BlackBox Challenge: Эволюционный Подход к Обучению Интеллектуального Агента

Victor Shevchenko VICTOR@NEXTMIND.ORG

Аннотация

BlackBox Challenge – это соревнование по созданию агента, действующего в игре с неизвестными правилами. Эта статья рассматривает различные методы, в особенности, эволюционный подход, к обучению агента принятию решений в условиях недетерминированной среды с неизвестными правилами. Рассматривается применение Эволюционной Стратегии Адаптации Ковариационной Матрицы (CMA-ES) и ее модификаций (LMCMA, VDCMA, ElitistCMA) к обучению нейронных сетей, а также скорость сходимости эволюционных методов, выбор целевой функции, подходы к увеличению обобщающей способности и скорости обучения многослойных нейронных сетей.

1. Введение

BlackBox Challenge[1] – это соревнование по машинному обучению. Задача участников заключается в написании интеллектуального агента, набирающего максимальное число очков в недетерминированной игре с неизвестными правилами. Даны обучающий и тестовый уровни, состоящие из 1,258,935 и 1,214,494 шагов соответственно. На каждом шаге доступны 36 переменных среды и 4 возможных действия. Валидационная и финальная выборки не доступны участникам, проверка результата на валидационной выборке проходит на серверах организаторов состязания. Результат на финальной выборке определяет конечных победителей соревнования.

2. Исследование среды

Исходя из данных о переменных среды были построены графики корреляции исследованы значения score-функции по действиям 1 и 2. Детальное рассмотрение среды позволило выявить ряд закономерностей:

- Уровень состоит из 10 подуровней
- Подуровни длятся около 150,000 шагов
- Начало подуровня определяется по обнулению 36-ой переменной среды
- Вознаграждение по модулю зависит от 36-ой переменной
- Значения 36-ой переменной изменяются в промежутке [-1.1;1.1]

1. http://blackboxchallenge.com

- Переменные 1-35 не зависят от действий агента
- Действия 1 и 2 противоположны
- Многократное выполнение 1-го действия увеличивает 36-ую переменную на 0.1
- Многократное выполнение 2-го действие уменьшает 36-ую переменную на 0.1
- Действие 3 эквивалентно бездействию
- Действие 0 изменяет значение 36-ой переменной в ту или иную сторону
- Награда начисляется с некоторой задержкой
- Предусмотрен штраф за бездействие

Ниже приведены график score-функции при выполнении действий 1 и 2, а также график корреляции переменных состояния.

