**Министерство науки и высшего образования Российской Федерации**
**Федеральное государственное бюджетное образовательное учреждение высшего образования**
**«Московский государственный технический университет имени Н.Э. Баумана**
**(национальный исследовательский университет)»**
**(МГТУ им. Н.Э. Баумана)**

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

# РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

## *К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ*

## *НА ТЕМУ:*

## *«Метод создания связных сцен из художественного видео»*

Студент группы ИУ7-81 _____ Домнин Е.О.
(Подпись, дата) (И.О. Фамилия)

Руководитель ВКР _____ Рудаков И.В.
(Подпись, дата) (И.О. Фамилия)

Нормоконтролер _____ _____
(Подпись, дата) (И.О. Фамилия)

Москва, 2020

**Министерство науки и высшего образования Российской Федерации**
**Федеральное государственное бюджетное образовательное учреждение**
**высшего образования**
**«Московский государственный технический университет имени Н.Э. Баумана**
**(национальный исследовательский университет)»**
**(МГТУ им. Н.Э. Баумана)**

УТВЕРЖДАЮ

Заведующий кафедрой ИУ7
(Индекс)
_____ _Рудаков И.В.
(И.О.Фамилия)
« 06 » 11 2019 г.

# З А Д А Н И Е
## на выполнение выпускной квалификационной работы магистра

Студент группы __ИУ7-43М_____

_____Домнин Егор Олегович_____
(фамилия, имя, отчество)

Тема квалификационной работы _____Метод создания связных сцен из художественного видео_____

_____

_____

Источник тематики (НИР кафедры, заказ организаций и т.п.)

_____НИР кафедры_____

_____

_____

Тема квалификационной работы утверждена распоряжением по факультету _ИУ_____№ _03.02.01-04.03/25_____от « _13__ » __ноября_____ 2019__ г.

***Часть 1. Аналитический раздел*** Анализ предметной области. Анализ современных алгоритмов создания связных сцен из видео. Анализ областей применимости алгоритмов.

***Часть 2. Конструкторский раздел*** Выбор форматов, обрабатываемых видео. Конструирование алгоритма создания связных сцен. Проектирование приложения, реализующего создание связных сцен из видео.

***Часть 3. Технологический раздел*** Обоснование выбора средств программной реализации. Описание функционала приложения, а также его интерфейса.

***Часть 4. Исследовательский раздел*** _Выбор тестовых наборов данных. Формулировка основных критериев сравнения результатов создания связных сцен, сравнительное тестирование реализованных алгоритмов, оценка корректности и скорости работы программного комплекса.

***Оформление квалификационной работы:***

Расчетно-пояснительная записка на __58___ листах формата А4.

Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.)

_____
_____
_____
_____
_____
_____

Дата выдачи задания « _10__ » ____09_____ 2019__ г.

В соответствии с учебным планом выпускную квалификационную работу выполнить в полном объеме в срок до « __25__ » _____05_____ 2020____ г.

**Руководитель квалификационной работы** _____ ____Рудаков И.В.____
(Подпись, дата) (И.О.Фамилия)

**Студент** _____ ____Домнин Е.О._____
(Подпись, дата) (И.О.Фамилия)

**Министерство науки и высшего образования Российской Федерации**
**Федеральное государственное бюджетное образовательное учреждение высшего образования**
**«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»**
**(МГТУ им. Н.Э. Баумана)**

**ФАКУЛЬТЕТ ИУ** УТВЕРЖДАЮ

**КАФЕДРА ИУ7** Заведующий кафедрой ИУ7
(Индекс)

**ГРУППА ___ИУ7-43М_____** _____ Рудаков И.В.
(И.О.Фамилия)

« 06 » 11 2019 г.

# КАЛЕНДАРНЫЙ ПЛАН
**выполнения выпускной квалификационной работы** студента:
_____Домнина Егора Олеговича_____
(фамилия, имя, отчество)

Тема квалификационной работы ____ Метод создания связных сцен из художественного видео

| № п/п | Наименование этапов выпускной квалификационной работы | Сроки выполнения этапов | | Отметка о выполнении | |
|---|---|---|---|---|---|
| | | план | факт | Должность | ФИО, подпись |
| 1. | Задание на выполнение работы. Формулирование проблемы, цели и задач работы | *24.10.2019* *Планируемая дата* | *24.10.2019* | Руководитель ВКР | |
| 2. | 1 часть Аналитический раздел | *24.12.2019* *Планируемая дата* | *24.12.2019* | Руководитель ВКР | |
| 3. | Утверждение окончательных формулировок решаемой проблемы, цели работы и перечня задач | *24.12.2019* *Планируемая дата* | *24.12.2019* | Заведующий кафедрой | |
| 4. | 2 часть Конструкторский раздел | *06.04.2020* *Планируемая дата* | *06.04.2020* | Руководитель ВКР | |
| 5. | 3,4 части Технологический и Исследовательский разделы | *18.05.2020* *Планируемая дата* | *18.05.2020* | Руководитель ВКР | |
| 6. | 1-я редакция работы | *20.05.2020* *Планируемая дата* | *20.05.2020* | Руководитель ВКР | |
| 7. | Подготовка доклада и презентации | *20.05.2020* *Планируемая дата* | *20.05.2020* | | |
| 8. | Заключение руководителя | *24.05.2020* *Планируемая дата* | *24.05.2020* | Руководитель ВКР | |
| 9. | Допуск работы к защите на ГЭК (нормоконтроль) | *27.05.2020* *Планируемая дата* | *22.06.2020* | Нормоконтролер | |
| 10. | Внешняя рецензия | *19.06.2020* *Планируемая дата* | *19.06.2020* | | |
| 11. | Защита работы на ГЭК | *29.06.2020* *Планируемая дата* | *29.06.2020* | | |

*Студент* *Домнин Е.О.* 10.09.2019 *Руководитель работы* *Рудаков И.В.* *10.09.2019*

# РЕФЕРАТ

Расчетно-пояснительная записка 54 с., 25 рис., 1 табл., 18 источников, 1 прил.

СОЗДАНИЕ СВЯЗНЫХ СЦЕН, ВРЕМЕННОЕ РАЗБИЕНИЕ, КЛАСТЕРИЗАЦИЯ, КОМПЬЮТЕРНОЕ ЗРЕНИЕ, НЕЙРОННЫЕ СЕТИ

Объектом разработки является программное обеспечение для создания связных сцен из художественного видео.

Цель работы — разработка метода создания связных сцен из художественного видео.

Поставленная цель достигается за счет анализа существующих алгоритмов и разработки нового метода на основе существующих.

Результатом работы является разработанный метод создания связных сцен из художественного видео, основной особенностью которого является новый подход к решению подзадачи кластеризации планов.

Программная реализация выполнена на языке Python версии 3.8 с использованием библиотек Keras, Tensorflow, OpenCV, Scipy и Numpy.

В качестве основных направлений для дальнейших исследований предлагается несколько вариантов: улучшение метода решения подзадачи поиска количества сцен в видео и улучшение метода решения задачи выделения особенностей из планов.

# СОДЕРЖАНИЕ

## Определения, обозначения и сокращения

**HSV –** Hue Saturation Value

**SVD** – Singular Value Decomposition

**VSD** – Video Shot Detection

**SBD** – Scene Boundary Detection

**CNN** – Convolutional Neural Network

**API -** application programming interface

# ВВЕДЕНИЕ

В данной работе производится разработка метода, решающего задачу создания связных сцен из художественного видео. Данная задача является промежуточной задачей анализа видео, однако может использоваться и отдельно для упрощения навигации по длительным видео.

В последние годы передача видео стала одним из основных потребителей интернет траффика и доступность потребления видео привела к большему его производству и расширению предоставления некоторых его подвидов, таких как обучающие видео. Также это увеличило вероятность повторного использования видео и потребности в навигации по имеющимся. Структуризация видео упрощает навигацию по нему и соответственно упрощает его повторное использование что обуславливает актуальность данной задачи.

Результатом работы метода является разбиение видео на относительно короткие фрагменты, анализ которых проще и позволяет извлечь ключевые моменты видео. Данный метод основывается на группе методов, решающих поставленную задачу через построение матрицы расстояний планов и кластеризацию планов в сцены на её основе. Идея модификации заключается в кластеризации планов с помощью поиска сцен на визуализации матрицы расстояний планов по аналогии с поиском лиц на изображении с помощью построения тепловой карты ограничивающих прямоугольников.

# 1 Аналитический раздел

## 1.1 Входные данные

На вход разрабатываемого метода подаётся цифровое видео. Цифровое видео — это последовательность кадров и соответствующая им аудиодорожка. Для извлечения полезной для дальнейшего анализа информации разрешения кадров должно быть не менее 299 пикселей по высоте и ширине [1], для извлечения полезной информации из аудиодорожки её разрешение должно быть не менее 64 килобит в секунду [1]. Как правило, цифровое видео подвергается компрессии для уменьшения занимаемого объёма, однако тип компрессии не является важным для разрабатываемого метода по причине того, что любой из них позволяет извлечь из видео составляющие его кадры и произвольные отрезки аудиодорожки.

Художественное видео имеет иерархическую структуру: кадры состоят из пикселей, планы из кадров, сцены из планов, а видео из сцен.

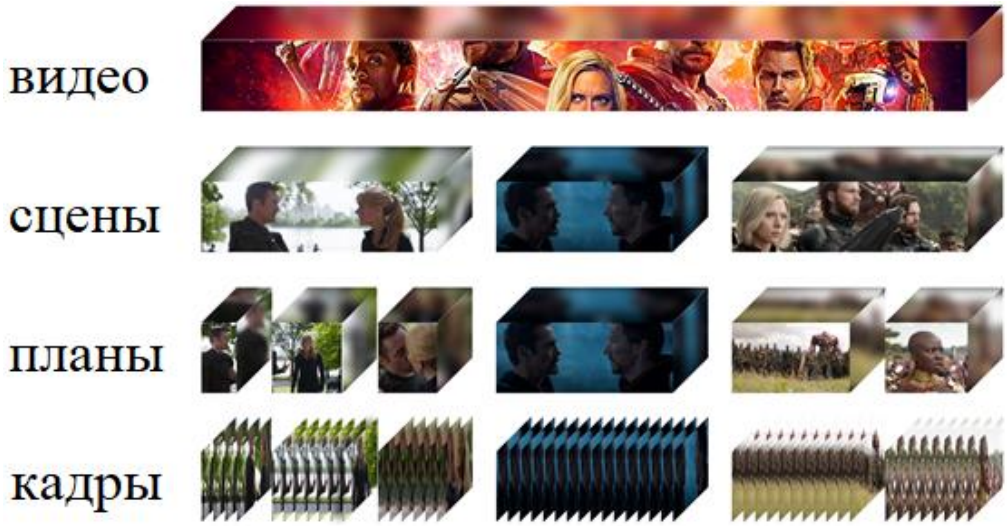



Рисунок 1 — Представление иерархической структуры видео

Кадр – статическая картинка. План – последовательность кадров от одной монтажной склейки до другой, то есть группа кадров, непрерывно снятая с одной камеры в случае художественного, не анимационного, видео. Сцена – по-

следовательность связных планов, то есть планов, характеризующихся единством места, времени и персонажей.

