ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ «ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

**Факультет информатики, математики и компьютерных наук**

Программа подготовки магистров по направлению

01.04.02 Прикладная математика и информатика

Богатова Маргарита Дмитриевна

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА МАГИСТРА**

на тему:

**«Исследование построения агентов на основе метода Active Inference»**

**Научный руководитель:**

профессор кафедры прикладной математики и информатики

Крылов Владимир Владимирович

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись

**Рецензент:**

д.ф.м.н, профессор кафедры прикладной математики и информатики

Калягин Валерий Александрович

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись

Нижний Новгород  
2022

АННОТАЦИЯ

Активный вывод (AIF) — это математическое понятие, возникшее в вычислительной нейронауке как теория того, как мозг реализует действие, восприятие и обучение. В основе активного вывода лежит принцип минимизации свободной энергии. На данный момент, в подавляющем большинстве работ по активному выводу рассматриваются игрушечные дискретные задачи. Поэтому вектор исследования в области активного вывода сейчас направлен в сторону расширения области применения на задачи с непрерывным пространством действий и состояний.

На данный момент есть ряд исследований, в которых активный вывод применятся для задач с пространством состояний большой размерности, однако исследований для многомерного пространства и состояний, и действий еще очень мало и многие из них не используют все возможности глубокого обучения. Таким образом **задача** адаптации активного вывода для непрерывного пространства действий, решающая **проблему** масштабирования этого метода для комплексных задач таких как управление роботом является **актуальной**.

**Объектом** исследования являются алгоритмы активного вывода, а также методы глубокого обучения для управления роботом. **Предметом** работы является исследование применения теории активного вывода для реальных задач большой размерности. **Цель** исследования состоит в изучении существующих подходов к построению AIF агентов и разработка на их основе алгоритма для решения задачи управления роботом.

Для достижения поставленной цели требуется решить следующие **задачи**:

1. Изучение теории активного вывода.
2. Исследование научных источников, посвященных проблеме масштабирования активного вывода.
3. Разработка алгоритма на основе активного вывода для решения задачи управления роботом.
4. Реализация этого алгоритма и проведение экспериментов для оценки качества работы агента, а также определения лучшей его конфигурации.

В процессе работы использовались следующие **методы**: анализ литературных источников, описывающих передовые достижения в области глубокого активного вывода, и обобщение полученной информации; проведение экспериментов по решению различных задач управления роботом deep active inference (dAIF) агентом и сравнение результатов, визуализация и анализ данных результатов проведенных экспериментов.

**Новизна** работы заключается в том, что в ней:

* Предложен алгоритм для обучения dAIF агента для окружений с многомерными разнотипными данными и непрерывными действиями.
* Обученный dAIF агент успешно применен для решения задачи управления роботом.

**Практическая значимость**: построенный алгоритм может быть использован в других прикладных задачах, а также может помочь исследователям в будущем разрабатывать новые подходы.

Active inference (AIF) is a mathematical concept that originated in computational neuroscience as a theory of how the brain implements action, perception, and learning. Active inference is based on the principle of minimizing free energy. Now, in the vast majority of works on active inference, toy discrete problems are considered. Therefore, the vector of research in the field of active inference is directed towards expanding the scope to problems with a continuous space of actions and states.

Now, there are a number of studies in which active inference is applied to problems with high-dimensional state space, however, there are still very few studies for high-dimensional space actions, and many of them do not use the full capabilities of deep learning. Thus, the task of adapting the active output for a continuous action space, solving the problem of scaling this method for complex tasks such as robot control, is relevant.

The object of research is active inference algorithms, as well as deep learning methods for robot control. The subject of the work is the study of the application of the theory of active inference for real complex problems. The purpose of this work is to study the existing approaches to the construction of AIF agents and develop an algorithm based on them for solving the robot control problem.

To achieve this goal, it is necessary to solve the following tasks:

1. Studying the theory of active inference.

2. Research of scientific sources devoted to the problem of active output scaling.

3. Propose algorithm based on active inference for robot control.

4. Implementation of this algorithm and conducting experiments to assess the quality of the agent's work, as well as to determine its best configuration.

The novelty of the work lies in the fact that it:

• Proposed an algorithm for training a dAIF agent for environments with multidimensional heterogeneous data and continuous actions.

• A trained dAIF agent has been successfully applied to solve the problem of robot control.

Practical significance: the constructed algorithm can be used in other applied problems, and can help researchers develop new approaches in the future.

Определения, обозначения, сокращения

AIF – active inference – активный вывод.

dAIF – deep active inference – глубокий активный вывод.

EFE – expected free energy, ожидаемая свободная энергия.

VFE – vibrational free energy, вариационная свободная энергия.

MDP - Markov decision process, Марковский процесс принятия решений

POMDP - Partially observable Markov decision process, частично наблюдаемый Марковский процесс принятия решений

MCTS - Monte Carlo tree search

RL – Reinforcement Learning

CEM - Cross-entropy method

SAC – Soft Actor Critic

HER - Hindsight Experience Replay

PER - Prioritized Experience Replay

MSE – Mean-Squared-Error

VAE - variational autoencoder, вариационный автоэнкодер

MLP - Multilayer perceptron

LSTM – Long-Short-Term-Memory

GRU - Gated Recurrent Unit

ОГЛАВЛЕНИЕ

[АННОТАЦИЯ 2](#_Toc104519956)

[Определения, обозначения, сокращения 4](#_Toc104519957)

[ОГЛАВЛЕНИЕ 5](#_Toc104519958)

[Введение 6](#_Toc104519959)

[Глава 1. Обзор литературы 9](#_Toc104519960)

[1.1. Вариационная свободная энергия 10](#_Toc104519961)

[1.1. Ожидаемая свободная энергия 11](#_Toc104519962)

[1.2. Связь с обучением c подкреплением 13](#_Toc104519963)

[1.3. Активный вывод в задачах непрерывного управления 13](#_Toc104519964)

[1.4. Глубокий активный вывод (dAIF) 14](#_Toc104519965)

[1.5. Анализ существующих работ 16](#_Toc104519966)

[Глава 2. Предлагаемое решение задачи 19](#_Toc104519967)

[2.1. Существующие решения этой задачи в обучении с подкреплением 19](#_Toc104519968)

[2.2. Модель и алгоритмы 20](#_Toc104519969)

[2.2.1. Определение актера 21](#_Toc104519970)

[2.2.2. Вывод формулы для подсчета ожидаемой свободной энергии 22](#_Toc104519971)

[2.2.3. Вывод формулы для подсчета вариационной свободной энергии 24](#_Toc104519972)

[2.2.4. POMDP и MPD версии 27](#_Toc104519973)

[Глава 3. Реализация и эксперименты 28](#_Toc104519974)

[3.1. Выбор симулятора 28](#_Toc104519975)

[3.2. Описание задач 29](#_Toc104519976)

[3.3. Описание и цель экспериментов 30](#_Toc104519977)

[3.4. Детали разработки 31](#_Toc104519978)

[3.5. Результаты вычислений 31](#_Toc104519979)

[3.5.1. FetchReach 31](#_Toc104519980)

[3.5.2. NeedleReach 32](#_Toc104519981)

[3.5.3. NeedleGrasp 33](#_Toc104519982)

[3.5.4. FetchPickAndPlace 34](#_Toc104519983)

[3.6. Анализ результатов 34](#_Toc104519984)

[Заключение 37](#_Toc104519985)

[Список литературы 38](#_Toc104519986)

Введение

Активный вывод (AIF) — это математическое понятие, возникшее в вычислительной нейронауке как теория того, как мозг реализует действие, восприятие и обучение [19]. В основе активного вывода лежит принцип минимизации свободной энергии, предполагающий, что агенты действуют, противодействуя силам окружающей среды, которые мешают им оставаться в ограниченном наборе желаемых состояний мира, и минимизируют неожиданность сенсорных наблюдений за счет минимизации аппроксимации этой неожиданности, которая называется свободной энергией. Согласно этому предположению, биологические агенты развивают различные навыки, такие как восприятие, действие, планирование и обучение, которые агенты постоянно адаптируют на протяжении всей своей жизни.

Активный вывод и принцип свободной энергии использовались для объяснения и моделирования нескольких сложных процессов в различных дисциплинах. Например, в психологии они использовались для распознавания эмоций [1]. В экономике принцип свободной энергии использовался для формулирования процесса оптимизации агентов с точки зрения их когнитивных искажений [1].

Недавние разработки в области глубокого обучения открыли новые горизонты для изучения и экспериментирования с различными теориями восприятия и поведения. Одним из популярных примеров в этом отношении является обучение с подкреплением (RL) [20], теория, которая связывает сигналы дофамина в мозге с сигналами вознаграждения, которые можно использовать для подкрепления правильного поведения [21], и описывает, как поведение агента может быть оптимизировано через максимизацию вознаграждения. Сочетание RL с моделями глубокого обучения для оценки функций позволило достичь успехов в построении искусственных агентов. Такие агенты нашли широкое применение в видеоиграх [22], настольных игры [23, 24] или в управлении роботами [25, 26]. Методы глубокого обучения также начинают возникать в контексте активного вывода [5,6,7]. Интегрированный с машинным обучением активный вывод называют deep active inference (dAIF).

Обучение с подкреплением и активный вывод призваны решать одни и те же задачи, которые обычно формулируются как марковский процесс принятия решений (MDP) или частично наблюдаемый марковский процесс принятия решений POMDP. Ввиду этой образующейся связи, активный вывод сейчас часто применятся для решения задач именно в такой формулировке, что позволяет внедрять передовые разработки в области RL в AIF. Как следствие и использовании глубокого обучения, dAIF достигают сравнимых с RL результатов.

Современные задачи управления роботами поставленные как POMDP\MDP успешно решаются с применением методов обучения с подкреплением. В свою очередь, в подавляющем большинстве работ по активному выводу рассматриваются игрушечные дискретные задачи. Вектор исследования в области активного вывода сейчас направлен в сторону расширения области применения на задачи с непрерывным пространством действий и состояний [8, 10, 12].

На данный момент есть ряд исследований, в которых активный вывод применятся для задач с пространством состояний большой размерности [5,6,7], однако исследований для многомерного пространства и состояний, и действий еще очень мало и многие из них не используют все возможности глубокого обучения. Таким образом **задача** адаптации активного вывода для непрерывного пространства действий, решающая **проблему** масштабирования этого метода для комплексных задач таких как управление роботом является **актуальной**.

**Объектом** исследования являются алгоритмы активного вывода, а также методы глубокого обучения для управления роботом. **Предметом** работы является исследование применения теории активного вывода для реальных задач большой размерности. **Цель** исследования состоит в изучении существующих подходов к построению AIF агентов и разработка на их основе алгоритма для решения задачи управления роботом.

