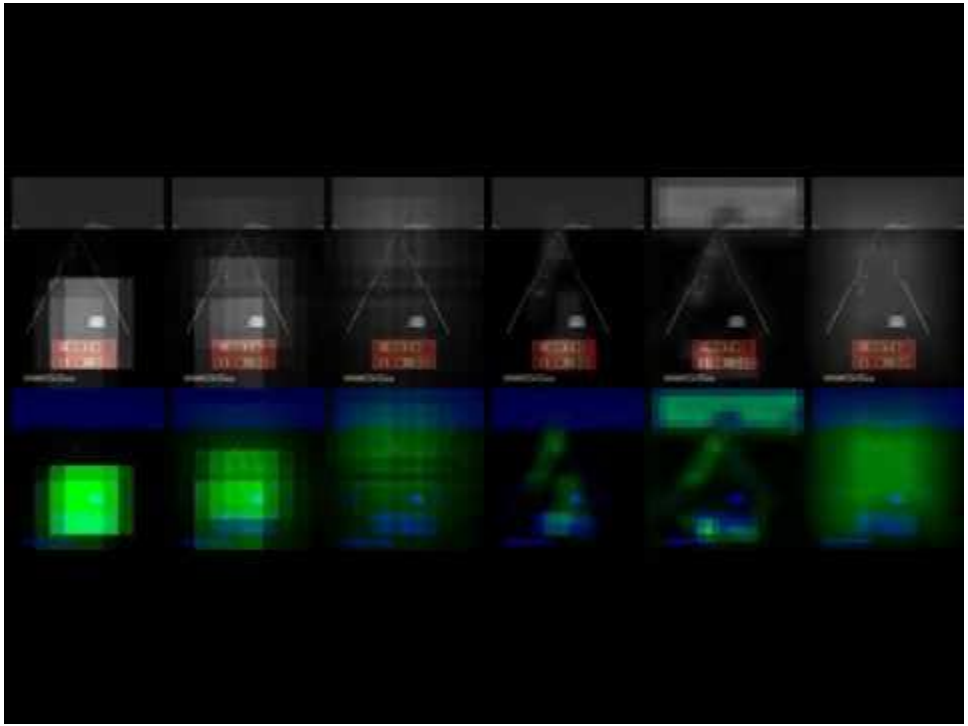
Experiment management in research: case study

Дмитрий Никулин
Samsung AI Center Moscow

Ситуация

- Проект в reinforcement learning
- Гоняем эксперименты на Atari
- Среда является датасетом ⇒ данные версионировать не нужно
- Очень много довольно похожих экспериментов
- Каждый эксперимент от 1 до 6 суток
- Код пишет и запускает один человек

Результат



https://arxiv.org/abs/1908.02511, under review at ICCV 2019 VXAI workshop

Эксперименты

Game	BeamRider	Breakout	Enduro	MsPacman	Seaquest	SpaceInvaders
Nature CNN [24]	6949±2569	618±209	3832±1661*	4874±1701	1920±37	3867±3627
DAQN [36]	701±205	601±201	2191±1088	3111±1165	1453±420	2096±1554
RS-PPO [42]	583±185	605±202	3813±1662*	3943±1435	1670±372	2562±1339
RS-PPO [42] w/o padding	823±432	591±199	3606±1669*	3950±1371	1710±379	2248±782
Sparse FLS	6634±2361	624±211	5094±1876	5421±1517	2440±382	9359±13230
Sparse FLS + sum-pooling	3356±1878	520±183	1918±1485	4317±1485	1150±385	1847±773
Sparse FLS + norm	6584±2159	598±200	4614±1821*	3409±1275	1161±348	11206±10441
Sparse FLS w/ 1×1 convs	6870±2413	621±207	4721±1924*	4887±1589	2252±336	5673±6344
Sparse FLS w/ $\mathrm{SoftPlus}_2$	6697±2261	612±208	4808±1851*	5242±1527	1908±29	6443±7684
Sparse FLS w/o final ReLU	6777±2242	589±207	4767±1947*	5049±1145	2013±185	2929±556
Sparse FLS w/o final ReLU + sum-pooling	1057±1061	480±158	2156±1529*	3365±1292	2356±1874	1999±899
Sparse + FLS after first conv layer	7468±2645	640±212	4976±2022*	4720±1455	2181±376	9285±13326
Sparse + FLS after each conv layer	6588±2348	633±217	5956±3735*	4978±1461	2083±314	2855±194
Dense FLS + sum-pooling	866±415	532±173	2113±2161	4977±1253	1368±517	1549±831
Dense FLS w/o final ReLU + sum-pooling	730±245	503±162	1335±1883*	4879±1282	10052±11853	1316±813

