 7.4   |
| BREAKOUT        | 30.5  | 1.7    | 16.4   | 9.8       | 1.9          | 7.3     | 4.9   | 16.1  | 2.7   |
| CHOPPER COMMAND| 7387.8| 811.0  | 1246.9 | 1033.3    | 861.8        | 624.4   | 1058.5| 780.3 | 765.0 |
| CRAZY CLIMBER   | 35829.4| 10780.5| 62583.6| 21327.8 | 16185.3      | 5430.6  | 12146.5| 20165.2| 7845.0|
| DEMON ATTACK    | 1971.0| 152.1  | 208.1  | 711.8     | 508.0        | 403.5   | 817.6 | 1113.4| 1360.9|
| FREeway         | 29.6  | 0.0    | 20.3   | 25.0      | 27.9         | 3.7     | 26.7  | 9.8   | 22.6  |
| FROSTBITE       | 4334.7| 65.2   | 254.7  | 231.6     | 866.8        | 202.9   | 1181.3| 331.1 | 1401.0|
| Gopher          | 2412.5| 257.6  | 771.0  | 778.0     | 349.5        | 320.8   | 669.3 | 363.6 | 814.7 |
| Hero            | 30826.4| 1027.0| 2656.6 | 6458.8    | 6857.0       | 2200.1  | 6279.3| 3736.3| 6944.5|
| JAMESBOND       | 302.8 | 29.0   | 125.3  | 112.3     | 301.6        | 133.2   | 471.0 | 236.0 | 308.8 |
| KANGAROO        | 3035.0| 52.0   | 323.1  | 605.4     | 779.3        | 448.6   | 872.5 | 940.6 | 650.0 |
| KRULL           | 2665.5| 1598.0| 4539.9 | 3277.9    | 2851.5       | 2999.0  | 4228.6| 4018.1| 3975.0|
| KUNG FU MASTER  | 22736.3| 258.5 | 17257.2| 5722.2    | 14346.1      | 2020.9  | 14307.8| 9111.0 | 12605.0|
| MS PACMAN       | 6951.6| 307.3  | 1480.0 | 941.9     | 1204.1       | 872.0   | 1465.5| 960.5 | 1397.5|
| PONG            | 14.6  | -20.7  | 12.8   | 1.3       | -19.3        | -19.4   | -16.5 | -8.5  | -17.3 |
| PRIVATE EYE     | 69571.3| 24.9  | 58.3   | 100.0     | 97.8         | 351.3   | 218.4 | -13.6 | 100.0 |
| QBERT           | 13455.0| 163.9 | 1288.8 | 509.3     | 1152.9       | 627.5   | 1042.4| 854.4 | 953.8 |
| ROAD RUNNER     | 7845.0| 11.5   | 5640.6 | 2696.7    | 9600.0       | 1491.9  | 5661.0| 8895.1| 11730.0|
| SEAQUEST        | 42054.7| 68.4  | 683.3  | 286.9     | 354.1        | 240.1   | 384.5 | 301.2 | 550.5 |
| UP N DOWN       | 11693.2| 533.4 | 3350.3 | 2847.6    | 2877.4       | 2901.7  | 2955.2| 3180.8| 3376.3|

For consistency, we directly use the average scores of all baseline models that are recorded in DrQ and CURL papers. It can be seen from Table 8 that CCLF can improve over CURL and DrQ on 11 and 18 out of 26 games, with about 18% and 39% performance improvements on average respectively. In particular, we have obtained 2.07×, 1.66×, and 1.65× mean scores than CURL on the Road Runner, Demon Attack, and Alien games. Compared to DrQ, CCLF outperforms it with 2.30×, 1.86×, and 1.83× mean scores on the Freeway, Hero, and Seaquest games. These results have demonstrated the desired sample-efficient capability of the proposed CCLF. In addition, CCLF even achieves superhuman performance on Krull and Road Runner tasks. Although CCLF attains state-of-the-art in 8 games, there are still some gaps, compared to the human performances and the top performing model-based method SimPLe on the other tasks. This shortcoming can also be found in both DrQ and CURL results, where further improvement towards human-level performance is desired.

C.3 Additional Discussion on MiniGrid

As shown in Figure 3 from the main paper, CCLF can significantly improve the sample efficiency as well as the ultimate learning performances, by extending both A2C and RE3 algorithms. In particular, incorporating CCLF directly on A2C is sufficient to outperform RE3 by reaching optimal performance levels faster with even higher scores in all three tasks. Meanwhile, integrating CCLF to RE3 can further improve the sample efficiency. In the DoorKey-8x8 task, our agent started to obtain non-trivial scores at around 700K steps, while conventional A2C has failed even at 2400K step and RE3 started to improve only at around 1200K steps. As RE3 has been empirically proven more effective than other curiosity-driven methods (ICM [Pathak et al., 2017] and RND [Burda et al., 2018]), we can conclude that our CCLF is sample-efficient and is capable of encouraging exploration effectively in the MiniGrid environments.

