Санкт-Петербургский государственный университет

Кафедра системного программирования

Шарганов Артем Николаевич

Разработка системы поиска аномалий при видеонаблюдении

Бакалаврская работа

Научный руководитель:
ст. преп. Я. А. Кириленко

Рецензент:
Старший консультант ООО "САП Лабс" В. М. Палкин

Санкт-Петербург
2018

SAINT PETERSBURG STATE UNIVERSITY

Software engineering

Artem Sharganov

Development of system for abnormal event detection in video surveillance

Graduation Thesis

Scientific supervisor:
senior lecturer I.A. Kirilenko

Reviewer:
Senior consultant at SAP Labs V.M. Palkin

Saint Petersburg
2018

Оглавление

Введение

Важными задачами любого крупного бизнес-центра или супермаркета являются обеспечение безопасности посетителей и защита собственного имущества. Данные задачи призвана выполнять охранная система. Одним из основных компонентов такой системы являются камеры видеонаблюдения, с помощью которых производится визуальный контроль за периметром. Видеонаблюдение предоставляет большое количество информации, но из-за того, что информация не структурирована, присутствует вероятность, что наблюдатель не уделит достаточно внимания важному событию.

Существует несколько способов решения данной проблемы, например, выделение движения в кадре, которое позволяет существенно снизить количество обрабатываемого видео. Также некоторые системы выделяют лица, людей, машины и т.д. Все эти улучшения, безусловно, облегчают работу наблюдателю, но они плохо применимы в многолюдных местах и действуют по заранее подготовленному шаблону.

Предугадать варианты конкретных нарушений в виду различия сцен заранее очень сложно, например, если мы находимся в супермаркете, то хаотичные движение людей — это нормально, чего нельзя сказать, если камера видеонаблюдения контролирует вход или выход в метро, офис, в данном случае движение в сторону противоположную основному потоку людей может означать попытку нарушения безопасности. Гораздо легче, а иногда намного полезнее выделять какие-либо аномальные события.

Аномалией будем называть событие, в нашем случае фрагмент видео, который выбивается за пределы общего фона. Стоит отметить, что аномалия не всегда носит отрицательный, криминальный характер, а, скорее, является неестественным событием для наблюдаемой сцены. Данный подход не предоставляет информации о том, что конкретно произошло, но помогает существенно сократить и структурировать информацию. Создание системы поиска аномалий при видеонаблюдении будет рассмотрено в данной работе.

1. Постановка задачи

Целью данной работы является разработка системы для поиска аномалий при видеонаблюдении. Для достижения данной цели были сформулированы следующие задачи.

1. Разработать требования к системе поиска аномалий при видеонаблюдении.
2. Проанализировать существующие алгоритмы для поиска аномалий в видео.
3. Разработать архитектуру системы поиска аномалий при видеонаблюдении.
4. Разработать прототип системы.
5. Провести апробацию.

2. Требования к системе

В этом разделе описаны требования к системе с обсуждением причин, по которым они были выдвинуты. Требования были разделены на функциональные и нефункциональные.

2.1. Функциональные требования

2.1.1. Требования к поиску аномалий

Система должна реализовывать функциональность для нахождения аномальных (неестественных) событий в видеопотоке. Аномальное событие в системах видеонаблюдения обладает критической важностью, поэтому система должна надежно реагировать на аномалии, даже если из-за этого повысится количество ложный срабатываний.

2.1.1.1. Ограничения

Данной системе не требуется определять категорию произошедшего события.

2.1.2. Требования к графическому интерфейсу

Основная задача данной системы — предоставить оператору системы видеонаблюдения наглядную структурированную информацию о произошедших или происходящих аномальных событиях. В связи с этим система должна включать графический интерфейс, в котором информация о найденных аномалиях будет представлена с помощью всплывающих оповещений, выделения аномалий на кадре в режиме реального времени, таблиц с историей.

2.1.2.1. Просмотра видео с камер видеонаблюдения

Система несет вспомогательную функцию, поэтому требуется отображение изначального видеопотока в графическом интерфейсе, для этого необходимо разработать гибко настраиваемую экранную форму, с

помощью которой оператор сможет просматривать изображение с интересующих его видеокамер.

2.1.2.2. Выделение аномалий на кадре

Выделение аномалий на кадре в режиме реального времени является важной частью графического интерфейса. Для этого должна быть реализована следующая функциональность:

- тепловые карты
- обозначение границ аномального объекта рамками.

