



PHDays - 2025

Чудиновских Николай
Младший Исследователь

Первое впечатление





Apache Arrow – быстрее, ниже, сложнее

ИТЕРАТОР ПО ВИДЕО

```
import pyarrow as pa
import pyarrow.compute as pc
from pyarrow import dataset as ds
import cv2 #opencv-python
from io import BytesIO
from PIL import Image
import numpy as np

dataset = ds.dataset(paths, filesystem=s3_fs, format="parquet")

condition = (
    pc.field("header", "stamp", "sec") * 1000 +
    pc.field("header", "stamp", "ms")
).isin([2500, 3500])

reader = dataset.scanner(filter=condition).to_reader()

try:
    while True:
        batch = reader.read_next_batch()
        for row_idx in range(batch.num_rows):
            array = batch["data"]
            PNG = array[row_idx].as_buffer() # ARROW ARRAY
            IMG = np.array(Image.open(BytesIO(PNG)))
            RES = cv2.do_some_work(IMG)
except StopIteration as e:
    pass
```

chunk_100500.parquet		
header.stamp		data
.sec	.ms	
0	0	PNG_BLOB_0
0	500	PNG_BLOB_1
1	0	PNG_BLOB_2
1	500	PNG_BLOB_3
2	0	PNG_BLOB_4
2	500	PNG_BLOB_5
3	0	PNG_BLOB_6
3	500	PNG_BLOB_7
4	0	PNG_BLOB_8

18

Apache Arrow – быстрее, ниже, сложнее

ОСНОВНЫЕ СТРУКТУРЫ

ТИП В ARROW

DataType

Buffer

Array

Table

RecordBatch

Schema

АНАЛОГ В PANDAS/SPARK

np.dtype

np.array

pd.Series / spark.df.col

pd.DataFrame

spark.RecordBatch

spark.df.schema

Apache Arrow – быстрее, ниже, сложнее

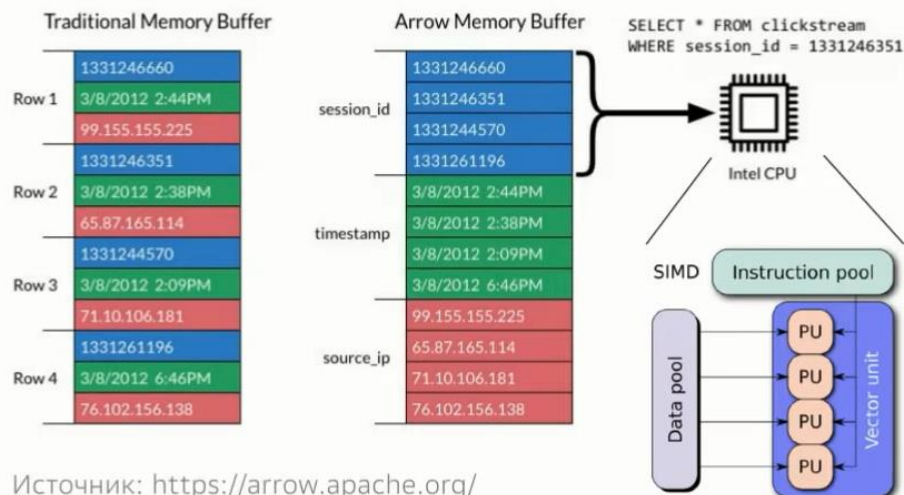
ARROW MEMORY BUFFER

- Последовательный участок памяти
- Типизирован
- Хранит данные одной колонки
- Выровнен по 8 (или даже 64) байтам
- Оптимален для SIMD*-инструкций в CPU

* https://en.wikipedia.org/wiki/Single_instruction,_multiple_data

Чтобы выйти из полноэкранного режима, нажмите Esc

	session_id	timestamp	source_ip
Row 1	1331246660	3/8/2012 2:44PM	99.155.155.225
Row 2	1331246351	3/8/2012 2:38PM	65.87.165.114
Row 3	1331244570	3/8/2012 2:09PM	71.10.106.181
Row 4	1331261196	3/8/2012 6:46PM	76.102.156.138