C.4 Effectiveness of the Proposed Components

One might wonder if the proposed CCLF benefits mainly from one or several curiosity-based components in practice. Hence, we empirically examine the effectiveness of all possible (15) combinations of the proposed four components on the Cartpole task from the DMC suite, averaging by 6 random runs. In this task, agents need to swing up a pole over the cart by continuously moving the cart around.
Adding One Component

We respectively incorporate each curiosity-based component (Selection Only, Prioritization Only, Regularization Only, and Reward Only) into the SAC base algorithm and compare their performances against our full CCLF. It can be observed from Figure 8(a) that all models approximately obtain the same performances at 500K regimes. For the sample efficiency, all four curves are significantly below the proposed CCLF at 100K. Meanwhile, there exists some sudden increase or decrease in the learning curves of all four models, indicating the occurrence of instability. Therefore, each component alone cannot achieve the desired sample efficiency and can even cause unstable learning qualitatively. In contrast, our collaborative CCLF can navigate all components together, which demonstrates effective collaboration.

| Model             | 100K Step Score | 500K Step Score |
|-------------------|-----------------|-----------------|
| Regularization Only | 551±146         | 857±10          |
| Selection Only    | 561±181         | 837±38          |
| Reward Only       | 668±105         | 858±27          |
| Prioritization Only | 670±127       | 858±16          |
| CCLF              | **799±61**      | **869±9**       |

Table 9: Results for the sample efficiency at 100K environment steps and asymptotic performance at 500K environment steps by adding only one proposed component.

In addition, we further compare and analyze the sample efficiency and ultimate performance quantitatively according to the results in Table 9. For the sample efficiency at 100K environment steps, Reward-Only and Prioritize-Only obtain similar performances, which outperform Regularize-Only and Select-Only. However, our proposed CCLF achieves even a 1.19 × mean score at 100K steps compared to the best performance among these four models. Meanwhile, we can observe that the standard deviations of these four models are much higher than our proposed CCLF at both 100K and 500K environment steps. This further implies that each component alone cannot resolve the instability issue that occurred in the learning process. In particular, Select-Only has the highest standard deviation because it selects the most challenging inputs that agents cannot accurately predict for learning. However, as the learning process is not properly adapted to capture the novel knowledge contained in the selected inputs, the instability issues cannot be avoided. CCLF collaboratively adapts the learning process with all four RL components and therefore its standard deviation is the lowest, where the agents can learn with contrastive curiosity. From the perspective of the ultimate performance, the proposed CCLF is also the best compared to the other four models, indicating an improved learning capability as the result of the efficient collaboration of all RL components as well.

Adding Two Components

Two proposed components are incorporated into the base algorithm in this sub-section. It can be observed from Figure 8(b) that all models approximately obtain the same performances at 500K regimes. For the sample efficiency, all six curves are significantly below the proposed CCLF at 100K. Among the six experimented models, Reward+Selection performs the best while Prioritization+Reward learns the worst. It seems that regularization and selection are more important to attain the desired performance. However, only incorporating these two components (Regularization+Selection) may instead introduce some instability, which should be addressed by the remaining two components as a whole. In contrast, our collaborative CCLF can navigate all components together, which demonstrates effective collaboration.

| Model                      | 100K Score | 500K Score |
|----------------------------|------------|------------|
| Prioritization+Reward      | 542±147    | 848±12     |
| Prioritization+Selection   | 655±104    | 859±9      |
| Regularization+Prioritization | 601±137    | 851±13     |
| Regularization+Reward      | 591±52     | 857±12     |
| Regularization+Selection   | 648±135    | 861±11     |
| Reward+Selection           | 654±74     | 858±22     |
| CCLF                       | **799±61** | **869±9**  |

Table 10: Results for the sample efficiency at 100K environment steps and asymptotic performance at 500K environment steps by adding two proposed components.

Quantitatively, we compare and analyze the sample efficiency and ultimate performance based on the results in Table 10. For the sample efficiency at 100K environment steps, Prioritization+Selection, Regularization+Selection and Reward+Selection obtain similar results, which outperform...
the others with Prioritization+Reward being the worst. However, our proposed CCLF achieves even a $1.22 \times$ mean score at 100K steps compared to the best performance among these six models. Meanwhile, we can observe that the standard deviations of these models except for Regularization+Reward are much higher than our proposed CCLF at 100K environment steps. It implies that CCLF can stably adapt the learning process by the contrastive curiosity that seamlessly connects all four components.

**Adding Three Components**

We respectively remove one component from the full CCLF and present the learning performances in Figure 8(c). In particular, removing only prioritization (No Prioritization) or reward (No Reward) only results in a slight downward shift in the learning curves while No Selection and No Regularization show significant performance downgrade in both sample efficiency and learning capabilities. It indicates that all four components are necessarily important to attain state-of-the-art results, where regularization and selection are more important than prioritization and reward.

| Model            | 100K Score | 500K Score |
|------------------|------------|------------|
| No Selection     | 536±141    | 832±35     |
| No Prioritization| 653±89     | 863±10     |
| No Regularization| 565±161    | 837±25     |
| No Reward        | 729±82     | 859±17     |
| CCLF             | 799±61     | 869±9      |

Table 11: Results for the sample efficiency at 100K environment steps and asymptotic performance at 500K environment steps by adding three (removing one) proposed components.

Quantitatively, we also compare and analyze mean scores at 100K and 500K steps, as shown in Table 11. For the sample efficiency at 100K environment steps, No Selection and No Regularization result in 33% and 29% return decreases compared to the full model; meanwhile, their standard deviations are much higher, indicating the occurrence of instability. Similarly, these two models have caused performance downgrades at even 500K steps, with larger standard deviations as well. As a result, we believe regularization and selection play more important roles in CCLF, but it still requires the four components to work collaboratively to obtain the desired sample efficiency and learning performances.