Разработка доверенных версий фреймворков машинного обучения TensorFlow и PyTorch

Андрей Федотов

25 мая 2023

ИСП РАН

Анализ фреймворков ИИ

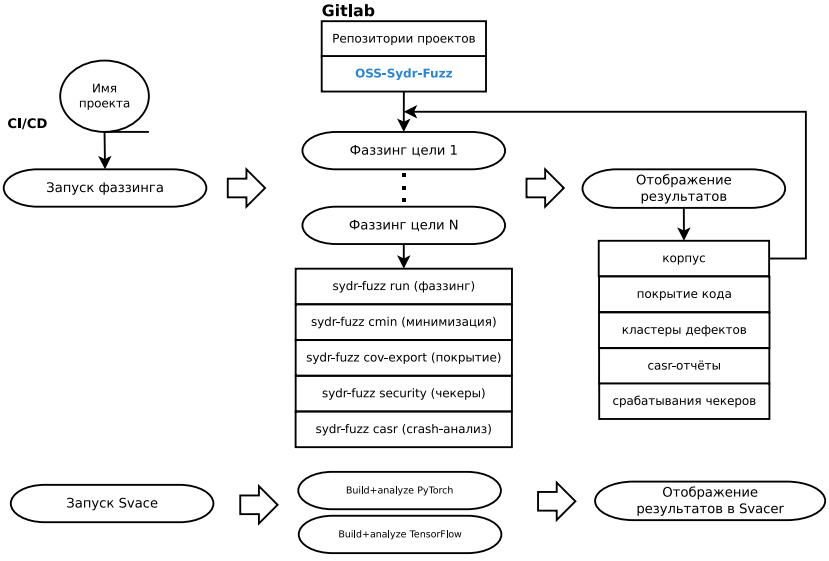
Почему это актуально?

- ML/DL фреймворки – фундаментальные библиотеки для разработки продуктов, использующих технологии искусственного интеллекта.
- **PyTorch** (Meta/Facebook) - не поддерживает автоматическое фаззинг-тестирование.
- **TensorFlow** (Google) - есть поддержка анализа в проекте OSS-Fuzz
  - наш патч github.com/google/oss-fuzz/pull/7704 возродил фаззинг TensorFlow в OSS-Fuzz.

Наша цель: непрерывный статический и динамический анализ фреймворков ИИ и сопутствующих проектов с помощью Svace и Sydr.

Система непрерывного анализа ИИ

ИСП РАН



Найденные ошибки в фреймворках ИИ

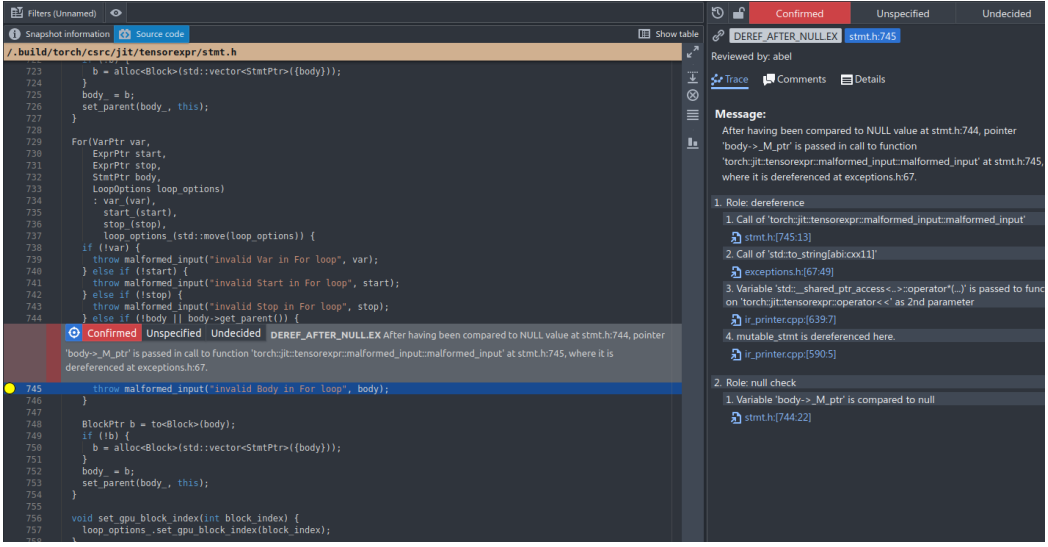
ИСП РАН

Проект	Ошибки (Sydr|Svace)	Исправлений	Принято в upstream
TensorFlow	25 (2|23)	25	24
PyTorch	24 (11|13)	24	23

9 ошибок было обнаружено в сторонних проектах (LLVM, oneDNN, miniz, torchvision, openjpeg). Подробности о найденных ошибках тут: github.com/ispras/oss-sydr-fuzz/blob/master/TROPHIES.md

Инструмент статического анализа Svace

ИСП РАН



Пример ошибки, обнаруженной Svace

ИСП РАН

tensorflow/pull/59282

```
3699  3699            }
3700  3700            lexer_.Lex();
3701  3701          break;
3702  3702        }
3703  3703        case TokKind::kRbrace: {
      3704  +       if (nest_level == 0) {
      3705  +         return TokenError("unexpected '}' token");
      3706  +       }
3704  3707          nest_level--;
3705  3708          if (elems_seen_per_dim[nest_level] != shape.dimensions(nest_level)) {
3706  3709            return TokenError(absl::StrFormat(
3707  3710                "expects %d elements in the %sth element, but sees %d",
```