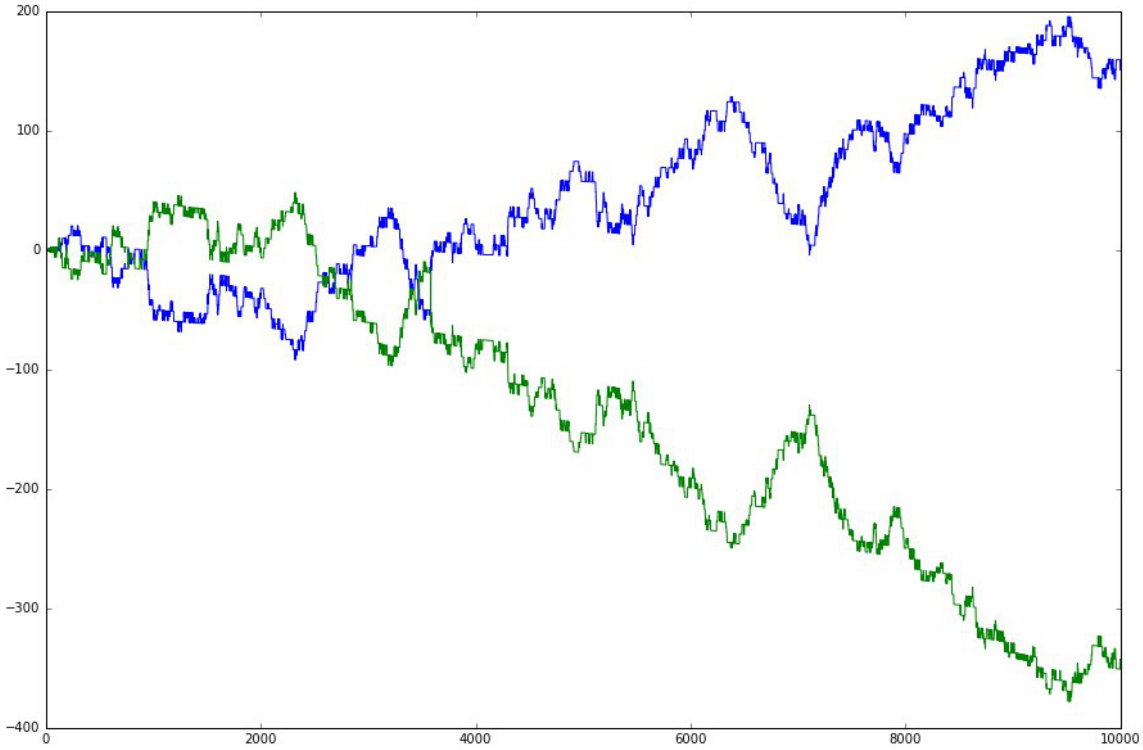

Рис. 1: Score-функция по действиям 1 и 2.

Рассмотрим график, приведенный на рис. 1. Заметно, что значения score-функции по действию 2 убывает быстрее роста значений score-функции по действию 1. Это указывает на наличие некоторой «Комиссии», что приводит к игре с отрицательным математическим ожиданием.

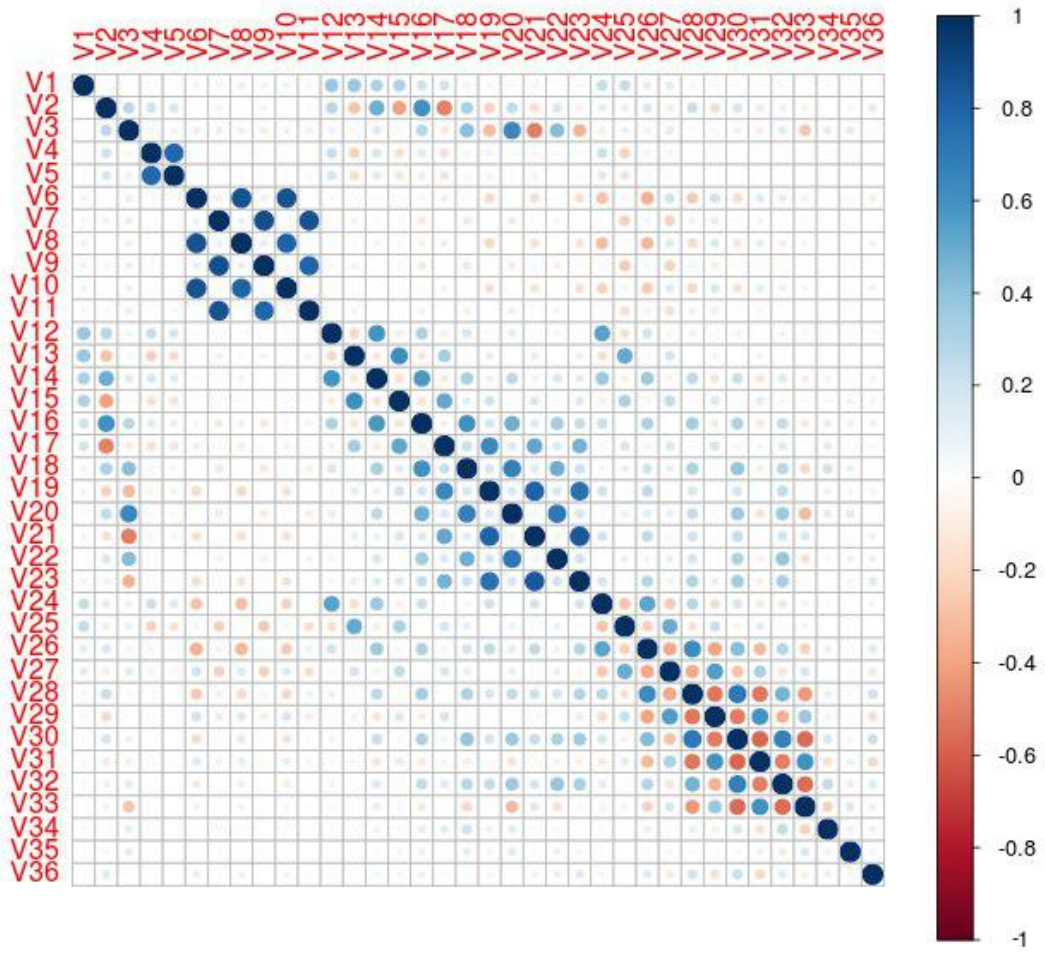

Рис. 2: Корреляция переменных состояния.