2.1.2.3. Отображение истории выявленных аномалий

Графический интерфейс должен включать оконную форму, которая будет отображать историю выявленных аномалий. По каждой аномалии необходимо предоставить следующую информацию: фрагмент видео с аномалией, локацию и время. Должна быть реализована возможность группировки по дате, локации.

2.1.3. Требование к хранению истории аномалий

Для удобной работы с историей, а также для обеспечения надежности, система должна:

- сохранять данные о найденных аномалиях в базу данных,
- предоставлять функциональность для выгрузки истории в файловую систему.

2.1.4. Поддержка работы с IP-камерами

Зачастую видеонаблюдение основано на IP-камерах, поэтому система должна предоставлять возможность работы с IP-камерами.

2.2. Нефункциональные требования

2.2.1. Требования к производительности

Система должна анализировать 3 видеопотока в разрешении 640x480 со скоростью 20 кадров в секунду при конфигурации с графической видеокартой NVidia GeForce GT 1060.

2.2.2. Портативность системы

Система будет использоваться людьми без особых навыков в сфере информационных технологий, поэтому она должна легко устанавливаться, а также ее внутреннее устройство должно быть максимально изолированно от конечного пользователя.

2.2.3. Требования к операционной системе

Целевой операционной системой является Ubuntu 16.04.

3. Литературный обзор

3.1. Обзор алгоритмов для поиска аномалий

Во многих областях машинного обучения одной из основных сложностей является отсутствие размеченных данных. Данная проблема достигает своего апофеоза в приложениях подобным охранным системам. Для таких систем характерна низкая частота появления аномальных событий, к тому же узнать заранее все возможные аномалии невозможно, поэтому в данном обзоре будут рассмотрены подходы, основанные на обучение без учителя.

Детекция аномалий - это задача бинарной классификации. Основная идея, на которой основываются алгоритмы, состоит в том, чтобы в начале извлечь особенности [15] из входных данных, составить из них вектор признаков (дескриптор), и с его помощью классифицировать событие. В независимости от природы особенностей, все алгоритмы детекции аномалий имеют примерно одинаковую структуру, изображенную на рис. 3.1.

3.1.1. Hand-crafted feature-based подходы

3.1.1.1. Методы, основанные на траекториях

В методах, основанных на траекториях, вычисляются траектории объектов в тренировочных данных, из которых в дальнейшем составляются дескрипторы, которые классифицируются, например, с помощью машины вспомогательных векторов [11]. Значительное отклонение траекторий тестовых данных будет означать аномалию. Хотя методы основанные на траекториях показывают хорошие результаты, особенно в сценах с маленьким количеством объектов, на данный момент нельзя говорит об их использовании в реальных приложениях. В основном это обусловлено тем, что задача трекинга объектов до сих пор не является полностью решенной, особенно для многолюдных сцен, а также для сцен с сложным освещением [13].

