Министерство науки и высшего образования Российской Федерации
Московский физико-технический институт
(национальный исследовательский университет)
Физтех-школа прикладной математики и информатики
Магистерская программа
«Методы и технологии искусственного интеллекта»

Вахитов Родион Радикович

Обучение с подкреплением на основе модели и дискретных моделей мира

Обзор исследований

Научный руководитель:
доцент, д-р физ.-мат. наук
Панов Александр Игоревич

Москва

2024 г.

Содержание

1 Введение **3**

2 Обзор современного состояния проблемы **4**

2.1 Формальная постановка задачи 5

3 Непрерывное латентное пространство **6**

3.1 MuZero 6

3.2 TD-MPC2 8

4 Дискретное латентное пространство **11**

4.1 DreamerV3 11

4.2 EfficientZeroV2 14

5 Объектно-центричное латентное пространство **16**

5.1 Focus 16

6 Заключение **19**

1 Введение

В последние годы обучение с подкреплением (RL) активно развивается, находя применение в задачах от робототехники до управления сложными системами. Одной из ключевых ветвей этой области является обучение с подкреплением на основе модели (model-based reinforcement learning, MBRL), которое показывает значительные преимущества по сравнению с подходами без модели (model-free reinforcement learning, MFRL).

Основное отличие MBRL заключается в использовании внутренней модели окружающей среды, или мира [1], что позволяет агенту прогнозировать будущие состояния и планировать действия, не полагаясь исключительно на взаимодействие с реальной средой. Это представлено на рисунке 1. Одним из ключевых элементов этого подхода является использование латентного, или скрытого, пространства — компактного представления исходных наблюдений, таких как изображения или сенсорные данные. Латентное пространство помогает существенно уменьшить размерность задачи, что упрощает вычисления и делает прогнозирование более эффективным. В модели MBRL наблюдения сначала кодируются в латентное пространство, и именно с этим сжатым представлением работает модель динамики для предсказания будущих состояний и построения плана действий. Такой подход позволяет ускорить процесс обучения, снижая требования к количеству итераций взаимодействия с реальной средой, а также повышает выборочную эффективность (sample efficiency) по сравнению с model-free подходами, которые требуют значительно больше данных и времени на обучение.

Важным направлением исследований в области обучения с подкреплением на основе модели является выбор способа представления латентного пространства. Латентное пространство может принимать различные формы, например, непрерывные или дискретные представления. Эти подходы оказывают значительное влияние на архитектуру модели, а также на способность агента к точному планированию и предсказанию. В задачах, где среда представлена в виде изображений, всё больше внимания уделяется объекто-центричному подходу, который фокусируется на представлении среды через набор отдельных объектов. Одним из таких активно развивающихся методов является слотовое представление, позволяющее декомпозировать изображение на объекты и считать их отдельными дискретными элементами структуры. Примером метода, реализующего такое представление, является SLATE [2]. Такие пред-

ставления способствуют более эффективному планированию за счет фокусировки на отдельных элементах сцены, что делает его особенно полезным в задачах, где важна объектно-ориентированная интерпретация состояния среды.

Еще одним важным аспектом MBRL является его способность к генерализации и переносу знаний на различные задачи. Модели среды, построенные в процессе обучения, могут использоваться повторно в новых задачах, что значительно увеличивает их универсальность. Это особенно важно в ситуациях, когда структура задач или динамика среды сохраняются схожими, что позволяет минимизировать необходимость в повторном обучении модели с нуля.

Таким образом, обучение с подкреплением на основе модели является перспективным направлением, предоставляющим возможность более эффективного и быстрого обучения агентов за счет использования прогнозов моделей среды.



Рис. 1: Сравнение model-free и model-based подходов

2 Обзор современного состояния проблемы

Исследования в области MBRL направлены на улучшение архитектур, которые строят и используют внутренние модели среды, чтобы сделать процесс обучения более эффективным и универсальным. Существует несколько подходов к использованию модели среды, которые включают различные способы сочетания планирования по модели среды и обучения стратегии. В подходах с известной моделью среды модель динамики точная и заранее известна, а обучение стратегии происходит через планирование, как, например, в методах на основе поиска по дереву Монте-Карло (MCTS), таких как AlphaZero [3], наследницей которой является MuZero [4], где модель среды уже обучаемая. Обучение с подкреплением с обучаемой моделью среды подразумевает одновременное обновление модели среды и функций стратегии или полезности. Планирование по обучаемой модели среды предполагает использование

обновляемой модели для локального планирования (на несколько шагов вперед) без обучения глобальной функции стратегии или полезности. Наконец, планирование без модели объединяет процедуру планирования и обучаемую модель в общий оптимизационный цикл, где они неразделимы в плане оптимизируемых функций.

Однако, несмотря на значительные преимущества, MBRL сталкивается с рядом недостатков и нерешённых проблем, которые являются активными направлениями исследований. Во-первых, слабая интерпретируемость нейросетевых моделей динамики приводит к необходимости разработки гибридных нейросимвольных решений, таких как модели на основе графовых нейронных сетей. Во-вторых, нестабильность латентного пространства требует создания более устойчивых методов представления, например, объектно-ориентированных подходов. В-третьих, сложность переноса обученной модели на новые задачи вызывает потребность в адаптивных подходах, учитывающих память и изменяющиеся условия. Наконец, низкая способность к обобщению требует исследования иерархических моделей динамики и представлений для повышения эффективности и универсальности алгоритмов.

Существует множество современных алгоритмов, таких как TD-MPC2 [5], MuZero, DreamerV3 [6], Efficient ZeroV2 [7], Focus [8], каждый из которых предлагает свои решения для повышения точности предсказаний и улучшения планирования. Эти алгоритмы будут рассмотрены в дальнейшем обзоре.

Цель данного обзора — исследовать современные методы обучения с подкреплением на основе модели, уделяя особое внимание представлению латентного пространства и анализу их роли в обеспечении обобщающей способности и эффективности алгоритмов.