Соответственно предоставленному ранее описанию и теме работы в качестве входного видео рассматривается только класс художественных видео, наиболее важными представителями которого являются фильмы, сериалы и телепередачи. Потоковые видео не подходят для анализа так как не имеют монтажных склеек и их длина заранее не известна. Бытовые видео, снятые единым куском без монтажа, также не подходят.

## 1.2 Постановка задачи

Цель работы — разработка метода создания связных сцен из художественного видео. Необходимо сформировать набор тестовых данных для проверки корректности работы ПО, реализующего метод, а также определить параметры, на основе которых можно будет оценивать результаты работы.

## 1.3 Оценка результата работы метода

Как было сказано ранее видео имеет иерархическую структуру и потому задача создания связных сцен является родственной задаче кластеризации. Для оценки результатов работы метода кластеризации используют различные метрики, которые оценивают степень соответствия между образцовым(истинным) разбиением на кластеры и разбиением, построенным алгоритмически.

Обсуждение метрик стоит предварить обсуждением принципов, которым эти метрики должны удовлетворять. В [2] очень подробно разбирается ряд принципов, их корректность и то, как различные классы метрик удовлетворяют этим принципам.

Будем считать, что метрика может принимать диапазон значений от 0 до 1, где 1 – означает идеальное разбиение, а 0 – худшее возможное.

Приведём четыре наиболее важных принципа:

1. Однородность кластеров

   Значение метрики качества должно уменьшаться при объединении в один кластер двух эталонных. Пример проиллюстрирован на рисунке 2.

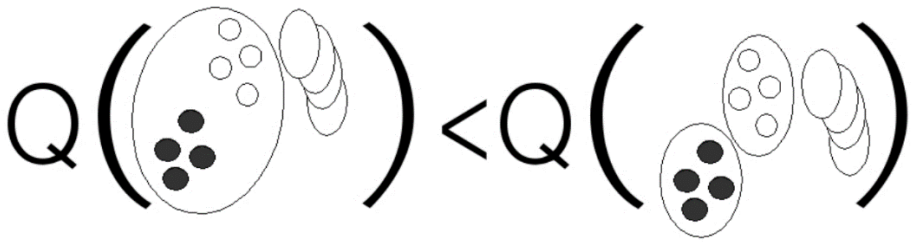


Рисунок 2

2. Полнота кластеров

   Это свойство, двойственное свойству однородности. Значение метрики должно уменьшаться при разделении эталонного кластера на части. Пример проиллюстрирован на рисунке 3.

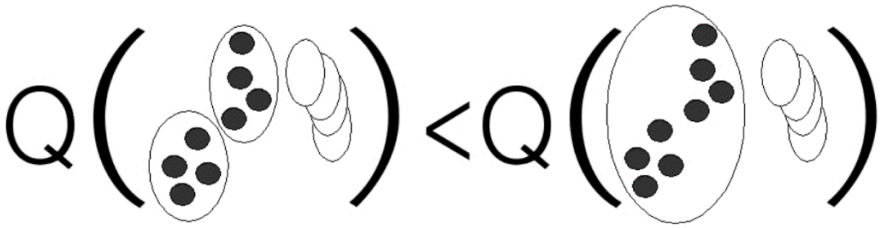
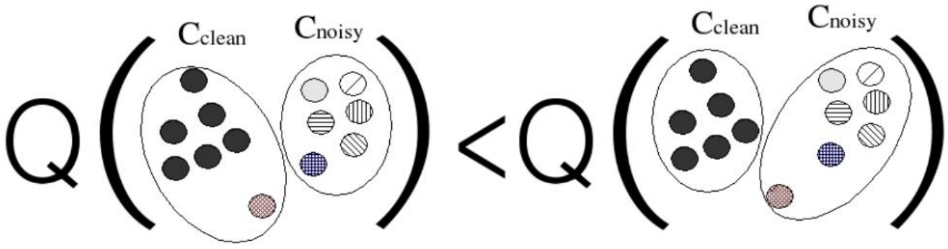


Рисунок 3

3. Кластер останков

   Пусть есть два кластера: чистый, содержащий элементы из одного эталонного кластера, и шумный («кластер останков»), где собраны элементы из большого числа различных эталонных кластеров. Тогда значение метрики должно быть выше у той версии кластеризации, которая помещает новый нерелевантный обоим кластерам элемент в шумный кластер, по сравнению с версией, которая помещает этот элемент в чистый кластер. Пример проиллюстрирован на рисунке 4.




$C_{clean}$ - чистый кластер, $C_{noisy}$ - шумный кластер

Рисунок 4

4. Размер важнее количества

   Значительное ухудшение кластеризации большого числа небольших кластеров должно сильнее занижать метрику, чем небольшое ухудшение кластеризации в крупном кластере. Пример проиллюстрирован на рисунке 5.

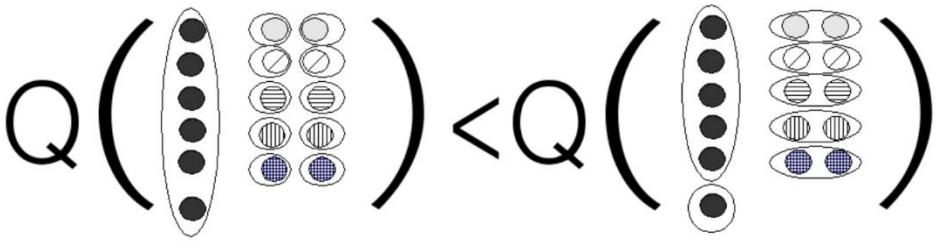



Рисунок 5

В имеющихся работах для оценки результата работы разработанных методов использовались метрики Coverage [3], Overflow [3], Purity [4], а также сочетание метрик Coverage и Overflow с использованием F-меры [5].

Рассмотрим названные ранее метрики:

1. Метрика Coverage измеряет наибольшее пересечение между кластерами идеального и сгенерированного разбиения, где каждому кластеру оригинального разбиения сопоставляется один сгенерированный.

   Для её расчёта используется следующая формула:

$$C\left(L_{new}, L_{orig}\right) = \frac{\sum^{n} maxintersec\left(L_{new}, L_{orig_i}\right)}{\sum^{n} len\left(L_{orig_i}\right)} \quad (1)$$

, где $L_{orig}$ это идеальное разбиение, $L_{new}$ это новое разбиение, $maxintersec(A, B)$ это количество элементов наибольшего пересечения кластера из A с кластером из разбиения B, а $len(A)$ количество элементов кластера A.

Проиллюстрируем как данная метрика выглядит в контексте задачи создания связных сцен на рисунке 6.

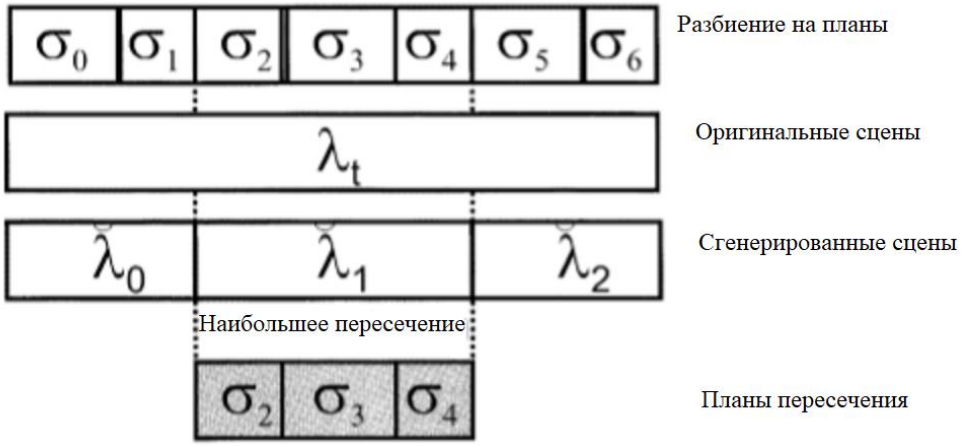



Рисунок 6

2. Метрика Overflow измеряет минимальное количество элементов, не попадающих в пересечение между кластерами идеального и сгенерированного разбиения, где каждому кластеру оригинального разбиения сопоставляется один сгенерированный.

   Для её расчёта используется следующая формула:

$$O\left(L_{new}, L_{orig}\right) = 1 - \frac{\sum^{n} overflow\left(L_{new}, L_{orig_i}\right)}{\sum^{n} len\left(L_{orig_i}\right)} \quad (2)$$

, где $L_{orig}$ это идеальное разбиение, $L_{new}$ это новое разбиение, $overflow(A, B)$ это минимальное количество элементов не попада-

ющих в пересечение кластера из A с кластером из разбиения B, а $len(A)$ количество элементов кластера A.

Проиллюстрируем как данная метрика выглядит в контексте задачи создания связных сцен на рисунке 7.

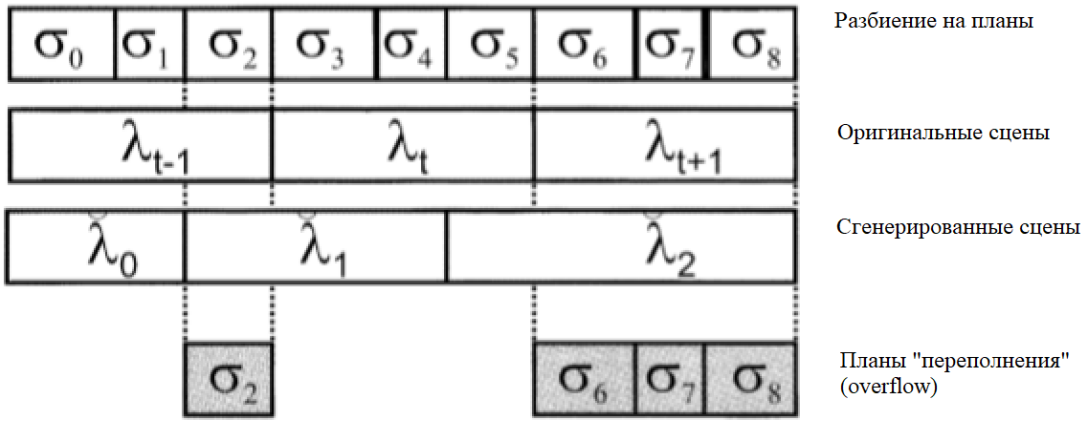



Рисунок 7

3. Метрика Purity означает насколько хороший эталон найдётся для сгенерированного кластера.