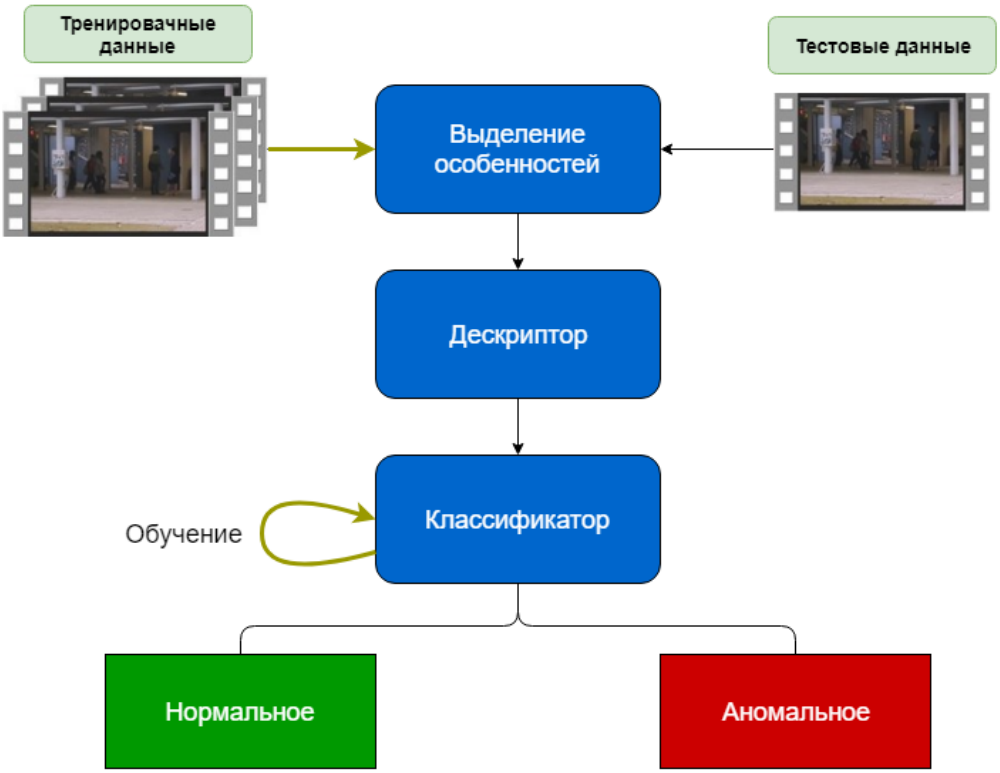


Рис. 1: Схема работа алгоритма детекции аномалии

3.1.1.2. Методы, основанные на низкоуровневых признаках

Группа подходов, основывающаяся на извлечение низкоуровневых признаков, на основе которых составляются дескрипторы. Популярными дескрипторами в таких методах являются гистограммы ориентированного потока (HOF) [3], гистограммы ориентированных градиентов (HOG) [2] и их комбинации, расширенные на временное измерение, и другие.

Так как эти подходы опираются на заранее определенные человеком признаки, данные методы иногда могут давать противоречивые результаты, так как не достаточно хорошо обобщаются на конкретную сцену.

3.1.2. Sparse coding

Основывается на идеи извлечения скрытых связей из данных. В процессе обучения с помощью тренировочных данных составляется сло-

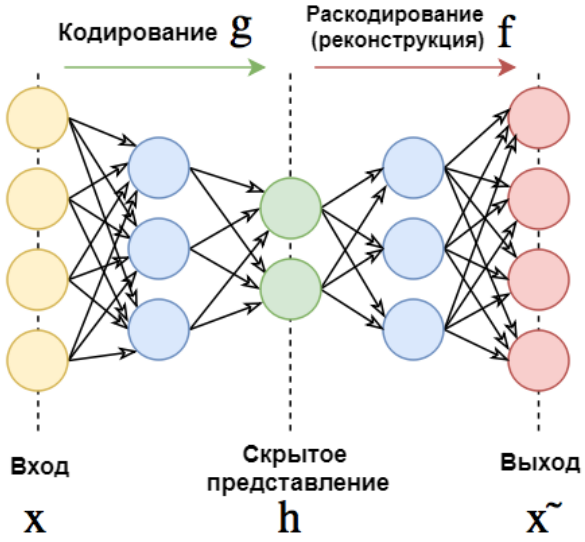
варь. Тестовые данные представляются как линейная комбинация кодовых слов из словаря, основываясь на данном представлении производится классификация. У данного подхода существует один большой минус - вычислительная сложность. Для надежной классификации требуется большой размер словаря, что требует поиска линейной комбинации в пространстве огромной размерности.

3.1.3. Методы глубоких нейронных сетей

Методы основанные на нейронных сетях, в особенности сверточных, зарекомендовали себя в задачах классификации изображений, поэтому их использование в задачах поиска аномалий в видео кажется логичным продолжением. Их популярность и мощь можно объяснить тем, что они не ограничены извлечением заранее установленных характеристик из изображений, а наоборот, находят интересные закономерности в самих данных. Также важным аспектом использования данного метода является хорошая производительность при использовании GPU.

3.1.3.2. Автоэнкодер

Рис. 2: Автоэнкодер



Автоэнкодер — тип нейронных сетей, состоящих из двух, как правило, симметричных частей (рис. 2), и «копирующий» свой вход на выход.

Первая половина, называемая кодировщик, сжимает входные данные в вектор меньшей размерности (скрытое представление).

Вторая часть — раскодировщик, пытается реконструировать оригинальные данные, полагаясь исключительно на скрытое представление.

В процессе обучения перед автоэнкодером стоит задача минимизировать разницу между входным и выходными данными. Таким образом, предполагается, что скрытое представление h будет содержать «существенные» признаки изначальных данных.

При работе с изображениями у обычного автоэнкодера есть проблема — избыточность параметров, каждый из которых применяется к конкретному региону входного изображению. Это приводит к медленной работе с картинками больших размеров, к тому же не учитывается пространственная структура.

3.1.3.2. Сверточный автоэнкодер

Сверточный автоэнкодер - автоэнкодер, состоящий целиком из сверточных слоев. Он отличается от обычного тем, что благодаря наличию свертки учитывает пространственную структуру картинки, что позволяет исследовать взаимосвязи между соседними пикселями. Также ему требуется значительно меньше параметров при том же качестве анализа, а значит скорость его работы выше.