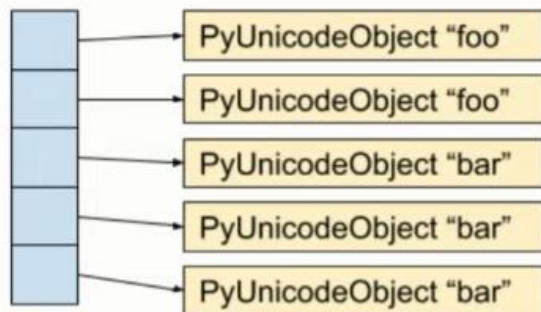
Источник: <https://arrow.apache.org/>
Авторы: ASF

Apache Arrow – быстрее, ниже, сложнее

ARROW STRING ARRAY VS NUMPY VS PANDAS (1.X)

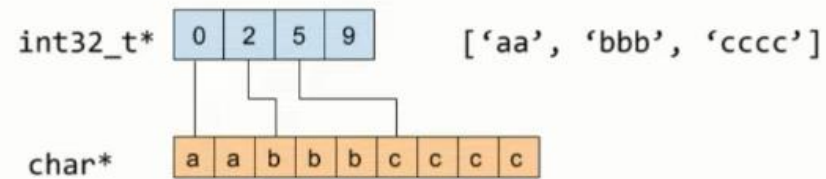
NUMPY/PANDAS:

- Строки как Python-объекты
- Можно применять любые python-функции
- Сколько памяти?
 - 8 (PyObject*) + 48 (Python C struct) + str_len + 1
 - 57 байт оверхеда на каждую строку!



ARROW:

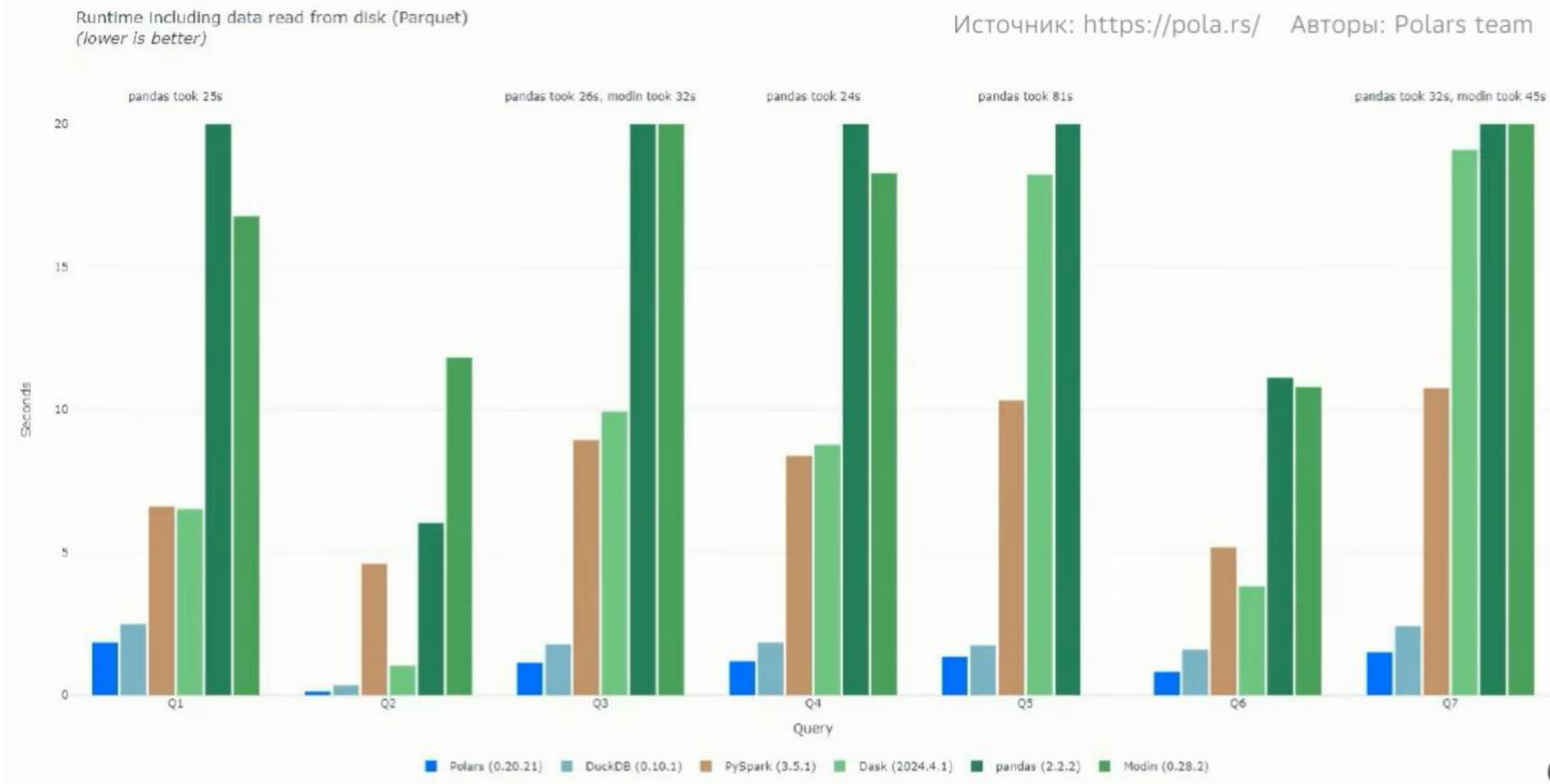
- Один массив из всех склеенных строк
- Один массив с оффсетами
 - 4 байта оверхеда на строку
- Битовая маска null
 - 1 бит оверхеда на строку