   Для её расчёта используется следующая формула:

$$Purity = \left( \sum_{i=1}^{N_g} \frac{\tau(s_i)}{T} \sum_{j=1}^{N_a} \frac{\tau^2(s_i, s_j^*)}{\tau^2(s_i)} \right) \cdot \left( \sum_{j=1}^{N_a} \frac{\tau(s_j^*)}{T} \sum_{i=1}^{N_g} \frac{\tau^2(s_i, s_j^*)}{\tau^2(s_j^*)} \right) \quad (3)$$

   , где T – общее количество элементов, $\tau$ – количество общих элементов между двумя кластерами, $s^*$- кластер из сгенерированного разбиения, $s$ – кластер из оригинального разбиения, $N_g$- число кластеров оригинального разбиения, $N_a$- число кластеров сгенерированного разбиения.

4. F-мера используется для объединения значений нескольких метрик в один параметр, в контексте данной задачи для оценки результатов работы с помощью F-меры объединяли метрики Coverage и Overflow.

Для её расчёта используется следующая формула:

$$F_\beta = \left(1 + \beta^2\right) \cdot \frac{\text{coverage} \cdot \text{overflow}}{\left(\beta^2 \cdot \text{coverage}\right) + \text{overflow}} \quad (4)$$

В формуле (4) $\beta$ определяет вес точности в метрике, и при значении равном 1 это среднее гармоническое, но мы будем использовать $\beta = 2$ чтобы при coverage=overflow=1 значение F-меры тоже было равно 1 для унификации метрик, как было предложено в [5].

## 1.4 Обзор существующих методов

В [6] проводится обзор существующих методов решения задачи создания связных сцен из видео на 2013й год, также в [7] и [8] описаны новые методы решения и предложены идеи для их улучшений.

Сцена определяется как набор последовательно идущих планов, которые характеризуются похожим аудиовизуальным составляющим.

Согласно [6] методы решения задачи создания связных сцен из видео могут быть классифицированы двумя способами. Первый способ классификации – по способу получения особенностей из видео, в таком случае есть 7 классов соответственно особенностям: визуальные, аудио, текстовые, аудиовизуальные, визуально-текстовые, аудио-текстовые и гибридные как проиллюстрировано на рисунке 8. В [9] показано что наибольший вклад оказывают визуальные особенности, аудио отдельно малопригодны к использованию, как и текстовые, а лучшие результаты даёт использование аудиовизуальных и гибридных особенностей. Следовательно, далее будут рассматриваться только методы использу-

ющие визуальные особенности, и отдельно, и в сочетании с другими.


Рисунок 8 Классификация методов по извлекаемым особенностям

Все существующие методы делят видео на планы, извлекают из них особенности и кластеризуют их. Так можно декомпозировать задачу создания связных сцен на три задачи: выделения планов из видео, получения особенностей из планов и кластеризация планов в сцены. Можно выделить пять подходов: методы, основанные на правилах, методы, основанные на графах, стохастические методы, методы, методы динамической оптимизации и методы модифицированной кластеризации.

Рассмотрим далее детальнее каждый из этих классов:

### 1.4.1 Методы, основанные на правилах

Данные методы основываются на определённых правилах, используемых при создании студийных видео, таких как фильмы, тв-передачи и сериалы. Проблемы возникают, когда две последовательные сцены похожи и следуют одним и тем же правилам. Также данные алгоритмы показывают неудовлетво-

рительные результаты в случае, когда режиссёр целенаправленно не соблюдает общепринятые правила съёмки

1. Правило 180 градусов – все камеры участвующие в сцене расположены на одной стороне от воображаемой линии, откуда они снимают планы с общим фоном
2. Правило направления действия – направление движения камер должно совпадать в двух последовательных планах снимающих движение актёров
3. Правило ритма в сцене – число планов, частота звуков и скорость движений в них определяет ритм сцены, он не должен меняться. Обычно быстрый ритм означает экшн-сцену.
4. Правило обратного плана – сцена может состоять из нескольких противоположных планов, классический пример – диалог между двумя персонажами, где планы поочерёдно показывают персонажей
5. Правило обзора и разбиения – сначала происходит обзор сцены где показывают локацию, персонажей и объекты в неё вовлечённые, после чего показывается серия планов, отображающих эти элементы детальнее. Данное правило часто сочетается с правилом обратного плана

### 1.4.2 Методы, основанные на графах

Один из первых, применённых к решению поставленной задачи подходов, что логично, учитывая иерархическую структуру видео. Листьями являются планы, узлами группы планов, ветви показывают схожесть. С помощью методов сегментации графов возможно получить подграфы, являющиеся сценами.

Данное семейство методов лучше всего подходит для видео, содержащих повторяющиеся типы сцен, например, тв-шоу или новостные передачи. К художественным фильмам оно хуже применимо, из-за большего разнообразия

планов в таком случае происходит излишнее дробление сцен, особенно в случае экшн-сцен.

### 1.4.3 Стохастические методы

Алгоритмы, основанные на стохастических методах, представляют проблему определения границ сцен с помощью стохастических моделей. Оптимальное решение аппроксимируется максимизируя апостериорную вероятность того что предсказанные границы будут истинными. С таким подходом может быть достигнута высокая точность, однако для создания стохастических моделей требуется большое количество данных для создания обучающих наборов данных.

Выбор обучающих наборов данных критичен. Так в случае если видео сильно отличается от тех что были в обучающем наборе. То результаты окажутся плохими. Так каждая стохастическая модель подходит для определённого класса видео, соответствующего обучающему набору для модели.

### 1.4.4 Методы динамической оптимизации

Методы динамической оптимизации рассматривают задачу создания связных сцен из видео как общую проблему оптимизации. Общий алгоритм таков:

1. Получение особенностей из N планов
2. Построение матрицы расстояний D между планами с помощью функции $D(x_i, x_j)$ где $x_j$ и $x_i$это вектора особенностей i-го и j-того плана
3. Производится оценка вероятного количества сцен K
4. Минимизируется функции стоимости $H_D^K$

Результаты работы методов данного семейства сильно зависят от выбранной функции стоимости. Так аддитивная функция стоимости имеет сложность $O(NK)$ [5], тогда как функция нормальной стоимости $O(NKN^2)$ [7]. Проблема в том, что функции стоимости, кроме функции нормальной стоимости имеют разную вероятность к дроблению видео в разных его частях.

Идеальная функция стоимости должна иметь два свойства:

1. Если видео состоит из одной сцены, то где бы мы не пытались разбить его на две части значение функции стоимости будет одинаковым
2. При правильном разбиении на сцены значение будет меньше, чем при неправильном

Всего на данный момент были предложены три вида функций стоимости:

$$H_{add}(t) = \sum_{\text{для каждой сцены}}[\text{сумма растояний внутри сцены}] \quad (5)$$

$$H_{avg}(t) = \sum_{\text{для каждой сцены}}[\text{среднее растояние внутри сцены}] \quad (6)$$

$$H_{norm}(t) = \sum_{\text{для каждой сцены}}[\text{сумма растояний внутри сцены}] \quad (7)$$

Особенности каждой функции можно наглядно проиллюстрировать с помощью матрицы расстояний, заполненной равномерным шумом, см рисунок 9.

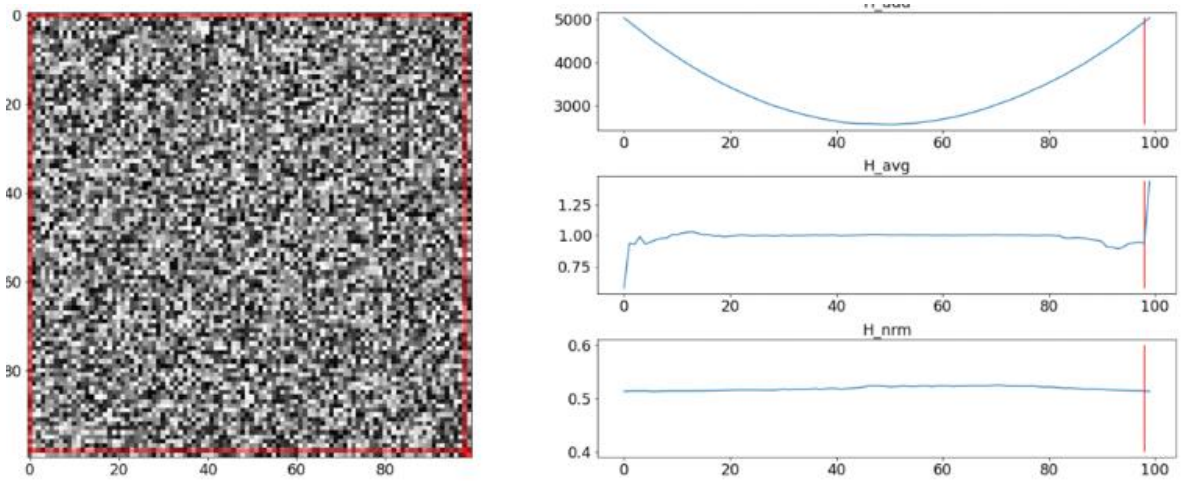



Рисунок 9 Различные функции стоимости на матрице равномерного шума

### 1.4.5 Методы модифицированной кластеризации

Методы модифицированной кластеризации рассматривают задачу создания связных сцен из видео как задачу кластеризации с дополнительными ограничениями [5,10]. Общий алгоритм таков:

1. Получение особенностей из N планов
2. Построение матрицы расстояний D между планами с помощью функции $D(x_i, x_j)$ где $x_j$ и $x_i$это вектора особенностей i-го и j-того плана
3. Производится оценка вероятного количества сцен K. Необязательный этап для методов кластеризации не требующих на вход количество кластеров
4. Кластеризация с дополнительным условием что объединять можно только соседние элементы

### 1.4.6 Возможность работы с видео по частям

Сложность некоторых методов нелинейно зависит от длины видео, что порождает вопрос возможности создания сцен из отдельных частей видео, а потом их объединения, однако это очевидно вызывает проблемы с большими сценами, которые оказываются поделены между несколькими частями. Эта модификация предлагалась в [5,7], но успешной реализации на данный момент не имеет.

### 1.4.7 Выводы

Основываясь на [5,6,7,8,9] можно сделать выводы что наиболее перспективным подходом являются методы динамической оптимизации и модифици-

рованной кластеризации. Первые три этапа методов аналогичны, ключевым отличием является способ кластеризации, получающий на вход матрицу расстояний план и создающий на её основе кластеры – сцены.

Таким образом можно декомпозировать задачу на пять подзадач:

1. Разбиение видео на планы
2. Извлечение особенностей из N планов
3. Построение матрицы расстояний D между планами
4. Оценка вероятного количества сцен K
5. Создание K сцен из матрицы D

Рассмотрим каждую из них детальнее.

## 1.5 Разбиение видео на планы

Задача разбиения на планы заметно проще разбиения на сцены так как у планов есть чёткие визуально и аудио различимые границы так как планы разделены монтажными переходами, монтажного перехода внутри плана быть не может по определению плана – непрерывная съёмка с одной камеры. Монтажные склейки бывают двух видов: резкая и плавная. Резкая склейка называется склейкой встык, плавных переходов много видов.