3. Модели и методы обучения

3.1 Случайный агент

Для определения сходимости моделей был написан агент, выполняющий случайные действия. Среднее значение, полученное по результатам 20 запусков на тестовой выборке, составило -11893 очков.

3.2 Базовый агент

Организаторами был написан агент, использующий линейную регрессию, демонстрирующий результат в 2090 очков на валидационной выборке. При обучении, вероятно, использовался один из методов Policy Gradient.

3.3 Reinforce

Для обучения однослойной нейросети был протестирован метод Reinforce (Williams (1992)). Метод показал умеренную сходимость, при числе шагов равному 10,000. С ростом количества шагов наблюдалось значительное ухудшение сходимости. Для выборок более 100,000 шагов сходимости достичь не удалось.

3.4 Decision Tree

Для построения дерева решений был использован алгоритм C4.5 (Quinlan (1993)). С помощью механизма чекпоинтов найдены «оптимальные» действия, обеспечивающие максимальное число очков. Таким образом задача была сведена к supervised learning. Метод показал хорошую сходимость на обучающей выборке, с лучшим результатом в 16,324 очков. Тестирование модели показало крайне неудовлетворительную обобщающую способность дерева решений, с результатом в -9864 очков на тестовой выборке.

3.5 Backpropagation

На основе данных об «оптимальных» действиях, полученных на основе чекпоинтов, для обучения многослойной нейронной сети был протестирован классический метод обратного распространения ошибки (D. E. Rumelhart and Williams (1986)) с использованием dropout (G. E. Hinton (2014)). Нейронная сеть не достигла сходимости.

3.6 Random Search

Для обучения однослойной нейронной сети был протестирован метод случайного поиска. Данный подход к обучению предполагает случайное изменение весов сети с целью максимизировать количество очков на обучающей выборке. Метод показал медленную, но уверенную сходимость на начальных этапах обучения. С ростом точности, из-за большого количества локальных максимумов, наблюдалась значительное снижение скорости сходимости метода, вплоть до полного прекращения.

3.7 Backprop. Search

Был разработан простой метод, обеспечивающий приемлемую скорость сходимости и обобщающую способность для многослойной нейронной сети. Суть метода состоит в чередовании шагов прюнинга и случайной мутации весов. Шаги прюнинга предполагают отсев итераций обратного распространения, не приводящих к росту точности на обучающей выборке. Шаги случайной мутации предполагают случайное изменение весов с сохранением изменений, приводящих к росту точности. Если при выполнении 50 итераций ни один из шагов не привел к росту точности, использовалась случайная мутация, что улучшало способности сети к преодолению локальных максимумов. Метод показал быструю по сравнению с Random Search сходимостью, с лучшим результатом в 2826 очков на обучающей и 2470 на тестовой выборках, после чего модель сталкивалась со стойким локальным максимумом.

3.8 Метод Нелдера-Мида

Для максимизации результата на обучающей выборке и поиска оптимального вектора весов нейронной сети был протестирован метод Нелдера-Мида (Nelder and Mead (1965)). В данной задаче, по скорости сходимости, метод уступает Backprop Search, плохо преодолевает множественные локальные максимумы. Увеличение размерности нейронной сети крайне негативно влияет на скорость сходимости.

4. Эволюционные методы

Наибольшую эффективность в данной задаче продемонстрировали эволюционные алгоритмы, в частности CMA-ES (эволюционная стратегия адаптации ковариационной матрицы) (N. Hansen (2004)) и модификации: ElitistCMA (C. Igel and Hansen (2006)), LMCMA (Loshchilov (2015)), VDCMA (Akimoto and Hansen (2014)) в реализации библиотеки Shark (C. Igel (2008)) , что потребовало реализацию интерфейса взаимодействия C++ и Python кода. Применение эволюционного алгоритма создало необходимость в быстром вычислении целевой функции, для чего была написана Cython-реализация нейронной сети CyMLP[2]. После оптимизации, время, затрачиваемое на одну итерацию (полный проход агента по уровню), для однослойной нейронной сети в 148 связей, составило 0.443667 секунд. Вычисления проводились на виртуальном сервере Google Compute Engine с Intel Xeon E5-2670v3.