2.1 Формальная постановка задачи

Формально, MBRL можно определить как задачу обучения агента в рамках марковского процесса принятия решений (Markov Decision Process, MDP), определяемого кортежем $(\mathcal{S}, \mathcal{A}, T, R, G, \gamma)$, где:

- $\mathcal{S}$ — пространство состояний;
- $\mathcal{A}$ — множество действий (дискретное или непрерывное);
- $T : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ — функция переходов;
- $R : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ — функция вознаграждения;

- $G : \mathcal{S} \to \{0, 1\}$ — функция цели, определяющая состояние завершения;
- γ — коэффициент дисконтирования.

Агент имеет доступ к обновляемой модели среды $M = \langle \hat{T}, \hat{R} \rangle$ и может строить план для достижения цели $G(s_{n+1}) = 1$, моделируя переходы:

$$\text{Plan} = \langle s_i, r_i, a_i, \hat{s}_{i+1}, \hat{r}_{i+1}, a_{i+1}, \ldots, a_{i+n}, \hat{s}_{i+n+1} \rangle \tag{2.1}$$

Цель агента — максимизировать ожидаемое кумулятивное вознаграждение, определяемое стратегией π:

$$\mathbb{E}\pi \left[\sum_{t=0}^{\tau} \gamma^t R(s_t, a_t) \right] \tag{2.2}$$

3 Непрерывное латентное пространство

Непрерывные латентные пространства используют непрерывные распределения для кодирования сложных наблюдений, что позволяет эффективно аппроксимировать плавные изменения в среде. Такие методы хорошо подходят для задач с непрерывной динамикой, например, управления роботами. Однако они требуют больше вычислительных ресурсов и менее эффективны в задачах с чёткими границами между состояниями или действиями.

3.1 MuZero

Алгоритм MuZero — это метод обучения с подкреплением, который эффективно объединяет планирование и обучение без явного знания динамики среды. MuZero использует MCTS, как и AlphaZero. Алгоритм MCTS используется для планирования, где агент выбирает наилучшие действия на основе модели предсказаний, которая производит оценку распределения следующих действий и значения фукнции ценности состоянияю. Модель динамики, в свою очередь, предсказывает вознаграждение в текущем состониии и следующее скрытое состояние. Также есть модель энкодера, который трансформирует наблюдения в скрытое состояние, и модель стратегии. Скрытое представление состояния представляет вектор из непреррывных значений,

полученный детерменированным способом. Данные модели обучается на основе следующей функции потерь:

$$l_\theta(\theta) = \sum_{k=0}^{K} l^r(u_{t+k}, r^k_{t+k}) + l^v(z_{t+k}, v^k_{t+k}) + l^\pi(\pi_{t+k}, \mathbf{p}_{t+k}) + c||\theta||^2 \tag{3.1}$$

где θ — вектор параметров модели, $l_\pi(u_{t+k}, r^k_{t+k})$ — функция потерь для вознаграждения, где u_{t+k} — реальная награда, полученная в момент времени $t+k$, r_{t+k} — оценка награды в момент времени $t+k$, $l_v(z_{t+k}, v_{t+k})$ — функция потерь для ценности состония, где z_{t+k} —реальное значение функции ценности состояния, а v_{t+k} — предсказанное значкение функции ценности состояния. $l_\pi(\pi_{t+k}, \mathbf{p}_{t+k})$ — функция потерь для стратегии, где π_{t+k} — целевая стратегия на шаге $t+k$, полученная через MCTS, и $\mathbf{p}_{t+k}$ — распределение вероятностей действий, предсказанная моделью стратеги. $c||\theta||^2$ — регуляризация для предотвращения переобучения, где c — коэффициент регуляризации, а $||\theta||^2$ — квадратичная норма параметров модели.Алгоритм MCTS является ключевым элементом MuZero для выбора оптимальных действий. На каждом шаге алгоритм MCTS выполняет следующие операции:

1. **Спуск по дереву.** Поиск начинается с корня дерева, соответствующего текущему состоянию. Для каждого узла выбирается действие a, максимизирующее ожидаемую ценность состояния действия $Q(s, a)$ с учетом Upper Confidence Bound (член, отвещающий за баланс межжу исследованеим среды и выполнения уже хорошо изученных траеткорий.

2. **Добавление узлов.** Когда достигается узел, для которого действия не были полностью исследованы, генерируется новое состояние s' на основе предсказания модели динамики. Этот узел добавляется в дерево, и для него вычисляются новые предсказания: стратегия π, ценность v и вознаграждение r.

3. **Симуляция.** После добавления нового состояния запускается симуляция (rollout), где модель динамики предсказывает последовательные действия и состояния, пока не будет достигнуто конечное состояние. Эта симуляция используется для оценки будущей ценности состояния, так как реальные данные о наградах и состоянии на этом этапе отсутствуют. Данная симуляция случайна и просходит с помощью модели стратегии.

4. **Подъем и обновление.** Результаты симуляции — предсказанное вознаграждение r и ценность состония v — передаются вверх по дереву. Статистика посе-

щений узлов обновляется, а значения ценности состояния-действия корректируются для улучшения будущих поисков.

Таким образом, MuZero сочетает предсказательные модели для стратегии, ценности и динамики с методом MCTS для поиска и планирования действий.Принципы работы схематично представлены на рисунке 2. MuZero имеет несколько преимуществ: он не требует явной модели динамики среды, что делает его подходящим для широкого круга задач, и эффективно использует MCTS в сочетании с предсказательными моделями для планирования. Этот алгоритм может обучаться в средах с неизвестной динамикой, что расширяет его области применения. Однако, MuZero имеет и недостатки: его вычислительная сложность велика из-за многократных симуляций в MCTS на каждом шаге, и алгоритм ограничен дискретными пространствами действий. Также существует проблема нестабильности скрытого состояния, что может привести к неточным предсказаниям динамики среды. Кроме того, для обучения и планирования MuZero требуется значительные вычислительные ресурсы.

