Архитектура сети

Вадим Кузьмин

18 июля 2018 г.

1 Описание модели

На вход модели подается изображение, описывающее состояние среды в текущий момент, и one-hot вектор длины *количество опций плюс один*, показывающий то, действие или какая опция выполнялась на предыдщуем шаге.

Изображение используется сверточными слоями для получения признакового описания состояния среды, которое далее используется как вход для других частей модели. В модели для реализации иерархии вводятся подзадачи. Каждая подзадача, кроме нулевой, определяет опцию. Нулевая задача пытается решить исходную проблему не зная об опциях. Для выбора задач используется менеджер, который, принимая на вход признаковое описание состояния среды, играет роль Q-функции над задачами, формируя вектор, орпеделяющий подзадачу.

По вектору предыдущего действия решается, передать управление менеджеру, так как до этого не выполнялась опция, или по тому, какая была опция, проверить, закончилась ли она. Если она закончилась, то управление передается менеджеру и формируется вектор, орпеделяющий подзадачу. Если не закончилась, то опция продолжается и вектор, определяющитй подзадачу, не меняется. Чтобы проверить, закончилась ли опция, используется отдельный классификтаор, который представляет собой стандартный перцептрон.

По вектору, определяющему подзадачу, выбирается выход нужной сети, соответствущей текущей подзадаче. Так формируется выход модели - вектор длины *кол-во базовых действий*, который представляет собой набор значний Q-функции для данного состояния и подзадачи.

Рассмотрим более подробно граф, визуализированный в TensorBoard. Предположим, что используется две опции.

Входные данные:

- input_image - RBG изображение состояния среды;
- prev_action - one-hot вектор длины *число опций плюс один*, показывает, какая задача выполнялась на предыдущем шаге (базовое действие или опция).

Обучаемые модули:

- convolution - сверточные слои, на входе - input_image, на выходе - вектор признаков;
- opt1_checker, opt2_checker - классификаторы для соответствующих опций, на входе классификатора - результат convolution, на выходе - 0/1 (данная опция закончилась или нет);
- task0, task1, task2 - Q-функции соответствующих задач, для каждой на входе - результат convolution, на выходе - вектор длины *число базовых действий*;
- manager - Q-функция над задачами, на выходе - результат convolution, на выходе - вектор длины *число опций плюс один*, определяющий задачу.

Программные модули:

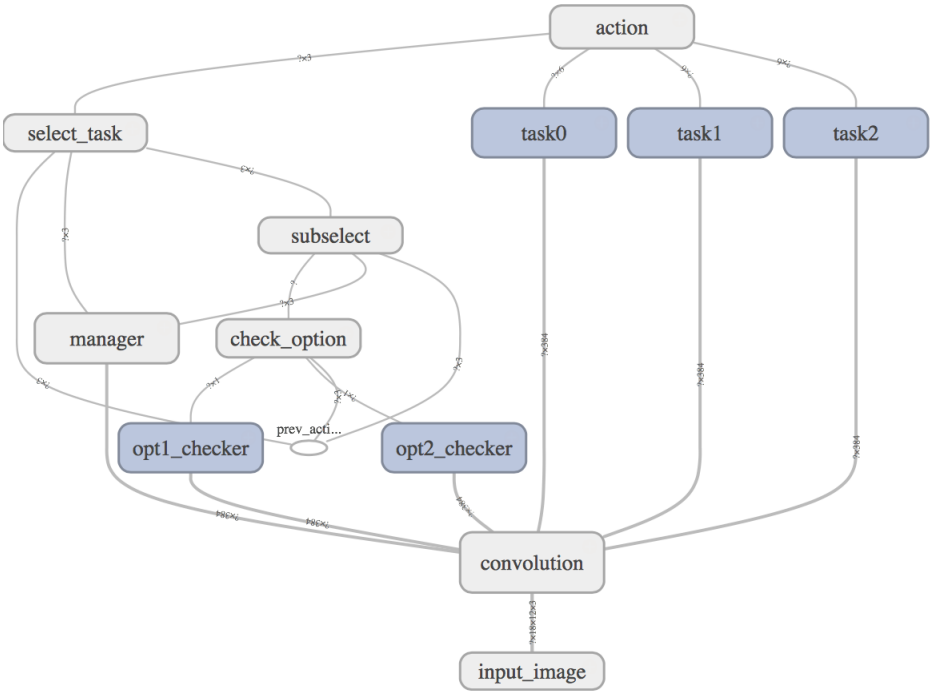

Рис. 1: Модель с двумя опциями

- action - по one-hot вектору, определяющему текущую задачу, выбирает нужный выход из task0, task1, task2;
- get_task - представляет собой операцию boolean mask, где маской как раз является результат работы select_task, приведенный у типу boolean. Данная операция оставляет выход только той сети, задача которой была выбрана;
- select_task - если по prev_action на предыдущем шаге было выполнено базовое действие, то выбирается выход manager, если выполнялась опция - результат работы subselect;
- subselect - выполняется, если на предыдущем шаге выполнялась опция, осуществляется проверка с помощью opt1_checker, opt2_checker, закончилась ли опция, если да, то выбирается выход manager, если не закончилась - выбирается prev_action, так как задача не меняется.

Выходные данные:

- результат выполенения action - вектор длины *число базовых действий*, из которого выбирается действие для агента;
- результат выполенения select_task - будет использован как prev_action.

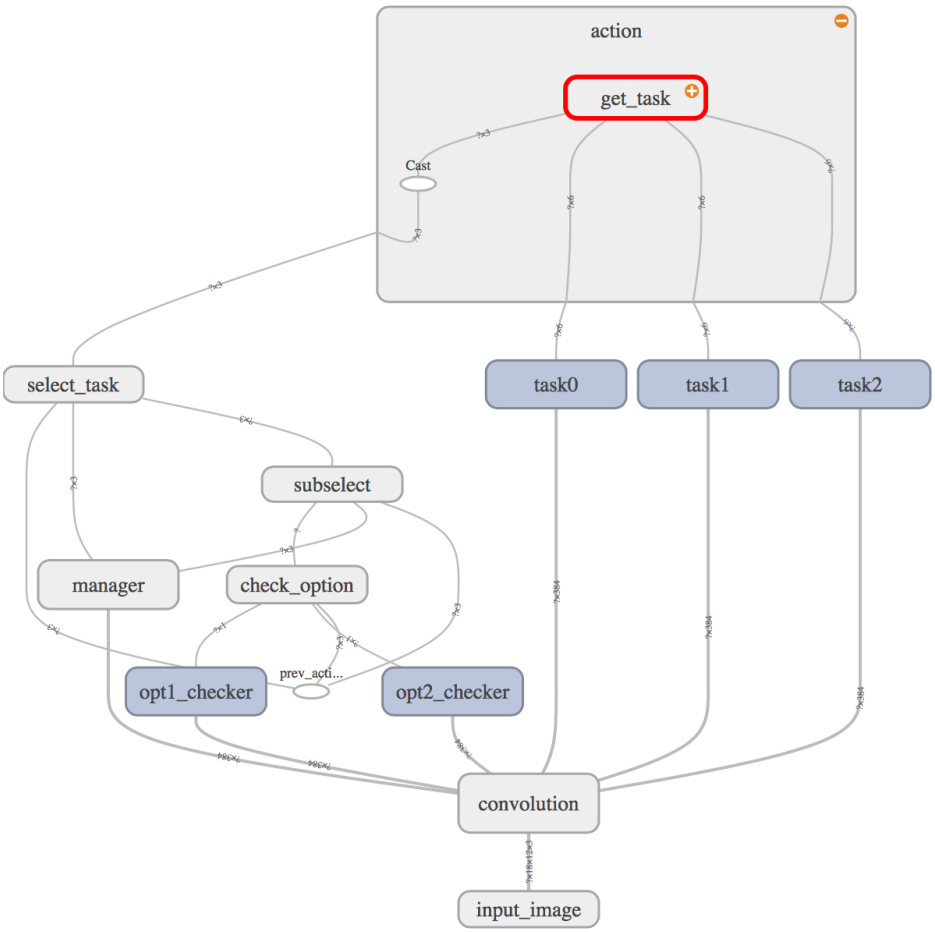

Рис. 2: Модель с двумя опциями, операция action