4.1 Целевая функция

Переобучение является ключевой проблемой в данной задаче. Был проанализирован ряд переобученных моделей, рассмотрена их стратегия взаимодействия со средой. Основные черты переобученных агентов:

- Агрессивная стратегия
- Резкий рост score-функции на определенных этапах игры, подуровнях
- Большие значения весовых коэффициентов

Исходя из этого была построена целевая функция:

$$\text{Maximize } \min(sublevel\_score(w)) - \eta \left\|w\right\|_1 \tag{1}$$

Где w – вектор весов нейронной сети агента, sublevel_score – количество очков, набранных агентом на подуровне, η - коэффициент, регулирующий размер штрафа, и L1-норма $\|w\|_1$, где $\|w\|_1 = \sum_i |w_i|$. Иными словами максимизируется результат на подуровне с минимальным числом набранных очков, со штрафом за рост абсолютной величины весовых коэффициентов. Это уменьшает агрессивность стратегий и не допускает безконтрольного роста значений весовых коэффициентов, что повышает обобщающую способность нейронной сети.

2. Fast Cython/C implementation: https://github.com/gncgroup/CyMLP

4.2 Сравнение CMA-методов

Исследование скорости сходимости алгоритма CMA-ES и его модификаций, проводилось на примере однослойной нейронной сети, инициализированной идентичными весовыми коэффициентами.

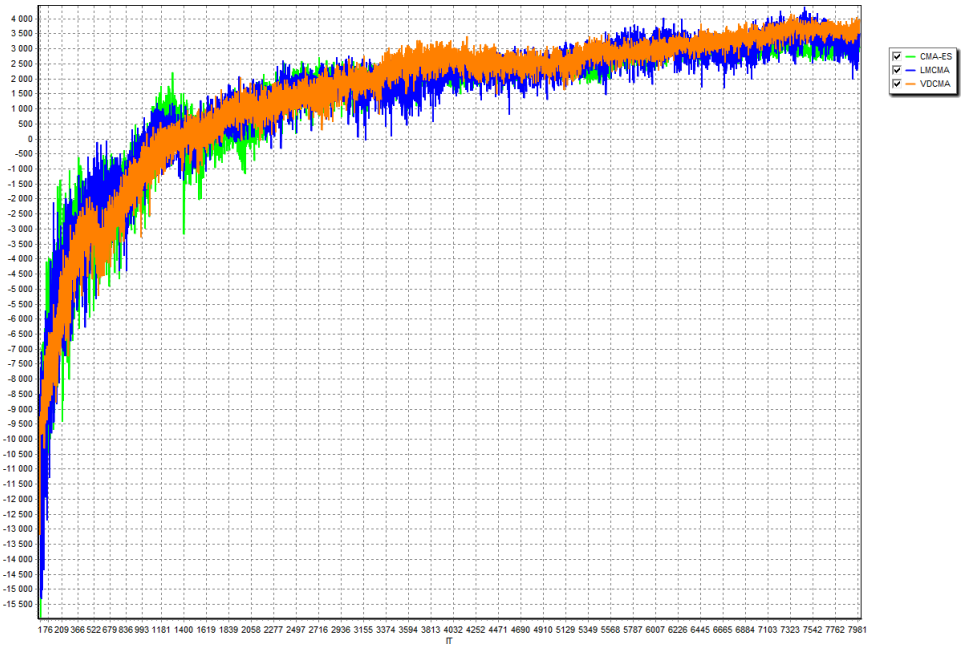

Рис. 3: Сравнение сходимости CMA-ES, LMCMA и VDCMA.

4.3 Обучение глубоких сетей. Проекционная стратегия оптимизации

Алгоритмы адаптации ковариационной матрицы демонстрируют уверенную сходимость при обучении сетей большой размерости, однако увеличение размерности нейронной сети неизбежно сказывается на скорости итерации. Метод проецирования в сеть большей размерности позволяет повысить скорость на начальных этапах обучения.

Алгоритм 1: **Projection optimization strategy**
1: given: nn_size, nsteps
2: initialize: nn(nn_size[1])
3: for s ← 1, 2, . . . , nsteps do
4: if s > 1 then
5: initialize new_nn(nn_size[s])
6: projection(new_nn,nn)
7: nn ← new_nn
8: repeat
9: CMA-ES optimization step
10: until stopping criterion is met

Даны nsteps – число шагов проецирования, nn_size – массив размерности сети (число нейронов на каждом слое) для каждого шага. На первом шаге инициализируется однослойная нейронная сеть, выполняется оптимизация весов пока не будет достигнут критерий остановки, на последующем шаге, сеть меньшей размерности проецируется в большую сеть, выполняется оптимизация весов, по достижению критерия остановки, выполняется следующая итерация, цикл повторяется до достижения nsteps. Проецирование сети представляет собой обучение классическим обратным распространением. Сеть большей размерности обучается отображению входного вектора в значения выходных нейронов проецируемой сети. Критерием остановки служит прекращение улучшения результатов на тестовой выборке за последние n итераций эволюционного алгоритма.

