Transfer learning
Meta-learning

Резяпкин Вячеслав

27.08.2019

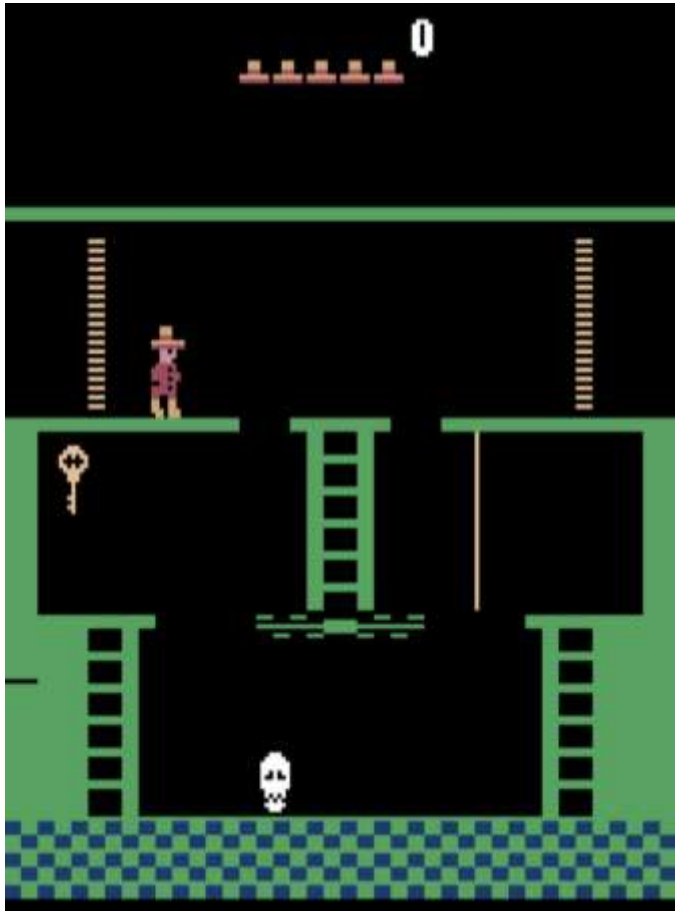
Tinkoff.ru

План

Transfer learning – перенос опыта решения некоторого множества задач (знания) для решения новой задачи

1. **Зачем**
2. **Откуда, куда и как**
    1. Fine-tuning, few-shot learning
    2. Progressive learning
    3. Multi-task learning
    4. Meta-learning

Transfer learning. Мотивация



Montezuma's Revenge

Знания, которые позволяют нам

быстрее учиться играть:

- По *лестнице* можно подниматься и опускаться
- Простая *физика*:

  когда под тобой ничего нет – ты падаешь
- *Ключ* может открывать замки
- *Череп*? Мы не знаем что точно он делает,

  но наверное что-то нехорошее

Transfer learning. Терминология

Transfer learning – использование опыта решения некоторого *множества задач* (знания) для решения *новой задачи*

Задача в RL - MDP

K-shot learning:
Shot – обучение на target domain

0-shot : сразу пускаем в бой

1-shot: даём один раз попробовать задачу

k-shot: даём k раз попробовать задачу

Что значит «*один раз попробовать задачу*»?
Например, *засэмплить* эпизод с фиксированной политикой и затем *поучиться* на нём

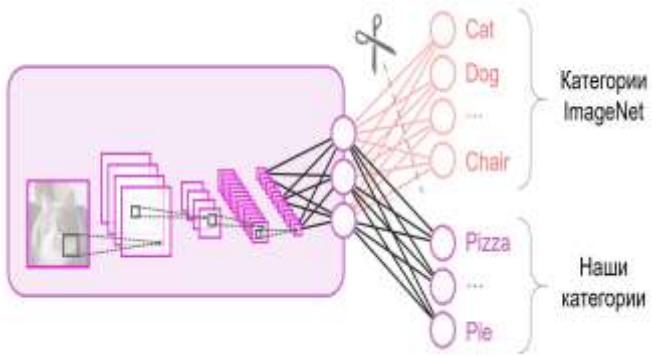
Transfer learning в Deep Learning

Transfer learning – использование опыта решения некоторого *множества задач* (знания) для решения *новой задачи*

Компьютерное зрение

- **Fine-tuning**
  Переносим информативные признаковые представления

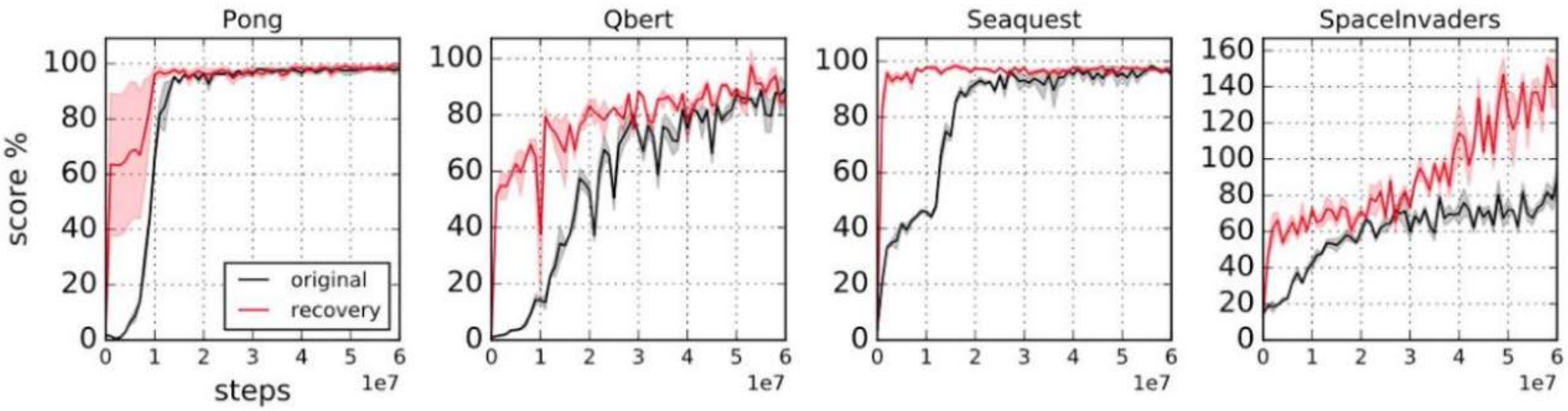
- Если классы в новой задаче (target domain) совпадают с классами исходной задачи (target domain), то просто дообучим сеть на наших данных.
- Если классы в новой задаче (target domain) не совпадают с классами исходной задачи (target domain), то прежде выбросим последний слой и сделаем новый