3.1.3.3 Применение сверточных автоэнкодеров в поиске аномалий в видео

В [8] рассматривается проблема поиска аномалий в контексте исследования видео на регулярность. Предполагается, что при использовании модели обученной на видео, в которых содержались только нормальные события, ошибка реконструкции кадров с регулярными событиями будет низкой, с аномальными — высокой.

Для достижения данной цели авторы рассматривают два подхода: на основе сверточных автоэнкодеров и автоэнкодеров, на вход которым подается hand-crafted дескриптор, описанный в [14]. В результате экспиреметов было показано, что для поставленной задачи, подходы

основанные целиком на нейронных сетях дают более высокую точность, чем при использование hand-crafted дескрипторов.

Стоит отметить что, в данной статье для анализа видео используются автокодировщики на основе сверточных сетей, однако существуют специальные типы для исследования временных зависимостей, такие как рекуррентные нейронные сети.

3.1.3.4. LSTM нейронные сети

Обычные нейронные сети плохо подходят для исследования длинных временных зависимостей ввиду проблемы затухания градиентов. Для таких целей используются специальные архитектуры, например, сети с долгой кратковременной памятью (LSTM) [6]. Их главной особенностью является наличие механизма вентилей, который позволяет распознавать события, разделенные временными промежутками с неопределённой продолжительностью и границами.

Модель со сверточной долгой кратковременной памятью — разновидность сетей с долгой кратковременной памятью, адаптированная для анализа временных зависимостей в видео. Благодаря использованию свертки, такой тип сетей требует меньшее количество параметров и лучше распознает пространственную информацию, заложенную в картинке.

3.1.3.5. Пространственно-временной автокодировщик

В [1] авторы объединили подход, описанный в [8] и conv-LSTM сети для надежного поиска временных зависимостей в данных, не теряя преимуществ от использования сверточных слоев.

Для исследования видео на регулярность используется модель нейронной сети, изображенная на рис. 3.1.3

Изначально, каждый кадр входного видео с помощью сверточного кодировщика сжимается в пространственный вектор признаков, который используется временным кодировщиком и раскодировщиком для анализа временных паттернов, далее полученная последовательность разжимается к исходной размерности с помощью сверточного раскоди-

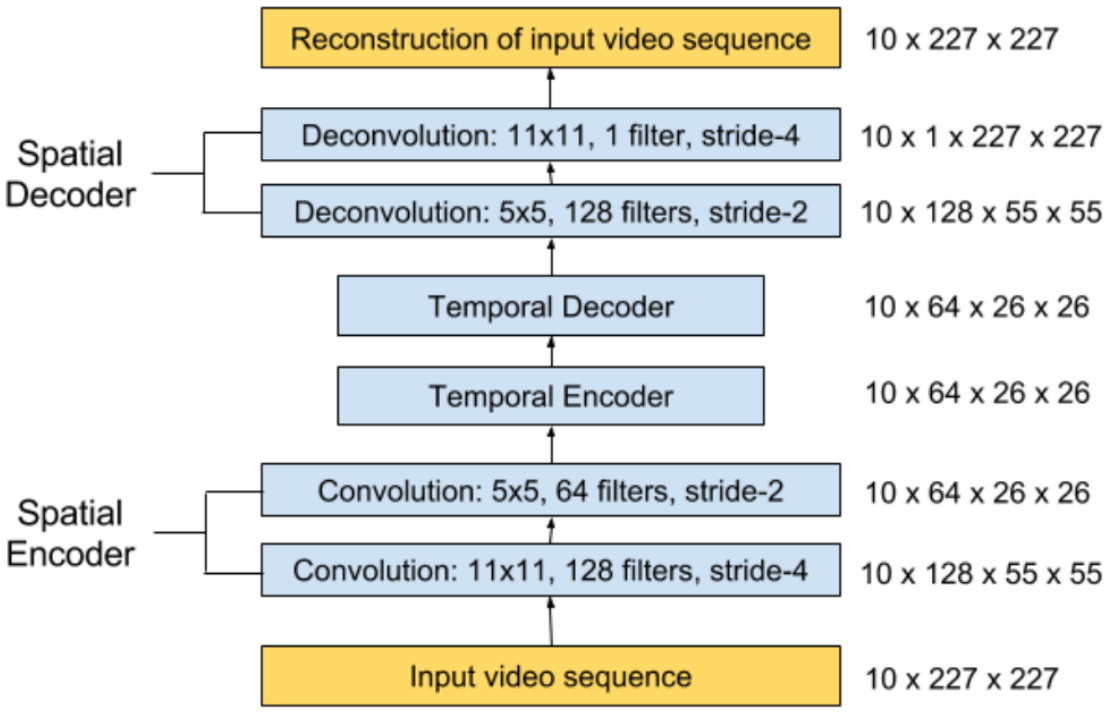


Рис. 3: Модель автокодировщика

ровщика.