Apache Arrow – быстрее, ниже, сложнее

Скорость работы на запросах TPC-H



Apache Arrow – быстрее, ниже, сложнее

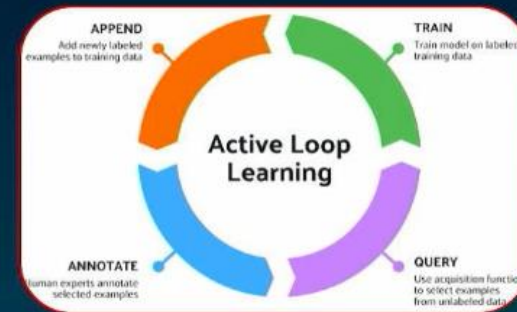
Применительно к нашим проектам:

Выполнение препроцессинга и фильтрации данных на Apache Arrow, в то время как EDA можно проводить в pandas

Современные подходы к управлению данными в эпоху AI и Big Data

Что такое активное обучение?

Активное обучение — это парадигма машинного обучения, при которой модель **целенаправленно выбирает наиболее информативные данные** для разметки, вместо того чтобы обучаться на случайной, полностью размеченной выборке



Definition

Key Concept

- Модель сама определяет, какие данные нужно разметить
- Это позволяет сократить расходы на аннотацию

Comparison

Passive vs. AL

- Высокие Затраты, Низкая Масштабируемость
- AC: низкие Затраты, Высокая Масштабируемость

Techniques

Sampling

- Примеры, в которых она наименее уверена
- Данные с наибольшими разногласиями

Современные подходы к управлению данными в эпоху AI и Big Data

Vision-Language Models



VLMs (Visual-Linguistic Models) — это модели ИИ, которые могут понимать и анализировать как визуальную (изображения, видео), так и текстовую информацию

Примеры возможностей VLM:

- Подписи к изображениям (Image Captioning)
- Визуальные вопросы и ответы (Visual Question Answering)
- Поиск изображений по текстовому запросу (Text-based Image Retrieval)