Считается что имеющиеся на данный момент методы определения планов в видео способны определять их с такой величиной ошибки что их можно использовать для дальнейшего анализа [1,5,11]. Основной задачей разработки новых методов в данном направлении является ускорение скорости обработки видео. В связи с тем, что имеющиеся методы определяют планы так что их можно использовать при дальнейшем анализе выберем один из современных методов, приведённый в статье [11], он определяет планы с использованием нейросетей что позволяет уменьшит время выполнения с помощью многоядерных процессоров и современных видеокарт. Его детальный разбор будет приведён в конструкторской части.

## 1.6 Извлечение особенностей из планов

Как было уже сказано, есть три вида особенностей, которые мы можем извлечь из плана: визуальные, аудио и текстовые.

В случае визуальных особенностей есть два разных возможных подхода: усреднение значений из всех кадров плана или определение ключевых кадров.

### 1.6.1 Извлечение ключевых кадров

Выбор правильного количества ключевых кадров важен потому что они должны быть максимально репрезентативными для плана, и оно может зависеть от типа плана, так статичный план с небольшим количеством движений можно достаточно достоверно представить с помощью одного кадра из его середины, а в случае с большим количеством передвижений на один план может потребоваться несколько ключевых кадров. В случае если для представления плана выбирается несколько ключевых кадров из каждого из них получаются особенности, значение которых затем усредняется и является представлением особенностей плана. Существует несколько разных подходов к их выбору, приведём несколько из них:

1. Усреднение данных между всеми кадрами в плане. Такой подход применим в некоторых случаях [10], однако в полученных данных получается много шума.
2. Выбор среднего кадра. Самый простой способ выбора и соответственно самый быстрый, однако наименее репрезентативный, но согласно [5,6,7] такой метод даёт результаты достаточные для дальнейшего анализа.
3. Выбор кадров с наибольшим количеством движений. Возможно на основе отличия кадров друг от друга выбрать те в которых отличия относительно соседних сильнее всего. Минусы такого подхода –

большие вычислительные затраты и вероятно смазанные кадры, на которых тяжело определять объекты [5].

4. Выбор наиболее отличающихся от всех остальных кадров в плане. Такой подход использовался в [12], он выбирает в качестве начального кадра средний, а далее сравнивает его со всеми остальными кадрами и если находит такой что он отличается от уже выбранных более чем на установленное пороговое значение, то он добавляется в список ключевых кадров. Данный подход позволяет получить из плана больше данных, чем выбор просто среднего кадра, имеет меньше шума и согласно [8,12] применим при использовании цветовых гистограмм в качестве извлечённых особенностей для сравнения, однако в случае использования нейронных сетей для извлечения особенностей его вычислительная сложность сильно возрастает.

### 1.6.2 Извлечение визуальных особенностей

В прошлом разделе было рассмотрено получение ключевых кадров из плана, в данном будет представлен обзор способов получения особенностей из кадра. Есть три ключевых подхода

1. Извлечение низкоуровневой информации кадра. К данному подходу относятся цветовые гистограммы, определение насыщенности, контрастности. Наиболее распространённым методом получения особенностей этого семейства являются HSV гистограммы. В цветовом пространстве HSV цвет точки кодируется тремя параметрами: цветовым тоном, насыщенностью и яркостью. Широко применялось ранее, но сейчас считается устаревшим [6].
2. Извлечение высокоуровневой информации. На кадрах возможно определять объекты, например, актёров, место действия, время суток, происходящее действие и различные окружающие предметы.

Чаще всего данный подход используют как дополнительный в случае методов, использующих графы, в иных случаях этих данных недостаточно [6].

3. Извлечение неявной информации с помощью нейронных сетей. В последние годы [7,8,9] всё более распространённым становится подход, в котором используются нейросети обнаруживающие объекты на изображении или классифицирующие его, однако вместо использования прямых выходных данных как в случае извлечения высокоуровневой информации в качестве особенностей берут данные с предпоследнего слоя нейронной сети что позволяет получить неявную информацию о содержимом кадра. Таким образом возможно извлечение высокоуровневой семантической информации, которая закодирована в кадре, однако её невозможно словесно описать. Наиболее распространённым подходом является использование для данной цели нейросетей обученных на таких банках фотографий как ImageNet, а наиболее распространённые семейства нейросетей для данной задачи это AlexNet, VGG, Inception и ResNet.

### 1.6.3 Извлечение аудио особенностей

Существует несколько классов аудиальных особенностей. На данный момент большинство из них определяются с помощью нейросетей которым на вход подаются аудиодорожка длинной от 10мс. Перечислим несколько видов, которые могут использоваться в качестве особенностей планов [6,9]:

1. Количество речи
2. Средняя громкость
3. Класс фонового шума. Как правило, если фоновый шум меняется, то происходит и смена сцены.

4. Классификация говорящих. Помогает определить смену персонажей, участвующих в действии.

Как правило, определение аудио особенностей менее затратное, чем определение визуальных. Однако использование их отдельно даёт заметно худшие результаты чем отдельное использование визуальных особенностей. Рекомендуется использование совместно с визуальными, это позволяет улучшить результаты создания связных сцен.

### 1.6.4 Извлечение текстовых особенностей

Иногда видео имеют связанные с собой текстовые метаданные, такие как субтитры или сценарий. Также существуют способы получения текстовых описаний видео\кадров, однако на данный момент их использование не даёт оптимистичных результатов. Имеющие у видео текстовые особенности также возможно использовать для получения особенностей планов, однако в связи с тем, что они присутствуют далеко не у каждого видео, а при наличии оказывают незначительное влияние на результат [6] создания связных сцен в данной работе мы не будем рассматривать способы извлечения и применения текстовых особенностей.

## 1.7 Построение матрицы расстояний между планами

Матрицей расстояний между планами называют матрицу, которая отражает сравнения каждого плана в видео с каждый. Очевидно, что каждая такая матрица положительная, квадратная, симметричная, так как порядок планов в сравнении не важен, и все её элементы на диагонали равны 1 так как каждый план полностью соответствует самому себе. Считая, что мы имеем на входе, массив планов и их особенностей необходимо выбрать функцию определяющую степень различности двух планов на основе их особенностей.

Важно учесть возможность наличия особенностей разных видов, например, аудио и визуальных. Есть два известных способа объединения особенностей:

1. Создание отдельных матриц расстояний и их последующие объединение. Данный способ сохраняет в некоторой мере различия, появившиеся в каждой модальности, хотя они и усредняются в результирующей матрице [7]. Объединение матриц производится путём их сложения и нормализации.
2. Объединение всех особенностей в единый вектор, на основе которого далее строится матрица расстояний [5]. Недостатком данного способа является невозможность придать различный вес различным модальностям, в отличие от первого способа где это возможно с помощью задания весовых коэффициентов при объединении матриц.

Рассмотрим наиболее распространённые способы измерения расстояния:

1. Различные математические функции [5,6,8,9]
2. Сиамские нейронные сети [13]
3. Нейронные сети с использованием Triplet Loss [14] [15]

## 1.8 Оценка вероятного количества сцен

Известное число сцен позволяет применять большее число методов кластеризации так как становится известно число кластеров. Также оно необходимо для методов динамической оптимизации [5,7]. Так как матрица расстояний планов аналогична классической матрице различий в задаче кластеризации к ней применимы классические методы определения количества кластеров, наиболее эффективным из которых является анализ разрывов [16], который учитывает влияние количества кластеров на расстояние между элементами внутри кластера.

Однако также был разработан метод специально для задачи создания связных сцен. Он основывается на том что в случае идеальной матрицы расстояния D (см. рис 10) между некоторыми строками есть линейная зависимость. В случае идеальной матрицы её ранг является числом сцен в видео. На практике всегда есть шум, однако мы можем создать другую матрицу, являющуюся низкоранговым приближением к существующей, используя SVD [17] мы можем выбрать наиболее близкое приближение.

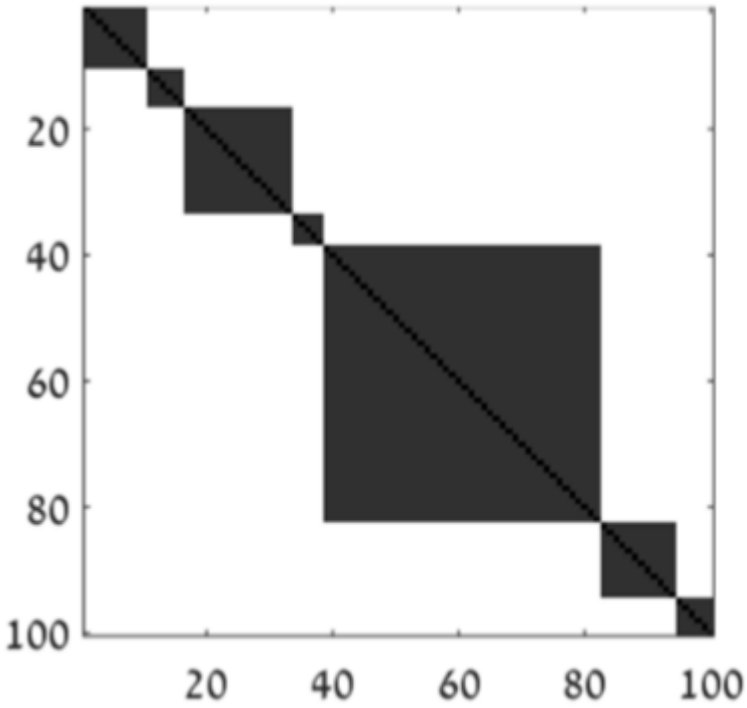



Рисунок 10 Пример идеальной матрицы расстояния между планами

Для выбора наиболее близкого приближения строится граф зависимости расстояний внутри кластеров в зависимости от их количества. На рисунке 11 визуализирован выбор количества кластеров, оно находится в месте перелома, “elbow index”.

