УДК 004

АНАЛИЗ ВИДЕОПОТОКА В РЕАЛЬНОМ ВРЕМЕНИ С ИСПОЛЬЗОВАНИЕМ НЕЙРОСЕТЕЙ

Пономарева А.А., студентка гр. ИИм-231, II курс,

Пузынин С.В., студент гр. ИИм-231, II курс

Кузбасский государственный технический университет имени Т.Ф. Горбачева

г. Кемерово

Анализ видеопотока в реальном времени – одна из ключевых задач компьютерного зрения и искусственного интеллекта. Современные системы видеонаблюдения уже не ограничиваются простым архивированием видеоданных — они становятся интеллектуальными, способными мгновенно анализировать происходящее и реагировать на события.

С ростом объемов видеоданных (по оценкам, 80% всей интернет-трафика составляют видео) и увеличением числа подключенных устройств (камеры видеонаблюдения, дроны, автомобильные регистраторы) возникла потребность в автоматическом анализе видео в режиме реального времени. Ручной просмотр оператором становится невозможным из-за огромного объема информации.

Несмотря на стремительное развитие технологий анализа видеопотока, существуют несколько ключевых проблем, которые препятствуют их повсеместному внедрению:

1) Ограниченные вычислительные ресурсы
   - Обработка видео в реальном времени требует мощных GPU или специализированных процессоров (TPU, FPGA).
   - Видеокамеры, работающие на периферии (edge devices), имеют ограниченные вычислительные возможности.
2) Задержки обработки
   - Классические модели глубокого обучения требуют значительных вычислений, что может приводить к задержкам в анализе.
   - В реальном времени критична минимальная задержка: в системах безопасности даже 1 секунда может стать разницей между предотвращением инцидента и его фиксацией постфактум.
3) Точность и надежность алгоритмов

- Ложные срабатывания могут приводить к ненужным вызовам служб безопасности.
- Недостаточная точность может приводить к пропуску важных событий.
- Данные с видеокамер могут быть зашумлены, с плохим освещением, что усложняет детекцию объектов.

4) Правовые и этические вопросы

- Использование технологий распознавания лиц сталкивается с проблемами конфиденциальности и защиты персональных данных.
- В некоторых странах законодательно ограничено применение систем автоматического наблюдения.

5) Масштабируемость систем

- В городах с тысячами камер возникает необходимость в распределенной обработке данных.
- Необходим баланс между локальной обработкой (на устройствах) и отправкой видео в облачные серверы.

Анализ видеопотока в реальном времени требует использования различных технологий, начиная от классических алгоритмов компьютерного зрения и заканчивая современными глубокими нейросетями. Разберем ключевые подходы к обработке видео.

1. Методы классического компьютерного зрения (OpenCV)

До появления глубокого обучения анализ видео выполнялся методами классического компьютерного зрения. OpenCV (Open Source Computer Vision Library) — одна из самых популярных библиотек для обработки изображений и видео.

Основные методы классического анализа видео:

1) Методы обнаружения движения
   - Разностный метод (frame differencing) – сравнение последовательных кадров для выявления изменений.
   - Фоновые модели (Background Subtraction) – построение модели фона для выделения подвижных объектов (например, алгоритмы MOG, KNN).
   - Оптический поток (Optical Flow) – анализ направления движения пикселей между кадрами.

2) Обнаружение и отслеживание объектов

   - Метод Хаара (Haar Cascades) – ранняя технология детекции лиц и объектов.

- HOG (Histogram of Oriented Gradients) – анализ градиентов изображения для распознавания форм (используется в детекции пешеходов).
- SURF/SIFT – методы обнаружения и сопоставления ключевых точек объектов.
- Коррелляционный трекинг – отслеживание объектов по шаблонам.

3) Обнаружение аномалий и действий

- Анализ статистики движения в кадре.
- Выявление аномального поведения на основе правил.

2. Глубокие нейросети для анализа видео

С появлением глубокого обучения и специализированных архитектур нейросетей анализ видео значительно продвинулся вперед. Нейросети позволяют не только обнаруживать объекты, но и классифицировать их, понимать контекст сцены, предсказывать события.