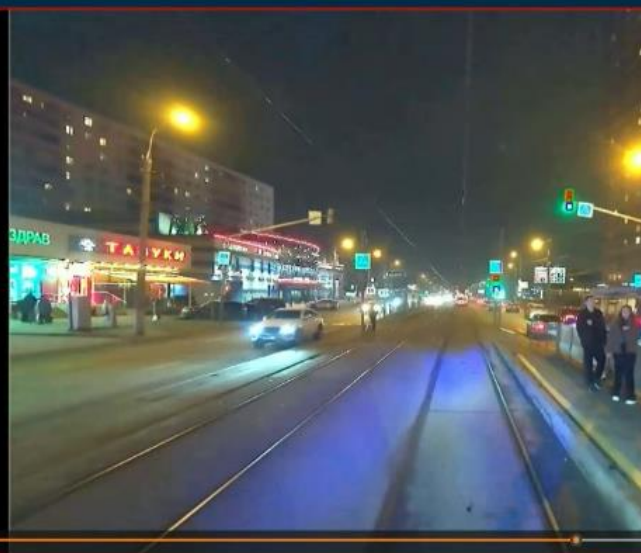
Современные подходы к управлению данными в эпоху AI и Big Data



Anything unusual is happening on the road?



Yes, there is a motorcycle lying on the road, which is unusual and may indicate an accident. Traffic cones are placed around it, likely to alert drivers and prevent further incidents.



How many people were on the bus stop?

Working as Planner... (3.7 seconds)

question: "How many people were on the bus stop?". Please think carefully and respond with your plan in JSON directly.

Planner Response:

```
[{"type": "answerer"}]
```

After browsing the video and the question. My plan to figure out the answer is as follows:

1. Analyze the whole video directly without cropping.

Working as Answerer... (7.0 seconds)

How many people were on the bus stop? Please think step by step and provide your response.

Answerer Response:

2.

After watching the whole video, my answer is: 2

Современные подходы к управлению данными в эпоху AI и Big Data

Active Learning & VLMs



Синергия

- > Выбора наиболее ценных пар данных
- > Visual-Linguistic Models (VLMs), которые понимают и обрабатывают как визуальные (изображения и видео), так и текстовые данные, могут дополнительно улучшить процесс активного обучения



Современные подходы к управлению данными в эпоху AI и Big Data

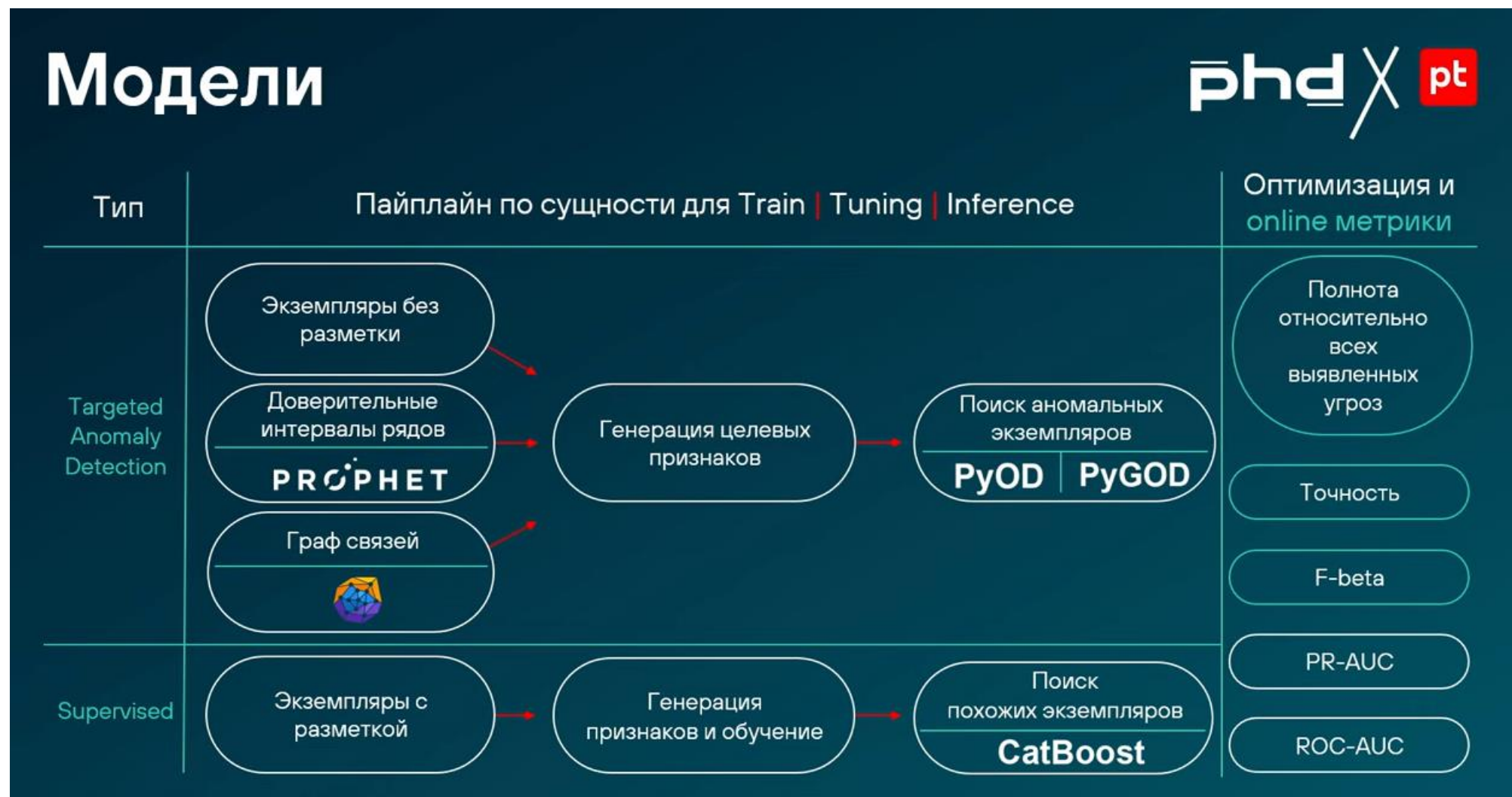
Применительно к нашим проектам:

Доразметка данных с помощью активного обучения

Использование VLM для распознавания документов и писем

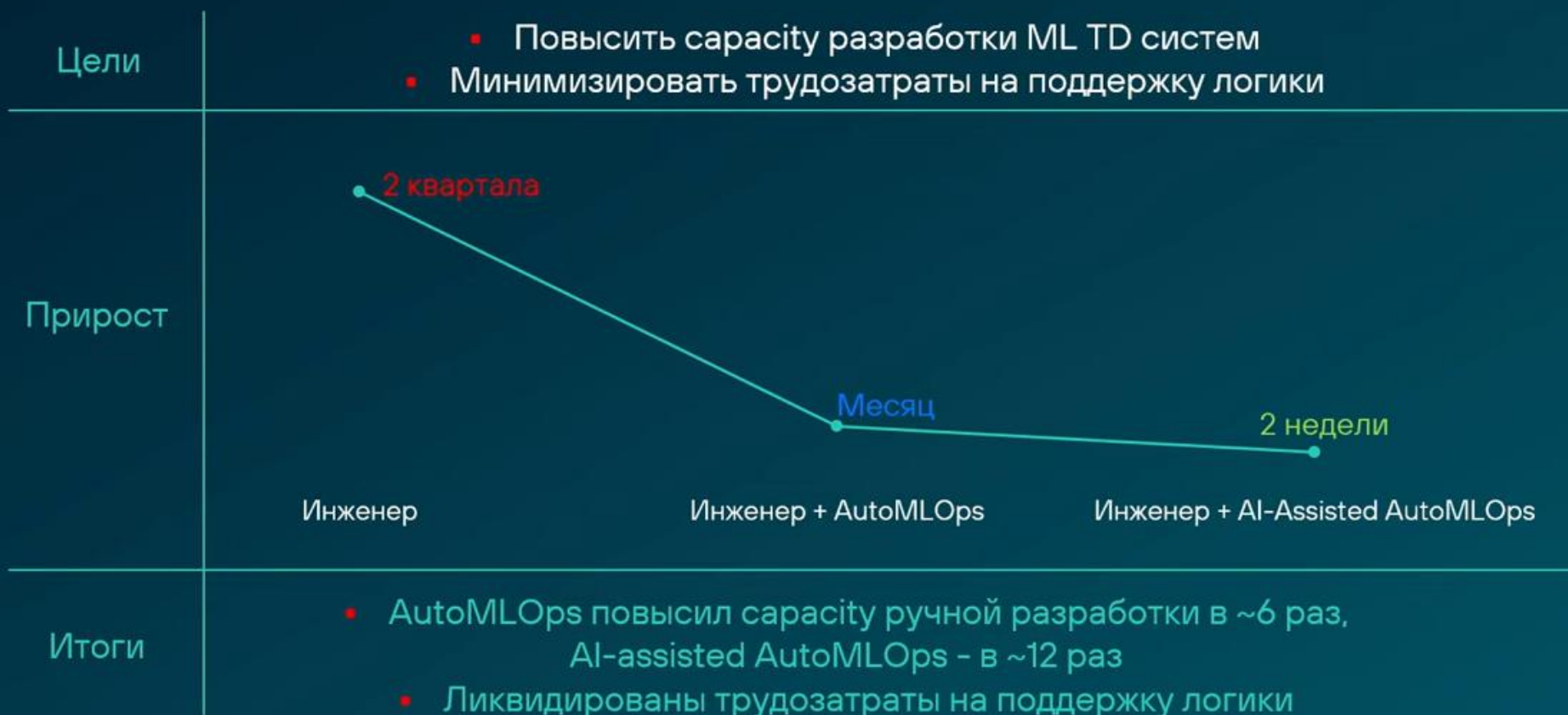


AutoML threat detection – повышаем уровень автономности SOC



AutoML threat detection – повышаем уровень автономности SOC

AutoMLOps Value для ML TD