Table 3: Evaluation scores. We trained 5 models with different random seeds for each (game, architecture) pair. Each model was evaluated 8192 times on environments initialized with a previously unseen random seed (except for *Enduro* runs marked with an asterisk; they are evaluated 256 times), with results aggregated across models. Colors correspond to Fig. 3.

```
$ ls ~/data/rl-saliency | wc -l
623
```

Ситуация (2)

- У компании нет готового инструмента для управления экспериментами и обработки их результатов
- Зато есть кластерная инфраструктура


aist

Требования

- Максимальная воспроизводимость (для публикации)
- Возможность посмотреть все параметры любого эксперимента
- Машиночитаемые результаты экспериментов
- Автоматизация рутины, защита от дурака

Воспроизводимость

Воспроизводимость: очевидное

- Код хранится в системе контроля версий (Git, SVN, Perforce, …)
- Код запускается в контролируемой среде (Docker, Singularity, Nix, …)

Воспроизводимость: Dockerfile в репозитории

```
$ cat Dockerfile | pygmentize -l docker
FROM nvidia/cuda:9.0-cudnn7-devel-ubuntu16.04
LABEL maintainer="Dmitry Nikulin <d.nikulin@samsung.com>"

# Connect deadsnakes PPA to enable installation of a recent Python version.
# Installed tools:
# - System
#     - curl: for installation of pip and fixuid
#     - git: for installation of Python packages from Github
#     - unzip: for unpacking source code within container
#     - time: for measuring resource usage during job execution
# - Python 3.6. Support for 3.7 was introduced in Tensorflow 1.13.1.
# - opencv-python dependencies (libglib2.0-0, libsm6, libxrender-dev)
# - ffmpeg as a moviepy dependency
# - fish for reducing pain in interactive session and man-db b/c fish uses apropos
# - pip

RUN echo 'deb http://ppa.launchpad.net/deadsnakes/ppa/ubuntu xenial main' >> /etc/apt/sources.list && \
    echo 'deb-src http://ppa.launchpad.net/deadsnakes/ppa/ubuntu xenial main' >> /etc/apt/sources.list && \
    apt-key adv --keyserver keyserver.ubuntu.com --recv-keys F23C5A6CF475977595C89F51BA6932366A755776 && \
    apt-get update && \
    apt-get upgrade -y && \
    apt-get install -y \
        curl git unzip time \
    # ...
```

Воспроизводимость: конфигурация в репозитории

```
$ git branch
  experiments
* master
$ cat config.json | pygmentize -l json
{
  "model": {
    "network": "cnn_sparse_fls",
    "frame_stack_size": 4
  },

  "train": {
    "env_name": "MsPacmanNoFrameskip-v4",
    "alg_name": "ppo2",
    "train_seed": 33,
    "num_processes": 8,
    "num_timesteps": 50000000,
    "num_steps": 128,
    "num_videos": 21
  },

  "eval": {
    "eval_num_envs": 16,
    "evals_per_env": 512,
    "eval_seed": 1001,
    "eval_max_eplen": 108000
  }
}
```

Воспроизводимость: коммиты для экспериментов

- Отдельная ветка `experiments` с изменениями конфига
- `git merge` из `master` в `experiments`, но не наоборот

```
$ git checkout experiments
Switched to branch 'experiments'
Your branch is up to date with 'origin/experiments'.
$ git log --oneline --decorate --graph -n 30
*   5b19ff5 (HEAD -> experiments, origin/experiments) Merge branch 'master' into experiments
|\
| * 2a0bd41 (origin/master, master) Add script for computing number of model params
| * a99f84c Set default eplen to 108k, as suggested in LTIAA
| * 99b2ee8 Support eval on CPU
| * 7514151 Save done_per_env in eval results
| * 333f264 Replace episode length limit of 10000 with 50000 everywhere
| * 726b52b Use VideoWriter based on OpenCV instead of Baselines
| * 50221fd Re-add cnn_attn_ltiaa_fixed network
| * 433b90e Use 50000 as default limit on eval episode length
* | 045d9bf Set train_seed = 33
* | b8accfe Set train_seed = 25
* | 99db242 Set train_seed = 17
* | 1e0e719 Set train_seed = 9
* | 07a0a8c Set train_seed = 1
* | e327c4e Set env=Seaquest
* | 027735c Set train_seed = 33
```

Воспроизводимость: обучение без CLI-параметров

```
$ python submit.py
```

Защита от дурака: незакоммиченный код

```
$ git status
On branch experiments
Your branch is up to date with 'origin/experiments'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git checkout -- <file>..." to discard changes in working directory)

        modified:   config.json

no changes added to commit (use "git add" and/or "git commit -a")
$ python submit.py
[2019-08-07 16:32:00,679] DEBUG: Running on laptop
Traceback (most recent call last):
  File "submit.py", line 330, in <module>
    main()
  File "submit.py", line 309, in main
    host_whitelist=args.host_whitelist,
  File "submit.py", line 194, in submit_laptop
    raise RuntimeError("Repo is dirty")
RuntimeError: Repo is dirty
```