При обучении перед градиентным спуском стоит задача минимизировать ошибку, определенную урав. 2, для каждого кадра, содержащего нормальные события, при этом предполагается, что на кадрах с аномалиями ошибка будет высокой.

$$e(t) = \|x(t) - f_W(x(t)\|_2 \tag{1}$$

3.2. Выбор алгоритма

Выбор алгоритма осуществлялся на основе следующих критериев: скорость работы, обобщаемость на различные сцены, точность, надежность.

Подход	Скорость работы	Обобщаемость	Точность	Надежность
HOG+HOF Features	+	-	+	+
Trajectory Features	-	+	+	-
Sparce coding	-	+	+	+
Сверточный автоэнкодер	+	+	-	+
Пространственно-временной автоэнкодер	+	+	+	+

Сравнительный анализ подходов к поиску аномалий в видео пред-

ставлен в таблице. Был выбран подход на основе пространственно-временного автоэнкодера. К реализации был выбран алгоритм из статьи [8]

3.3. Обработка данных

В задачах машинного обучения на качество модели очень сильно влияют данные, поэтому для улучшения результатов применялись шаги, описанные ниже.

Задачей данного этапа является приведение исходных данных к представлению, которое может быть подано на вход нейронной сети в процессе обучения.

Каждый кадр из тренировочного видео сжимается до размера 227x227. Для того, чтобы убедиться, что все картинки обладают одним и тем же масштабом, значение пикселей приводятся к промежутку [0,1]. Для нормализации из каждого кадра вычитается среднее по всему обучающему видео. После этого полученные картинки конвертируются в черно-белое представление. В результате изображения имеют нулевое среднее и единичную дисперсию.

3.3.1. Аугментация данных

Данная модель имеет большое количество параметров, следовательно для ее обучения требуется большое количество тренировочных данных. Так как выбранный подход опирается на то, что в тренировочных данных содержаться только нормальные события, все возможные аномалии необходимо удалить вручную, что требует большого количества человеческих ресурсов, которые зачастую очень сильно ограничены. С целью увеличить количество уже имеющихся тренировочных данных проводится аугментация. На вход модели поступают временные отрезки по 10 кадров, данные отрезки формируются не только из подряд идущих кадров (1, 2, 3, ...), но также из кадров взятых с шагом равным двум (1, 3, 5, ...) и трем.

3.4. Библиотеки для машинного обучения

3.4.1. Tensorflow

TensorFlow — открытая библиотека для машинного обучения. Благодаря гибкой архитектуре позволяет производить вычисления на различных платформах (CPU, GPU, TPU) без внесения изменений в исходный код программы.

3.4.2. Keras

Keras — библиотека-обертка, предоставляющая высокоуровневый API для работы с нейронными сетями, может работать поверх TensorFlow.

3.5. Библиотеки для обработки данных

3.5.1. OpenCV

OpenCV (Open Source Computer Vision Library) — библиотека для работы с компьютерным зрением и обработки видео.

4. Архитектура системы

С учетом заявленных требований была разработана базовая архитектура системы поиска аномалий при видеонаблюдении, которая основана на принципе модульности и реализует паттерн MVC. Архитектура изображена на рис. 4