Для достижения поставленной цели требуется решить следующие **задачи**:

1. Изучение теории активного вывода.
2. Исследование научных источников, посвященных проблеме масштабирования активного вывода.
3. Разработка алгоритма на основе активного вывода для решения задачи управления роботом.
4. Реализация этого алгоритма и проведение экспериментов для оценки качества работы агента, а также определения лучшей его конфигурации.

В процессе работы использовались следующие **методы**: анализ литературных источников, описывающих передовые достижения в области глубокого активного вывода, и обобщение полученной информации; проведение экспериментов по решению различных задач управления роботом dAIF агентом и сравнение результатов, визуализация и анализ данных результатов проведенных экспериментов.

**Новизна** работы заключается в том, что в ней:

* Предложен алгоритм для обучения dAIF агента для окружений с многомерными разнотипными данными и непрерывными действиями.
* Обученный dAIF агент успешно применен для решения задачи управления роботом.

**Практическая значимость**: построенный алгоритм может быть использован в других прикладных задачах, а также может помочь исследователям в будущем разрабатывать новые подходы.

Работа имеет следующую **структуру**: в первой главе содержится описание основных принципов активного вывода, а также обзор и анализ существующих AIF работ для его масштабирования. Во второй главе приводится ход разработки алгоритма для решения задачи. Третья глава содержит описание окружений, конкретных задач, а также проведенных экспериментов. В заключении содержаться выводы о проделанной работе.

Глава 1. Обзор литературы

Теория активного обучения была предложена К.Дж. Фристоном как единая теория мозга Главная концепция активного обучения состоит в минимизации функционала свободной энергии.

В активном выводе предполагается, что окружение, в котором действует агент описывается как частично наблюдаемый марковский процесс принятия решений (POMDP), который обозначается в виде набора M = {S, A, T, Ω, O, γ}, где S — множество ненаблюдаемых состояний, A — множество действий, T — функция перехода между состояниями, также называемая динамики окружающей среды, Ω — множество наблюдений, O — множество условных вероятностей наблюдения, γ — коэффициент температуры. Такая формальность обусловлена тем, что агент не имеет «доступа» к истинному состоянию мира. Используя прошлые наблюдения, он строит **убеждение** об истинном состоянии.

Агент получает наблюдения из среды, имеющей скрытые состояния, которые агент пытается вывести. Он также может предпринимать действия, которые изменяют состояние среды и, следовательно, будущие наблюдения.

Поведение агента в среде определятся его генеративной моделью своих наблюдений, состояний и действий, которые можно факторизовать следующим образом:

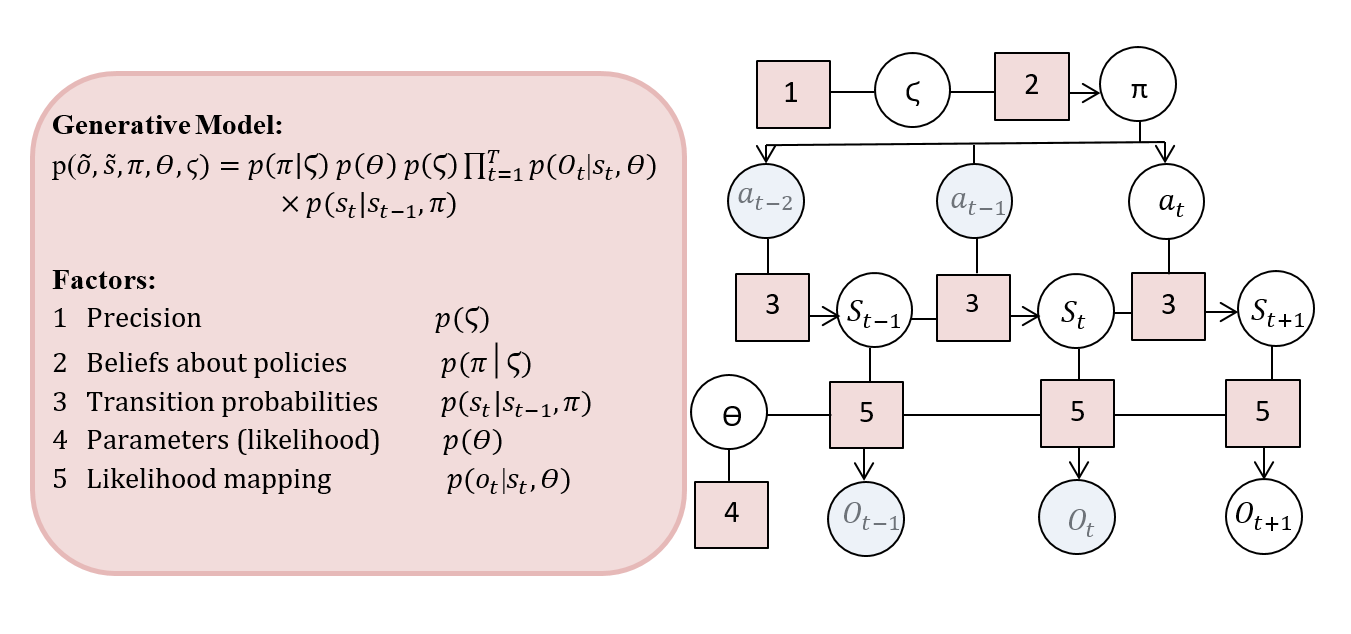


Рисунок 1. Генеративная модель в виде фактор-графа.

Путем оптимизации генеративной модели агент улучшает свою способность выяснить, какое действие привело к наблюдению. В каждый момент времени агент вычисляет ожидаемую свободную энергию (EFE) каждой политики (траектории, последовательности действий), которая количественно определяет вариационную свободную энергию (VFE) и изменяет среду, чтобы свести к минимуму VFE.

* 1. Вариационная свободная энергия

Ключевой смысл принципа свободной энергии состоит в том, что агент стремится найти равновесие между своей внутренней моделью и внешним миром. В терминах Баесовского вывода, агент максимизирует обоснованность (evidence) модели, приближая внутреннее состояние, описываемое плотностью q(s), к апостериорному миру p(s|o). Помимо интуиции, введение вариационного распределения необходимо по причине того, что p(s|o) в большинстве реальных случаев нельзя посчитать точно.

Для того, чтобы подобрать лучшую аппроксимацию q(s) минимизируют расстояние Кульбака-Лейблера (KL) между распределениями q(s) и p(s|o). Для простоты записи, рассмотрим упрощенную генеративную модель .

Из этого определения вытекает формула вариационной свободной энергии (VFE):

Эта формула представляет собой функцию внутренних убеждений агента об окружающей среде, и означает верхнюю границу неожиданности наблюдений. VFE измеряет меру расхождения между вариационной плотностью q(s) и генеративной моделью .

Далее, формула свободной энергии может быть переписана следующим образом:

С одной стороны, минимизация свободной энергии подразумевает максимизацию точности (второе уравнение, второе слагаемое) при одновременном снижении сложности апостериорного распределения (второе уравнение, первое слагаемое). С другой стороны, это подразумевает оптимизацию обоснованности модели (первое уравнение).

Очередной раз преобразуем формулу: используем условие марковского процесса о том, что распределение вероятностей будущих состояний процесса зависит только от нынешнего состояния и заменим на .

Это наиболее часто используемый в литературе вид формулы свободной энергии для урезанной генеративной модели. В статье [5] была выведена формула для генеративной модели, изображенной на схеме 1, которая берет во внимание распределения действий агента. Выглядит она следующим образом:

*Воспользуемся формулой расстояния Кульбака-Лейблера :*

* 1. Ожидаемая свободная энергия

«Активной» составляющей агента активного вывода является взаимодействие с окружением. Теперь необходимо взять во внимание то, что агент совершает действия. Для того, чтобы действовать в среде наиболее оптимально, в подходе предлагается помимо минимизации вариационной свободной энергии, также минимизировать ожидаемую свободную энергию (EFE), которая представляет собой комбинацию VFE всех ожидаемых траекторий в будущем с учетом текущего состояния и действия.

Стоит отметить, что в стандартной литературе по активному выводу рассматривается генеративная модель , где называется политикой, однако далее будет сделано допущение, что .

Общая формула ожидаемой свободной энергии обычно обозначается так:

Где , , а - та самая вариационная плотность. Следующее преобразование формулы представлено в источнике [12], рассмотрим его, так как оно хорошо отражает смысл ожидаемой свободной энергии.

*=*

*Допустим, что*

*Пусть действия распределены равномерно:*

*Перегруппируем части уравнения:*

Интуитивно, выведенная формула означает, что агент будет выбирать такие действия, для которых наблюдения содержат больше информации о состояниях, которые будут вести к получению предпочтительных результатов и в то же время сохраняя действия настолько вариативными, насколько это возможно.

Таким образом, мы определили приближенные формулы, которые обычно являются основой для описания различных подходов. Теперь определим основные для активного вывода понятия:

* – модель переходов (*transition model*).
* – предпочтения агента *(agent preferences)*.
* - условное распределение действий, обычно в литературе называет моделью действий.
* – модель наблюдений *(observation model)*. Также в литературе называется репрезентацией. Предполагается, что генеративная модель агента способна генерировать воображаемые наблюдения.
  1. **Связь с обучением c подкреплением**

Активный вывод был предложен как альтернатива обучению с подкреплением. Эти подходы призваны решать одни и те же задачи, и за иногда даже оперируют одними и теми же формулами, однако идеи, лежащие в основе, у них разные. Например, концепция «вознаграждения» в активном выводе сильно отличается от вознаграждения в RL, поскольку вознаграждение — это не сигналы, используемые для привлечения агента, а скорее сенсорные состояния, которые агенты стремятся часто посещать, чтобы минимизировать свою свободную энергию. Авторы активного обучения утверждают, что здесь функция награждения совсем не обязательна и агенты смогут изучать свою собственную.

* 1. Активный вывод в задачах непрерывного управления

Фокус этой работы направлен на применение активного вывода для решения задачи непрерывного управления. Особенности этой задачи состоит в том, что при ее рассмотрении как POMDP, внутреннее состояние системы и действия задаются следующим образом: и . Например, **задача управления роботизированной** рукой – наблюдения и действия в такой модели — это вектора, значения которого варьируются от -1 до 1, которые внутри симуляции трансформируются в углы между подвижными частями робота.

Реализация агента активного вывода, который бы действовал в окружении с непрерывным пространством состояний и действий на текущий момент открытая проблема. Потому что стандартные [13] формулы для расчета и обучения AIF агента применимы только для дискретных случаев. Так, в [13, 14] плотности в уравнении свободной энергии задаются матрицами. Например, кодируется набором матриц для каждого возможного действия, строки и столбцы которых задают плотность для каждой возможной пары состояний.