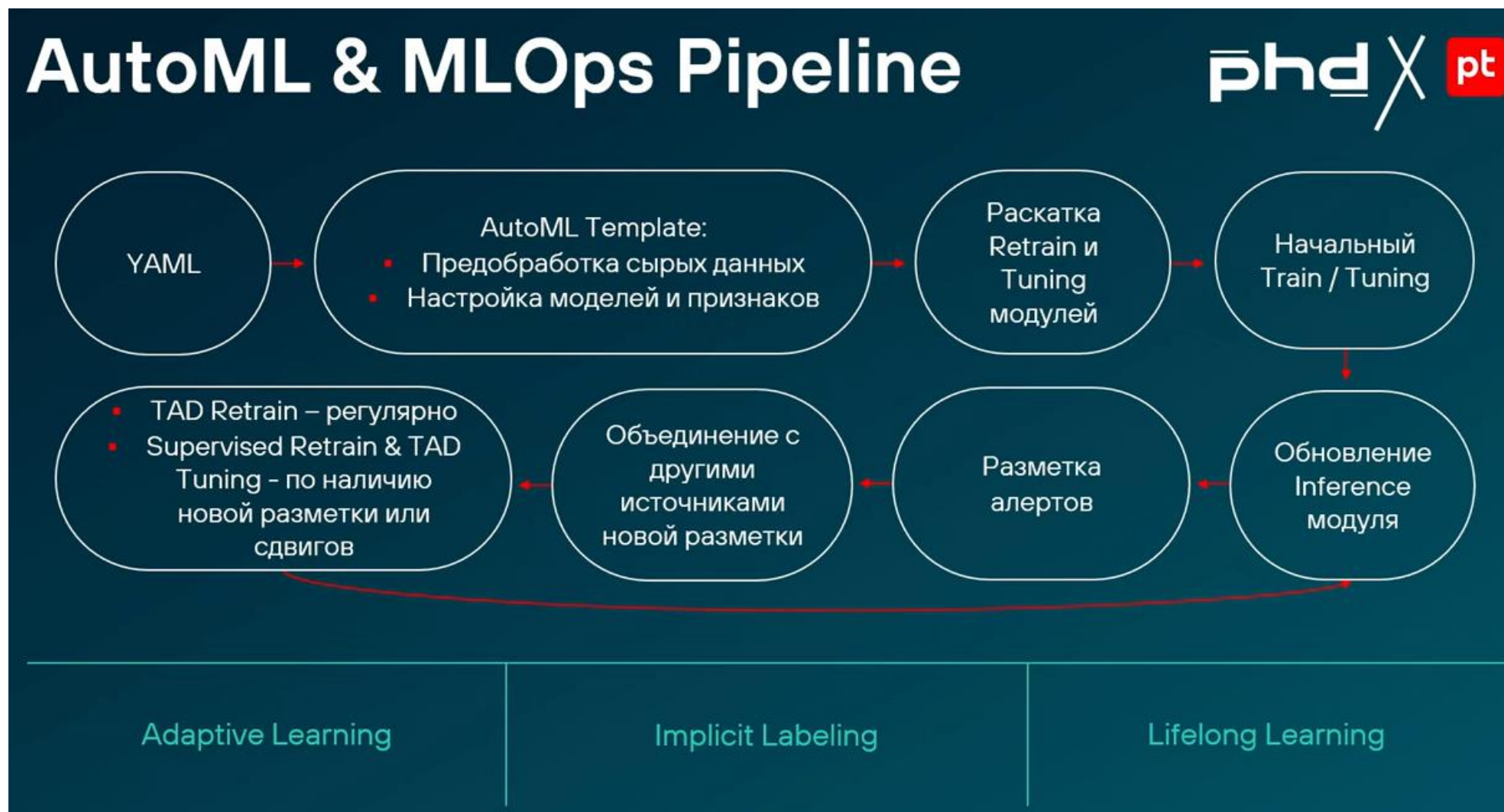


AutoML threat detection – повышаем уровень автономности SOC





AutoML threat detection – повышаем уровень автономности SOC





AutoML threat detection – повышаем уровень автономности SOC





AutoML threat detection – повышаем уровень автономности SOC