NLP– эмбеддинги слов

Важность признакового представления

Разделим обучение **признакового представления** от обучения с подкреплением

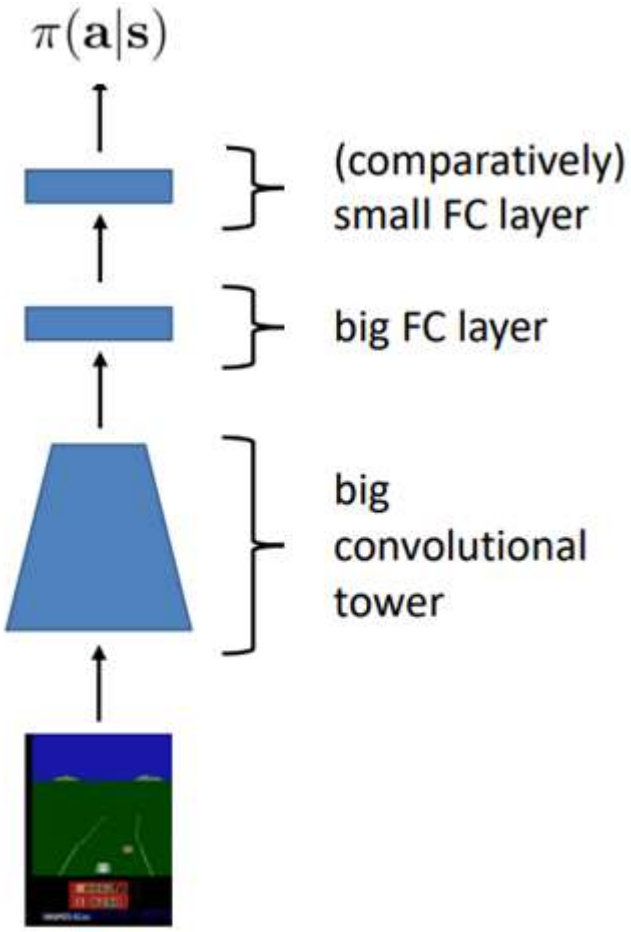


1. Инициализировали RL сеть
2. Обучили - original
3. Сбросили веса последнего слоя
4. Обучили recovery

Recovery – fine-tuned политика и value-функция на выученных признаковых представлениях

Loss is its own Reward: Self-Supervision for Reinforcement Learning

Fine-tuning



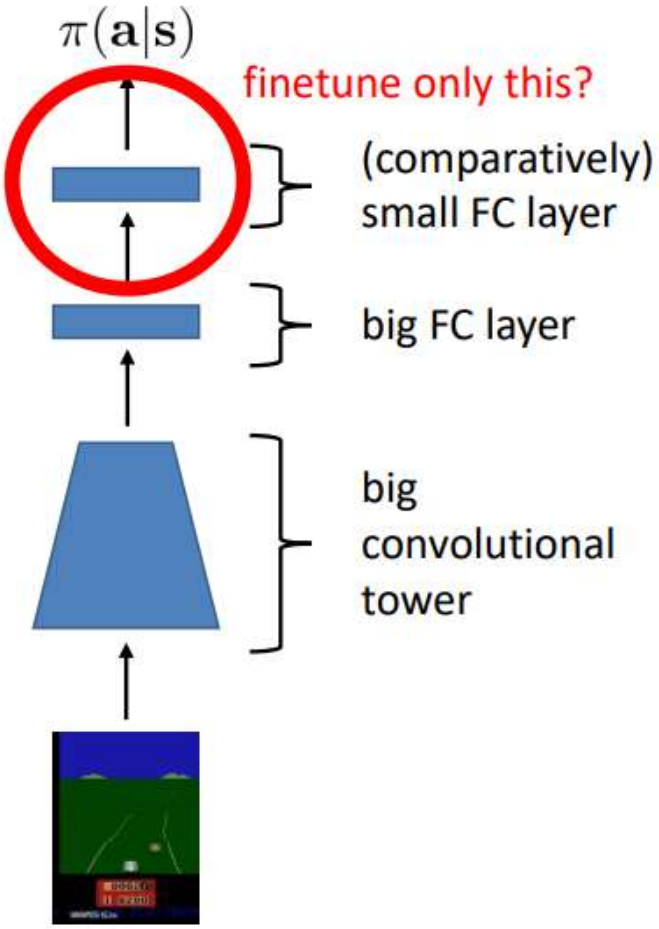
Проблема с обычным fine-tuning:

1. Глубокие нейронные сети хорошо работают, когда содержат много параметров
2. В *target domain* вероятно немного примеров, раз мы хотим прибегнуть к fine-tuning
3. Обучение нейросети на небольшом количестве примеров => переобучение
4. Можем ли мы уменьшить число обучаемых параметров?

Идея:

Учить веса ***только*** последнего слоя, остальные веса фиксируем

Fine-tuning

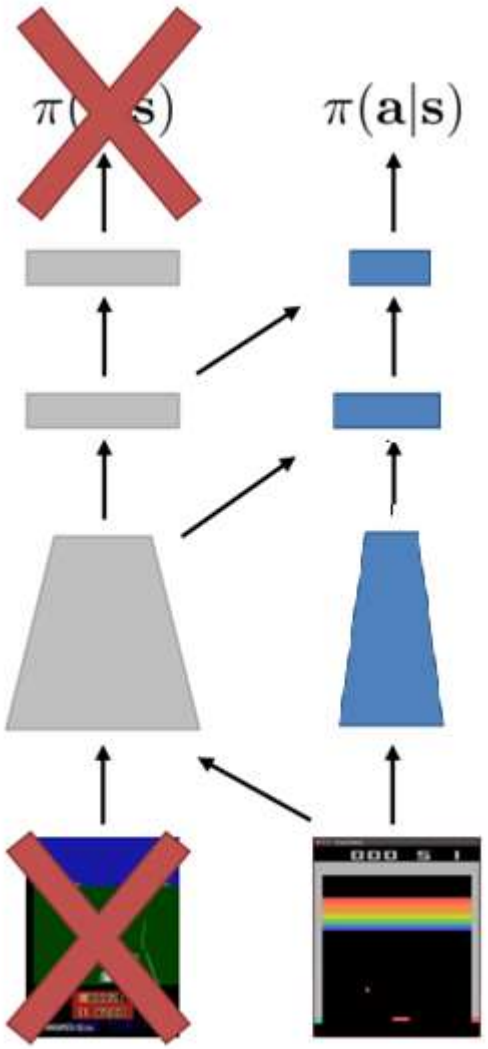


Проблема с обычным fine-tuning:

1. Глубокие нейронные сети хорошо работают, когда содержат много параметров
2. В *target domain* вероятно немного примеров, раз мы хотим прибегнуть к fine-tuning
3. Обучение нейросети на небольшом количестве примеров => переобучение
4. Можем ли мы уменьшить число обучаемых параметров?

Идея:
Учить веса ***только*** последнего слоя, остальные веса фиксируем
Тоже так себе:
Ограниченная выразительность
Признаки с исходной игры могут быть не так актуальны в новой игре

Fine-tuning



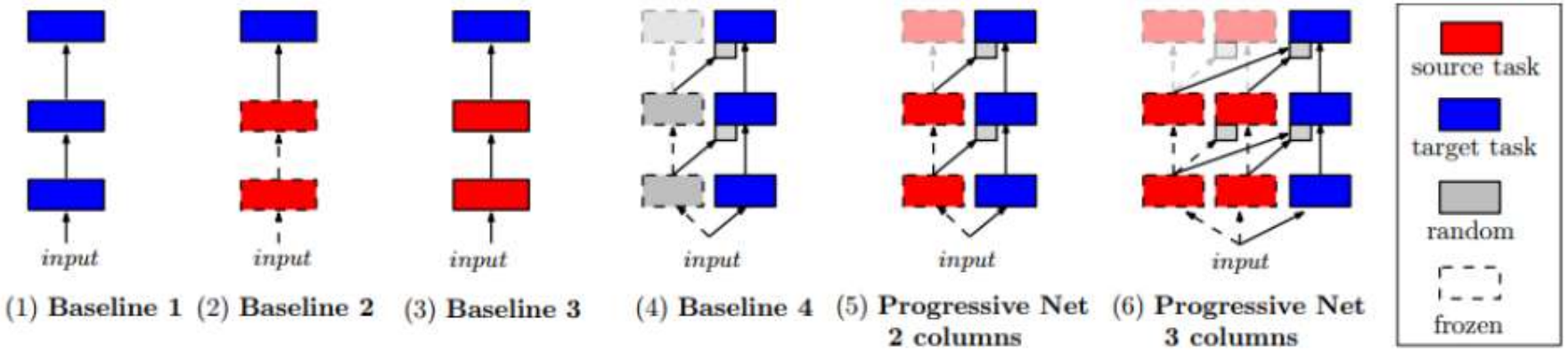
Проблема с обычным fine-tuning:

1. Глубокие нейронные сети хорошо работают, когда содержат много параметров
2. В *target domain* вероятно немного примеров, раз мы хотим прибегнуть к fine-tuning
3. Обучение нейросети на небольшом количестве примеров => переобучение
4. Можем ли мы уменьшить число обучаемых параметров?

Идея:
Учить веса ***только*** последнего слоя, остальные веса фиксируем
Тоже так себе:
Ограниченная выразительность
Признаки с исходной игры могут быть не так актуальны в новой игре

Идея 2:
Сохранить старые веса, но и использовать сырые признаки тоже

Progressive network



(1) **Baseline 1** (2) **Baseline 2** (3) **Baseline 3** (4) **Baseline 4** (5) **Progressive Net 2 columns** (6) **Progressive Net 3 columns**

	Pong Soup		**Atari**		**Labyrinth**	
	Mean (%)	Median (%)	Mean (%)	Median (%)	Mean (%)	Median (%)
Baseline 1	100	100	100	100	100	100
Baseline 2	35	7	41	21	88	85
Baseline 3	181	160	133	110	235	112
Baseline 4	134	131	96	95	185	108
Progressive 2 col	209	169	132	112	**491**	**115**
Progressive 3 col	**222**	**183**	140	111	—	—
Progressive 4 col	—	—	**141**	**116**	—	—

Table 1: Transfer percentages in three domains. Baselines are defined in Fig. 3.

Что ещё интересного?

Transfer learning может быть особенно полезен в задачах реального мира

Например, учить беспилотные автомобили с нуля может оказаться дорого.
Было бы здорово сначала поучить их ездить в симуляции, а затем перенести опыт на реальные дороги

CAD2RL: Real Single-Image Flight without a Single Real Image

Дрона обучили летать в искусственной среде, затем без дообучения отправили в реальный мир

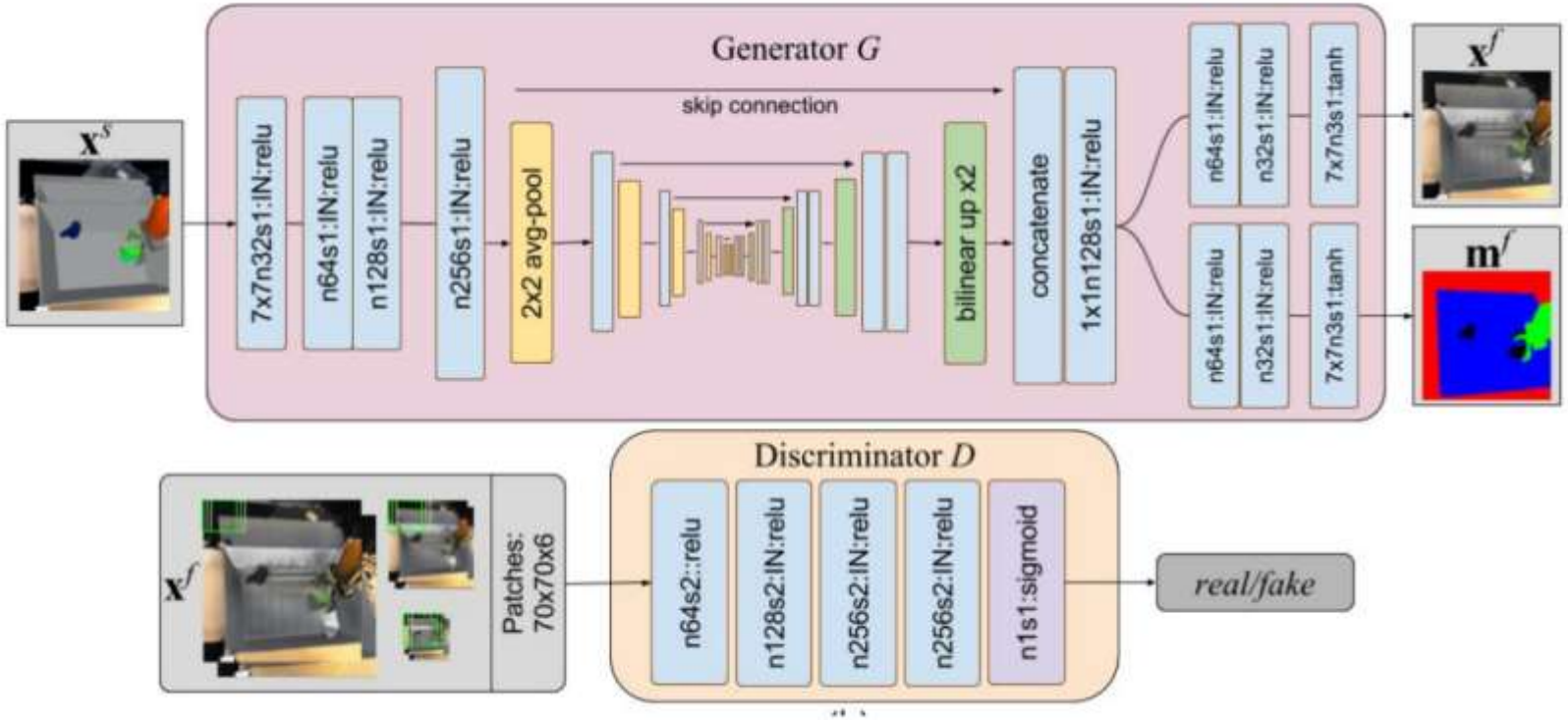
CAD2RL: Real Single-Image Flight without a Single Real Image

Дрона обучили летать в искусственной среде, затем без дообучения отправили в реальный мир