5. Конечное решение

Методами CMA-ES, LMCMA, VDCMA было обучено множество нейронных сетей. Сети, набирающие наибольшее число очков на тестовой выборке, проецировались в большую размерность, где происходило дообучение, процесс повторялся пока наблюдался рост обобщающей способности, которая определялась по результатам на тестовой и валидационной выборках. Далее, по результатам на тестовой и валидационной выборках, отобраны 4 сети размерностями 36-64-4 и 16-4, демонстрирующие лучшую обобщающую способность. На основе критерия уверенности (разности значений двух наиболее активных выходных нейронов) был построен ансамбль нейронных сетей[3]. Активность переобученных выходных нейронов сетей ансамбля блокировалась, что способствовало получению дополнительного прироста точности.

6. Дальнейшее развитие

В условиях множественного количества целей методы одноцелевой оптимизации, включая CMA-ES и его модификации, не являются оптимальным решением. Максимизация минимальной ошибки увеличивает обобщающую способность сети, однако на поздних этапах обучения это не помогает преодолеть запоминание score-функции. Одним из способов получения сети с максимальной обобщеющей способностью, является поиск Парето-эффективных решений. Суть Парето-эффективности заключается в поиске решений, приводящих к улучшению показателей по всем или нескольким целям, без ухудшения других. Целевая функция для многокритериальной оптимизации примет следующий вид:

$$\text{Maximize} \begin{cases} sublevels\_scores(w) \\ -\|w\|_1 \end{cases} \tag{2}$$

Целевая функция состоит в максимизации количества очков на всех подуровнях и минимизации вектора весов. Для поиска Парето-оптимального решения предлагается использование эволюционного метода многокритериальной оптимизации MOCMA (C. Igel (2007)).

3. Final solution: https://github.com/gncgroup/BlackBox-Solver

7. Заключение

Эволюционные алгоритмы адаптации ковариационной матрицы показывают уверенную сходимость в задачах поиска оптимальных весовых коефициентов нейронных сетей. Особенно полезным эволюционные методы оказываются в случае, когда вычисление производных целевой функции затруднительно или невозможно. Проекционная стратегия оптимизации позволяет ускорить обучение глубоких нейронных сетей. Правильный выбор целевой функции является ключевым фактором. L1 регуляризация позволяет избежать роста весовых коефициентов и увеличить обобщающую способность нейронной сети. Дальнейшим направлением развития является применение методов многокритериальной оптимизации, в частности, MOCMA, позволяющих найти Парето-оптимальное решение, способного обеспечить лучшую обобщающую способность.

Благодарности

Выражается благодарность участникам форума и telegram-канала за идеи, наблюдения, а также предоставленные графики (см. рис. 1 и 2). Отдельная признательность кураторам и организаторам состязания.

Список литературы

A. Auger Akimoto, Y. and N. Hansen. Comparison-based natural gradient optimization in high dimension. *GECCO*, 2014.

N. Hansen C. Igel, T. Suttorp. Steady-state selection and efficient covariance matrix update in the multi-objective cma-es. *Evolutionary Multi-Criterion Optimization*, pages 171–185, 2007.

T. Glasmachers C. Igel, V. Heidrich-Meinser. Shark. *JMLR*, (8):993–996, 2008.

T. Suttorp C. Igel and N. Hansen. Computational efficient covariance matrix update and a (1+1)-cma for evolution strategies. *GECCO*, 2006.

G. E. Hinton D. E. Rumelhart and R. J. Williams. Learning representations by back-propagating errors. *Nature*, (323):533–536, 1986.

A. Krizhevsky I. Sutskever R. R. Salakhutdinov G. E. Hinton, N. Srivastava. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*, (15):1929–1958, 2014.

I. Loshchilov. Lm-cma: an alternative to l-bfgs for large scale black-box optimization. *Evolutionary Computation*, 2015.

S. Kern N. Hansen. Evaluating the cma evolution strategy on multimodal test functions. *LNCS*, pages 282–291, 2004.

J. A. Nelder and R. Mead. A simplex method for function minimization. *Computer Journal 7*, (7):308–313, 1965.

J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufman Publishers, 1993.

R. J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, (8):229–256, 1992.