Еще более сложная ситуация с непрерывным пространством действий. Чтобы сделать вывод о своей политике, агент должен вычислить ожидаемую свободную энергию (EFE) для каждой траектории. На практике это означает, что для *каждой* *возможной* *политики* агенту необходимо на определенный горизонт времени вперед сгенерировать состояния и наблюдения, в которых он будет, если будет следовать этой политике. И далее вычисляется свободная энергия этих состояний и оценивается ожидаемая свободная энергия для каждой политики. Когда доступных политик и действий неопределенно много, такой способ подсчета ожидаемой свободной энергии также становится экспоненциально сложным.

Такой подход называют **табличным активным выводом**, и он был довольно хорошо изучен [15,16,17]. Однако эти методы, хотя и дают представление об идее активного вывода и важности различных параметров, не масштабируются из-за их экспоненциальной сложности, и они не применялись ни к чему, кроме простых игрушечных задач.

* 1. Глубокий активный вывод (dAIF)

В некоторых недавних научных работах были предприняты попытки масштабирования активного вывода для более комплексных и реальных задач. В последние годы для построения MDP/POMDP агентов большую популярность имеют методы глубокого обучения с подкреплением. Именно применение нейросетей позволило достигнуть значительных успехов в этой области. Эта тенденция не обошла стороной и теорию активного вывода - использования нейронных сетей для аппроксимации плотностей положило начало развитию активного обучения в этом направлении.

Насколько это известно, [4] является первой работой, в которой предлагается аппроксимировать модели наблюдений и переходов с помощью глубоких нейронных сетей. Авторы представляют все плотности в виде диагонального гаусовского распределения, а для аппроксимации параметров этих распределений используют однослойные сети с шестнадцатью нейронами. Также именно они ввели понятие *deep active inference* или *dAIF.*

Большое влияние на область dAIF также оказала работа [5]. Авторы использовали последние на то время разработки в области обучения с подкреплением и создали смешанный алгоритм. Основываясь на этом подходе были разработаны такие алгоритмы как [6] и [7]. Вкратце, авторы аппроксимировали с помощью нейронных сетей модели наблюдений и переходов, но что самое интересное, позволили агенту самому предсказывать значение ожидаемой свободной энергии для сразу всех доступных действий, и сделано это тем же способом, как это сделано в обучении с подкреплением при аппроксимации Q-функции.

Возвращаясь к проблеме масштабирования активного вывода, далее статьи будут рассматриваться с точки зрения того, какие именно методы предлагаются и предлагаются ли вообще для работы с непрерывными пространствами состояний и действий. Так, в упомянутых ранее [5,6,7] реализуют масштабирование модели наблюдений с помощью вариационного автоэнкодера, однако VFE считают для всех возможных действий.

Наиболее интересной в этом плане является работа [8], в которой авторы преследуют такие же цели что и в текущем исследовании. Все плотности они рассматривают как нормальные распределения, а их параметры аппроксимируют с помощью нейронных сетей. Модель наблюдений строится на основе вариационного автоэнкодера, что позволяет применять метод в случае многомерных наблюдений (например, изображений). Ключевой особенностью здесь является то, как реализован выбор политики и планирование. В каждый момент времени они выбирают такое действие, которое будет математическим ожиданием модели действий, которая также представлена как нормальное распределение. Для это из нее сэмплируется конечная последовательность действий, для каждой такой последовательности производится оценка свободной энергии и далее модель действий оптимизируется с помощью метода кросс-энтропии (CEM) [9]. Авторы показывают, что такой вид dAIF является конкурентным с передовыми методами. Некоторым продолжением и в своем роде упрощением предыдущего подхода стала работа [10], в которой авторы попробовали усилить алгоритмы обучения с подкреплением с помощью идей активного обучения. На основе принципа ожидаемой свободной энергии была разработана целевая функция, минимизируя которую мы могли бы получить оптимальную модель действий. Благодаря такому упрощению, этот подход можно использовать и для задач с большим входом и непрерывными действиями.

В исследовании [11] попытались оптимизировать подход к выбору оптимального действия (то есть минимизации ожидаемой свободной энергии в пределах некоторого горизонта времени), чтобы избежать полного перебора всех возможных траекторий. Авторы предлагают использовать MCTS для выбора оптимального действия в дереве решений, которое образуется, если рассматривать задачу минимизации ожидаемой свободной энергии для конечного числа возможных действий.

Наконец, из наиболее релевантных последних исследований выделяется [12], авторы которого адаптировали стандартный алгоритм обучения с подкреплением - актер-критик - для предсказания действий, и тем самым создали модель действий, которая применима для непрерывного пространства. Помимо этого, было предложено использовать так называемое контрастное обучение, а в качестве репрезентации использовали обычную нейронную сеть. Как результат, алгоритм, который работает с непрерывным пространством состояний и действий, превосходящий другие dAIF методы на дискретных задачах.

* 1. Анализ существующих работ

Резюмируя обзор статей из прошлого пункта, выделим основные подходы к масштабированию моделей наблюдений и действий.

Так, **для репрезентации** наблюдений большой размерности самым популярным подходом оказался вариационный автоэнкодер, и это оправдано – так как этот метод идеально подходит для аппроксимации с теоретической точки зрения, и при этом является не требовательным к виду входных данных – наблюдения могут быть как изображениями, так и, например, текстом или любым числовым вектором. Однако эта модель довольно тяжела для обучения, и поэтому некоторые исследования [12, 10] стремятся найти другие архитектуры сетей.

С вычислительной точки зрения наиболее сложный аспект минимизации ожидаемой свободной энергии состоит в том, как выбрать действия, которые в будущем приведут к более правильному убеждению агента. На практике оптимизация VFE трансформируется в поиск по дереву среди наиболее вероятных политик. Тем не менее, поиск может быть дорогостоящим, особенно в многомерных пространствах и с непрерывными действиями.

В обзоре [1] выделяют три пути для организации планирования и выбора действия:

1. Наивная реализация активного вывода, где политика представляет собой некоторый план действий . Каждому такому плану ставится в соответствие ожидаемая свободная энергия, и следующее действие выбирается из лучшего плана в соответствии с уравнением общим уравнением, описанным в пункте 2.



Рисунок 2. Различные подходы для выбора оптимального действия. Источник [1].

1. Построение отображения состояние-действие (), которое также называется политикой привычек. Работает по принципу нахождения оптимальной политики, которая бы обеспечивала ожидаемое наилучшее действие для каждого состояния. Эта формулировка была вдохновлена методами в обучении с подкреплением и применялась в работах [5,6,7,12].
2. Объединение первых двух вариантов – сначала для конкретного состояния и действия оценивается ожидаемая свободная энергия, а затем производится поиск по дереву с помощью эвристических алгоритмов.

Оценим три описанных подхода с точки зрения выбора оптимального действия для непрерывного управления. Первый, наивный, вариант уже рассматривался – перебор всех возможных траекторий неэффективен. Третий вариант - это также в своей наивной реализации требует перечисления всех возможных действий, что также невозможно в контексте рассматриваемой задачи. Обычно исследования, которые применяют этот подход, используют MCTS [11]. Однако, недавно были разработаны расширения MCTS для комплексных пространств действий [18]. А это значит, что *третий вариант* также *потенциально может быть применен* для задачи управления роботом. Однако, наиболее подходящим вариантом кажется второй – потому что в текущих передовых RL подходах наиболее успешными являются алгоритмы, где непрерывное вещественное действие так или иначе предсказывается нейронной сетью. То есть перебор действий и как следствие долгосрочное планирование отсутствуют.

Таким образом, за основу решения задачи управления роботом в текущей работе возьмем вариационный автоэнкодер как модель наблюдений и политику привычек как модель действий. Такой выбор основывается на том, ***во-первых***, автоэнкодеры широко используются и позволяют работать с любыми типами наблюдений, и ***во-вторых***, именно такой подход к выбору оптимального действия агентом доказал свою эффективность в области обучения с подкреплением.

Также важно отметить, что среди рассмотренных алгоритмов глубокого активного вывода лишь два [10, 12] были разработаны с перспективой применения их для комплексных задач и по факту применили свою разработку к таким задачам и показали результаты. Однако более сложные задачи, такие как управление роботом, еще не рассматривались в данном контексте. Из этого и вытекает цель этой работы – разработка алгоритма глубокого активного вывода для непрерывных пространств действий и решение с помощью него задачи управления роботом.

Глава 2. Предлагаемое решение задачи

В текущей работе предлагается использовать новейшие разработки из области обучения с подкреплением для масштабирования глубокого активного вывода. За основу берется алгоритм и код, разработанные в [5, 7] и расширяются для решения проблем, где пространство действий непрерывно.

2.1. Существующие решения этой задачи в обучении с подкреплением

Путь улучшения dAIF через уже существующие RL алгоритмы выбран не случайно: последние достигли высоких результатов в решении таких комплексных задач как управление роботами. Разработка текущего решения была вдохновлена таким подходом как как soft actor-critic [21].

Стандартная актер-критик архитектура агентов обучения с подкреплением предполагает, что модуль, называемый актером, некоторым образом генерирует действия, а другой модуль, называемый критиком, обучается предсказывать и стремиться увеличивать «полезность» этих действий. Soft actor-critic (SAC) развивает эту архитектуру. Авторы рассматривают случайного агента и предполагают, что критик будет максимизировать value-функцию, которая зависит не только от награды, но и от энтропии предсказанной актером политики.

Для того, чтобы оптимизировать параметры распределения актера, минимизируют расстояние Кульбака-Лейблера между распределением действий и между аппроксимацией неизвестного оптимального распределения value-функцией.

На этом этапе видны черты сходства между формулой ожидаемой свободной энергии и value-функцией, а также способ аппроксимации политики напоминает из формулы вариационной свободной энергии.

2.2. Модель и алгоритмы

На рисунке 3 изображена общая схема работы dAIF агента. Взаимодействуя с окружением через действия, агент получает наблюдение, награду и флаг шага. На основе полученных данных, модель актер предсказывает следующее действие, и здесь образуется цикл, число итераций которого зависит от решаемой задачи.

Определенное количество последних наблюдений вместе с выведенными из них действиями сохраняется в так называемый replay buffer, который используется для хранения эпизодов. Далее, во время обучения из этого буффера случайно выбираются траектории, для которых будет уменьшаться свободная энергия. По опыту обучения с подкреплением, такой подход позволяет более эффективно использовать предыдущий опыт и при нем наблюдается лучшая сходимость. На схеме показано, что при извлечении данных из буффера, каждый эпизод — это не отдельный шаг, а последовательность шагов. Это необходимо, например, для рекуррентных нейронных сетей.

Также важным элементом рассматриваемой модели является модуль Hindsight Experience Replay (HER)[27], цель которого сделать выборку эпизодов более полезной для обучения. Он применяется для задач, в которых можно определить цель агента на основе наблюдений. Идея этого метода очень проста и гениальна – для заранее заданной части неуспешных эпизодов из обучающей выборки данные наблюдений и наград изменяются так, словно они были удачными. Этот подход очень часто применяется в задачах управления роботами – так как обычно они сформулированы как задачи достижения конкретной цели.