Защита от дурака: незакоммиченный код

```
def has_uncommitted_changes(path: Path = None):
    command = ['git', 'diff-index', '--quiet', 'HEAD', '--']
    return bool(subprocess.call(
        command,
        cwd=str(path) if path is not None else path
    ))

# ...

if git.has_uncommitted_changes():
    raise RuntimeError("Repo is dirty")
```

Защита от дурака: не та ветка

```
def current_branch(path: Path = None):
    command = ['git', 'rev-parse', '--abbrev-ref', 'HEAD']
    return subprocess.check_output(
        command,
        cwd=str(path) if path is not None else path,
        stderr=subprocess.STDOUT,
        encoding='ascii'
    ).rstrip('\n')

# ...

if git.current_branch() != 'experiments':
    raise RuntimeError("Wrong branch")
```

Защита от дурака

- Эксперименты запускаются только из коммита
- Доставка кода на кластер делается через `git pull` ⇒ нет возможности случайно запустить незакоммиченный код
- `submit.py` автоматизирует всё это и ещё много чего

Docker

- Образ на основе `nvidia/cuda:9.0-cudnn7-devel-ubuntu16.04`
- Python 3.6 из deadsnakes PPA (conda слишком медленная)
- `pip` (`pipenv` бесконечно медленный)
- `fixuid` для запуска не от `root`

```
create_command = [
   'nvidia-docker',
   'create',
   '--volume', f'{output_dir!s}:{container_output_dir!s}:rw',
   '-t',  # allocate TTY
   '--rm',  # do not leave containers lying around
   f'--user={os.getuid()}:{os.getgid()}',  # run as current user
   container_tag,
   'sh', '-c', ' && '.join(container_command)
]
```

Docker: пересборка образа

Сборка образа запускается на каждый эксперимент (используется встроенный в Docker кэш):

```
build_command = [
    'nvidia-docker', 'build',
    '-t', container_tag,
    '.'
]

with open('Dockerfile', 'r') as fp:
    subprocess.check_call(build_command, stdin=fp)
```

Docker: запуск коммита в контейнере

- Для обучения запускается код, соответствующий конкретному коммиту

```
$ python submit.py --help
  --commit-hash <commit>
                        Commit hash to use for submission
```

- В контейнер кладётся `git archive` коммита, который распаковывается и запускается уже внутри

```python
def dump_archive(output_path: Path, path=None, ref: str = 'HEAD', format='zip'):
    command = [
        'git', 'archive',
        '--format', format,
        ref,
        '-o', str(output_path)
    ]
    subprocess.check_call(command, cwd=str(path) if path is not None else path)
```

Сохранение результатов: формат Tensorboard

- Встроено в OpenAI Baselines
- Сам Tensorboard бесконечно медленный
- Логи можно прочитать и без него

```
import pandas as pd
from tensorboard.backend.event_processing import event_accumulator

def read_tag(tb_path: Path, tag):
    ea = event_accumulator.EventAccumulator(str(tb_path))
    ea.Reload()
    assert tag in ea.Tags()['scalars']
    return ea.Scalars(tag)

eprewmean = read_tag(path, 'eprewmean')
eprewmean = pd.DataFrame.from_records(
    [pb._asdict() for pb in eprewmean],
    index='step',
)['value']
```

Сохранение результатов: JSON

- Эксперименты однотипные, поэтому схема одинаковая для всех
- Чтение JSON очень быстрое: 450 файлов суммарно на 177 Мб читаются за 1.22 сек
- Человекочитаемый формат, удобно отлаживать

И что теперь?

- Задача решена: эксперименты проведены, статья принята на ревью
- Если бы я начинал сейчас заново, я бы больше использовал готовые инструменты (DVC или MLFlow), но следовал бы тем же принципам:
  - Для каждого эксперимента сохранять как можно больше мета-информации (хэш коммита, параметры запуска, среда запуска)
  - В частности, запускать только закоммиченный код
  - Результаты сохранять в машиночитаемом и (желательно) человекочитаемом формате
  - Автоматизировать как можно больше, в том числе защиту от дурака

И что теперь?

- Неочевидно, как без сохранения конфига в git и запуска без CLI-параметров убедиться, что все важные параметры сохранены
- Неочевидно, как обойтись без парсинга кода, если ваш код внезапно превратился в параметры
- Неочевидно, как компактно записывать архитектуру нейросетей для агрегации результатов в табличку (DSL?)
- Неочевидно, насколько всё это будет работать, если код пишет >1 человека

Questions?