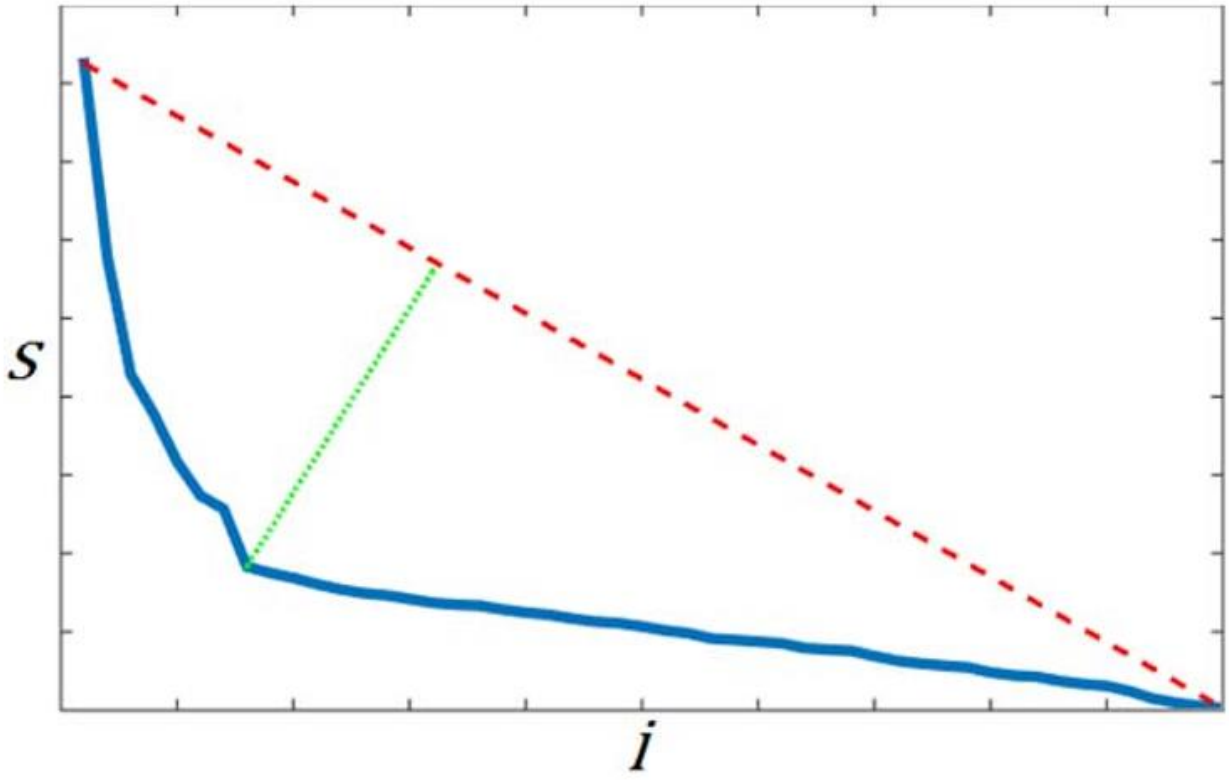



Рисунок 11 Визуализация поиска точки перелома, где s – количество особенных точек по методу SVD, а I – количество кластеров.

Так как число особенных точек уменьшается по экспоненте посчитаем десятичный логорифм от тех чьё значение выше 1. Построим такой граф и найдём точку, наиболее удалённую от линии, соединяющей его начало и конец. Если диагональ графа описывается как $H \triangleq [m-1,\ s_m - s_1]^T$ , то точка перелома находится как

$$\text{elbow\_index} = \underset{i}{\text{argmax}} \left\{ I_i - \frac{I_i^T H}{\|H\|} \cdot \frac{H}{\|H\|} \right\} \qquad (8)$$

При оценке результатов работы данного метода была замечена сильная корреляции точности результатов в зависимости от общего количества планов [7] что естественно.

# 2 Конструкторский раздел

В данном разделе приведено детальное описание разработанного метода, алгоритмов необходимых для его реализации, а также рассматривается архитектура разрабатываемого программного обеспечения.

## 2.1 Общая структура метода

На основе анализа, проведённого в аналитической части разбора существующих методов решения поставленной задачи было решено использовать метод динамической оптимизации с аддитивной функцией.

Общая структура подобных методов была описана в аналитическом разделе, проиллюстрируем её на рис 12 и 13.

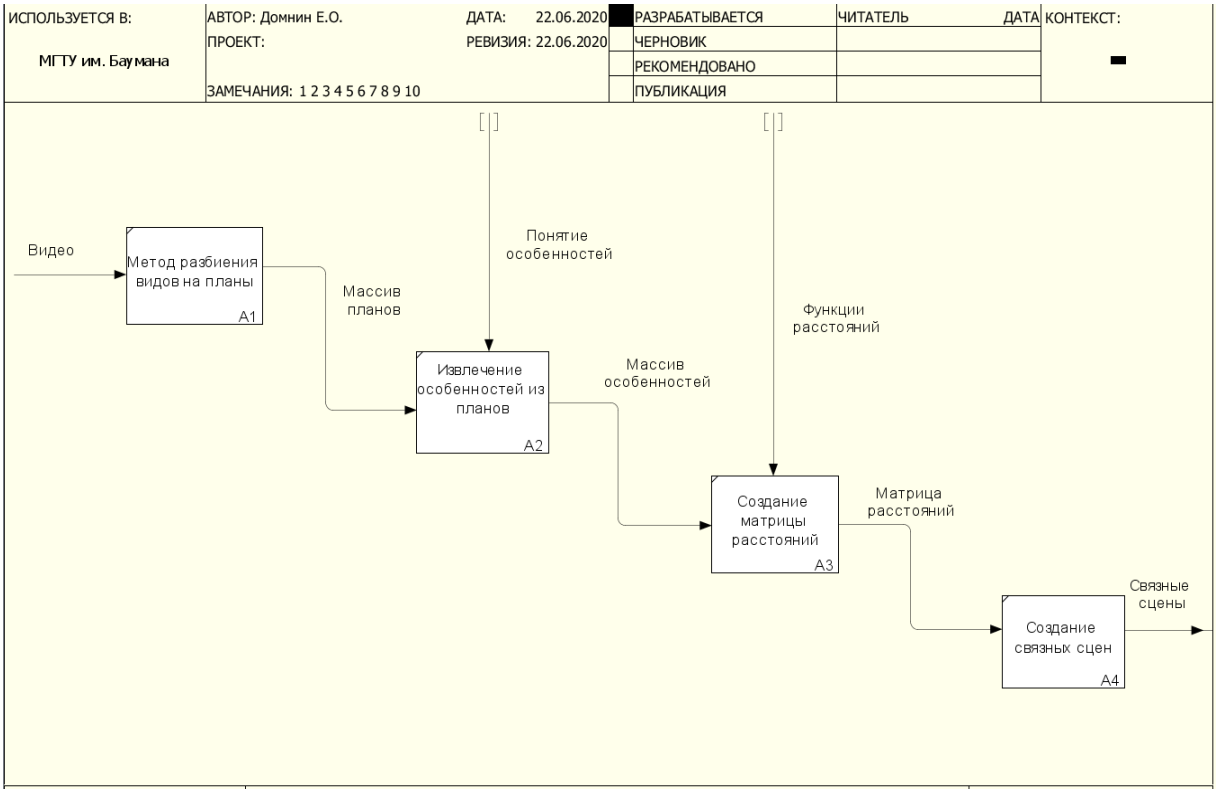



Рисунок 12 IDEF0 диаграмма работы метода

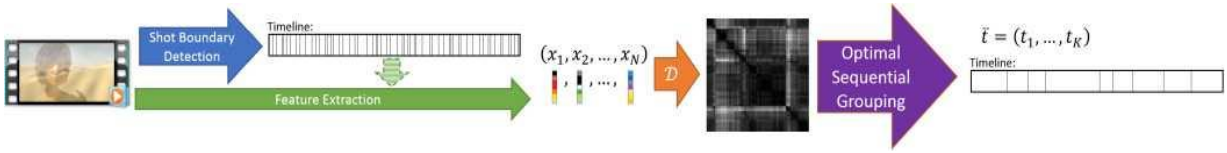



Рисунок 13 Иллюстрация этапов работы метода

Таким образом метод состоит из четырёх этапов:

1. Разбиение видео на планы
2. Извлечение особенностей из планов
3. Создание матрицы расстояний
4. Создание связных сцен

Рассмотрим далее детальнее каждый из этих этапов

## 2.2 Разбиение видео на планы

Предложенная архитектура TransNet [11] (рис. 14) следует работе стандартных свёрточных архитектур. В качестве входных данных сеть затем использует последовательность из N последовательных видеокадров и применяет серию трехмерных сверток, возвращающих прогноз для каждого кадра во входном сигнале. Основной строительный блок модели (разветвленная ячейка DCNN, Dilated CNN) спроектирован как четыре свёрточных операции 3D 3 × 3 × 3.

Каждая свёрточная операция отличается по значениям порога расстояния и их выходам, которые соединены в одном измерении размеров. Множество DDCNN-ячеек поверх друг от друга предшествуют максимальному пулингу что формирует комбинировванную ячейку DCNN (SDCNN – Stacked DCNN). TransNet состоит из нескольких SDCNN блоков, каждый последующий блок, работающий на меньшем пространственном разрешении, увеличивает размер канала, еще больше увеличивая выразительное поле и, следовательно, целевое поле сети.

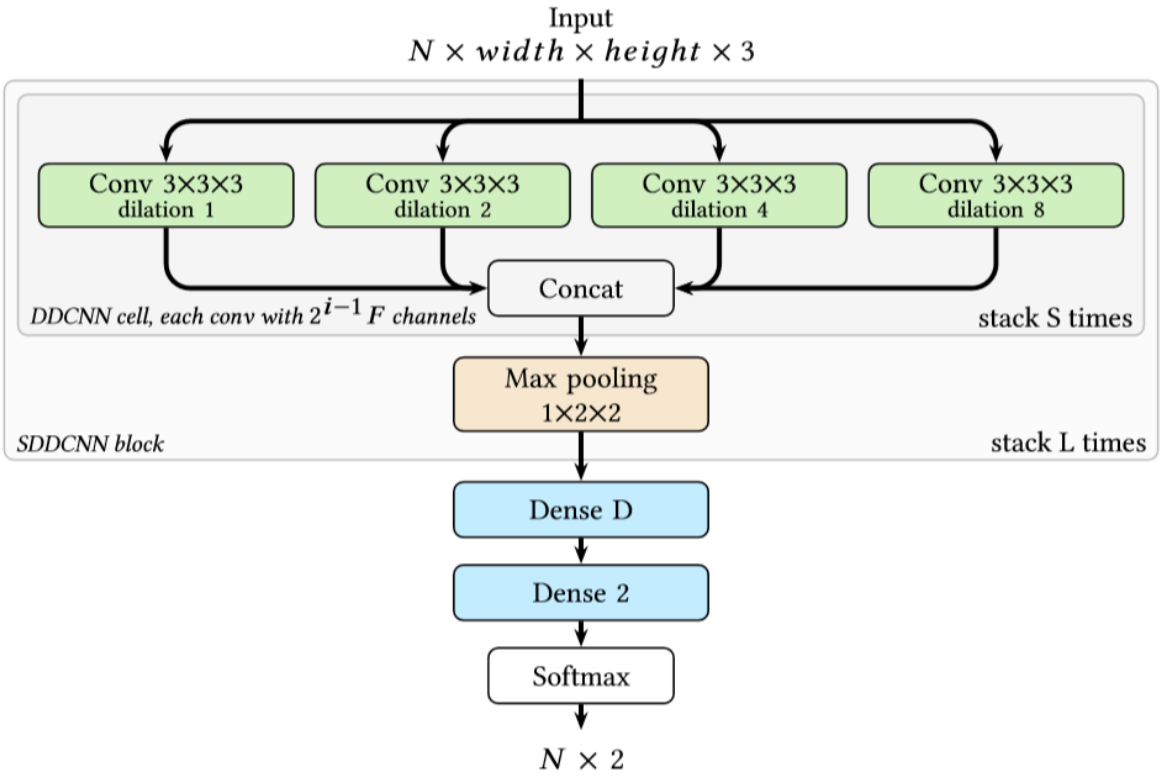



Рисунок 14 Архитектура сети TransNet

Два соединенных между собой слоя уточняют характеристику, извлеченную сверточными слоями, и предсказывают возможные границы плана для каждого кадра независимо от остальных кадров (весовые коэффициенты слоев разделяются). ReLU активация используется во всех слоях с единственным исключением, из-за того, что последний полностью подключенный слой имеет функцию активации softmax. Stride равен 1 и padding «same» используют все сверточные слои.