Ключевые направления нейросетевого анализа видео:

1) Обнаружение объектов (Object Detection)
   - Используются сверточные нейросети (CNN), такие как YOLO (You Only Look Once), Faster R-CNN, SSD.
   - Позволяют находить объекты на кадре и определять их координаты.
   - YOLO – один из самых популярных алгоритмов, способный обрабатывать видео в реальном времени (~30-60 FPS на GPU).
2) Распознавание действий (Action Recognition)
   - Используются архитектуры 3D-CNN, RNN, Transformers (например, MoViNet, SlowFast, I3D).
   - В отличие от Object Detection, анализируют не только один кадр, но и временной контекст.
   - Применяются в системах безопасности (обнаружение драк, падений, курения).
3) Сегментация объектов (Semantic & Instance Segmentation)
   - Алгоритмы Mask R-CNN, DeepLab, U-Net позволяют выделять контуры объектов.
   - Используется в медицине (анализ медицинских видео), автопилотах.

Архитектура системы анализа видео

Анализ видеопотока в реальном времени требует грамотной архитектуры системы, обеспечивающей минимальные задержки, высокую точность детекции объектов и возможность масштабирования. В этом разделе разберем ключевые компоненты такой системы, выбор аппаратного обеспечения и методы оптимизации обработки данных.

Полноценная система анализа видеопотока включает несколько ключевых компонентов, которые обеспечивают сбор, обработку и интерпретацию данных.

Компоненты системы анализа видео:

1) Источник видеопотока
2) Модуль предобработки видео
3) Нейросетевой анализ видеопотока
4) Логика принятия решений

Реализация на практике

Создание системы анализа видеопотока в реальном времени требует интеграции нескольких ключевых технологий: нейросетевых моделей, библиотек обработки видео и серверных решений для масштабируемости. В этом разделе мы разберем, как можно построить такую систему, начиная от выбора инструментов и заканчивая запуском рабочего прототипа.

1. Выбор технологий и инструментов

Для построения системы анализа видео рассмотрим три основных компонента:

1) Обработка видеопотока
   - OpenCV – базовая библиотека для работы с видео.
   - GStreamer – продвинутый фреймворк для потоковой обработки видео.
   - FFmpeg – мощный инструмент для конвертации и потоковой передачи видео.
2) Модели машинного зрения
   - Обнаружение объектов: YOLOv8, Faster R-CNN, SSD.
   - Отслеживание объектов: DeepSORT, ByteTrack.
   - Распознавание действий: MoViNet, SlowFast.
3) Серверная часть
   - Фреймворки: FastAPI (быстрая асинхронная обработка), Flask (легковесное API).
   - Базы данных: PostgreSQL, MongoDB (хранение метаданных событий).
   - Очереди задач: Redis, Kafka (для обработки видеопотока в нескольких процессах).

2. Захват и предобработка видео

Первым шагом является захват видеопотока и его предобработка перед подачей в нейросетевую модель.

```python
# Подключение к IP-камере или веб-камере
video_source = "rtsp://user:password@camera_ip/stream"
cap = cv2.VideoCapture(video_source)
while cap.isOpened():
    ret, frame = cap.read()
    if not ret:
        break
     # Уменьшение размера кадра для ускорения обработки
    frame = cv2.resize(frame, (640, 480))

    cv2.imshow('Video Stream', frame)

    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()
```

Рисунок 1 – Пример кода на Python с OpenCV

3. Обнаружение объектов с YOLOv8

Для анализа сцены мы используем модель YOLOv8, которая обеспечивает высокую точность и скорость работы.

```python
from ultralytics import YOLO
import cv2

# Загрузка модели YOLOv8
model = YOLO("yolov8n.pt")

# Захват видео
cap = cv2.VideoCapture(0)

while cap.isOpened():
    ret, frame = cap.read()
    if not ret:
        break

    # Запуск модели YOLO
    results = model(frame)

    # Отображение результатов
    for result in results:
        for box in result.boxes:
            x1, y1, x2, y2 = map(int, box.xyxy[0])
            cv2.rectangle(frame, (x1, y1), (x2, y2), (0, 255, 0), 2)

    cv2.imshow("YOLO Detection", frame)

    if cv2.waitKey(1) & 0xFF == ord("q"):
        break

cap.release()
cv2.destroyAllWindows()
```