0-shot learning

Адаптация к реальному миру

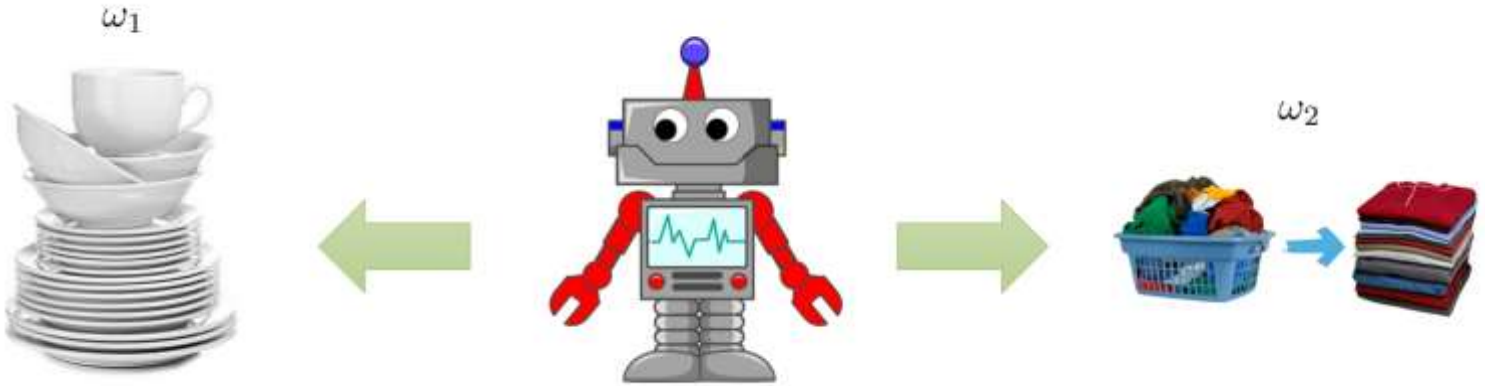
Что если у нас есть фотографии реального мира?
Сделаем симуляцию похожей на реальность с помощью GAN



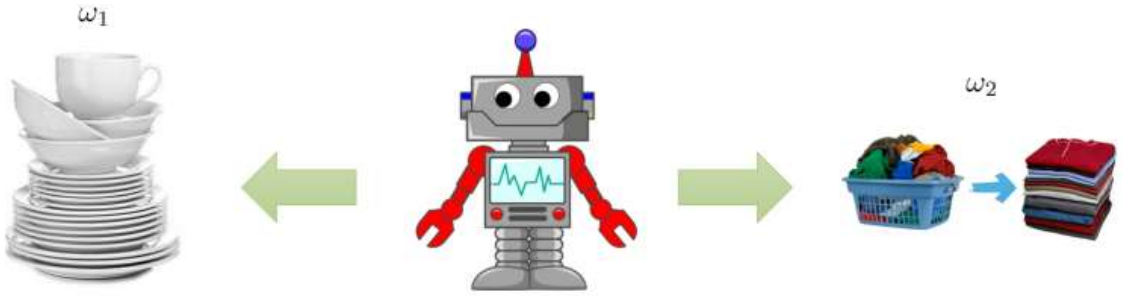
Bousmalis et al., "Using Simulation and Domain Adaptation to Improve Efficiency of Deep Robotic Grasping"

Multi-task learning

До сих пор мы рассматривали только one-task learning
Учимся **на одной** задаче, переносим опыт в другую задачу
Пусть, теперь, мы учимся **на нескольких** задачах
Что это вообще значит – одним агентом в одной среде учиться нескольким задачам?



Multi-task learning



Как сформулировать такую задачу?

Дополним состояние (здесь - картинка) **контекстом** ω:

$$s' = (s, \omega)$$

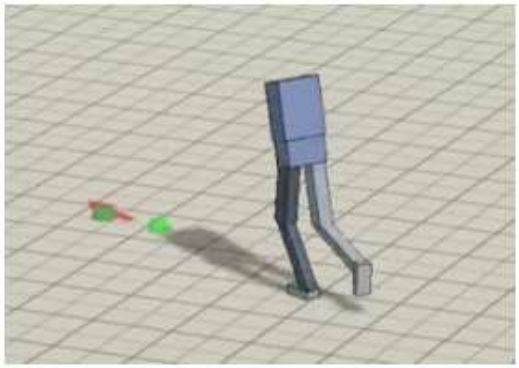
По **контексту** агент сможет определять, какую именно задачу он должен делать.

Как можно задавать **контекст**:

- **ОНЕ**, если конечное небольшое количество задач
- **Картинка**, если задач много. Например, фотография полки, которую нужно протереть
- …

Multi-task learning

ω: stack location



ω: walking direction

ω: where to hit puck

Как сформулировать такую задачу?

Дополним состояние (здесь - картинка) **контекстом** ω:

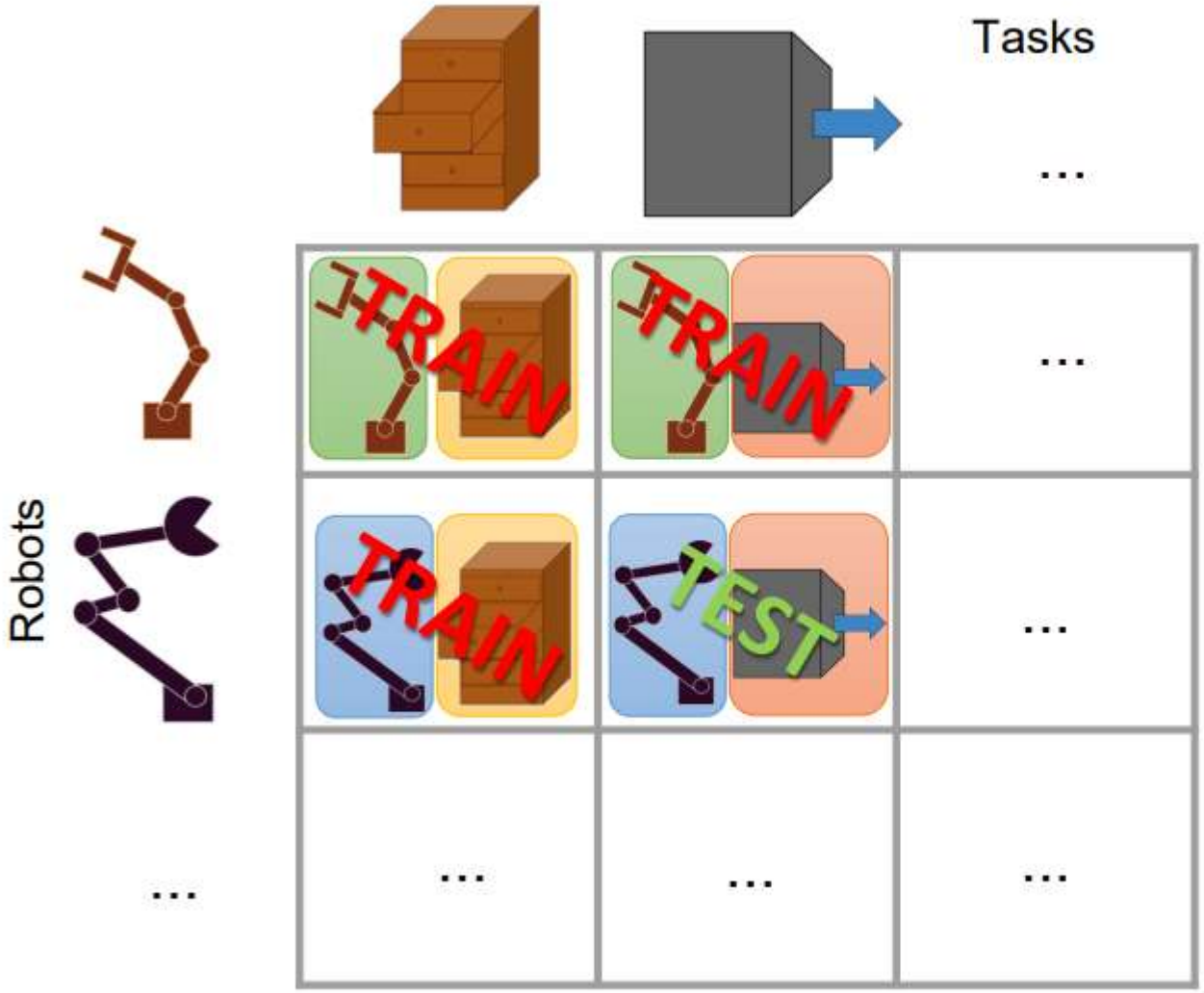
$$s' = (s, \omega)$$

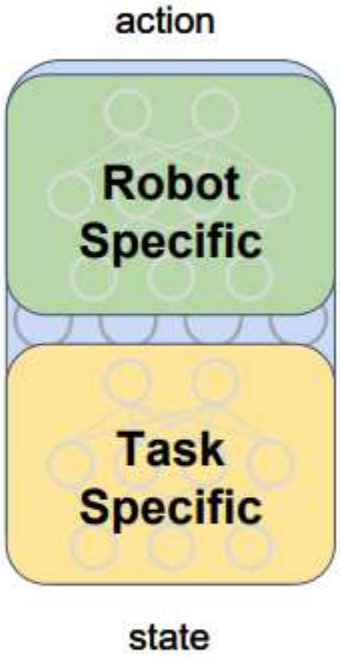
По **контексту** агент сможет определять, какую именно задачу он должен делать.

Как можно задавать **контекст**:

- **ОНЕ**, если конечное небольшое количество задач
- **Картинка**, если задач много. Например, фотография полки, которую нужно протереть
- …

Modular Networks





Learning Modular Neural Network Policies for Multi-Task and Multi-Robot Transfer

Задача:

I. Инициализируется точка в начале координат
II. Задача – прийти в состояние-цель
III. Цель – точка на координатной плоскости (например (-10, -10))
IV. Действие – шаг в координатной плоскости
V. Награда – минус расстояние до цели

Цель можно задавать **явно** в виде контекста Тогда $\text{state} = (x, y, goal.x, goal.y)$	Цель можно не задавать явно. Хочется, чтобы агент понимал где его цель на этот раз с помощью предыдущих состояний, действий, наград Для представления состояния можно использовать RNN

Задача:

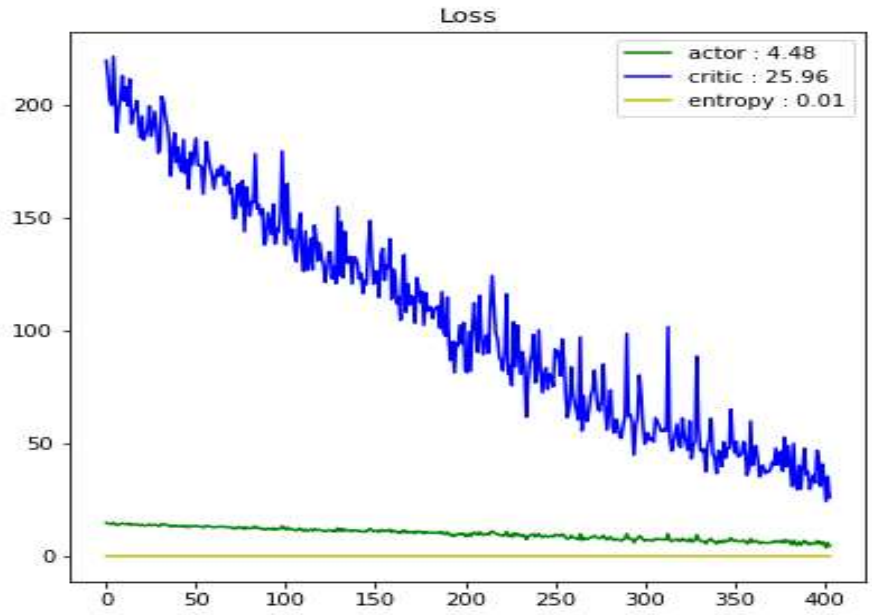
I. Инициализируется точка в начале координат
II. Задача – прийти в состояние-цель
III. Цель – точка на координатной плоскости (например (-10, -10))
IV. Действие – шаг в координатной плоскости
V. Награда – минус расстояние до цели

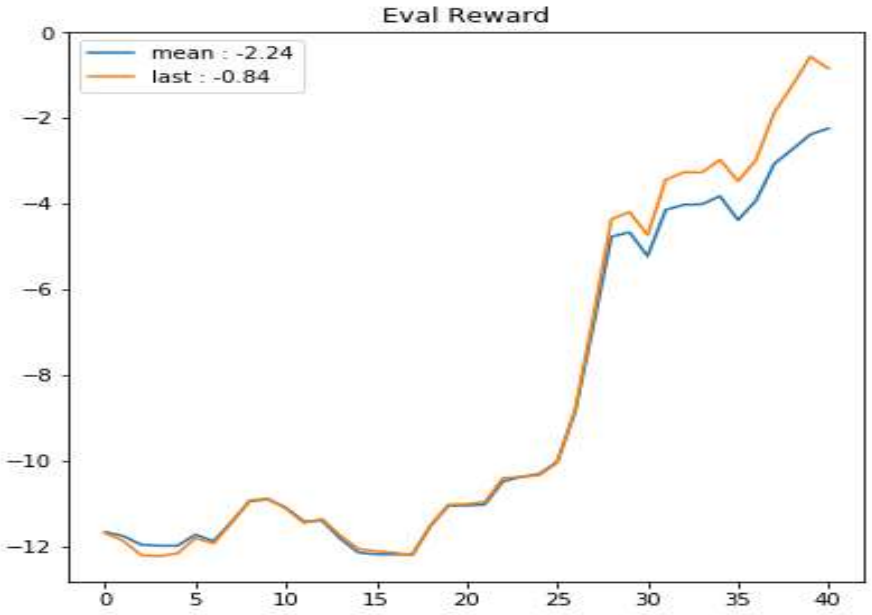
Цель можно задавать **явно** в виде контекста
Тогда $\text{state} = (x, y, goal.x, goal.y)$

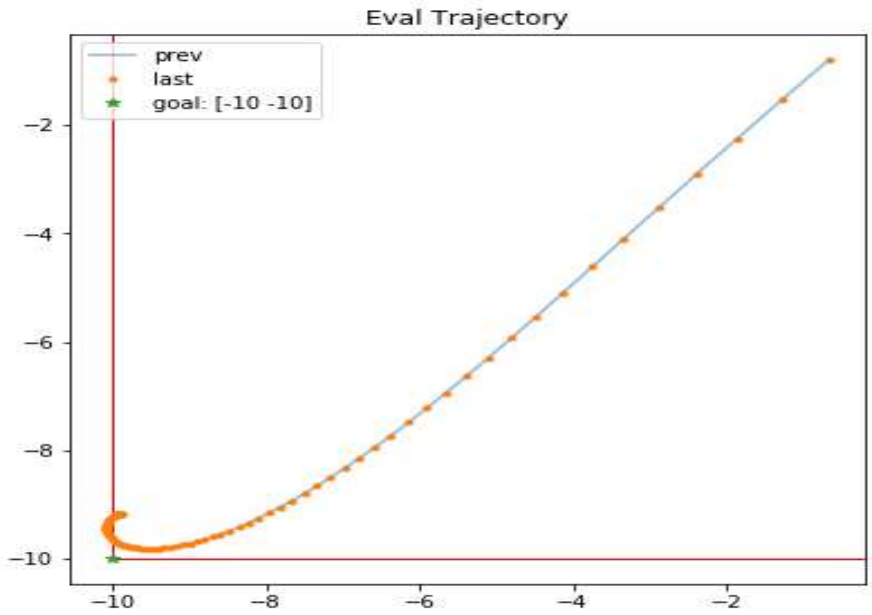
- Цель можно не задавать явно
- Хочется, чтобы агент понимал где его цель на этот раз по предыдущим состояниям, действиям, наградам
- Для представления состояния можно использовать RNN