Для обучения использован набор данных TRECVID IACC.3, поскольку он снабжен набором предопределенных временных сегментов. Следовательно, пары предопределенных сегментов могут быть случайно выбраны из пула для автоматического создания переходов для целей обучения. В частности, были рассмотрены сегменты 3000 IACC.3 случайно выбранных видео. Кроме того, сегменты с менее чем 5 кадрами были исключены, и из них были выбраны все остальные сегменты, в результате чего было выделено 5448 сегментов.

Обучающие примеры были генерировались с помощью того что во время обучения случайным образом выбирались два плана соединялись случайным типом перехода. Только переходы вида склейки встык и плавного перехода, были учтены в процессе обучения. Для подтверждения моделей дополнительно были добавлены 100 IACC.3 видео (из набора, отличающегося от обучающего), в результате чего было получено 3800 снимков.

Эффективность расширенных 3D сверток была показана на основе набора данных RAI. TransNet показывает результаты на уровне других современных алгоритмов, без какой-либо дополнительной постобработки и с малой долей возможных параметров. Использование одного графического ускорителя уровня Nvidia GeForce K80 позволяет достичь скорости в 100 раз превышающей скорость видео.

### 2.3 Извлечение особенностей планов

В последние годы сверточные нейронные сети (CNN) показали большой прогресс в масштабном распознавании изображении и видео, приводя к значительным увеличениям результатов в различных тестах [6,7,8,9]. Архитектуры CNN, такие как AlexNet, VGG, Inception и ResNet, могут быть обучены на больших наборах данных, таких как ImageNet, а затем использованы в качестве мощных экстракторов особенностей, которые будут включать полезную семантическую информацию. Точно так же было показано, что архитектуры CNN могут также предложить очень многообещающие результаты по классификации аудио.

Для обеспечения эффективного обнаружения связных сцен, функции, извлеченные из различных снимков, должны кодировать высокоуровневую семантическую информацию о видео. Учитывая, что низкоуровневая информация в видео может сильно отличаться даже при небольших сдвигах камеры, может очень полезно описать план концепциями высокого уровня, которые сделают сравнение с соседними планами более значительным. Таким образом, наша ги-

потеза в этой работе заключается в том, что особенности, извлечённые с помощью свёрточных нейронных сетей (кодирующие визуальную и звуковую информацию видео) могут оказаться полезным также для задачи обнаружения видео сцены.

Более конкретно, для кодирования визуальной информации мы используем различные архитектуры, предварительно обученные на базе фотографий ImageNet, и применяем их к ключевым кадрам в видео для извлечения семантической информации, представляющей каждый план. На основе анализа, проведённого в аналитической части в качестве способа выбора ключевого кадра было решено использовать средний кадр плана так как это даёт баланс между скоростью работы метода и точностью результатов. Получение особенностей из нейронной классифицирующей сети делается путем удаления окончательного уровня классификации сети и извлечения предпоследнего уровня CNN, чтобы получить представление высокого уровня. Результирующий многомерный вектор для плана позволяет измерять расстояния между планами семантическим и более значимым способом.

Кроме того, оценим результаты при добавлении также аудио функций для получения мультимодального представления особенностей видео. Для этого используем модель нейронной сети подобную VGG, которая представляет каждые 0,96 секунды звука в качестве 128-мерного вектора особенностей.

## 2.4 Создание матрицы расстояния планов

Для создания матрицы плана при использовании нескольких видов особенностей необходимо определиться с мерой расстояния и способом объединения различных особенностей.

На основе анализа, проведённого в аналитической части для объединения особенностей мы будем строить матрицу для каждой модальности и усреднять их значения, после чего нормализировать. Отметим что это объединение не эквивалентно усреднению нормализованных векторов признаков вместе, так как

матрицы уже включают в себя сходства и различия между векторами. Слияние по существу усредняет относительное сходство между каждой парой планов, позволяя обеим модальностям внести свой вклад в предпочтение разбиения.

Наиболее распространёнными мерами расстояния для данной задачи являются:

1. Расстояние Бхаттачариа
2. Расстояние Хеллигера
3. Косинусное расстояние
4. Евклидово расстояние
5. Мера $L_2 norm$

В наиболее современной работе по методам создания связных сцен с помощью динамической оптимизации использовалось косинусное расстояние, потому решено и в данной работе использовать его.

$$D = \cos(S_1, S_2) = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}} \tag{9}$$

## 2.5 Создание связных сцен

При расчете расстояний между всеми планами мы можем объединить значения в матрицу: $\mathrm{D}(\mathrm{j}, \mathrm{j}') = \mathrm{D}(\mathrm{x_j}, \mathrm{x_{j'}})$. Если планы выбраны правильно, так что в одной и той же сцене между планами низкое расстояние, а между планами из разных сцен высокое. На основе этого, можно предположить, что матрица расстояния D выглядят аналогично изображению на рис. 10. Блоки по диагонали представляют сцены, в которых группы последовательных объектов имеют небольшие значения расстояния между ними, но большие расстояния до других объектов в видео. Основная диагональ имеет значения нулевого расстояния, поскольку $\mathrm{D}(\mathrm{x_j}, \mathrm{x_j}) = 0$. Оптимальное значение будет получено при раз-

мещении делений таким образом, чтобы целевая функция суммировала блоки с малыми характеристическими расстояниями.

Учитывая указанную выше метрику расстояния и принимая во внимание определение сцены, определи целевую функцию $H_D^K: \Omega^K \rightarrow R$, которая присваивает оценку любому возможному разделению в отношении функции стоимости D и планов K. Мы представляем целевую функцию, которая принимает форму:

$$H_D^K(t) = \sum_{i=1}^{K} \left\{ \frac{\left( \sum_{j=t_{i-1}+1, j'=t_{i-1}+1}^{t_i} D\left(x_j, x_{j'}\right) \right)}{S_i} \right\} \quad (10)$$

Эта целевая функция суммирует внутригрупповые расстояния для всех объектов, сгруппированных в сцену i, и суммирует эти групповые расстояния для всех сцен, где $S_i$ является необязательным весовым коэффициентом (например, для включения специфичного для домена коэффициента, если таковой существует). Этот подход похож на некоторые классические формулировки для кластеризации, но с добавленной внутренней группировкой только последовательных элементов.

Для процесса оптимизации далее обобщим, используя дополнительный верхний индекс $H_D^{n,K}$, чтобы указать, что мы используем только подмножество кластеров-сцен от: n до N. В этом случае область изменяется на $\Omega^{n,k}$, которая является множеством всех возможных монотонно увеличивающихся серий длины k начинающихся не ранее n и заканчивающиеся N. Когда H пишется только с одним верхним индексом, подразумевается, что $n = 1$.

Теперь нужно найти t, что минимизирует $H_D^{n,K}$. Поскольку формулировка не является типичным определением кластеризации, классические решения кластеризации не могут быть применены. Предположим схему динамического программирования для достижения оптимального решения $t^*$ эффективно. В

дальнейшем мы считаем $S_i = 1$ для упрощения. Теперь опишем этапы метода динамического программирования:

Определим таблицу затрат $C(n, k)$ как оптимальное значение целевой функции $H_D^{n,K}$. Тогда оптимальное значение целевой функции для всей задачи находится при $C(1, K)$. Поскольку само значение целевой функции не содержит информации о разбиении, определим $J(n, k)$ как $t_1$ подзадачи разделения $N - n + 1$ особенностей планов на $k$ сцен. Таким образом $J(n, k)$ указывает, где лучше разместить следующую границу. Инициализируем таблицы значениями

$$C(n, 1) = \sum_{j=n}^{N} \sum_{j'=n}^{N} D(x_j, x_{j'}) \quad (11)$$

и

$$J(n, 1) = N \quad (12)$$

Поскольку разделение оставшихся элементов на один раздел состоит только из одного возможного решения $C(n, 1) = H_D^{n,1}(t = t_1 = N)$.

Используя линейность внешней суммы$H_D^{n,K}$ в (10), остальная часть таблицы определяется рекурсивно для каждого $1 < k \leq K$:

$$C(n, k) = \min_i \left\{ \sum_{j=n}^{i} \sum_{j'=n}^{i} D(x_j, x_j) + C(i + 1, k - 1) \right\} \quad (13)$$

$$J(n, k) = \operatorname{argmin}_i \left\{ \sum_{j=n}^{i} \sum_{j'=n}^{i} D(x_j, x_j) + C(i + 1, k - 1) \right\} \quad (14)$$

Таблицы C и J должны быть построены последовательно с $k$ от 2 до. Границы сцен можно воспроизвести, вычислив последовательно от 1 до K следующую формулу:

$$t_i^* = J(t_{i-1} + 1, K - i + 1) \quad (15)$$

Для дальнейшей эффективности можно вычислить промежуточные суммы отдельно. Построим таблицу E размера N*N, которая содержит суммы для быстрого их использования без повторных вычислений.

$$E(i,i') = \sum_{j=i}^{i'} \sum_{i'=i}^{i'} D\left(x_j, x_j\right) \quad (16)$$

Эта таблица не обязана быть сразу инициализирована, а вместо этого сама может быть рассчитана с использованием динамического программирования: инициализируем $E(i,i) = D(x_i, x_i) = 0$ и затем рекурсивно остальную часть таблицы по следующей формуле:

$$E(i,i') = E(i-1,i') + E(i,i'-1) - E(i-1,i'-1) + D(x_i, x_{i'}) + D(x_{i'}, x_i) \quad (17)$$

которая позволяет построить E вдоль диагоналей от главной диагонали и в порядке возрастания.

# 3 Технологический раздел

Технологический раздел обосновывает выбор средств программной реализации, включая выбор языка программирования, дополнительных библиотек и модулей, а также описание основных моментов программной реализации и методики тестирования созданного программного обеспечения.

## 3.1 Выбор средств программной реализации

### 3.1.1 Выбор языка программирования

В качестве языка программирования выбран Python версии 3.7 для операционной системы Windows. Python является современным языком программирования, поддерживающим объектно-ориентированную парадигму программирования. Кроме того, язык Python имеет большое количество библиотек, связанных с машинным обучением и нейронными сетями, что важно для извлечения особенностей из планов. Так как в данной работе используются нейронные сети рационально использовать библиотеку, которая позволяет использовать графический процессором. Выбор такой сторонней библиотеки будет обоснован в следующем разделе.

Разработанное программное обеспечение не предполагает использования неподготовленным пользователем так как реализует концепт алгоритма, используемого в промежуточной фазе анализа видео.