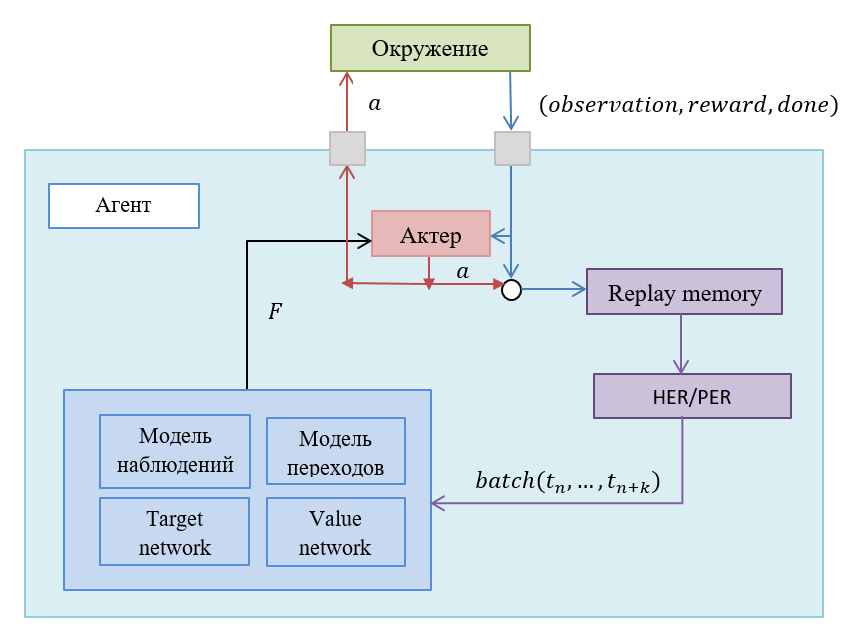


Рисунок 3. Высокоуровневая схема работы алгоритма.

В случае, если достижение той или иной цели определить невозможно – например, наблюдения – это изображения, то тогда тестируется использование другого улучшения воспроизведения опыта - Prioritized Experience Replay (PER) [28], цель которого, задействовать как можно больше различных элементов из буфера. Это реализовано как назначение приоритета попадания в обучающую выборку каждому шагу эпизода, и чем чаще эпизод будет использоваться для обучения, и чем меньше «на нем» будет ошибка, тем меньше будет его приоритет.

2.2.1. Определение актера

Начнем с прямого описания того, как действует агент. Действия генерируются из условного распределения . Это распределение представляется как нормальное распределение, математическое ожидание и логарифм стандартного отклонения которого предсказываются полносвязной нейронной сетью, но с одной особенностью, которая была позаимствована из упомянутого ранее SAC. Авторы предложили способ для ограничения границ действий, которые сэмплируются из такого распределения. Для этого они применяют обратимую функцию сжатия (tanh) к сэмплам из нормального распределения и редактируют формулу для вычисления логарифма вероятностей полученных ограниченных действий:

Для сравнения, в работах, на которых мы основываемся, роль генератора действий выполняет так называемая policy network. В их реализации эта сеть предсказывает значения от нуля до единицы для каждого возможного действия и потом этот массив представляется как аппроксимация искомого распределения. В текущей же работе будем называть эту часть actor-network, и нейронная сеть в ней будет предсказывать два вектора – математическое ожидание и логарифм стандартного отклонения. Действия же сэмплируются с помощью трюка репараметризации.

2.2.2. Вывод формулы для подсчета ожидаемой свободной энергии

Следующим этапом определим способ подсчета ожидаемой свободной энергии. Эта часть является самым нетривиальным моментом в текущей задаче.

Для начала, обратимся к формуле, которая выглядит иначе чем представленная в пункте 1.2. Эта формула была выведена [5] для подсчета свободной энергии лишь на один шаг вперед для конечного количества действий. Наша задача расширить ее для непрерывных действий.

Для конечного горизонта времени формула выглядит так:

Тогда, выделяя нулевое слагаемое из суммы, авторы получают:

В оригинальной статье называют target-network и предлагают аппроксимировать нейронной сетью, как это обычно делают в RL алгоритмах. Если и далее проводить аналогию с методом обучения с подкреплением, – это q-value-network, которая обучается предсказывать EFE на два шага вперед. Функция ошибки для этих двух сетей:

По факту, в реализации это нейронная сеть, которая принимает на вход следующее состояние и предсказывает ожидаемую свободную энергию для каждого возможного действия. То есть, выход сети — это n-мерный вектор, где n – количество действий. И в дальнейшем этот выход взвешивается вектором плотностей каждого действия, полученного из policy-network. Здесь опять возникает ограничение для непрерывных действий – мы не можем предсказать свободную энергию для всех действий сразу.

Таким образом, получается, что единственным возможным способом является предсказание ожидаемой свободной энергии для конкретного состояния и действия. Для внедрения этого в формулу EFE было рассмотрено три варианта:

1. Аппроксимация математического ожидания выборкой и взвешивание с помощью вероятности.

Проблемный элемент заменяется на , а само математическое ожидание аппроксимируется выборкой.

Однако, на практике, такой вариант будет обучаться очень нестабильно – это происходит потому что элемент обычно рассчитывается как возведение в экспоненту предсказанного логарифма вероятности, и в итоге, как будет показано далее в практической части, генерирует либо слишком большие, либо слишком маленькие значения и дестабилизирует обучение.

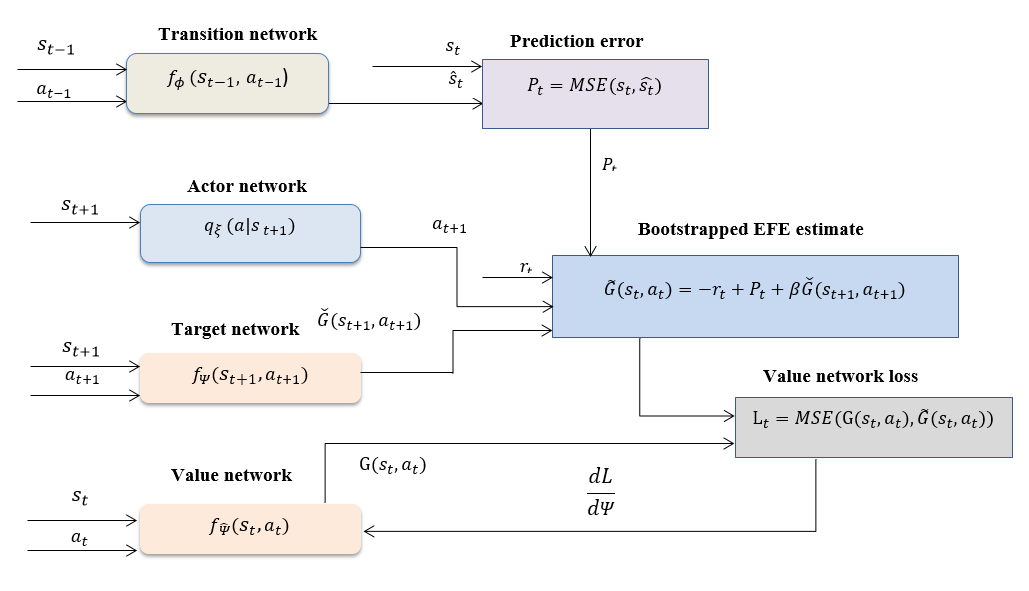


Рисунок 4. Схема расчета EFE.

1. Отказ от «весов» перед EFE следующего шага. Тогда получается стандартная critic-сеть из обучения с подкреплением.

Схема расчета по этой формуле показана на рисунке 4.

1. Попытка учесть распределение действий на подобие тому, как это сделано в SAC и soft q-value.

Идея различных soft-value методов в RL состоит в том, что агент будет искать такую политику, которая максимизирует энтропию в конкретном состоянии, а значит приведет агента в состояния, о которых у него меньше информации. Это преследует ту же идею, что и теория AIF - агент будет стремиться уменьшить свою неопределённость и собирать как можно больше информации.

Этот способ расчета ожидаемой свободной энергии больше походит на улучшение метода SAC, потому что, как будет рассмотрено далее при расчете VFE, также присутствует термин энтропии действий с противоположным знаком.

Второй и третий вариант будут протестированы на реальных задачах и сравнены. Важно заметить, что аппроксимируется наградой в момент времени t. Для расчета используется модель переходов.

2.2.3. Вывод формулы для подсчета вариационной свободной энергии

Для адаптации расчета вариационной свободной энергии обратимся к формуле, полученной ранее в пункте 1.1.

Первый элемент аппроксимируется моделью переходов, второй – вариационным автоэнкодером в случае рассмотрения задачи как POMDP, однако с третьим возникают проблемы. В оригинальной работе, авторы раскладывают расстояния Кульбака-Лейблера по определению:

Как можно заметить, здесь идет перебор всех возможных действий в сумме, и также, как и в EFE, умножение на значение плотности, которое является экспонентой. И аппроксимация в свою очередь предполагает, что есть вектор свободных энергий для каждого возможного действия, и поэтому может быть использована функция softmax.

Для адаптации этого расчета для непрерывных действий, воспользуемся выводом очень похожего расстояния Кульбака-Лейблера, проведенным в SAC, но слегка адаптируем формулы под текущую задачу.