Применительно к нашим проектам:

- Возможное применение AutoML для простых задач ИБ (ускорение разработки)
- Использование инструментов мониторинга и автоматизации экспериментов

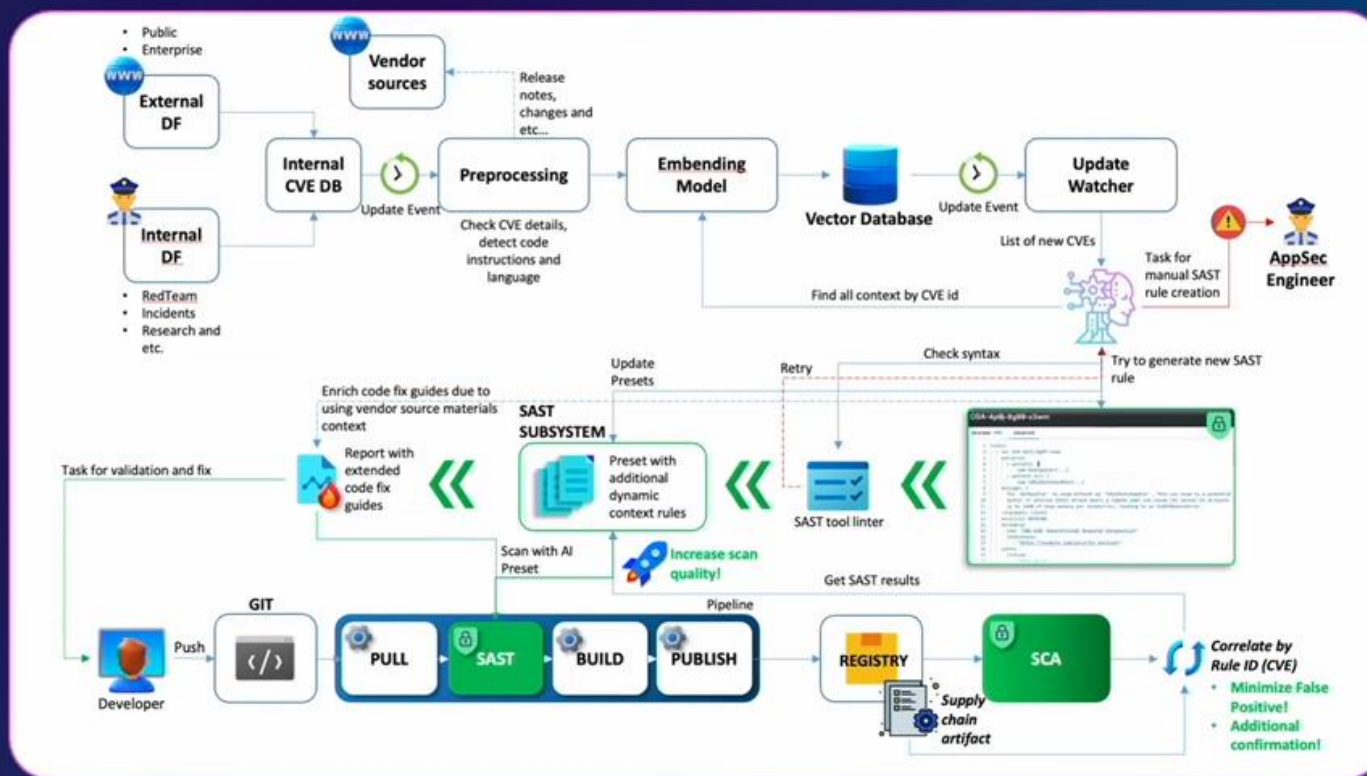
Повышение точности SAST за счет динамической генерации правил AI агентом