В качестве IDE для разработки была выбрана PyCharm, которая предоставляет множество инструментальных средств для разработки на языке Python. PyCharm включает в себя графический редактор кода с автодополнением, встроенный интерпретатор языка Python, средства отладки и тестирования разрабатываемого ПО.

### 3.1.2 Выбор библиотек для разработки

Необходимо выбрать библиотеки, которые упростят извлечение частей видео, использование нейронных сетей для создания планов и извлечения особенностей.

Для работы с видео была выбрана популярная кроссплатформенная библиотека OpenCV так как она может использоваться из Python, однако сама по себе она реализована на C\C++ что даёт относительно высокую скорость работы, также она поддерживает ускорение некоторых функций с помощью графических процессоров.

Для Python есть ряд библиотек, упрощающих работу с нейронными сетями, наиболее популярные из которых:

1. Tensorflow - Библиотека, разработанная корпорацией Google для работы с тензорами, используется для построения нейросетей. Поддержка вычислений на видеокартах имеет версию для языка C++. На основе данной библиотеки строятся более высокоуровневые библиотеки для работы с нейронными сетями на уровне целых слоев. Так, некоторое время назад популярная библиотека Keras стала использовать Tensorflow как основной бэкенд для вычислений вместо аналогичной библиотеки Theano. Для работы на видеокартах NVIDIA используется библиотека cuDNN. Если вы работаете с картинками (со сверточными нейросетями), скорее всего, придется использовать данную библиотеку.
2. Keras - Библиотека для построения нейросетей, поддерживающая основные виды слоев и структурные элементы. Поддерживает как рекуррентные, так и сверточные нейросети, имеет в своем составе реализацию известных архитектур нейросетей (например, VGG16). Некоторое время назад слои из данной библиотеки стали доступны внутри библиотеки Tensorflow. Существуют готовые функции для

работы с изображениями и текстом (Embedding слов и т.д.). Интегрирована в Apache Spark с помощью дистрибутива dist-keras.

3. Caffe - Фреймворк для обучения нейросетей от университета Беркли. Как и TensorFlow, использует cuDNN для работы с видеокартами NVIDIA. Содержит в себе реализацию большего количества известных нейросетей, один из первых фреймворков, интегрированных в Apache Spark (CaffeOnSpark).
4. pyTorch - Позволяет портировать на язык Python библиотеку Torch для языка Lua. Содержит реализации алгоритмов работы с изображениями, статистических операций и инструментов работы с нейронными сетями. Отдельно можно создать набор инструментов для оптимизационных алгоритмов (в частности стохастического градиентного спуска).
5. PlaidML - портативный тензорный компилятор. Тензорные компиляторы устраняют разрыв между универсальными математическими описаниями операций глубокого обучения, таких как свертка, и специальным кодом платформы и чипа, необходимым для выполнения этих операций с хорошей производительностью. Внутри PlaidML использует TDS eDSL для генерации кода OpenCL, OpenGL, LLVM или CUDA. Это позволяет проводить глубокое обучение на устройствах, где доступное компьютерное оборудование либо плохо поддерживается, либо доступный программный стек содержит только фирменные компоненты. Например, он не требует использования CUDA или cuDNN на оборудовании Nvidia при достижении сопоставимой производительности.

Как видно почти все библиотеки поддерживают ускорение только с помощью CUDA, а не OpenCL, так как тестовая платформа поддерживает CUDA будут выбраны наиболее простые для использования библиотеки: Tensorflow и Keras. Tensorflow для реализации разбиения видео на планы, так как в оригинальной статье о данном методе использовалась именно Tensorflow [11]. Keras

будет использоваться для извлечения особенностей из планов тк имеет в своём составе реализации многих нейросетей подходящих для классификации изображений [18].

Также для ускорения математических вычислений и реализации функций расстояния между планов будут использоваться библиотеки:

1. NumPy - Библиотека с открытым исходным кодом для выполнения операций линейной алгебры и численных преобразований. Как правило, такие операции необходимы для преобразования датасетов, которые можно представить в виде матрицы. В библиотеке реализовано большое количество операций для работы с многомерными массивами, преобразования Фурье и генераторы случайных чисел. Форматы хранения numpy де-факто являются стандартом для хранения числовых данных во многих других библиотеках (например, Pandas, Scikit-learn, SciPy).
2. SciPy - Довольно обширная библиотека, предназначенная для проведения научных исследований. В ее состав входит большой набор функций из математического анализа, в том числе вычисление интегралов, поиск максимума и минимума, функции обработки сигналов и изображений. Во многих отношениях данную библиотеку можно считать аналогом пакета MATLAB для разработчиков на языке Python. С ее помощью можно решать системы уравнений, использовать генетические алгоритмы, выполнять многие задачи по оптимизации.

Для визуализации будет использоваться библиотека mathplotlib.

## 3.2 Формат тестовых данных

Как было указано в аналитическом разделе, для извлечения особенностей разрешение видео должно быть не меньше 299 пикселей по горизонтали и вер-

тикали. Непосредственно формат видео не принципиален так как библиотека OpenCV поддерживает широкий диапазон форматов видеозаписей. Поэтому ограничения больше зависят от имеющихся наборов тестовых данных, под которые будет добавлена поддержка в ПО.

## 3.3 Условия запуска программного обеспечения

Так как у разрабатываемого ПО нет зависимостей от внешних служб и программ, установленных в операционной системе, а также отсутствует необходимость в модификации системных и пользовательских настроек, было принято решение не разрабатывать инсталлятор. Установка производится копированием папки с программой в нужную директорию. Для запуска программы необходимо иметь установленные в системе библиотеки Tensorflow 2.0, Keras 2.3.1, SciPy 1.3.1, Numpy 1.18.3, OpenCV-Python 4.2 и Python версии 3.7. Для запуска программы необходимо запустить интерпретацию файла VSDmain.py командой python VSDmain.py .

Однако ввиду использования библиотек Tensorflow и Keras накладываются ограничения на программное и аппаратное обеспечение:

- 64 битная версия Windows не старше Windows 7
- Процессор с поддержкой AVX инструкций
- Для поддержки ускорения графическим процессором грубуется графический процессор с поддержкой CUDA версии 3.5 и новее, установленная версия CUDA 10.1, cuDNN SDK версии 7.6 или новее

## 3.4 Тестирование и отладка программы

В ходе разработки программного обеспечения было произведено модульное и системное тестирование программного обеспечения, которые

позволили выявить и своевременно устранить недостатки. Автоматические тесты не применялись, выполнялось ручное тестирование.

## 3.5 Выводы

В результате проделанной работы был выбран подходящий язык программирования, а также необходимый набор библиотек для реализации разрабатываемого программного обеспечения. Был конкретизирован формат тестовых данных. Был выбран способ тестирования и отладки написанного кода, а также необходимые для этого инструменты.

# 4 Экспериментальный раздел

В данном разделе проводится ряд экспериментов с использованием различных способов получения особенностей из планов для оценки результатов работы разработанного метода и сравнения получаемых результатов. Также в рамках экспериментального раздела проводится сравнение результатов работы приложения в зависимости от входных данных.

## 4.1 Условия эксперимента

### 4.1.1 Тестовая платформа

Тестирование и эксперименты проводились на портативном компьютере со следующими характеристиками:

- Процессор AMD Ryzen 5 1600x CPU 4GHz 6 ядер 12 потоков;
- Два графических процессора Nvidia GTX 1070 8ГБ в режиме SLI high-bandwidth с поддержкой CUDA Compute Capability версии 6.1
- Оперативная память DDR4-3200 16ГБ
- ОС Windows 10.

### 4.1.2 Входные и выходные данные

Несмотря на то что само количество видео в свободном доступе достаточно велико для них недоступно разбиение на планы и сцены. Ручному разбиению могут помочь главы у фильмов или иные метаданные видеозаписи, однако всё равно данная задача остаётся чрезвычайно трудоёмкой. Потому было решено использовать уже имеющиеся наборы данных. Таких было найдено трое:

1. OVSD набор данных [5] – состоит из видео распространяющихся по открытой лицензии, состоящий из любительских полнометражных и короткометражных фильмов, а также анимационных фильмов от Blender Institute. Продолжительность видео от 10 до 105 минут. Форматы Mpeg4, avi и mov. Общее количество видео 21.
2. RAI набор данных [11] – состоит из набора итальянских новостных телепередач длинной около 10 минут каждая. Формат Mpeg4. Общее количество видео 10.
3. BBC набор данных [8] – состоит выпусков телепередачи ББС Планета Земля на английском языке продолжительностью около 50 минут каждое. Формат Mpeg4. Общее количество видео 11

Итого имеем тестовый набор из 42 видео, часть из которых является новостными передачами, часть телепередачами, часть анимационными фильмами, часть короткометражными и часть полнометражными фильмами. Форматы Mpeg4, Avi и Mov поддерживаются OpenCV.

### 4.1.3 Критерии оценки

Возможные метрики оценки были описаны в аналитической части данной работы. Мы будем использовать метрики Coverage, Overflow и Purity так как они позволяют сравнить полученные результаты с уже существующими методами на предоставленных в них наборах данных [5,7,10]. Также будет измерено время выполнения, так как это тоже важный параметр, учитывая темпы создания новых видео в алгоритмах чьё время исполнения на современной аппаратной платформе нет практического смысла [6].

## 4.2 Результаты эксперимента

На основе полученных результатов можно сделать вывод что в случае ускорения нейронных сетей с использованием графического процессора с поддержкой CUDA время выполнения метода динамической оптимизации с адди-

тивной функцией не сильно отличается от аналогичного метода, использующего в качестве особенностей цветовые гистограммы, однако в случае использования нейронных сетей для извлечения особенностей метрики показывают лучшие значения. В приложении А показаны визуализации матриц расстояний для одного из тестовых видео.