*Заменяем softmax экспонентой, как это сделано в SAC, однако оставляем знак минус из активного вывода, так как в отличии от SAC здесь value-функция минимизируется.*

*Распишем по определению:*

*Константа Z заменяется на*

*Преобразуем в математическое ожидание:*

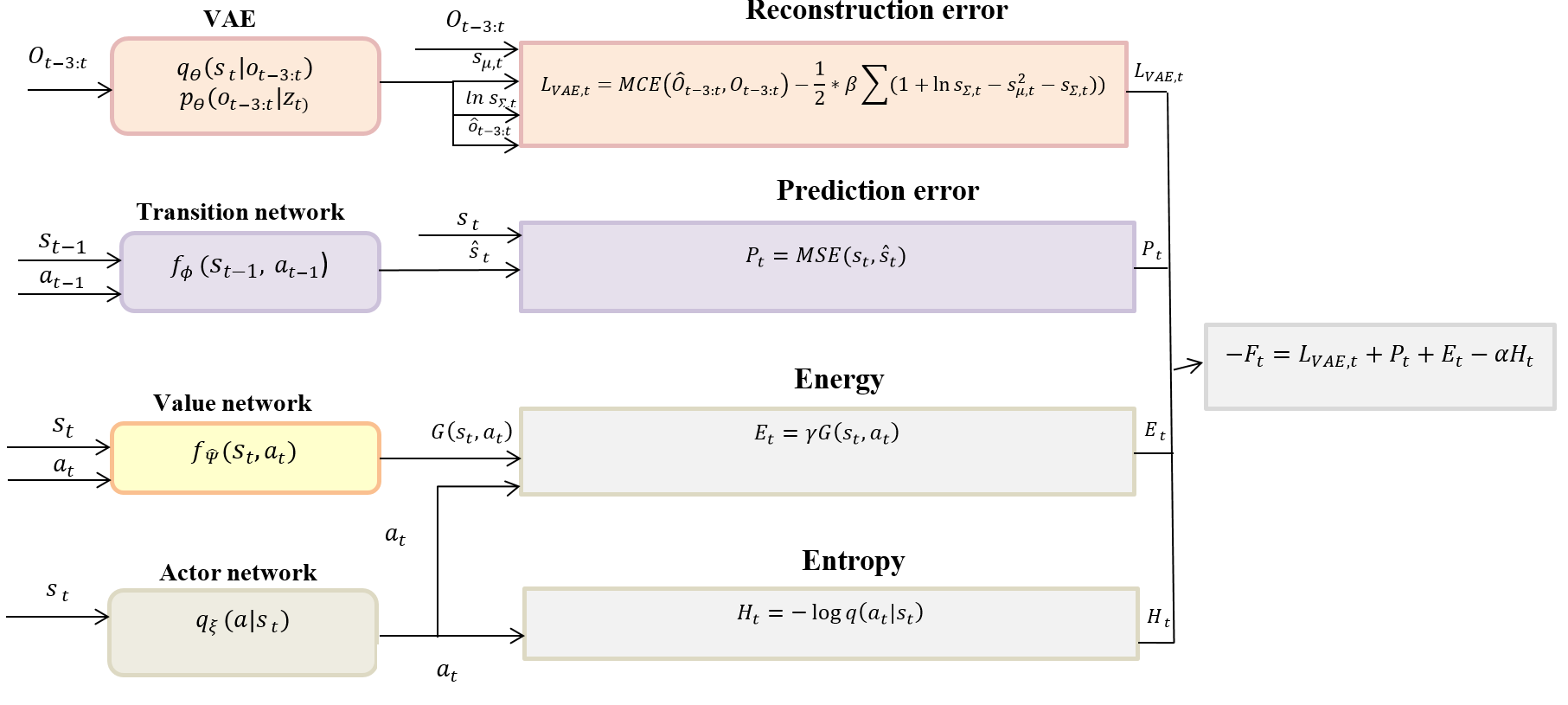
Учитывая это, формула VFE для непрерывных действий может быть переписана следующим образом:

Здесь, все математические ожидания, как это принято называть, аппроксимируются выборкой. Это популярный прием в обучении с подкреплением при оптимизации через SGD, чтобы не считать математическое ожидание вручную. Например, так было сделано в [22, 25].

Так, теперь можно рассчитать все части формулы VFE: первая часть уравнения может быть интерпретирована как ошибка предсказания состояния, и будет рассчитана как Mean-Squared-Error (MSE) между выходом модели переходов и истинным состоянием. Такой способ расчета был предложен в [7]. В этой работе рассматривается два разных варианта предсказания следующего состояния: обычная полносвязная сеть и реккурентная сеть(lstm)

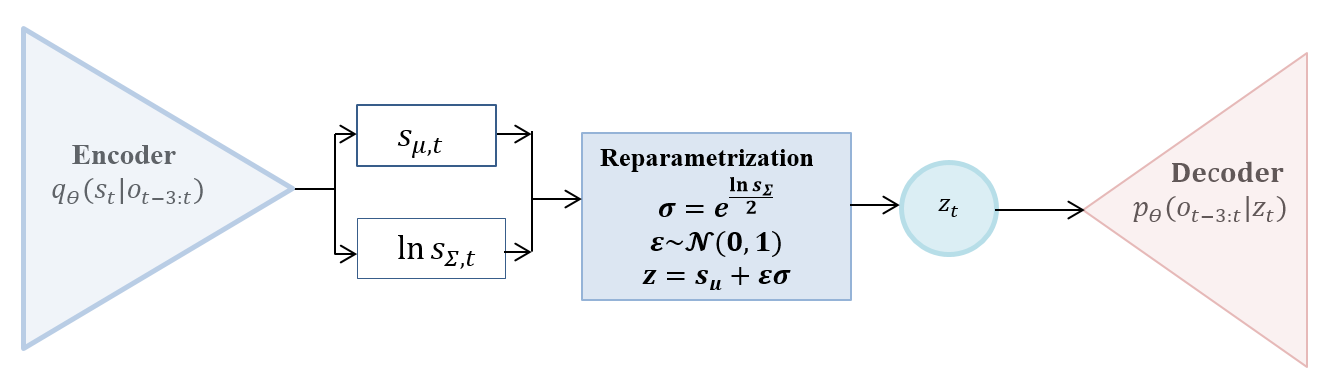
Вторая часть описывает маппинг наблюдений во внутренние состояния. Здесь мы идем общепринятым путем и аппроксимируем это значение с помощью вариационного автоэнкодера (VAE). Если рассмотреть это более детально, VAE состоит из сети, которая кодирует наблюдения как распределение над внутренними состояниями (encoder) - , и из сети, которая строит деконструкцию входа исходя из внутренних состояний (decoder) - . Общая схема изображена на рисунке 6. В этой работе рассматривается beta-VAE [29], который является модификацией вариационного автоэнкодера с особым упором на обнаружение распутанных скрытых состояний, то есть таких состояний, которые связанны с конкретным признаком наблюдения.

Рисунок 5. Схема расчета VFE. POMDP.



Третья часть свободной энергии рассчитывается следующим образом: предсказывается нейронной сетью агента, как это было описано в 2.2.1., а – это выход value-network, оптимизация которой описывалась в предыдущем пункте. При подсчете VFE, действие сэмплируется через репараметризацию и передается на вход value-network.

Рисунок 6. Архитектура VAE для аппроксимации маппинга из наблюдений на внутренние состояния.



2.2.4. POMDP и MPD версии

Аналогично [7], в этой работе будет рассмотрено две версии алгоритма: для случая, когда состояние мира известно (MDP) и для случая, когда состояние неизвестно (POMDP). Все, что было описано выше, касается второго случая. Первый же способ – упрощение - предполагается, что агент имеет доступ к истинному состоянию мира и в реконструкции не нуждается. Тогда из филомелы свободной энергии с рисунка 5 пропадает элемент .

При рассмотрении задачи как POMDP, агент может наблюдать лишь изображения, и получать из них внутренние скрытые состояния путем кодирования через вариационный автоэнкодер. При таком типе наблюдений применить HER невозможно. Поэтому в качестве замены рассматривается PER, как попытка улучшить качество данных для обучения.

Глава 3. Реализация и эксперименты

Практическая цель этой работы – обучить агента активного вывода для решения задачи управления роботами. Как более узкая область этой объемной задачи было выбрано управление хирургическими роботами [31]. В этой главе будет описан выбор симулятора и задач на которых был протестирован предлагаемый алгоритм, а также проведен анализ различных вариаций алгоритма и его сравнение с методом soft-actor-critic.

Код представлен в GitHub репозитории https://github.com/bogatovam/daif-surgical-robot-control.

* 1. Выбор симулятора

Для проведения экспериментов, необходимо выбрать симулятор среды, в которой будет действовать агент. На момент работы над исследованием, было известно три симулятора различных хирургических задач, наиболее продвинутыми из которых являются dVRL[31] и SurRol[32]. В таблице [1] приведено сравнение известных сред. Во многом при этом сравнении выигрывает SurRol. Это более продвинутый симулятор, в котором поддержано целых 10 различных задач, и он обладает гораздо лучшей переносимостью на реального робота. Особенно важно то, что для его запуска требуется только python, так как он использует физический движок PyBullet. Как покажет дальше практика, этот симулятор не оптимизирован и довольно сложен для решения, однако это лучшее что есть на данный момент и, возможно, является компромиссом между точностью и удобством.

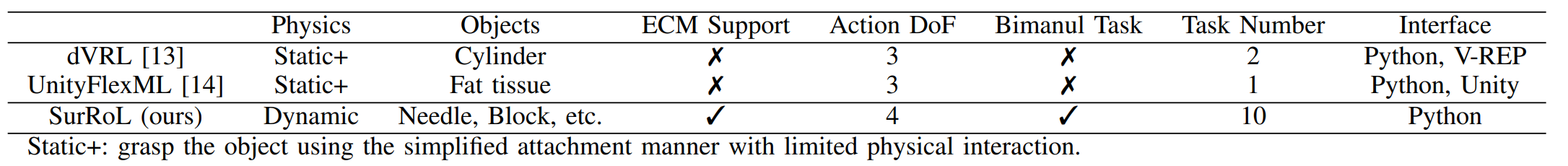


Таблица 1. Сравнение сред моделирования хирургических задач. Источник [32]

Также, в качестве теста, который покажет базовую работоспособность агента, была выбрана задача FetchReach из симулятора MuJoCo [30]. Это необходимо, так как запускать некоторые конфигурации модели сразу на SurRol. слишком дорого по времени, и эта задача помогла отбросить сразу нерабочие варианты.

* 1. Описание задач

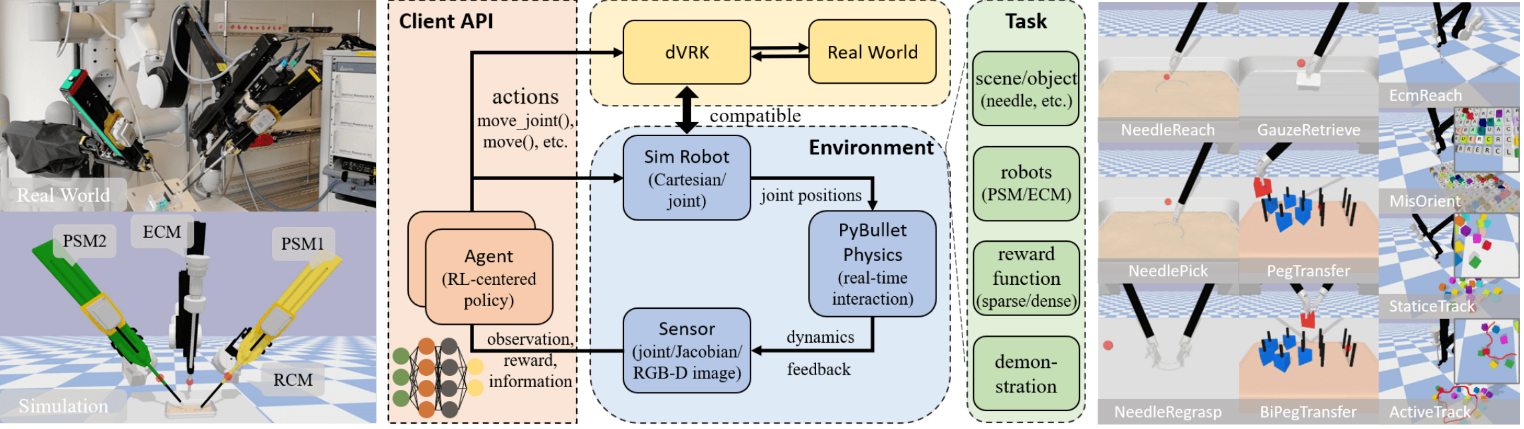


Рисунок 7. Дизайн SurRoL. Симулятор предоставляет совместимые с реальным роботом dVRK среды моделирования для обучения хирургических роботов (слева), с интерфейсами типа OpenAI Gym для разработки алгоритмов обучения с подкреплением и с десятью хирургическими задачами различных уровней сложности (справа).