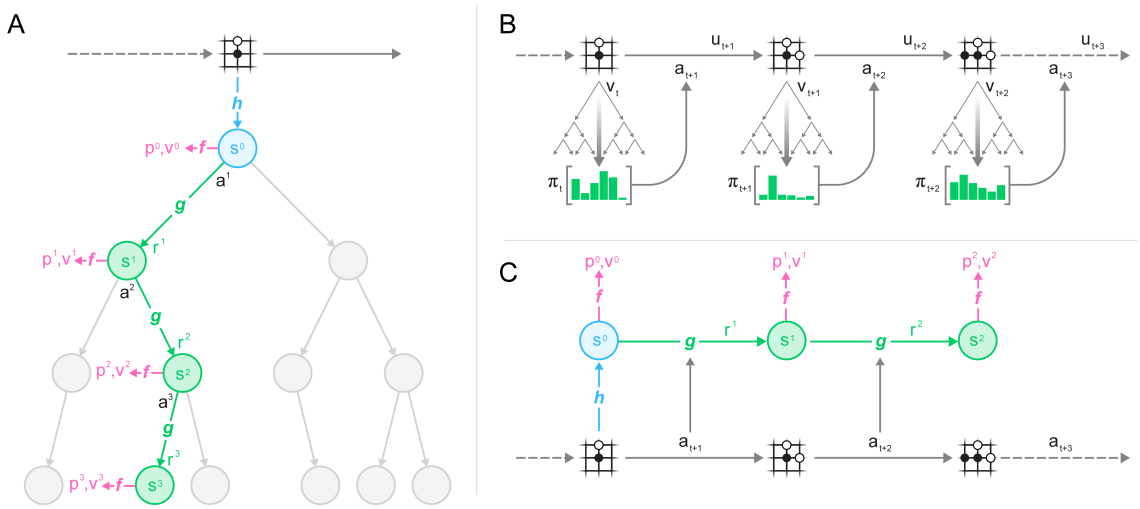

Рис. 2: **(A)** MCTS с моделями динамики, стратегии и предсказания. **(B)** MuZero взаимодействует с окружением через MCTS, выбирая действия по стратегии π_t для каждого состояния. **(C)** Параметры моделей обучаются совместно с использованием обратного распространения ошибки сквозь время для предсказания ключевых величин.

3.2 TD-MPC2

Алгоритм TD-MPC2, являющийся развитием TD-MPC [9], предназначен для решения задач непрерывного управления с помощью обучения с подкреплением на

основе модели, которая обучается с помощью взаимодействий с окружающей средой, объединяя предсказания векторного представления состояния, предсказания вознаграждений и обучение с временным различием. Этот подход позволяет эффективно моделировать большие объемы данных, не требуя декодирования наблюдений. В данном алгоритме скрытое представление состояния является непрерывным. Архитектура алгоритма состоит из следующих компонентов (также она схематично представлена на риунке 3):

$$\begin{aligned} \text{Энкодер:} \quad & \mathbf{z} = h(\mathbf{s}, \mathbf{e}) \\ \text{Модель динамики:} \quad & \mathbf{z}' = d(\mathbf{z}, \mathbf{a}, \mathbf{e}) \\ \text{Оценка награды:} \quad & \hat{r} = R(\mathbf{z}, \mathbf{a}, \mathbf{e}) \\ \text{Оценка ценности состояния-действия:} \quad & \hat{q} = Q(\mathbf{z}, \mathbf{a}, \mathbf{e}) \\ \text{Стратегия:} \quad & \hat{\mathbf{a}} = p(\mathbf{z}, \mathbf{e}) \end{aligned} \tag{3.2}$$

где z — это скрытое векторное представление состояния, а e — векторное представление задачи, выполняемой агентом, а — действие агента в текущем состоянии, $\hat{r}$ — предсказанная награда, $\mathbf{z}'$ — следующее скрытое векторное представление состояния, $\hat{q}$ — предсказанное значение функции ценности состояния-действия, $\hat{a}$ — действие, предсказанное стратегией.

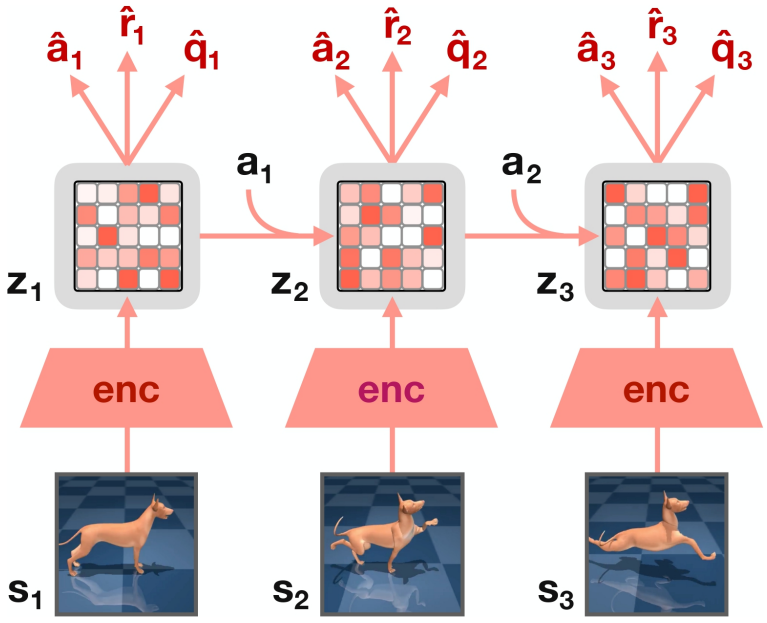

Рис. 3: Архитектура TD-MPC2

Компоненты h, d, R, Q совместно оптимизируются для минимизации общей

функции потерь:

$$\mathcal{L}(\theta) \doteq \mathbb{E}_{(s,a,r,s')/\mathcal{H}\sim\mathcal{B}}\left[\sum_{t=0}^{H}\lambda^t \left\|\mathbf{z}'_t - \text{sg}(h(\mathbf{s}'_t))\right\|_2^2 + \text{CE}(\hat{r}_t, r_t) + \text{CE}(\hat{q}_t, q_t)\right], \tag{3.3}$$

где sg — это оператор стоп-градиента, $(z'_t, \hat{r}_t, \hat{q}_t)$ определены в уравнении (3.2), $q_t \doteq r_t + \gamma\hat{Q}(z'_t, p(z'_t))$ — это целевая переменная TD на шаге t, $\lambda \in (0, 1]$ — коэффициент дисконтирования, а CE — кросс-энтропия. $\hat{Q}$, представляет собой экспоненциально взвешенное скользящее среднее функции ценности состояния-действия Q.