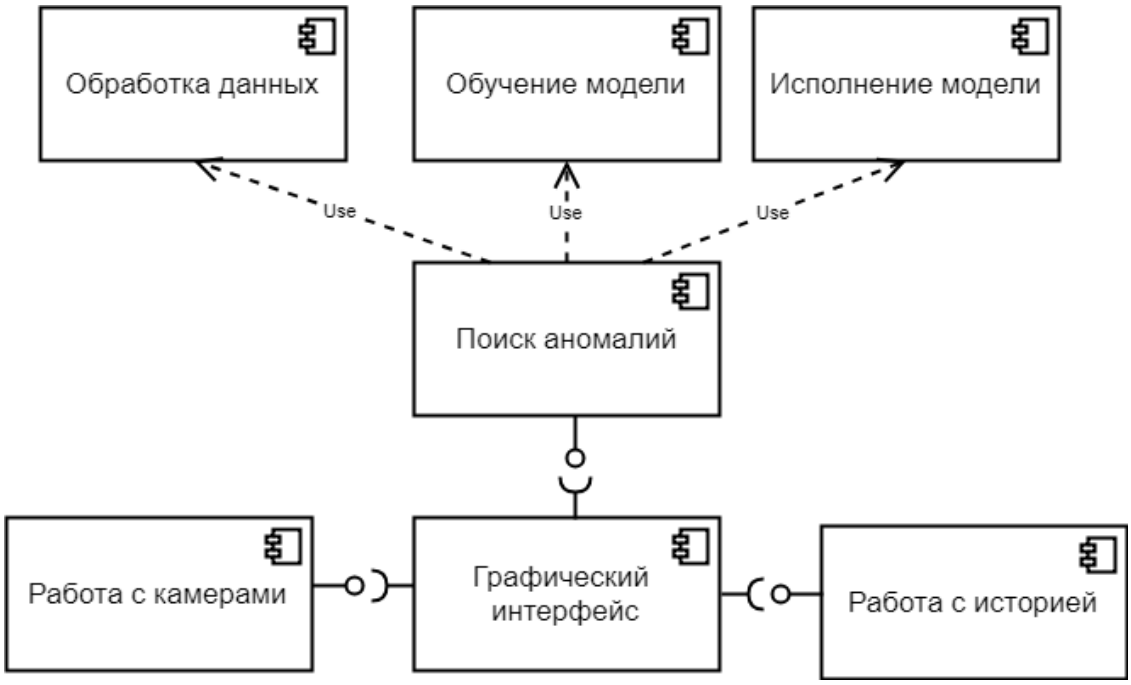


Рис. 4: Диаграмма компонент

Вся система разделяется на четыре основных компонента, которые связаны друг с другом посредством предоставляемых интерфейсов. Далее будет рассмотрен каждый компонент по отдельности.

4.1. Компонент поиска аномалий

Компонент поиска аномалий отвечает за тренировку модели, обработку данных, а также исполнение модели в процессе использования системы.

4.2. Компонент работы с камерами

Необходимым требованием к данной системе является наличие возможности работы с IP-камерами, также в процессе разработки и тести-

рования системы оказалось необходимым использование USB-камер. Данный компонент предоставляет функциональность для подключения к удаленным камерам по IP-адресу, к локальным по номеру, зарегистрированному в системе.

4.2.1. Компонент работы с историей

Важным требованием к системе являлось наличие функциональности для работы с историей выявленных аномалий. В данном компоненте реализованы контроллеры, обеспечивающие чтение и запись истории в базу данных, а также модели, которые используются графическим интерфейсом для отображения данных о выявленных аномалиях в виде таблиц.

4.2.2. Графический интерфейс

Графический интерфейс содержит различные представления для отображения данных о происходящих и произошедших аномалиях, состоянии подключенных камер.

5. Особенности реализации

В данном разделе будет описаны решения, принятые при реализации системы.

Изначально для разработки системы планировалось использовать язык Python, но на этапе разработки графического интерфейса и компонента для работы с камерами появилась необходимость использования параллельных вычислений, что оказалось труднореализуемо с помощью выбранного языка. Поэтому для разработки было решено использовать язык C++, при этом сохранив часть компонентов на Python.

5.1. Реализация алгоритма

Реализация компонента для обучения модели и обработки данных выполнена на языке Python с использованием библиотеки Keras.

Компонент для исполнения модели был разработан на C++ с использованием библиотеки TensorFlow.

5.1.1. Конвертация моделей из формата HDF5 в PB

Обучение модели для поиска аномалий производится с помощью библиотеки Keras, а ее исполнение с помощью библиотеки TensorFlow, так как Keras не имеет C++ API. Поэтому при работе системы требуется конвертировать обученную модель из формата HDF5 (Hierarchical Data Format v.5) [4], используемого Keras, в формат PB (Protocol Buffers) [12], который используется TensorFlow. Для этого была использована утилита [7]

5.1.2. Ускорение обучения нейронных сетей

Так как количество времени необходимого на обучение модели для конкретной камеры является критически важным для скорости развертывания системы, необходимо было сократить время обучения, для этого использовался подход описанный в [8]. Идея состоит в том, чтобы предобучить нейронную сеть, используя открытые наборы данных

(dataset), до развертывания системы. При использовании системы модель необходимо дотренировать на данных с конкретных камер видеонаблюдения. По результатам тестов, данная методика позволяет сократить время обучения модели в 2-3 раза.