Для оценки качества алгоритма, были решены следующие задачи:

1. *FetchReach*, MuJoCo - Задача состоит в том, чтобы переместить робо-руку в целевое положение. Награда – бинарная, то есть в случае успеха 0, в случае неудачи -1.
2. *FetchPickAndPlace,* MuJoCo– цель робота - достигнуть кубика, произвести его захват и перенести в определенную точку.
3. *NeedleReach*, SurRol – задача достижения цели в пространстве, аналогично предыдущей, однако сложнее – так как нужно исследовать большее пространство. Награда – расстояние до цели.
4. *NeedleGrasp* – в этой задаче необходимо клешнями робо-руки захватить иглу. Награда бинарная.

Все эти задачи имеют общую структуру: роботизированная рука выполняет действия в пространстве. Наблюдения этих окружений имеют следующую структуру.

* *observation*: фактическое наблюдение за окружающей средой, например, состояние робота и положение объектов.
* *desired\_goal*: Цель, которую агент должен достичь. В случае FetchReach это будет трехмерная целевая позиция.
* *achieved\_goal*: цель, которую агент достиг на самом деле. В случае FetchReach это положение клешни робота. В идеале значение должно равняться *desired\_goal*, как можно быстрее. Не используется при обучении, только при HER-сэмплировании.

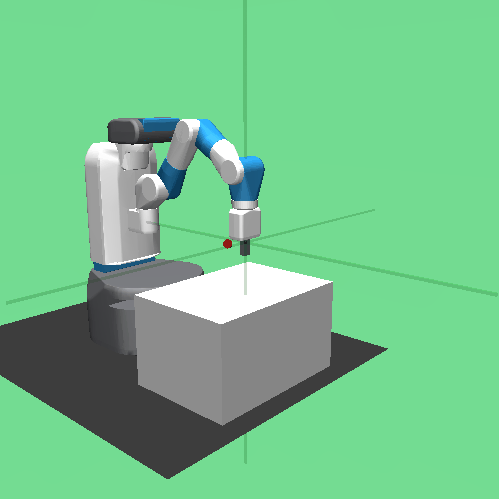
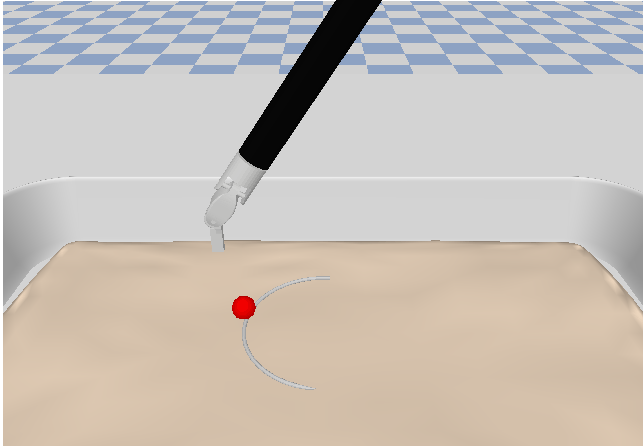


Рисунок 8. Как выглядят окружения. Слева – Mujoco, справа - SurRol

* 1. Описание и цель экспериментов

Итак, цель проведения экспериментов – доказать способность решать задачу и сравнить производительность с тем методом, на котором был основа алгоритм – с SAC

**MDP**

Для полностью наблюдаемой версии алгоритма, проводится ряд экспериментов, во-первых, с различными типами моделей переходов – MLP, LSTM, GRU. Во вторых, с различными версиями подсчета EFE: с учетом энтропии в и без (см. пункт 2.2.2.). Для удобства, первый вариант назовем ***soft-continuous-dAIF***, второй – ***vanilla-continuous-dAIF.***

**POMDP**

Частично наблюдаемая версия алгоритма лишь расширяет MDP версию. Однако она имеет одну особенность – в ней отсутствует HER модель, он заменен на PER. Это потому, что в наблюдениях не содержится информации о достижении цели.

* 1. Детали разработки

Все эксперименты с окружением SurRol проведены на центральном процессоре AMD Ryzen 7 Pro 475U, а задача из Mujoco тестировалась в среде Google Colab, MDP – на центральном процессоре, POMDP на графическом.

Весь код реализован на языке Python с помощью фреймворка PyTorch, который был выбран из-за его гибкости при разработке сложных и нетривиальных процессов обучения нейронных сетей.

* 1. Результаты вычислений
     1. FetchReach

Рисунок 9. График средней награды за эпизод.

Версия ***vanilla-continuous-dAIF*** для задачи FetchReach в сравнении с

SAC (желтая линия). MLP – фиолетовая. LSTM – красная. GRU – зеленая.

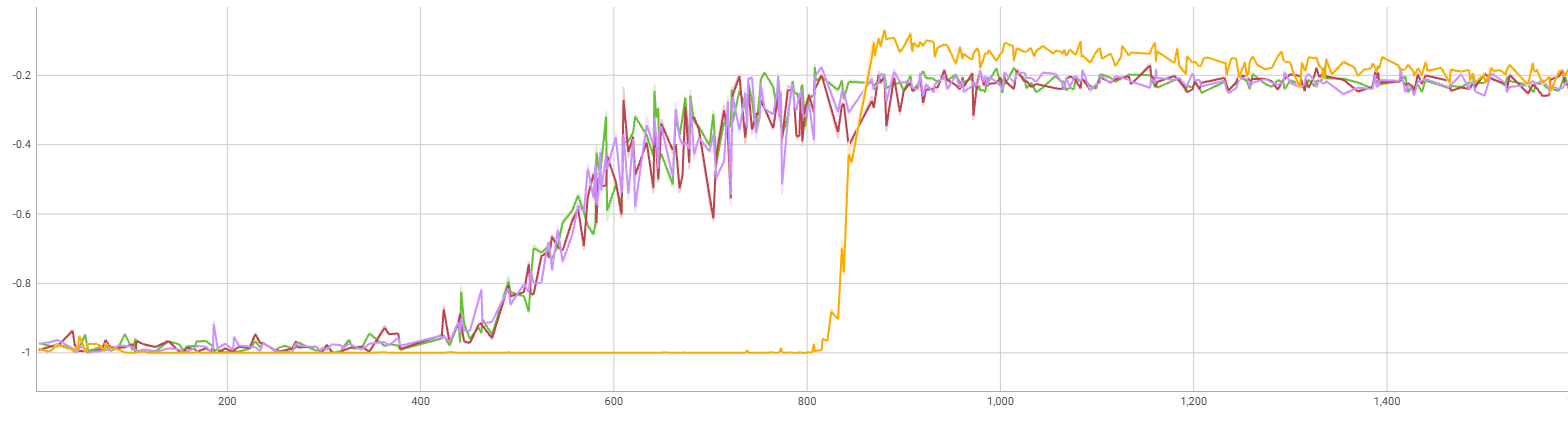
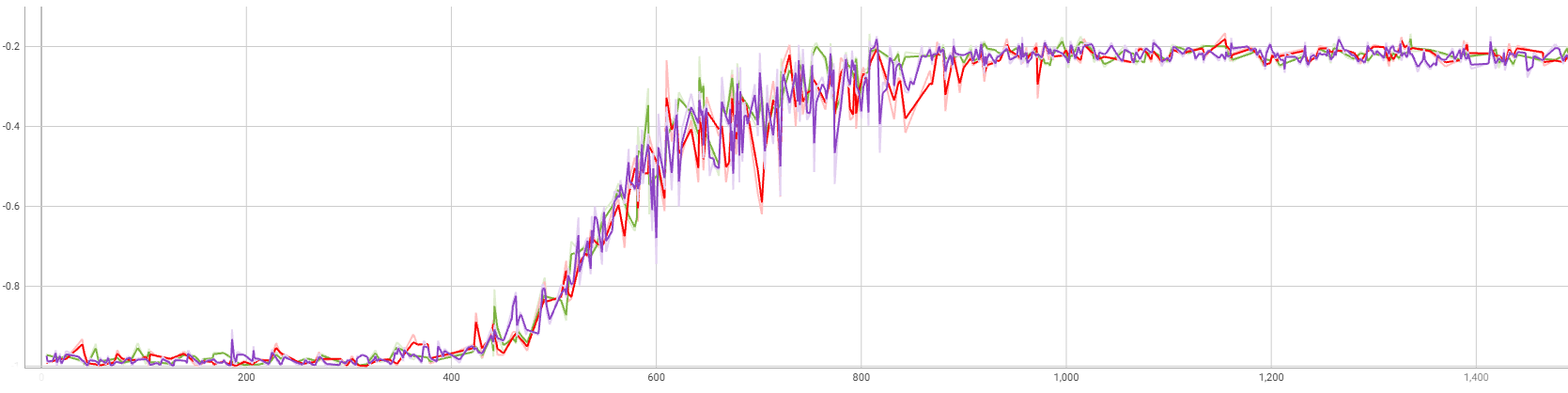


Рисунок 10. График средней награды за эпизод.

Версия ***soft-continuous-dAIF*** для задачи FetchReach в сравнении с

SAC (желтая линия). MLP – фиолетовая. LSTM – красная. GRU – зеленая.