Для управления агентом в среде TD-MPC2 использует метод Model Predictive Path Integral (MPPI), который позволяет находить приближенно оптимальные последовательности действий без вычисления градиентов.

На каждом шаге агент планирует свои действия на горизонт H. Эти последовательности выбираются случайным образом из многомерного гауссовского распределения с параметрами μ и σ, где $\mu \in \mathbb{R}^{H\times m}$ - вектор средних значений действий, $\sigma \in \mathbb{R}^{H\times m}$ - стандартное отклонение, m - размерность пространства действий.

Задача планирования формулируется как максимизация ожидаемой совокупной награды:

$$\mu^*, \sigma^* = \arg\max_{\mu,\sigma} \mathbb{E}_{(\mathbf{a}_t,\ldots,\mathbf{a}_{t+H})\sim\mathcal{N}(\mu,\sigma^2)}\left[\gamma^H Q(\mathbf{z}_{t+H}, \mathbf{a}_{t+H}) + \sum_{h=t}^{H-1}\gamma^h R(\mathbf{z}_h, \mathbf{a}_h)\right] \tag{3.4}$$

где $Q(z_{t+H}, a_{t+H})$ - оценка ценности состояния z_{t+H} после выполнения действия a_{t+H}, $R(z_h, a_h)$ - вознаграждение, полученное в состоянии z_h после выполнения действия a_h.

Ожидаемая совокупная награда, μ и σ оценивается путем моделирования траекторий в латентном пространстве с использованием обученной модели мира на основе взвешенного среднего.

После завершения процесса планирования агент выполняет в среде только первое действие $a_t \sim \mathcal{N}(\mu_t^*, \sigma_t^*)$. Затем процесс повторяется для следующего шага.

Для ускорения сходимости планирования часть последовательностей действий генерируется с использованием модели стратегии, обучаемой на основе функции потерь:

$$\mathcal{L}_{\pi_p} = \sum_{t=0}^{T}\lambda^t\left[\alpha\cdot Q(z_t, \pi_p(z_t)) - \beta\cdot H(\pi_p(a_t|z_t))\right] \tag{3.5}$$

где H — энтропия стратегии, которая способствует исследованию пространства действий.

В TD-MPC2 есть несколько ключевых преимуществ и недостатков. Среди плюсов — это способность эффективно комбинировать планирование и обучение через стратегии (MPPI и π_p), что позволяет динамично адаптироваться к задачам. Модель использует компактные скрытые представления и эмбеддинг задачи, что помогает лучше обобщать задачи и уменьшать требования к данным. Благодаря регуляризации через энтропию, стратегия приоритета способствует улучшению исследования пространства действий.

Однако одним из недостатков является высокая вычислительная сложность MPPI, особенно при длинных горизонтах планирования. Кроме того, качество обучения зависит от точности динамической модели и может ухудшаться, если модель недостаточно хорошо предсказывает будущие состояния.

4 Дискретное латентное пространство

Дискретные латентные пространства представляют собой категориальные распределения, где состояния среды кодируются конечным набором дискретных значений.Этот подход позволяет эффективно работать с задачами, где можно выделить чёткие, различимые классы состояний. Преимущества дискретного представления включают лучшую интерпретируемость и более быструю обработку данных. Рассмотрим алгоритмы, которые используют такой формат скрытого представления состояния.

4.1 DreamerV3

DreamerV3 также представляет MBRL подход, DreamerV3 основан на работах [10], [11] и [12]. В основе DreamerV3 лежит разделение процесса обучения на две ключевые компоненты: обучение модели мира (в данном случае Recurrent State-Space Model, RSSM) и обучение модели стратегии, процесс обучения представлен на рисунке 4. Модель мира строит предсказания о будущем состоянии среды на основании наблюдений, что позволяет агенту "воображать"возможные траектории без необходимости прямого взаимодействия с реальной средой. Стратегия агента обучается исключительно на этих "воображаемых"траекториях, что снижает зависимость от

непосредственных проб и ошибок в реальном мире. Это позволяет значительно экономить вычислительные ресурсы и уменьшать количество взаимодействий с окружением. Более того, сжатие высокоразмерных наблюдений в компактные представления помогает улучшить эффективность использования данных и ускорить процесс обучения.

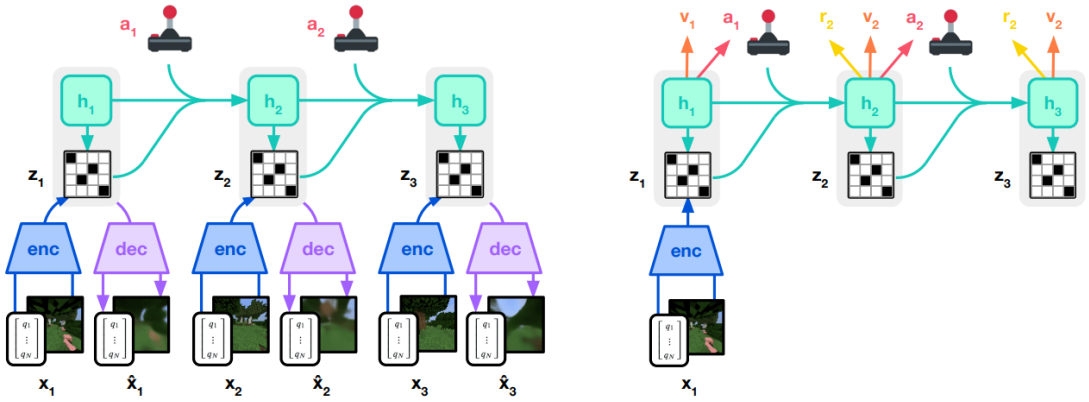

Рис. 4: Процесс обучения DreamerV3

RSSM представляет собой рекуррентную модель, в данном случае GRU, которая сочетает последовательную природу наблюдений и скрытых состояний для предсказания динамики среды. Основные компоненты модели описаны следующим образом:

$$\text{RSSM} \begin{cases} \text{Рекуррентная модель:} \quad h_t = f_\varphi(h_{t-1}, z_{t-1}, a_{t-1}) \\ \text{Энкодер:} \quad z_t \sim q_\varphi(z_t \mid h_t, x_t) \\ \text{Модель динамики:} \quad \hat{z}_t \sim p_\varphi(\hat{z}_t \mid h_t) \\ \text{Оценка награды:} \quad \hat{r}_t \sim p_\varphi(\hat{r}_t \mid h_t, z_t) \\ \text{Предсказание продолжение эпизода:} \quad \hat{c}_t \sim p_\varphi(\hat{c}_t \mid h_t, z_t) \\ \text{Декодер:} \quad \hat{x}_t \sim p_\varphi(\hat{x}_t \mid h_t, z_t) \end{cases} \tag{4.1}$$

где h_t — рукуррентное представление состояния на шаге t, z_t — стохастическое представление состояния, кодирующее текущее наблюдение x_t и рекуррентное состояние h_t, $\hat{z}_t$ — предсказание схохастического представления состояния моделью динамики в следующий момент времени, $\hat{r}_t$ — предсказание награды моделью $p_\varphi(\hat{r}_t \mid h_t, z_t)$, $\hat{c}_t$ — предсказание терминальности состояния моделью $p_\varphi(\hat{c}_t \mid h_t, z_t)$, $\hat{x}_t$ — восстановленное наблюдение декодером $p_\varphi(\hat{x}_t \mid h_t, z_t)$. Конкетанация рекуррентного состояния

и стохастического представление состояния дает скрытое представление состояния среды, с помощью которого производится предсказние награды, оконачания эпизода (для предсказаний используется многослойные перцептроны). Обучается энкодер в парадигме автоэнкодера с декодером, которы испольщуют сверточные нейронные сети.

RSSM обучается по представленной ниже функции потерь:

$$\mathcal{L}(\varphi) \doteq \mathbb{E}_{q_\varphi}\left[\sum_{t=1}^{T}(\beta_{\text{pred}}\mathcal{L}_{\text{pred}}(\varphi) + \beta_{\text{dyn}}\mathcal{L}_{\text{dyn}}(\varphi) + \beta_{\text{rep}}\mathcal{L}_{\text{rep}}(\varphi))\right] \tag{4.2}$$

Функция потерь $\mathcal{L}(\varphi)$ минимизирует три компоненты: потерю предсказания $\mathcal{L}_{\text{pred}}$, потерю динамики $\mathcal{L}_{\text{dyn}}$ и потерю представления $\mathcal{L}_{\text{rep}}$, с весами $\beta_{\text{pred}} = 1$, $\beta_{\text{dyn}} = 1$, и $\beta_{\text{rep}} = 0.1$. Потеря предсказания $\mathcal{L}_{\text{pred}}$ используется для обучения декодера, модели предсказания награды и модели предсказания продолжения эпизода. Декодер и модель предсказания награды обучаются с помощью квадратичной ошибки с использованием симметричной логарифмической потери (symlog), а моделль прдедсказывания продолжения эпизода — через логистическую регрессию. Потеря динамики $\mathcal{L}_{\text{dyn}}$ обучает модель динамики предсказывать следующее стохастическое представление состояния, минимизируя KL-дивергенцию между предсказанным латентным состоянием и истинным стохастическим представлением в момент времени t. Это позволяет модели лучше предсказывать будущее на основе текущих состояний. Потеря представления $\mathcal{L}_{\text{rep}}$ улучшает качество рекуррентный представлений, делая их более предсказуемыми и пригодными для использования в факторизованной модели динамики, что ускоряет процесс обучения на воображаемых траекториях. Отличие этой потери от потери динамики заключается в использовании оператора остановки градиента $\text{sg}(\cdot)$ и различиях в масштабировании этих двух компонент. Стратегия обучается в парадигме актор-критик.

DreamerV3 обладает высокой универсальностью и может быть применён к различным задачам без необходимости значительной настройки гиперпараметров. Алгоритм эффективно использует данные, обучаясь на "воображаемых"траекториях, что минимизирует количество взаимодействий с реальной средой. Он также демонстрирует хорошую масштабируемость, работая как с простыми, так и с более сложными задачами, включая задачи с непрерывными и дискретными действиями. Вдобавок, DreamerV3 способен обучаться в сложных и динамичных средах, таких как Minecraft, без использования предварительных данных. Основным недостатком является зависимость от точности модели мира, поскольку ошибки в этой модели могут привести

к ухудшению качества стратегии. Алгоритм также требует значительных вычислительных ресурсов, особенно при решении более сложных задач.