5.2. Реализация графического интерфейса

Графический интерфейс разработан на C++ с помощью библиотеки Qt [5]. В качестве средства общения с остальными модулям выступают программные интерфейсы этих модулей и механизм сигналов/слотов библиотеки Qt.

5.2.1. Многопоточность и Qt

Данная система должна анализировать информацию, поступающую с нескольких камер видеонаблюдения, для этого каждый анализатор работает в отдельном потоке и передает графическому интерфейсу проанализированные кадры и информацию о найденных аномалиях с помощью механизма сигналов/слотов. Для безопасной общения между разными потоками в библиотеке Qt используются специальные очереди, в которые сообщения попадают до того как будут обработаны потоком-получателем. При использовании большого количества потоков сообщения из данных очередей обрабатываются неравномерно, что может привести к неустойчивой скорости отображению кадров в графическом интерфейсе, с этой целью с был разработан специальный механизм буферов, которые стабилизируют количество кадров, отображаемых в секунду.

5.3. Реализация работы с историей

Для сохранения информации о найденных аномалиях было решено использовать базу данных SAP HANA, которая подключается к остальной системе по интерфейсу ODBC [10].

5.4. Реализация работы с камерами

Реализация работы с камерами выполнена с помощью библиотеки OpenCV.

5.5. Nvidia-docker

Согласно требованиям система должна быть портативной, поэтому было решено разработать Nvdia-docker [9] образ, который бы включал все необходимые зависимости и данные для работы системы, а также полностью скрывал внутреннее представление. Такой подход требует установки на конечный компьютер только docker-сервера, что значительно сокращает время и сложность развертывания системы.

6. Апробация

Целью апробации алгоритма было проверить, что аномальные события надежно детектируются в общем видеопотоке, и алгоритм удовлетворяет требованиям по быстродействию. Для этого было собраны данные с камер видеонаблюдения в офисе, пример наблюдаемого пространства на рис. 5

Рис. 5: 1,2 - входы с допустимым проходом, 3,4 - "закрытые" входы, движение в сторону 5 - запрещено, 6 - источник помех

Чтобы проверить весь спектр возможностей алгоритма, на наблюдаемую сцену рис.5 было наложено несколько искусственных ограничений, также в тестовую выборку были включены неестественные объекты.

Ошибка реконструкции всех пикселей в в кадре t определяется как Евклидово расстояние 2 между входным и рекоструированным кадром,

$$e(t) = \|x(t) - f_W(x(t)\|_2 \tag{2}$$

где f_W— веса, полученные в результате обучения

.

Определим регулярность как в урав. 3

$$s_r(t) = 1 - \frac{e(t) - e(t)_{min}}{e(t)_{max}} \tag{3}$$

Входе экспериментов было решено, что аномалией будет считаться фрагмент видео со значением метрики 3 больше 0.8