Инструмент динамического анализа Sydr

Инструмент динамической символьной интерпретации (DSE) Sydr:

Sydr-Fuzz обеспечивает полный цикл непрерывного динамического анализа:

1. Гибридный фаззинг
2. Минимизация корпуса
3. Проверка предикатами безопасности
4. Дедупликация и кластеризация найденных ошибок
5. Сбор достигнутого покрытия

Гибридный фаззинг

Наши фаззинг-цели и контейнеры для сборки PyTorch и TensorFlow доступны здесь: github.com/ispras/oss-sydr-fuzz

- Поиск фаззинг-целей в коде фреймворков
- Сборка библиотек и фаззинг-целей с санитайзерами и инструментацией для фаззинга/сбора покрытия
- Гибридный фаззинг C++-кода
  - Комбинированный запуск Sydr + libFuzzer
  - Комбинированный запуск Sydr + AFLplusplus
- Фаззинг Python-кода с помощью Atheris
- Минимизация и зацикливание корпуса

Исправление ошибки в TensorFlow

ИСП РАН

tensorflow/pull/56455

```
86   86    // Handles moving the data index forward, validating the arguments, and avoiding
87   87    // overflow or underflow.
88        - Status IncrementOffset(int old_offset, size_t increment, size_t max_size,
     88   + Status IncrementOffset(int old_offset, int increment, size_t max_size,
89   89                            int* new_offset) {
90   90      if (old_offset < 0) {
91   91        return errors::InvalidArgument("Negative offsets are not allowed: ",
92   92                                       old_offset);
93   93      }
     94   +  if (increment < 0) {
     95   +    return errors::InvalidArgument("Negative increment is not allowed: ",
     96   +                                   increment);
     97   +  }
94   98      if (old_offset > max_size) {
95   99        return errors::InvalidArgument("Initial offset is outside data range: ",
96  100 +                                   old_offset);
```

Исправление ошибки в TensorFlow ч.2

ИСП РАН

tensorflow/pull/60082

```
94   94     if (increment < 0) {
95   95       return errors::InvalidArgument("Negative increment is not allowed: ",
96   96                                       increment);
97   97     }
98   98     if (old_offset > max_size) {
99   99       return errors::InvalidArgument("Initial offset is outside data range: ",
100  100                                      old_offset);
101  101    }
102      -   *new_offset = old_offset + increment;
103      -   if (*new_offset > max_size) {
     102 +   int64_t sum = old_offset + increment;
     103 +   if (sum > max_size) {
104  104      return errors::InvalidArgument("Data too short when trying to read string");
105  105    }
106  106    // See above for the check that the input offset is positive. If it's negative
107  107    // here then it means that there's been an overflow in the arithmetic.
108      -   if (*new_offset < 0) {
109      -     return errors::InvalidArgument("Offset too large, overflowed: ",
110      -                                    *new_offset);
     108 +   if (sum < 0) {
     109 +     return errors::InvalidArgument("Offset too large, overflowed: ", sum);
111  110    }
     111 +   *new_offset = sum;
112  112    return OkStatus();
```

Символьные предикаты безопасности

ИСП РАН

- Целенаправленный поиск ошибок выхода за границы массива, целочисленного переполнения, деления на ноль.
- Предикаты применяются после фаззинга на минимизированном корпусе с хорошим покрытием;
- Срабатывания верифицируются на санитайзерах (asan+ubsan).
- Точность обнаружения ошибок предикатами безопасности составляет 96% на наборе тестов Juliet.

Срабатывание предиката безопасности в зависимости PyTorch:
https://github.com/richgel999/miniz/pull/238

```
SUMMARY: UndefinedBehaviorSanitizer
/pytorch_fuzz/third_party/miniz-2.0.8/miniz.c:3654:41:
undefined-behavior in mz_zip_reader_read_central_dir

runtime error: unsigned integer overflow:
4294967295 * 46 cannot be represented in type 'unsigned int'

miniz.c:3654 - imul ecx, eax, 0x2e - unsigned integer overflow
miniz.c:3654 - jb 0x1c72032 - error sink
```

Целочисленное переполнение в OpenJpeg (OpenCV) ИСП РАН

github.com/opencv/opencv/issues/22284

opencv/3rdparty/openjpeg/openjp2/image.c:134:

```
l_y1 = p_cp->ty0 + (p_cp->th - 1U) * p_cp->tdy; /* can't overflow */
```

Can't overflow? But we can!

Срабатыване символьного чекера Sydr:

opj_image_comp_header_update:/opencv/3rdparty/openjpeg/openjp2/image.c:134 - imul r15d, eax - **unsigned integer overflow**

Автоматическое подтверждение дефекта санитайзерами:

/opencv/3rdparty/openjpeg/openjp2/image.c:134:40: runtime error: **unsigned integer overflow**: 2 * 4278190076 cannot be represented in type 'unsigned int'

github.com/ispras/casr

```
Stacktrace:
#0 0x5761e0 in __sanitizer::internal_memmove(...)
                  ....
#7 0xfd5dd4a in vision::image::decode_png(at::Tensor const&, ...)

CrashLine: /torchvision/csrc/io/image/cpu/decode_png.cpp:61
Source:
    59      [](png_structp png_ptr, png_bytep output, png_size_t bytes) {
    60        auto reader = static_cast<Reader*>(png_get_io_ptr(png_ptr));
--->61        std::copy(reader->ptr, reader->ptr + bytes, output);
    62        reader->ptr += bytes;
CrashSeverity: NOT_EXPLOITABLE SourceAv
```

Спасибо за внимание!