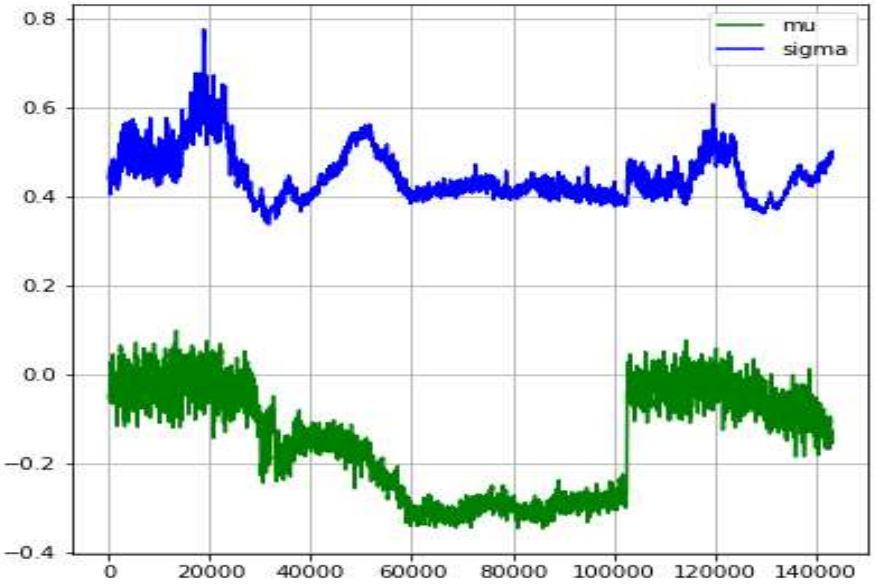
Meta-learning:
Проверим обобщающую способность агента.
Для этого разобъем множество целей на train и test
Таким образом проверим качество агента на отложенной выборке

Steps done: 99,
Epoch done: 403,
Working time: 0h: 6m: 8s,









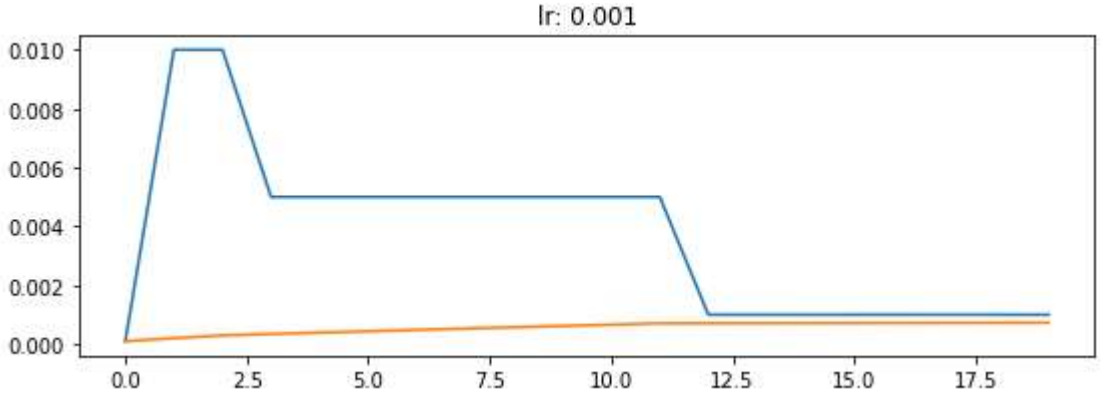
Learning to learn

Постановка задачи в терминах обучения с подкреплением

Пространство действий
$A = \{10^{-4}, 5 * 10^{-4}, 10^{-3}, 5 * 10^{-3}, 10^{-2}\}$
Состояние $s = (i, lr, loss_{last}, loss_{prev})$ — номер итерации, η на предыдущем шаге, значения функции на двух предыдущих шагах
Награда $r = -loss$
Политика $\pi : s \to a$
Траектория $\tau = (s_0, a_0, s_1, a_1, \ldots, a_{T-1}, s_T)$
Задача $J(\pi) = \mathbb{E}_{\tau \sim \pi} \sum_{i=1}^{20} r_i \to \max_\pi$
Геометрический смысл — минимизации площади под кривой функции ошибки