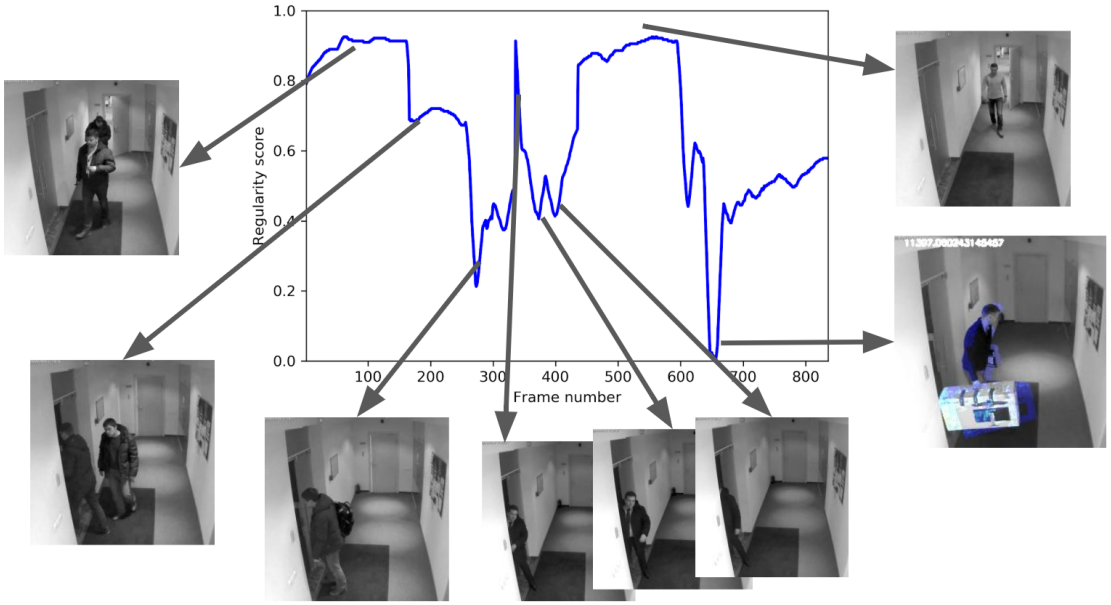


Рис. 6: Regularity score

Как видно из графика 6 алгоритм надежно отделяет аномальные события такие как проход в дверь по одному пропуску двух человек или перемещение необычного объекта от простого перемещения.

В ходе тестирования было вычислено, что система может одновременно анализировать видеопоток с 3 видеокамер со скоростью 15-20 кадров в секунду, что соответствует заявленным требованиям.

Заключение

В ходе выполнения данной выпускной квалификационной работы были достигнуты следующие результаты.

- Разработаны требования к системе поиска аномалий при видеонаблюдении.
- Сделан обзор существующих алгоритмов поиска аномалий в видео.
- Разработана архитектура системы основанная на при принципе модульности.
- Реализован прототип системы, включающий следующие компоненты:
-
  - графический интерфейс
  - компонент для поиска аномалий
  - компонент для работы с историей найденных аномалий
  - компонент для работы с камерами видеонаблюдения
  - компонент для обучения модели на пользовательских данных.
- Проведена апробация системы.

Список литературы

[1] Chong Yong Shean, Tay Yong Haur. Abnormal Event Detection in Videos using Spatiotemporal Autoencoder // CoRR. — 2017. — Vol. abs/1701.01546. — 1701.01546.

[2] Dalal N., Triggs B. Histograms of oriented gradients for human detection // 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05). — Vol. 1. — 2005. — June. — P. 886–893 vol. 1.

[3] Dalal Navneet, Triggs Bill, Schmid Cordelia. Human Detection Using Oriented Histograms of Flow and Appearance // Computer Vision – ECCV 2006 / Ed. by Aleš Leonardis, Horst Bischof, Axel Pinz. — Berlin, Heidelberg : Springer Berlin Heidelberg, 2006. — P. 428–441.

[4] Hierarchical Data Format. — URL: `https://www.hdfgroup.org/solutions/hdf5/` (online; accessed: 30.05.2018).

[5] Hierarchical Data Format. — URL: `https://www.qt.io/` (online; accessed: 30.05.2018).

[6] Hochreiter Sepp, Schmidhuber Jürgen. Long Short-term Memory. — 1997. — 12. — Vol. 9. — P. 1735–80.

[7] Keras to TensorFlow model converter. — 2017. — URL: `https://github.com/bitbionic/keras-to-tensorflow` (online; accessed: 30.05.2018).

[8] Learning Temporal Regularity in Video Sequences / Mahmudul Hasan, Jonghyun Choi, Jan Neumann et al. // CoRR. — 2016. — Vol. abs/1604.04574. — 1604.04574.

[9] Nvidia-docker. — URL: `https://github.com/NVIDIA/nvidia-docker` (online; accessed: 30.05.2018).

[10] Open Database Connectivity. — 2017. — URL: https://en.wikipedia.org/wiki/Open_Database_Connectivity (online; accessed: 30.05.2018).

[11] Piciarelli Claudio, Micheloni Christian, Foresti G.L. Trajectory-Based Anomalous Event Detection. — 2008. — 12. — Vol. 18. — P. 1544 – 1554.

[12] Protocol buffers. — URL: https://developers.google.com/protocol-buffers/docs/overview (online; accessed: 30.05.2018).

[13] Reddy K. R., Priya K. H., Neelima N. Object Detection and Tracking – A Survey // 2015 International Conference on Computational Intelligence and Communication Networks (CICN). — 2015. — Dec. — P. 418–421.

[14] Wang Heng, Schmid Cordelia. Action Recognition with Improved Trajectories // ICCV - IEEE International Conference on Computer Vision. — Sydney, Australia : IEEE, 2013. — . — P. 3551–3558. — URL: https://hal.inria.fr/hal-00873267.

[15] Wikipedia. Features in computer vision. — URL: https://en.wikipedia.org/wiki/Feature_(computer_vision) (online; accessed: 30.05.2018).