4.2 EfficientZeroV2

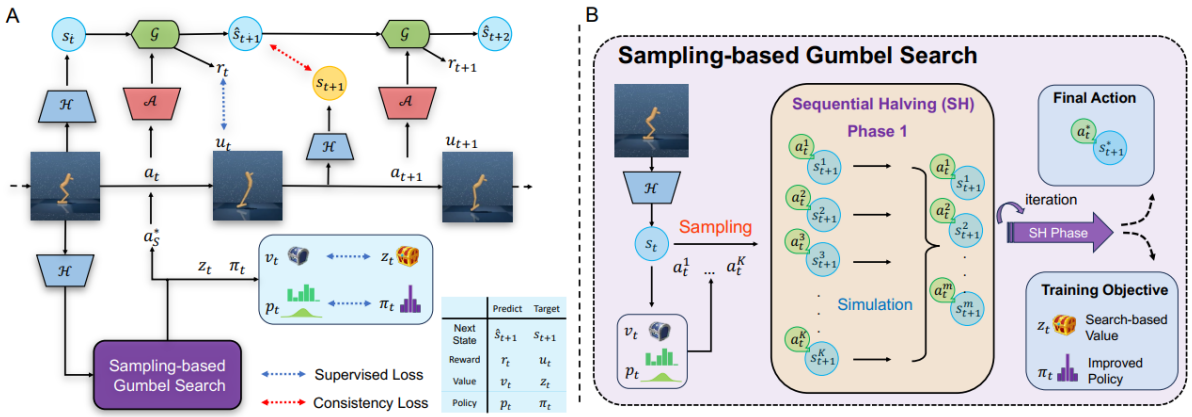

Рис. 5: Процесс обучения EZ-V2

EfficientZero V2 (EZ-V2) — это усовершенствованный алгоритм обучения с подкреплением на основе моделли, созданный для эффективного решения задач управления с ограниченным количеством данных. Основной акцент сделан на повышении эффективности обучения с точки зрения выборки, что особенно важно для реальных задач, где сбор большого количества данных может быть затруднен. В алгоритме сделаны улучшения, позволяющие работать как с дискретными, так и с непрерывными действиями, а также с визуальными и низкоразмерными входными данными.

EZ-V2 является идейным продолжением DreamerV3, MuZero и EfficientZero [13], сочетая ключевые особенности данных алгоритмов с некоторыми улучшениями и адапитационными модификациями. Модель мира в EZ-V2 имеет стандратные компоненты, которые присутствуют и в других алгоритмах:

$$\begin{cases} \text{Энкодер:} & s_t = \mathcal{H}(o_t) \\ \text{Модель динамики:} & \hat{s}_{t+1}, r_t = \mathcal{G}(s_t, a_t) \\ \text{Стратегия:} & p_t = \mathcal{P}(s_t) \\ \text{Функция ценности состояния:} & v_t = \mathcal{V}(s_t) \end{cases} \tag{4.3}$$

где s_t — скрытое состояние в момент времени t, o_t — наблюдение, $\hat{s}_{t+1}$) — предсказанное следующее скрытое состояние $\hat{s}_{t+1}$, (r_t) - вознаграждение, p_t — распределение

вероятностей действий p_t на основе текущего состояния s_t, v_t - ожидаемая суммарная награда для состояния. Также стоит отметить, что модель динамики принимает на вход векторного представление действия, которое получается после блока A на рисунке 5. к

EZ-V2 обучается, используя общую функцию потерь:

$$L_t = \lambda_1 L_R(u_t, r_t) + \lambda_2 L_P(\pi_t, p_t) + \lambda_3 L_V(z_t, v_t) + \lambda_4 L_G(s_{t+1}, \hat{s}_{t+1}) \quad (3) \tag{4.4}$$

где u_t обозначает награду, полученную из среды, π_t — целевую стратегию, полученную с помощью модернизированной версии MCTS., а z_t представляет фактическое значение функции цеенности состояния. L_R, L_P и L_V представляют собой функции потерь обучения с учителем. L_G — это функция потерь временной согласованности, которая повышает сходство между предсказанным следующим состоянием $\hat{s}_{t+1}$ и следующим состоянием s_{t+1}, что стабилизирует процесс обучения. Схематично процесс обучения представлен слева на рисунке 5.

Одним из главных нововведений EfficientZero V2 является использование Поиск Гумбеля на основе выборки вместо традиционного метода MCTS. Хотя MCTS хорошо справляется с планированием, он сталкивается с проблемами в средах с высокоразмерными пространствами действий. Поиск Гумбеля в EfficientZero V2 использует трюк Gumbel-Top-k (технически по смыслу заменяет Upper Confidence Bound) для выборки действий и метод Sequential Halving (при rollout отсекаем по половине траекторий с худшей совокупной наградой) для их оценки. Это позволяет эффективнее исследовать среду и обрабатывать задачи с непрерывными и большими пространствами действий, обеспечивая баланс между исследованием и эксплуатацией решений. Принципы работы поиска Гумбеля представлен справа на рисунке 5.

В совокупности данные улучшения делают EfficientZero V2 более эффективным для решения задач как с дискретными, так и с непрерывными действиями, значительно повышая его производительность по сравнению с предыдущими алгоритмами.

Метод Gumbel-Top-k с Sequential Halving, используемый в EfficientZero V2, обладает рядом важных преимуществ. Он эффективно работает с большими и сложными пространствами действий, что делает его более подходящим для задач с высокой размерностью, где традиционные методы, такие как UCB-MCTS, не всегда показывают хорошую производительность. Кроме того, добавление шума Гумбеля к логитам действий помогает сбалансировать исследование и эксплуатацию, что способствует улучшению общей производительности алгоритма за счет оптимизации процесса выбора

действий. Метод Sequential Halving также значительно сокращает вычислительные затраты, так как на каждом шаге отсеиваются менее перспективные действия, что позволяет быстрее находить оптимальные решения.

Однако у метода есть и свои недостатки. Одним из них является нестабильность обучения. В некоторых случаях добавление шума Гумбеля и использование агрессивного отбора траекторий может привести к колебаниям в процессе обучения, особенно в условиях высокоразмерных или нестабильных сред. Это может затруднить стабильность и скорость сходимости алгоритма, требуя дополнительных корректировок для достижения оптимальных результатов. В некоторых ситуациях также может наблюдаться недоисследование, если отбор траекторий оказывается слишком агрессивным, что снижает шансы на выявление более эффективных, но менее очевидных действий.

5 Объектно-центричное латентное пространство

Объектно-центрированные представления можно рассматривать как дискретные структуры, где состояние среды кодируется через отдельные объекты и их взаимодействия. Такой подход упрощает обобщение между различными задачами, где ключевую роль играют одни и те же объекты, и повышает эффективность планирования.