Learning to learn

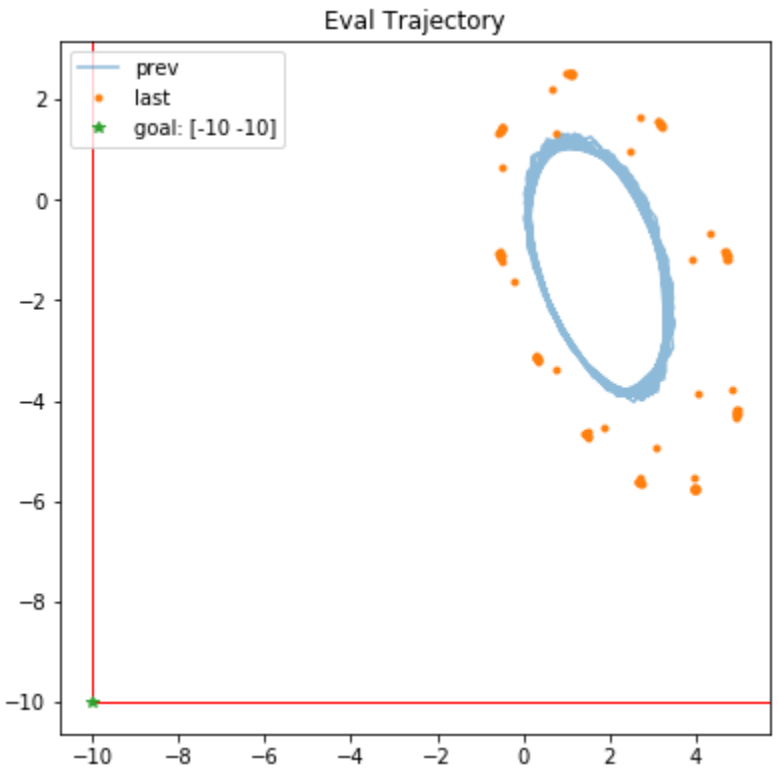
Выученная политика

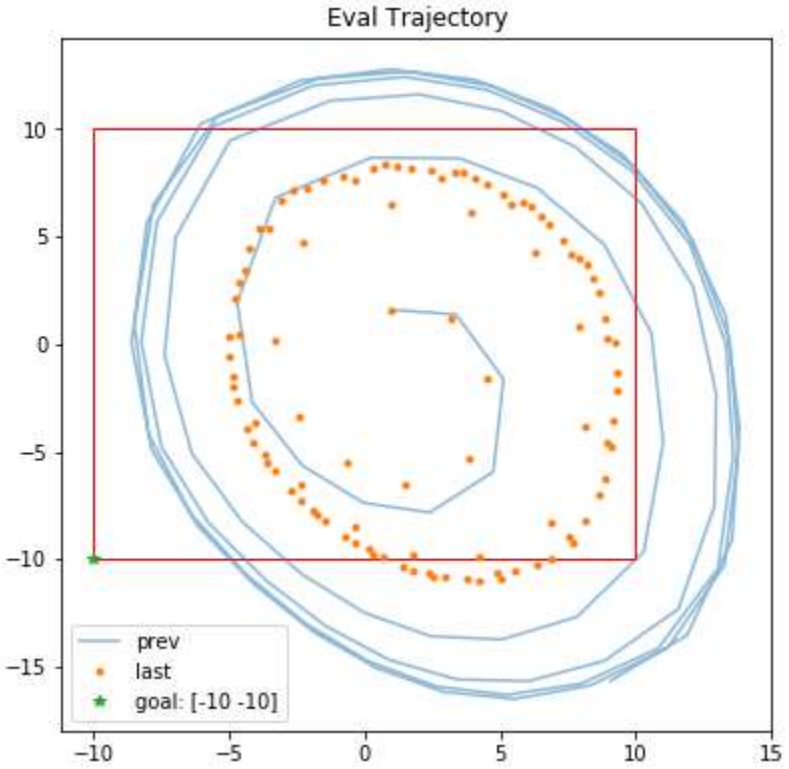


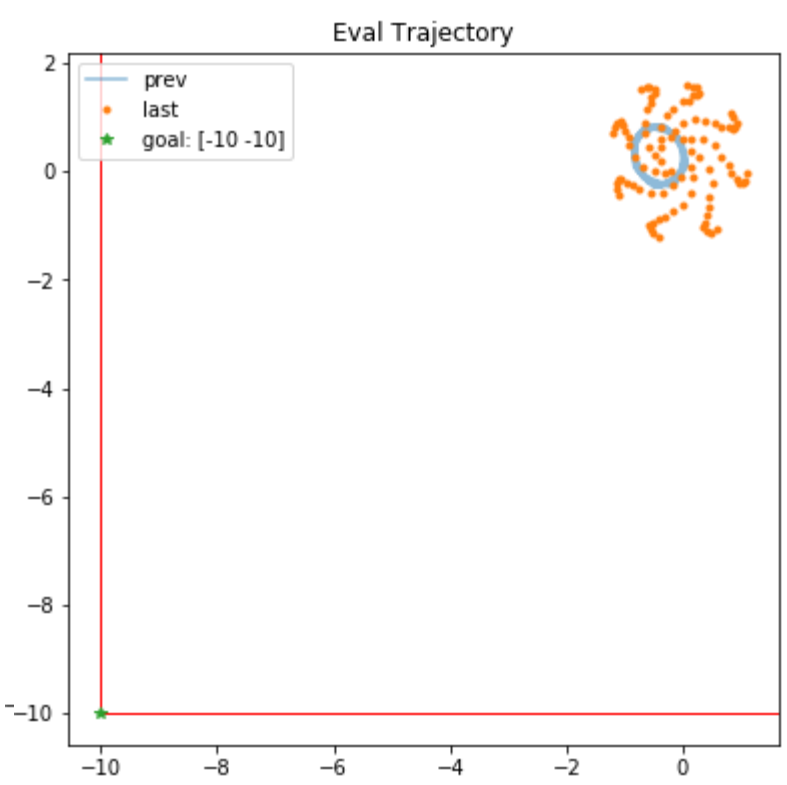
Сравнение с бейзлайнами (фиксированные learning rate)

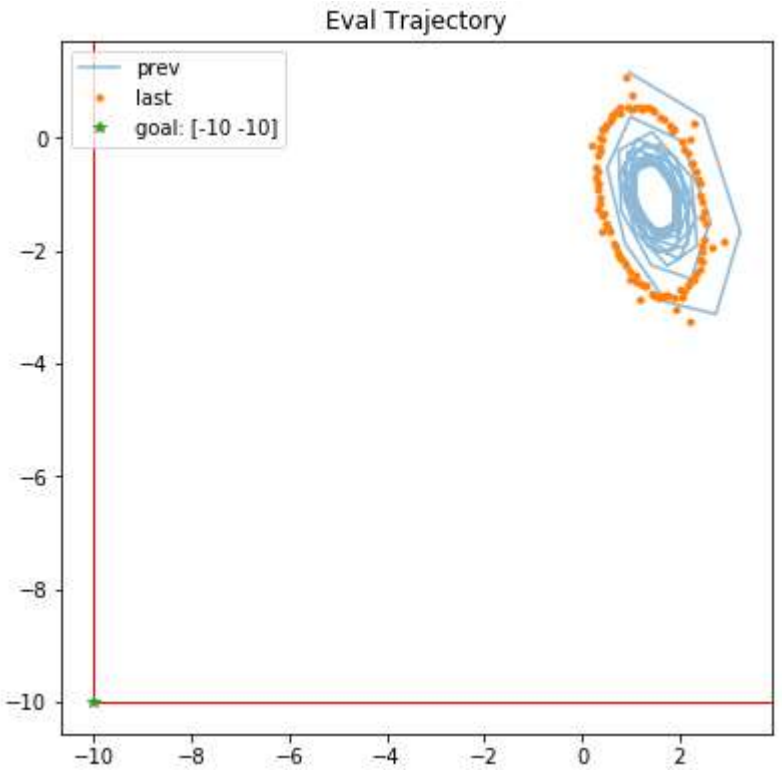
η	accuracy, %	sum loss	final loss
0.0001	41	-44	2.21
0.001	93	-12	0.36
0.01	78	-18	0.92
агент	95	-9	0.14