* + 1. NeedleReach

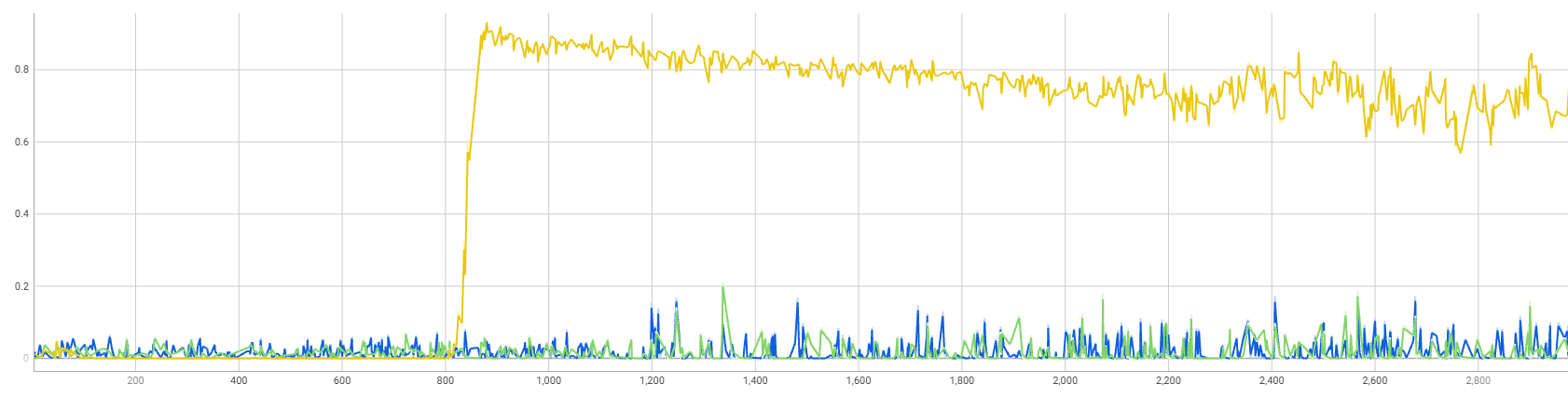


Рисунок 11. График средней награды за эпизод. Версия ***POMDP*** для задачи FetchReach в сравнении с

MDP версией SAC (желтая линия). ***vanilla-continuous-dAIF*** – зеленая. ***soft-continuous-dAIF*** – синяя.

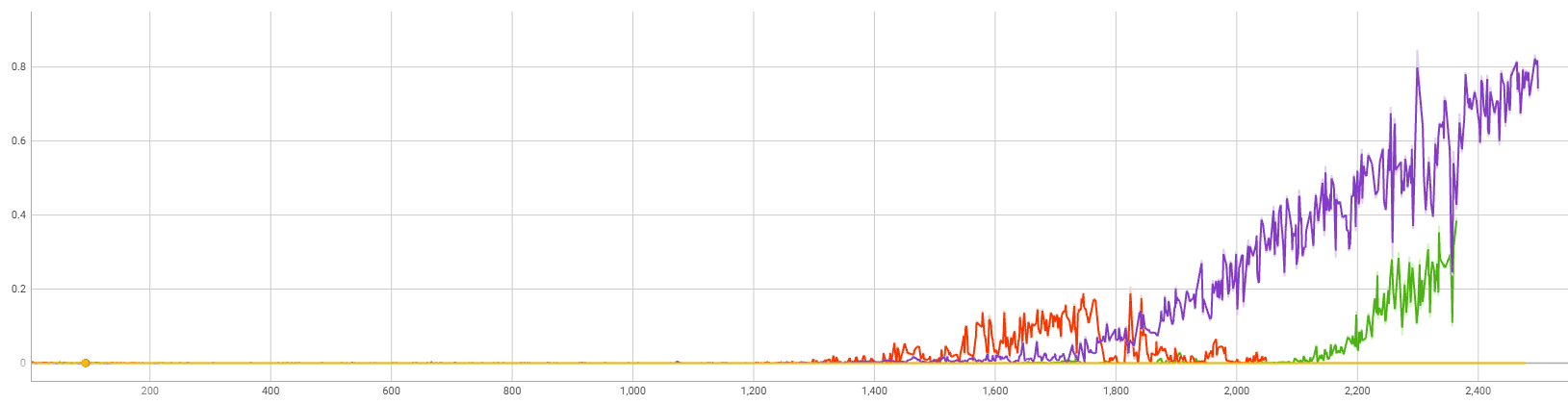


Рисунок 12. График средней награды за эпизод (сверху), график количества успешных шагов за эпизод (снизу). Версия ***soft-continuous-dAIF*** для задачи *NeedleReach* в сравнении с

SAC (желтая линия). MLP – фиолетовая. LSTM – красная. GRU – зеленая

* + 1. NeedleGrasp

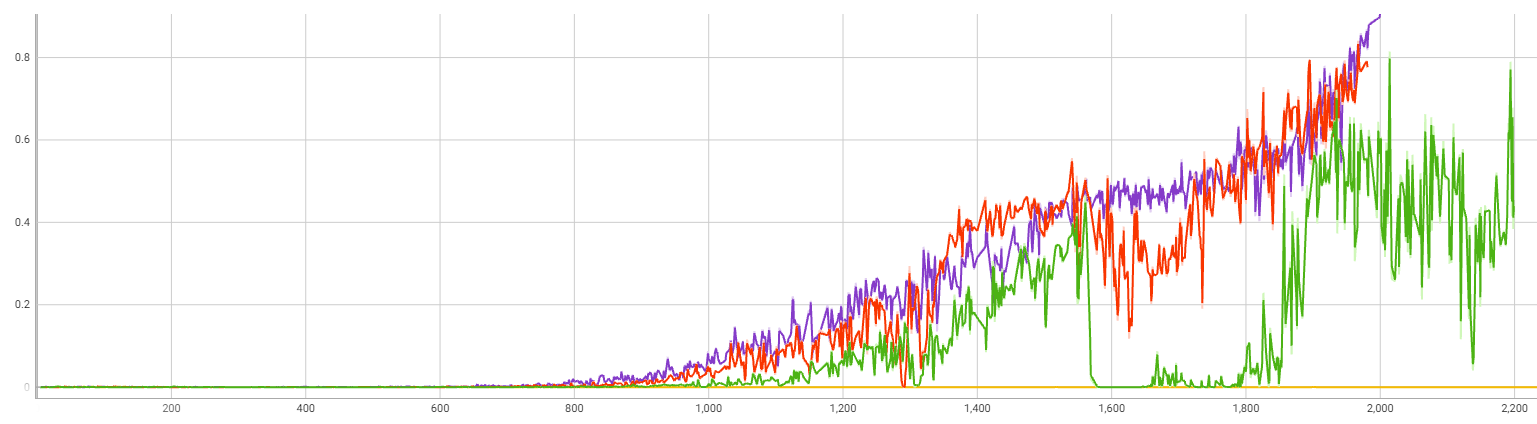
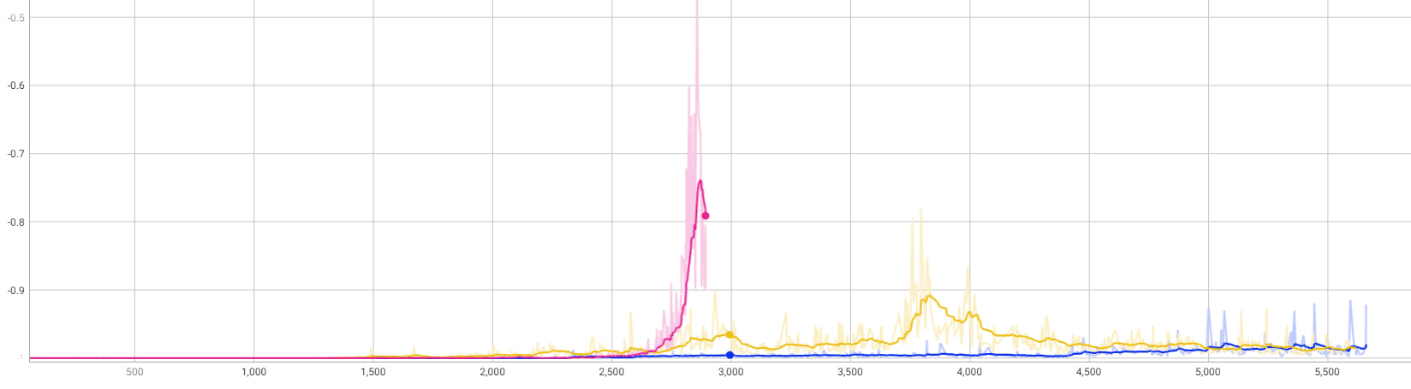


Рисунок 13. График средней награды за эпизод (сверху), график количества успешных шагов за эпизод (снизу). Версия ***vanilla-continuous-dAIF*** для задачи *NeedleReach* в сравнении с

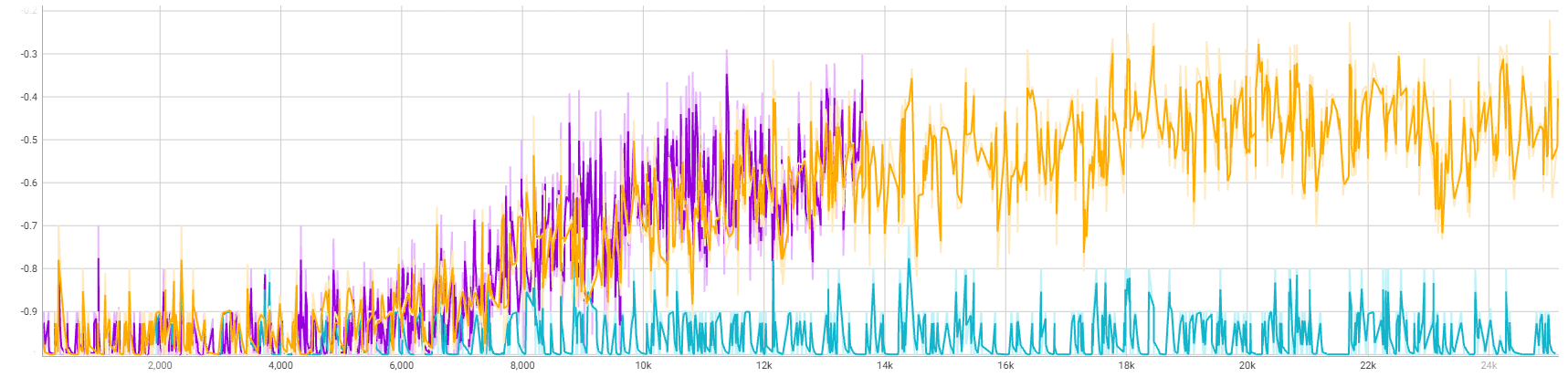
SAC (желтая линия). MLP – фиолетовая. LSTM – красная. GRU – зеленая

Рисунок 14. (Будет обновлено) График средней награды за эпизод. Версии ***vanilla-continuous-dAIF***(розовая линия) и ***soft-continuous-dAIF*** (синяя линия) для задачи *NeedleGrasp* в сравнении с SAC (желтая линия)



* + 1. FetchPickAndPlace

Рисунок 15. График средней награды за эпизод. Версии ***vanilla-continuous-dAIF***(фиолетовая линия) и ***soft-continuous-dAIF*** (голубая линия) для задачи *FetchPickAndPlace* в сравнении с SAC (желтая линия)



* 1. Анализ результатов

В первую очередь, оценим результаты выполнения базового теста FetchReach. Разницы между обеими версиями MDP dAIF и SAC практически нет - все три агента сходятся к одной и той же приемлемой величине награды (рис. 9, 10). Это значит, имеет смысл исследовать применение этих агентов для более сложных задач, в отличии от версии POMDP dAIF(рис.11). Получается так, что изображение не содержит достаточно информации для того, чтобы генерировать правильные действия. Также проблема может быть в том, что автоэнкодер во внутренних состояниях сохраняет много бесполезной для решения задачи информации, например, как цвет фона. Здесь также отсутствует HER, который критически важен для подобного рода задач [27]. Поэтому далее этот вид постановки задачи не рассматривается.