5.1 Focus

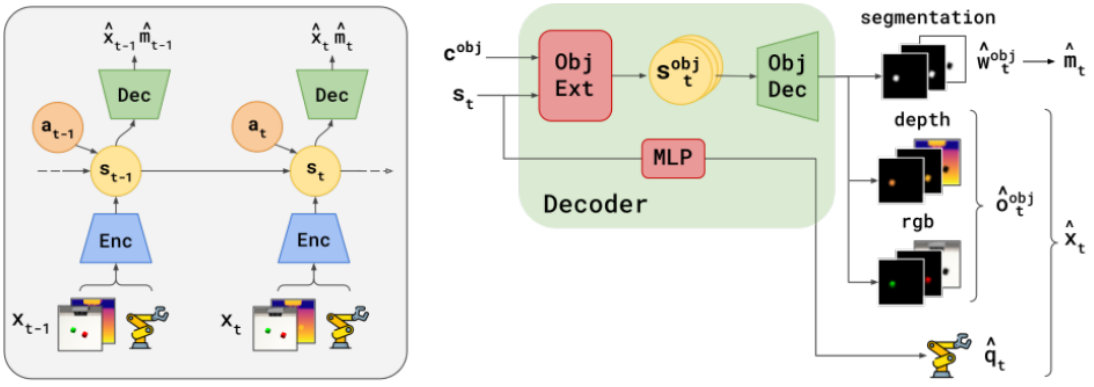

Рис. 6: Архитектура FOCUS

FOCUS — это модель, ориентированная на объектное-центричное представление мира, предназначенная для решения задач манипуляции с объектами в робототехни-

ке. Основная идея заключается в обучении робота взаимодействовать с окружающими его объектами. В отличие от традиционных подходов, где агент оперирует только с пикселями, FOCUS использует объектно-центричное представление, что способствует лучшему пониманию структуры среды и улучшает взаимодействие с объектами, такими как их перемещение, вращение и изменение состояния. На рисунке 6 слева представлена общая схема работы алгоритма, справа схема работы энкодера.

Агент наблюдает за окружающей средой через входные данные $x_t = \{o_t, q_t\}$, которые он получает при каждом взаимодействии, где мы можем различать (визуальные) наблюдения o_t, например, RGB-изображение с камеры и глубину, от проприоцептивной информации q_t, то есть состояния и скорости суставов робота. Эта информация обрабатывается агентом через модель энкодера $e_t = f_\varphi(x_t)$, которая может быть реализована как конкатенация выходов CNN для высокоразмерных наблюдений и MLP для низкоразмерных проприоцептивных входных данных.

Модель мира стремится захватить временную динамику входных данных в скрытом состоянии s_t. Для разделения информации, специфичной для объекта, на отдельные скрытые представления s_t^{obj} создаем два компонента: экстрактор скрытых представлений объекта и декодер объекта. Модель мира состоит из компонент:

$$\begin{aligned}
\text{Энкодер:} \quad & e_t = f_\varphi(x_t), \\
\text{Апостериорное распределение:} \quad & p_\varphi(s_{t+1}|s_t, a_t, e_{t+1}), \\
\text{Априорное распределение:} \quad & p_\varphi(s_{t+1}|s_t, a_t), \\
\text{Проприоцептивный декодер:} \quad & p_\theta(\hat{q}_t|s_t), \\
\text{Экстрактор скрытых представленийг:} \quad & p_\theta(s_t^{obj}|s_t, c^{obj}), \\
\text{Декодер объекта:} \quad & p_\theta(\hat{o}_t^{obj}, w_t^{obj}|s_t^{obj}),
\end{aligned} \tag{5.1}$$

где φ и θ обозначают набор обучаемых параметров. Используется RSSM для компонентов динамики, то есть априорного и апостериорного распределений, которая извлекает скрытое состояние s_t, состоящее из детерминированной и стохастической составляющих. Проприоцептивная информация $\hat{q}_t$ декодируется из скрытого состояния s_t с помощью многослойного перцептрона. Для каждого объекта на сцене экстрактор скрытых представлений объекта получает скрытое состояние модели мира s_t и вектор, идентифицирующий объект c^{obj}, и извлекает скрытое представление объекта, центрированное на объекте s_t^{obj}. При наличии таких скрытых векторов объекта декодер объекта выдает два значения: информацию о наблюдении, связанную с объектом $\hat{o}_t^{obj}$, и веса сегментации w_t^{obj}.

Информация о наблюдении, связанная с объектом $\hat{o}_t^{obj}$, представляет собой маскированную версию визуальных входных данных o_t, где вся нерелевантная для объекта информация маскируется с помощью маски сегментации. Одномерные ненормализованные веса w_t^{obj} представляют собой специфичные для объекта логиты для каждого пикселя для сегментации объектов. Общая сегментация сцены получается путем применения softmax ко всем весам объектов, где каждый пиксель присваивается объекту, который выдает наибольший вес.

Модель обучается end-to-end путем минимизации следующей функции потерь:

$$\mathcal{L}_{\text{wm}} = \mathcal{L}_{\text{dyn}} + \mathcal{L}_{\text{proprio}} + \mathcal{L}_{\text{obj}}. \tag{5.2}$$

Динамическая модель минимизирует дивергенцию Кульбака-Лейблера между апостериорным и априорным распределениями:

$$\mathcal{L}_{\text{dyn}} = D_{\text{KL}}[p_\varphi(s_{t+1}|s_t, a_t, e_{t+1})||p_\varphi(s_{t+1}|s_t, a_t)]. \tag{5.3}$$