Решение на базе AI-агента



AI открывает окно возможностей

- CVE из набора фидов сохраняется в БД
- Далее препроцессинг и оценка возможности проверки на уровне кода
- AI-агент генерирует правила для SAST на основе CVE
- Генерируем пресет для SAST
- Работаем с результатами
- Добавляем процессы и учитываем больше контекста



Повышение точности SAST за счет динамической генерации правил AI агентом

Как сделать, чтобы работало



- GenAI + RAG = контекстное понимание уязвимости
- БД гибко расширяется
- Возможность учитывать больше фактов и добавлять свои метрики и индексы
- AI-агент будет точнее выполнять команды на генерацию целого пресета по 1 текстовому запросу
- Динамический пресет и фильтрация, добавление индексов и генерация новых пресетов on-demand
- Фидбек-петля: на основе обратной связи от команд и ручной выборочной валидации возможна корректировка классификаторов
- Использование линтеров в простом сценарии проверки правила и полную автоматизацию на базе AI-агента для формирования задания на проверку работы сгенерированных правил



Повышение точности SAST за счет динамической генерации правил AI агентом

Риски и ограничения



Риск	Причина	Митигация
False Positive	Слишком широкий паттерн	QA-корпус + автоматизированная генерация проверки работы правила AI-агентом
False Negative	Сложная логика	Пополнение базы, расширение RAG
Нерабочие правила после генерации	Низкое качество описания CVE	Более сложный процесс препроцессинга
Latency in CI	+N тысяч правил замедляют build	Preset-slicing, инкрементальный анализ
Prompt-Injection	CVE-текст ломает YAML	Sanitizer + schema validation
Legal & IP	Права на сгенерированный код	OSS-лицензия/CLA для правил

Повышение точности SAST за счет динамической генерации правил AI агентом

Метрики



MTT-Rule

Mean Time to Rule — время от публикации CVE до генерации готового правила. Это ключевой показатель «реактивности» системы. Мы ориентируемся на **< 24 ч**, чтобы критичные уязвимости покрывались до релиза.

Precision / Recall

Точность и полнота срабатываний правила. Precision 0.9 означает, что 90 % алертов — верные; Recall 0.7 — что 70% случаев уязвимости действительно детектируются. Это стандартная пара метрик для оценки эффективности бинарных детекторов, и они дают сбалансированную картину качества.

Δ CI-Time

Изменение времени сборки CI при подключении автоген-пресета. Меньше +5 % при 1 000 правил считается отличным компромиссом между глубиной проверки и скоростью пайплайна.

Developer like-rate

Метрика доверия и полезности: насколько разработчики отмечают полезность найденных проблем (например, 👍 в PR-комментарии или auto-issue). $\geq 80\%$ — хороший сигнал, что система не раздражает, а помогает.

Автоген-покрытие

Доля критичных CVE, на которые было успешно сгенерировано правило. Цель $\geq 60\%$ за 6 месяцев показывает, что метод покрывает значимую часть угроз, а не единичные кейсы. Эта метрика доказывает применимость подхода в масштабе.



AutoML threat detection – повышаем уровень автономности SOC

Применительно к нашим проектам:

- Проверка кода на уязвимости с помощью LLM
- Интересные метрики для мониторинга