Перейдем к следующей по сложности задачи – NeedleReach. Здесь отчетливо видно, что dAIF агенты справляются лучше и быстрее, чем SAC. Причем, значительно лучших результатов достигла версия ***vanilla-continuous-dAIF.***

Сравнивая результаты ***soft-continuous-dAIF*** и SAC, можно сделать вывод о важности наличия модели переходов: именно наличие ошибки предсказания следующего состояния отличает эти подходы, и позволяет быстрее решить поставленную задачу. Однако, как модель переходов лучше всего работала обычная многослойная нейронная сеть, в отличии от рекуррентных нейронных сетей. Это объясняется тем, что рассматривались достаточно простые архитектуры LSTM и GRU.

Несмотря на то, что цели задач FetchReach и NeedleReach одинаковые, у них разный тип награды: у первого бинарная, такой тип называется разреженный, у второго вещественная, такой тип называется непрерывный. Можно заметить, что агент dAIF агент обладает хорошей способностью к исследованию новых состояний и способен достичь результата как в разреженной среде, так и в непрерывной.

Разницу между непрерывной наградой и разреженной видно при решении задачи NeedleGrasp – для того, чтобы робо-рука достигла цели и начала пытаться захватить ее, нужно гораздо больше эпизодов. Но если посмотреть на график расстояния до цели из этой задачи (рисунок 16) – видно, что определить верное направление движения для агента не проблема.

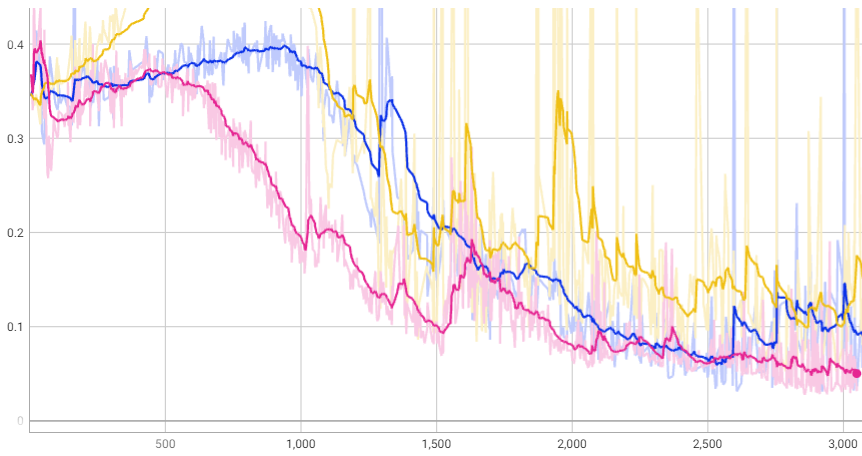


Рисунок 16. График среднего расстояния до цели за эпизод.

Версии ***vanilla-continuous-dAIF*** (розовая линия) и ***soft-continuous-dAIF*** (синяя линия) для задачи *NeedleGrasp* в сравнении с SAC (желтая линия)

Также, сравнивая между собой SAC и dAIF, заметна разница в поведении обученного агента. В отличии от SAC, dAIF более плавно исследует неизвестные состояния. Слишком большая вариативность - известная проблема стохастических агентов, и для SAC даже разрабатываются отдельные методы, помогающие сделать поведение более плавным [33]. В dAIF эта проблема так менее выражена, это видно при решении некоторых задач. Например, для NeedleGrasp:

<https://github.com/bogatovam/daif-surgical-robot-control/tree/main/doc/gifs/NeedleGrasp/sac>

<https://github.com/bogatovam/daif-surgical-robot-control/tree/main/doc/gifs/NeedleGrasp/mdp-daif-soft/mlp>

В этой работе предлагается следующее обоснование такого явления: так, как и ожидаемая и свободная энергия содержат в себе ошибку предсказания следующего состояния, и агент стремиться ее уменьшить, то это влечет за собой поощрение выбора такого действия, которое приведет в предсказуемое, последовательное состояние.

Также дополнительно была рассмотрена задача *FetchPickAndPlace,* с которой не справился алгоритм ***soft-continuous-dAIF***. Производительность остальных двух, аналогично FetchReach, не сильно отличается.

Результаты работы всех описанных агентов можно найти в открытом доступе в репозитории https://github.com/bogatovam/daif-surgical-robot-control

Заключение

В этой работе была предложена адаптация метода глубокого активного вывода для задач с непрерывным пространством действий.

В результате анализа предметной области и исследования существующих работ по теме масштабирования активного вывода, был сформулирован новый способ расчета ожидаемой и вариационной свободной энергии при условии, что количество и размерность действий агента неизвестны. В последствии этот способ реализован и успешно применен для решения задачи управления хирургическим и обычным роботом.

Для оценки способности dAIF агента решать задачу были проведены эксперименты, а также сравнение производительности с методом SAC. MDP версия алгоритма показала сравнимые с SAC результаты, а в некоторых задачах даже превосходящие. В свою очередь POMDP оказалась низко производительной даже на самых простых тестовых задачах. Из этого можно сделать вывод, что обычного вариационного автоэнкодера недостаточно для решения комплексных задач с непрерывными пространствами действий. В качестве развития предложенного подхода необходимо искать другие способы репрезентации скрытых состояний агента, например, как в [12]. Еще одним вариантом улучшения является реализация планирования. Комбинация текущей работы с такими подходами как [11] и [18] является перспективной областью для будущих исследований.

Список литературы

1. Pietro Mazzaglia, Tim Verbelen , Ozan Çatal and Bart Dhoedt, The Free Energy Principle for Perception and Action: A Deep Learning Perspective. 2022.
2. Pablo Lanillos, Cristian Meo, Corrado Pezzato, Ajith Anil Meera, Mohamed Baioumy, Wataru Ohata, Alexander Tschantz, Beren Millidge, Martijn Wisse, Christopher L. Buckley, and Jun Tani. Active Inference in Robotics and Artificial Agents: Survey and Challenges. 2021.
3. How Active Inference Could Help Revolutionise Robotics 2022
4. Ueltzhöffer K. Deep active inference. Biol. Cybern. 2018;112:547–573. doi: 10.1007/s00422-018-0785-7
5. Millidge B. Deep Active Inference as Variational Policy Gradients. arXiv. 2019 doi: 10.1016
6. Aleksey Zelenov, Vladimir Krylov, Deep active inference in control tasks, 2021.
7. Otto van der Himst, Pablo Lanillos, Deep Active Inference for Partially Observable MDP
8. Alexander Tschantz, Manuel Baltieri, Anil. K. Seth, Christopher L. Buckley. Scaling active inference. 2019
9. Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. arXiv:1811.04551 [cs, stat], 2018
10. Alexander Tschantz, Beren Millidge, Anil K. Seth, Christopher L. Buckley. Reinforcement Learning through Active Inference 2020
11. Zafeirios Fountas, Noor Sajid, Pedro A.M. Mediano, Karl Friston. Deep active inference agents using Monte-Carlo methods, 2020
12. Pietro Mazzaglia, Tim Verbelen, Bart Dhoedt, Contrastive Active Inference. 2022
13. Ryan Smith, K. Friston, Christopher Whyte. A Step-by-Step Tutorial on Active Inference and its Application to Empirical Data. 2021.
14. Noor Sajid, Philip J. Ball, Thomas Parr, Karl J. Friston. Active inference: demystified and compared. 2020.
15. Friston, K., Rigoli, F., Ognibene, D., Mathys, C., Fitzgerald, T., and Pezzulo, G. (2015). Active inference and epistemic value. Cognitive neuroscience, 6(4):187–214.
16. Friston, K., Samothrakis, S., and Montague, R. (2012). Active inference and agency: optimal control without cost functions. Biological cybernetics, 106(8-9):523–541
17. Parr, T. and Friston, K. J. (2018b). The anatomy of inference: Generative models and brain structure. Frontiers in computational neuroscience
18. Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Mohammadamin Barekatain, Simon Schmitt, David Silver. Learning and Planning in Complex Action Spaces. 2021
19. Friston K, FitzGerald T, Rigoli F, Schwartenbeck P, O Doherty J, Pezzulo G. Active inference and learning. Neurosci Biobehav Rev. 2016 Sep;68:862-879. doi: 10.1016
20. Sutton R.S., Barto A.G. Reinforcement Learning: An Introduction. MIT Press; Cambridge, MA, USA: 2018
21. Glimcher P.W. Understanding dopamine and reinforcement learning: The dopamine reward prediction error hypothesis. Proc. Natl. Acad. Sci. USA. 2011
22. Mnih V., Kavukcuoglu K., Silver D., Graves A., Antonoglou I., Wierstra D., Riedmiller M. Playing Atari with Deep Reinforcement Learning
23. Silver D., Hubert T., Schrittwieser J., Antonoglou I., Lai M., Guez A., Lanctot M., Sifre L., Kumaran D., Graepel T., et al. Mastering Chess and Shogi by Self-Play with a General Reinforcement Learning Algorithm
24. Schrittwieser J., Antonoglou I., Hubert T., Simonyan K., Sifre L., Schmitt S., Guez A., Lockhart E., Hassabis D., Graepel T., et al. Mastering Atari, Go, chess and shogi by planning with a learned model.
25. Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, Sergey Levine. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor 2018
26. Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, Daan Wierstra. Continuous control with deep reinforcement learning. 2015.
27. Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, Wojciech Zaremba. Hindsight Experience Replay. 2017.
28. Tom Schaul, John Quan, Ioannis Antonoglou, David Silver. Prioritized Experience Replay. 2016.
29. Irina Higgins, et al. "-VAE: Learning basic visual concepts with a constrained variational framework." ICLR 2017.
30. Matthias Plappert, Marcin Andrychowicz, Alex Ray, Bob McGrew, Bowen Baker, Glenn Powell, Jonas Schneider, Josh Tobin, Maciek Chociej, Peter Welinder, Vikash Kumar, and Wojciech Zaremba. Multi-Goal Reinforcement Learning: Challenging Robotics Environments and Request for Research. 2018
31. Jiaqi Xu, Bin Li, Bo Lu, Yun-Hui Liu, Qi Dou, Pheng-Ann Heng. SurRoL: An Open-source Reinforcement Learning Centered and dVRK Compatible Platform for Surgical Robot Learning. 2021.
32. Florian Richter, Ryan K. Orosco, Michael C. Yip. Open-Sourced Reinforcement Learning Environments for Surgical Robotics. 2020.
33. Raffin, Antonin & Stulp, Freek. (2020). Generalized State-Dependent Exploration for Deep Reinforcement Learning in Robotics.