Декодер проприоцептивных данных обучается восстанавливать проприоцептивные состояния, минимизируя отрицательную логарифмическую вероятность (NLL):

$$\mathcal{L}_{\text{proprio}} = -\log p_\theta(\hat{q}_t|s_t) \tag{5.4}$$

Декодер объектов обучается восстанавливать информацию, центрированную на объектах, выдавая веса объектов для сегментационной маски и восстанавливая наблюдения, связанные с объектами. Наблюдение маскируется с помощью маски, специфичной для объекта m_t^i, чтобы сосредоточиться только на информации об i-м объекте в функции потерь. Потерю декодера объектов можно выразить следующим образом:

$$\mathcal{L}_{\text{obj}} = -\underbrace{\log p(\hat{m}_t)}_{\text{mask}} - \log \sum_{i=0}^{N} \underbrace{m_t^i p_\theta(\hat{x}_t^i|s_t^i)}_{\text{masked reconstruction}} \tag{5.5}$$

где общая сегментационная маска получается как:

$$\hat{m}_t = \text{softmax}(w_t^1, ..., w_t^N) \tag{5.6}$$

где N - количество всех экземпляров объектов, присутствующих на сцене. Минимизируя маскированную NLL реконструкции, декодер объектов гарантирует, что скрытое

представление каждого объекта s_t^i фокусируется на захвате только релевантной информации, поскольку реконструкции, полученные из скрытого представления, маскируются для каждого объекта. Более того, объекты конкурируют за занятие своего правильного места на сцене (в пиксельном пространстве) посредством потери маски. Таким образом, скрытый вектор каждого объекта "знает"о положении объекта на сцене.

Преимущества алгоритма: Объектно-центричный подход FOCUS позволяет агенту эффективно "рассуждать"об отдельных объектах и их взаимодействиях, что способствует лучшему обучению и обобщению в задачах манипуляции. Встроенный механизм неявной сегментации упрощает архитектуру и обучение, а также способствует более целенаправленному исследованию среды.

Недостатки алгоритма: Несмотря на преимущества, FOCUS имеет ряд ограничений. Эффективность алгоритма напрямую зависит от качества сегментации, которая, будучи неявной, может быть неточной, особенно в сложных сценах с перекрывающимися объектами или сложным фоном. Кроме того, FOCUS предполагает фиксированное число объектов, что ограничивает его применимость в реальных сценариях.

6 Заключение

В данном обзоре были рассмотрены современные методы обучения с подкреплением на основе модели, с акцентом на различные подходы к представлению латентного пространства. Были проанализированы алгоритмы, использующие непрерывные латентные пространства, такие как MuZero и TD-MPC2, а также алгоритмы, основанные на дискретных представлениях, включая DreamerV3 и EfficientZeroV2. Кроме того, был рассмотрен объектно-центрированный подход, представленный алгоритмом FOCUS.

Каждый из рассмотренных методов имеет свои преимущества и недостатки. Непрерывные представления хорошо подходят для задач с плавной динамикой, но могут быть менее эффективны в задачах с дискретными состояниями. Дискретные представления, в свою очередь, обеспечивают лучшую интерпретируемость и более эффективную обработку данных, но могут столкнуться с проблемами в задачах с непрерывными действиями. Объектно-центрированные подходы позволяют агенту эффективно "рассуждать"об отдельных объектах и их взаимодействиях, что способ-

ствует лучшему обобщению, но их эффективность напрямую зависит от качества сегментации.

Выбор оптимального подхода к представлению латентного пространства зависит от конкретной задачи и ее особенностей. Дальнейшие исследования в этой области направлены на разработку более устойчивых и универсальных методов создания векторных представлений, которые смогут эффективно работать в различных условиях и обеспечивать высокую обобщающую способность алгоритмов обучения с подкреплением. Важным направлением является также разработка методов, объединяющих преимущества различных подходов, например, гибридных архитектур, сочетающих непрерывные и дискретные представления.

Список литературы

1. *Ha D.*, *Schmidhuber J.* World models // arXiv preprint arXiv:1803.10122. — 2018.
2. *Singh G.*, *Deng F.*, *Ahn S.* ILLITERATE DALL-E LEARNS TO COMPOSE // 10th International Conference on Learning Representations, ICLR 2022. — 2022.
3. Mastering chess and shogi by self-play with a general reinforcement learning algorithm / D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel [и др.] // arXiv preprint arXiv:1712.01815. — 2017.
4. Mastering atari, go, chess and shogi by planning with a learned model / J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D. Hassabis, T. Graepel [и др.] // Nature. — 2020. — Т. 588, № 7839. — С. 604—609.
5. *Hansen N.*, *Su H.*, *Wang X.* TD-MPC2: Scalable, Robust World Models for Continuous Control // The Twelfth International Conference on Learning Representations. — 2024.
6. Mastering diverse domains through world models / D. Hafner, J. Pasukonis, J. Ba, T. Lillicrap // arXiv preprint arXiv:2301.04104. — 2023.
7. EfficientZero V2: Mastering Discrete and Continuous Control with Limited Data / S. Wang, S. Liu, W. Ye, J. You, Y. Gao // Forty-first International Conference on Machine Learning. — 2024.
8. FOCUS: Object-Centric World Models for Robotic Manipulation / S. Ferraro, P. Mazzaglia, T. Verbelen, B. Dhoedt // Intrinsically-Motivated and Open-Ended Learning Workshop @NeurIPS2023. — 2023.
9. *Hansen N. A.*, *Su H.*, *Wang X.* Temporal Difference Learning for Model Predictive Control // International Conference on Machine Learning. — PMLR. 2022. — С. 8387—8406.
10. Mastering Atari with Discrete World Models / D. Hafner, T. P. Lillicrap, M. Norouzi, J. Ba // International Conference on Learning Representations. — 2021.
11. Dream to Control: Learning Behaviors by Latent Imagination / D. Hafner, T. Lillicrap, J. Ba, M. Norouzi // International Conference on Learning Representations. — 2019.

12. Learning latent dynamics for planning from pixels / D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, J. Davidson // International conference on machine learning. — PMLR. 2019. — С. 2555—2565.

13. Mastering Atari Games with Limited Data / W. Ye, S. Liu, T. Kurutach, P. Abbeel, Y. Gao // Advances in Neural Information Processing Systems / под ред. A. Beygelzimer, Y. Dauphin, P. Liang, J. W. Vaughan. — 2021.