Траектории во время обучения

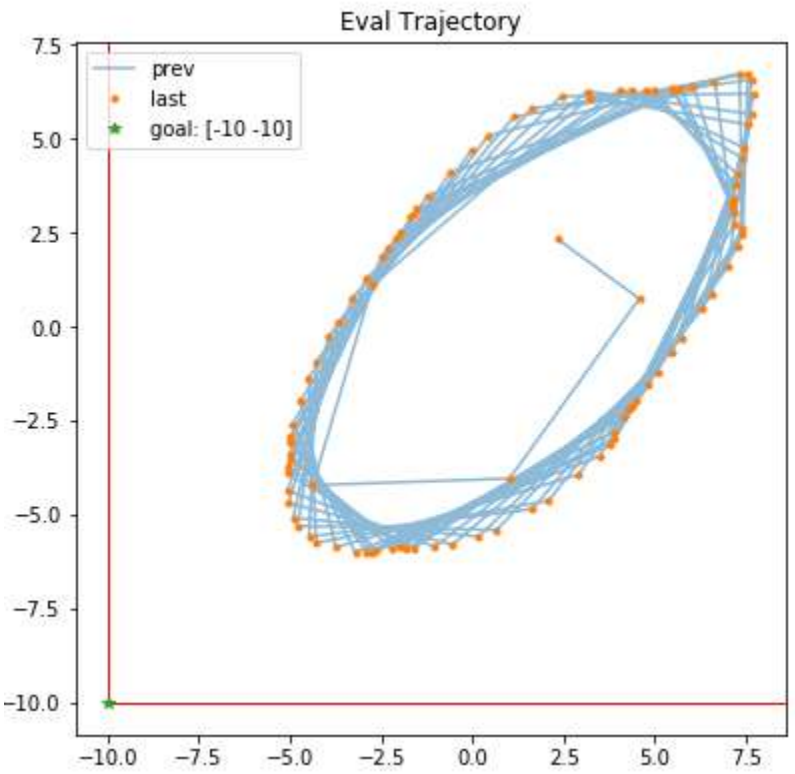


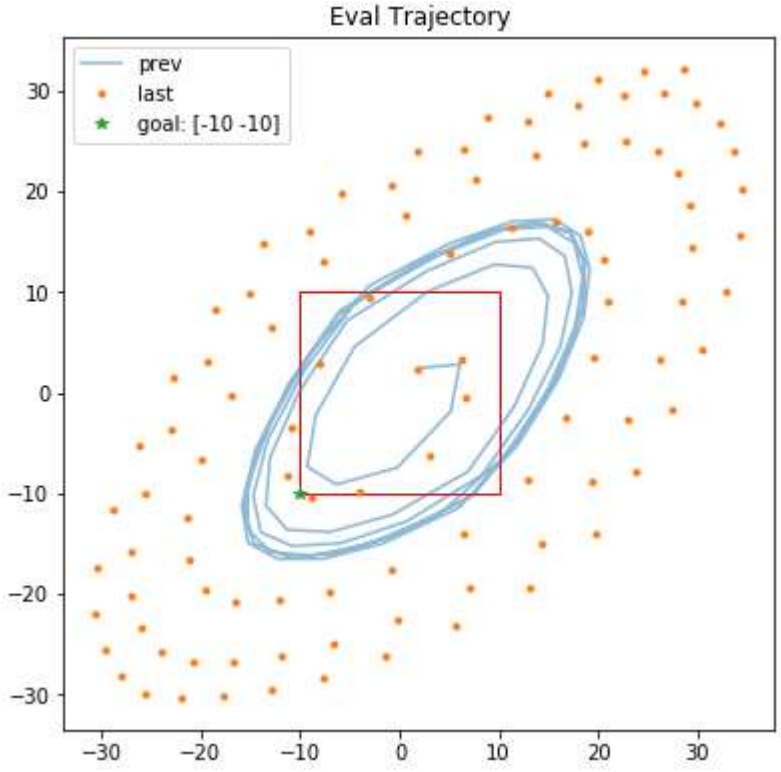


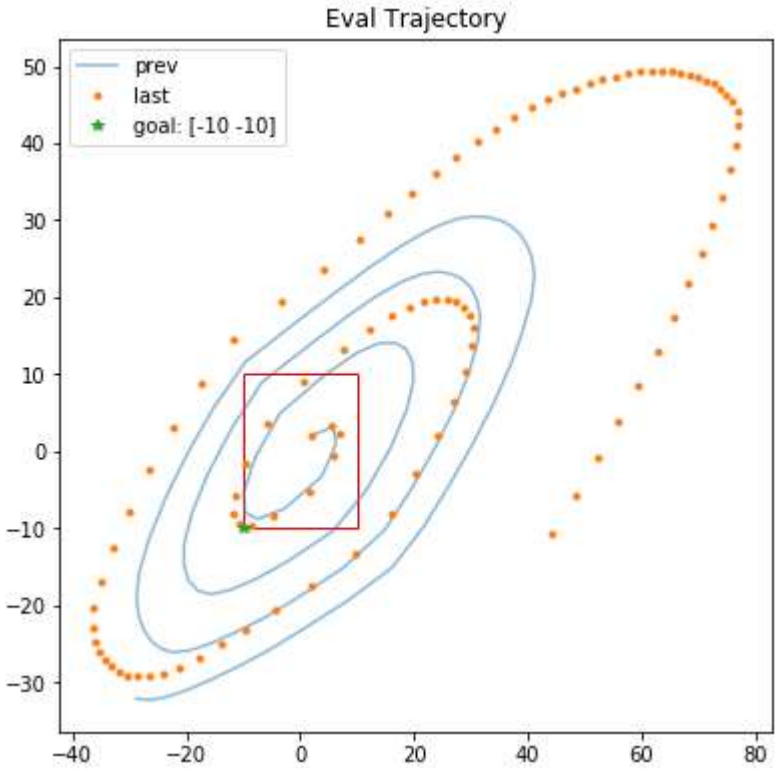

Eval Trajectory
prev
last
goal: [-10 -10]
2
0
−2
−4
−6
−8
−10
−10
−8
−6
−4
−2
0

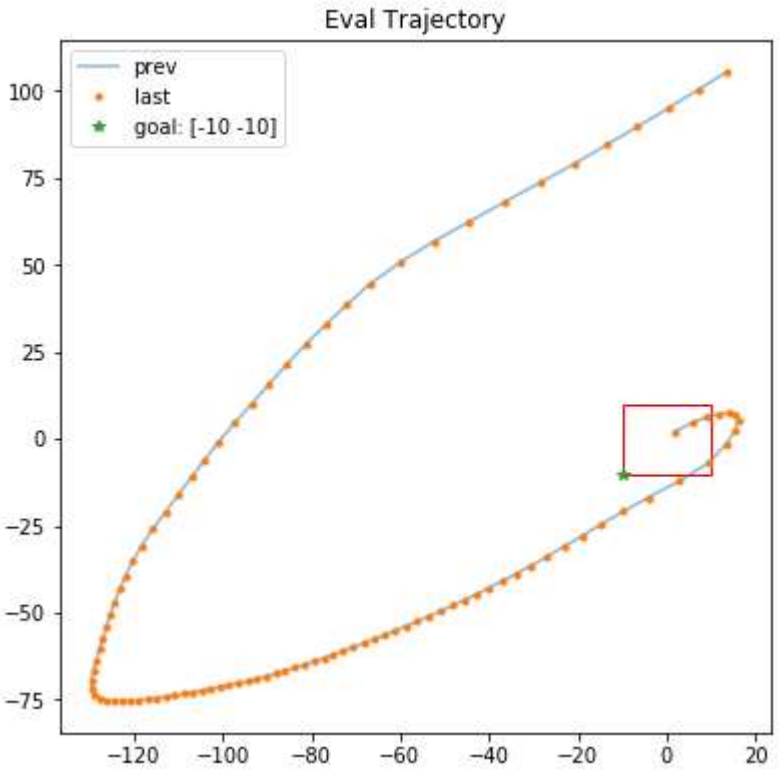

Eval Trajectory
prev
last
goal: [-10 -10]
0
−2
−4
−6
−8
−10
−10
−8
−6
−4
−2
0
2

Зачем останавливаться на достигнутом?






Eval Trajectory
prev
last
goal: [-10 -10]
50
40
30
20
10
0
−10
−20
−30
−40
−20
0
20
40
60
80


Eval Trajectory
prev
last
goal: [-10 -10]
100
75
50
25
0
−25
−50
−75
−120
−100
−80
−60
−40
−20
0
20

Ссылки

- Berkeley Deep RL. Лекция 19 и 20.
- CAD2RL
- Using Simulation and Domain Adaptation to Improve Efficiency of Deep Robotic Grasping
- PHYRE – RL среда с физическими головоломками