Рисунок 2 – Пример кода на Python

4. Отслеживание объектов с DeepSORT

Обнаружение объектов на каждом кадре не позволяет отслеживать их перемещение. Для этого используется алгоритм DeepSORT, который позволяет идентифицировать объекты между кадрами.

```python
from deep_sort_realtime.deepsort_tracker import DeepSort
tracker = DeepSort(max_age=5)  # Трекер удаляет объекты, если они пропали
while cap.isOpened():
    ret, frame = cap.read()
    if not ret:
        break

    results = model(frame)

    detections = []
    for result in results:
        for box in result.boxes:
            x1, y1, x2, y2 = map(int, box.xyxy[0])
            conf = box.conf[0]
            class_id = box.cls[0]
            detections.append(([x1, y1, x2, y2], conf, class_id))

    tracks = tracker.update_tracks(detections, frame=frame)

    for track in tracks:
        if not track.is_confirmed():
            continue
        x1, y1, x2, y2 = map(int, track.to_tlbr())
        track_id = track.track_id
        cv2.rectangle(frame, (x1, y1), (x2, y2), (255, 0, 0), 2)
        cv2.putText(frame, f"ID {track_id}", (x1, y1 - 10), cv2.FONT_HERSH

    cv2.imshow("Tracking", frame)

    if cv2.waitKey(1) & 0xFF == ord("q"):
        break
cap.release()
cv2.destroyAllWindows()
```

Рисунок 3 – Пример кода интеграция DeepSORT с YOLO на Python

Этот код добавляет уникальные ID для отслеживаемых объектов, производит фильтрацию ложных срабатываний и привязку объектов к траекториям.

5. Оптимизация и развертывание

После создания системы важно оптимизировать ее для работы в реальном времени.

FastAPI позволяет обрабатывать видео через API – например, получать JSON с детекциями объектов.

Практическая реализация системы анализа видео включает три этапа: захват и предобработку видеопотока, анализ нейросетями (YOLO + DeepSORT), а затем развертывание сервиса. Оптимизация с помощью TensorRT, параллельной обработки и Edge Computing позволяет работать в реальном времени. Таким образом, мы можем создать эффективную систему для видеонаблюдения, безопасности и анализа поведения в сложных сценах.

```python
from fastapi import FastAPI, File, UploadFile
import cv2
import numpy as np

app = FastAPI()

@app.post("/detect/")
async def detect(file: UploadFile = File(...)):
    contents = await file.read()
    nparr = np.frombuffer(contents, np.uint8)
    img = cv2.imdecode(nparr, cv2.IMREAD_COLOR)

    results = model(img)

    objects = []
    for result in results:
        for box in result.boxes:
            x1, y1, x2, y2 = map(int, box.xyxy[0])
            objects.append({"x1": x1, "y1": y1, "x2": x2, "y2": y2})

    return {"detections": objects}
```

Рисунок 4 – Пример кода развертывания через FastAPI на Python

Заключение

Системы анализа видеопотока находят применение в самых разных сферах – от безопасности и транспорта до медицины и ритейла.

Например, Amazon использует камеры с ИИ для контроля за перемещением грузов и прогнозирования задержек, на заводах Tesla используются камеры с ИИ для проверки качества сборки автомобилей в реальном времени и т.д.

Будущее технологий анализа видеопотока связано с улучшением качества данных, адаптацией моделей и повышением скорости обработки, но также требует решения важных этических и правовых вопросов.

Список литературы

1. Сзелиски, Р. Компьютерное зрение: Алгоритмы и приложения. - Бином, 2010
2. Гудфеллоу, И., Бенгио, Ю., & Курвиль, А. Глубокое обучение.. - MIT Press, 2016
3. OpenCV: Библиотека компьютерного зрения с открытым исходным кодом. // Документация OpenCV. (н.д.) URL: https://opencv.org/ (дата обращения: 19.03.2025).
4. Редмон, Дж., Диввала, С., Гиршик, Р., & Фархадди, А. Вы только посмотрите один раз: унифицированное обнаружение объектов в реальном времени: автореф. дис. 2016