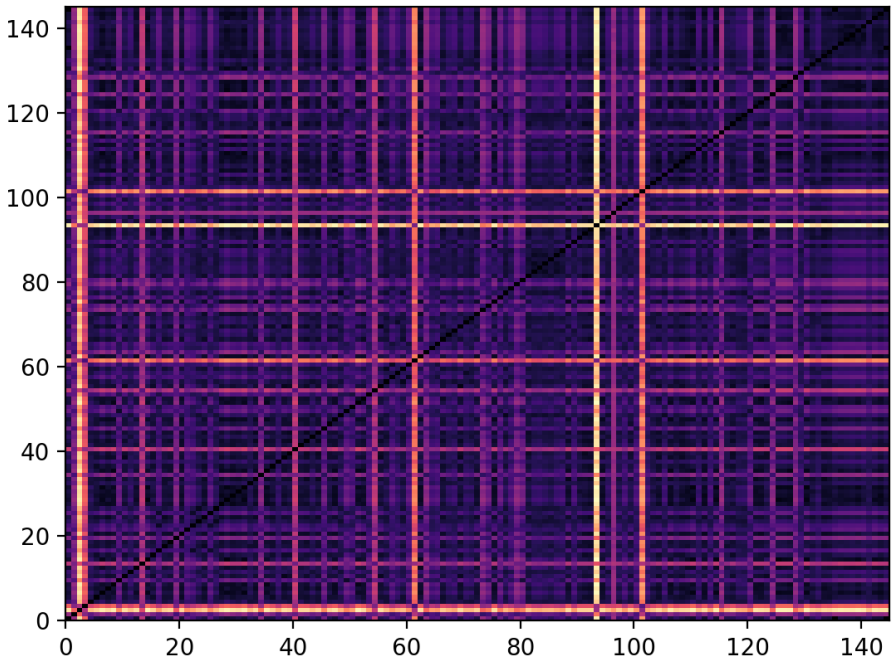



Рисунок 15 Визуализация матрицы расстояния планов для особенностей извлечённых с помощью Xception для видео Tears od Steel

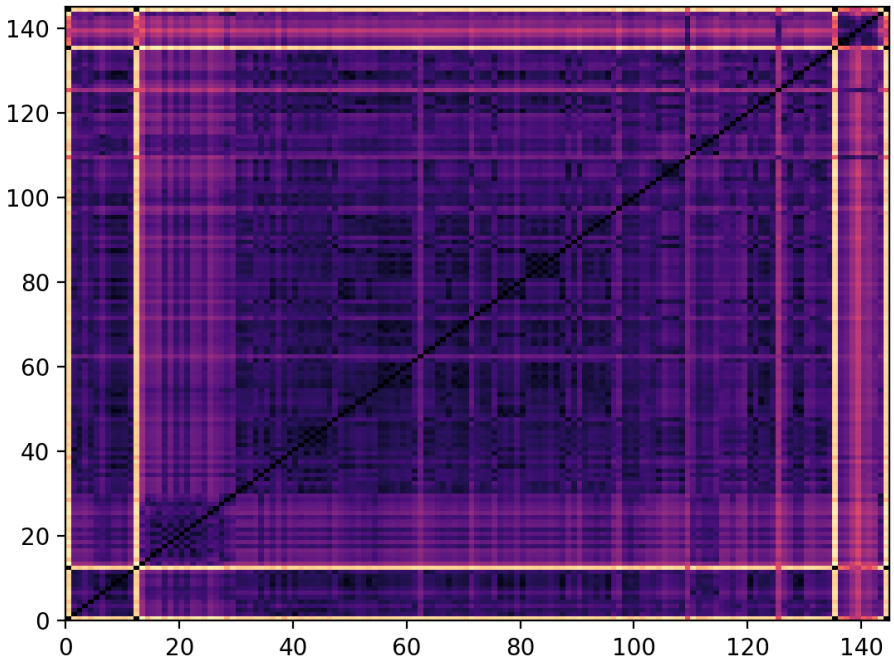



Рисунок 16 Визуализация матрицы расстояния планов для особенностей извлечённых с помощью HSV гистограмм для видео Tears od Steel

В приведённых выше примерах можно заметить затемнения квадратной формы рядом с побочной диагональю, это и есть предполагаемые сцены.

Таблица 1 – Сравнение результатов реализованного метода для различных видов особенностей. Через запятую перечислены усреднённые значения метрик Purity, Coverage, Overflow усреднённые для соответствующих наборов данных.

| Набор данных / Метод | BBC | RAI | OVSD |
|---|---|---|---|
| HSV | 0.43,0.72,0.71 | 0.4,0.85,0.37 | 0.5,0.8,0.5 |
| XCeption | 0.46,0.74,0.7 | 0.42,0.9,0.37 | 0.53,0.86,0.67 |
| Inception | 0.43,0.66,0.72 | 0.38,0.8,0.33 | 0.49,0.86,0.6 |
| VGG19 | 0.44,0.72,0.7 | 0.4,0.8,0.35 | 0.48,0.83,0.6 |
| MobileNet | 0.43,0.7,0.7 | 0.41,0.87,0.35 | 0.45,0.87,0.58 |
| ResNetV2 | 0.42,0.76,0.7 | 0.38,0.81,0.39 | 0.4,0.84,0.65 |
| DenseNet | 0.43,0.72,0.7 | 0.37,0.82,0.38 | 0.5,0.79,0.5 |
| Усреднение 2-7 способов | 0.44,0.7,0.69 | 0.36,0.84,0.32 | 0.51,0.86,0.51 |

В случае с наборами данных BBC и RAI в связи с однородностью данных результаты от видео к видео отличаются слабо, а в наборе данных OVSD в связи с разнообразностью видео результаты от видео к видео отличаются сильнее, однако, как можно заметить, общая тенденция зависимости метрик от способа к способу сохраняется.

## 4.3 Выводы

В экспериментальном разделе был проведён ряд экспериментов для различных наборов тестовых данных и способов получения особенностей. По результатам можно сказать что из всех рассмотренные способы получения особенностей для создания связных сцен по совокупности метрик в среднем лучшие результаты показал способ получения особенностей с помощью нейронной сети XCeption обученной на наборе фото ImageNet. Однако нельзя сказать, что в принципе нейронные сети лучше классического способа извлечения особенностей с помощью HSV гистограмм, сети ResNet, DenseNet и MobileNetV2 показали худшие результаты.

# ЗАКЛЮЧЕНИЕ

Разработанный в рамках данной работы метод удовлетворяет предъявленным ему требованиям. По результатам экспериментального раздела он обладает достаточной невысокими ресурсозатратами, поддерживает ускорение с использованием CUDA и результатом его работы являются связные сцены, подходящие для дальнейшего анализа.

В результате работы нам проектом были решены следующие задачи:

- Был произведён анализ существующих методов создания связных сцен, что позволило выбрать наиболее перспективный подход.
- Задача была формализована.
- Был разработан комбинированный алгоритм, сочетающий использование нейронных сетей для разбиения видео на планы и извлечения из них особенностей, а для создания на их основе связных сцен использовался метод динамической оптимизации
- Было разработано ПО реализующее алгоритм и демонстрирующее его работу.
- Было проведено сравнительное исследование работы реализованного алгоритма с современными существующими аналогами

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ


1. Alan F Smeaton, Paul Over, and Aiden R Doherty. 2010. Video shot boundary detection: Seven years of TRECVid activity
2. Enrique Amig´o Julio Gonzalo Javier Artiles Felisa Verdejo. 2010. A comparison of Extrinsic Clustering Evaluation Metrics based on Formal Constraints
3. Jeroen Vendrig, Marcel Worring. 2002. Systematic Evaluation of Logical Story Unit Segmentation
4. Alessandro Vinciarelli, Sarah Favre. 2007. Broadcast News Story Segmentation Using Social Network Analysis and Hidden Markov Models
5. Daniel Rotman, Dror Porat, Gal Ashour. 2016. Robust and Efficient Video Scene Detection Using Optimal Sequential Grouping
6. Manfred del Fabro, Laszlo Boszormenyi. 2013. State-of-the-Art and Future Challenges in Video Scene Detection: A Survey
7. Daniel Rotman, Dror Porat, Gal Ashour, Udi Barzelay. 2018. Optimally Grouped Deep Features Using Normalized Cost for Video Scene Detection
8. Lorenzo Baraldi, Costantino Grana, Rita Cucchiara. 2015. A Deep Siamese Network for Scene Detection in Broadcast Videos
9. Lorenzo Baraldi, Costantino Grana, Member, Rita Cucchiara. 2014. Recognizing and Presenting the Storytelling Video Structure with Deep Multimodal Networks
10. Lorenzo Baraldi, Costantino Grana, Rita Cucchiara. 2015. Shot and scene detection via hierarchical clustering for re-using broadcast video
11. Tomáš Souček, Jaroslav Moravec, Jakub Lokoč. 2019. TransNet: A deep network for fast detection of common shot transitions
12. Zeeshan Rasheed and Mubarak Shah. 2005. Detection and Representation of Scenes in Videos
13. Shawn Hershey, Sourish Chaudhuri, Daniel P. W. Ellis. 2017. CNN Architectures for Large-Scale Audio Classification

14. EladHoffer, NirAilon. 2015. Deep Metric Learning Using Triplet Network
15. Lorenzo Baraldi, Costantino Grana, Rita Cucchiara. 2017. A Video Library System using Scene Detection and Automatic Tagging
16. Robert Tibshirani, Guenther Walther, Trevor Hastie. 2001. Estimating the number of clusters in a dataset via the gap statistic
17. I. Markovsky. 2011. Low rank approximation: algorithms, implementation, applications.
18. Keras Applications. Режим доступа: https://keras.io/api/applications/

# ПРИЛОЖЕНИЕ А

## Примеры матриц расстояний для различных способов извлечения особенностей для тестового видео Elephants Dream

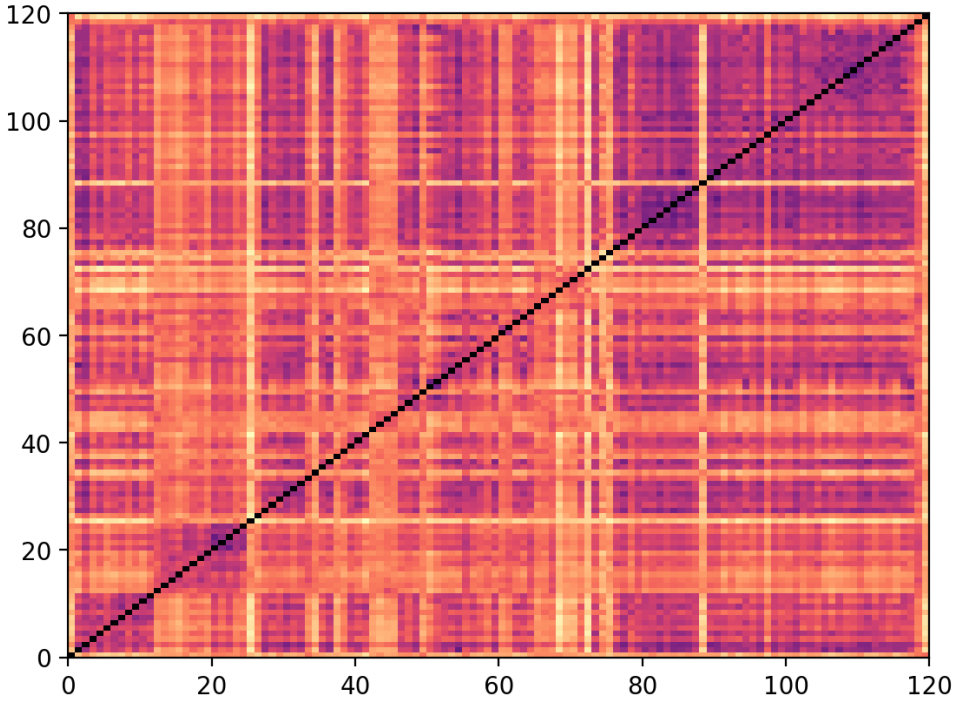



Рисунок 17 Особенности извлечённые с помощью MobileNet

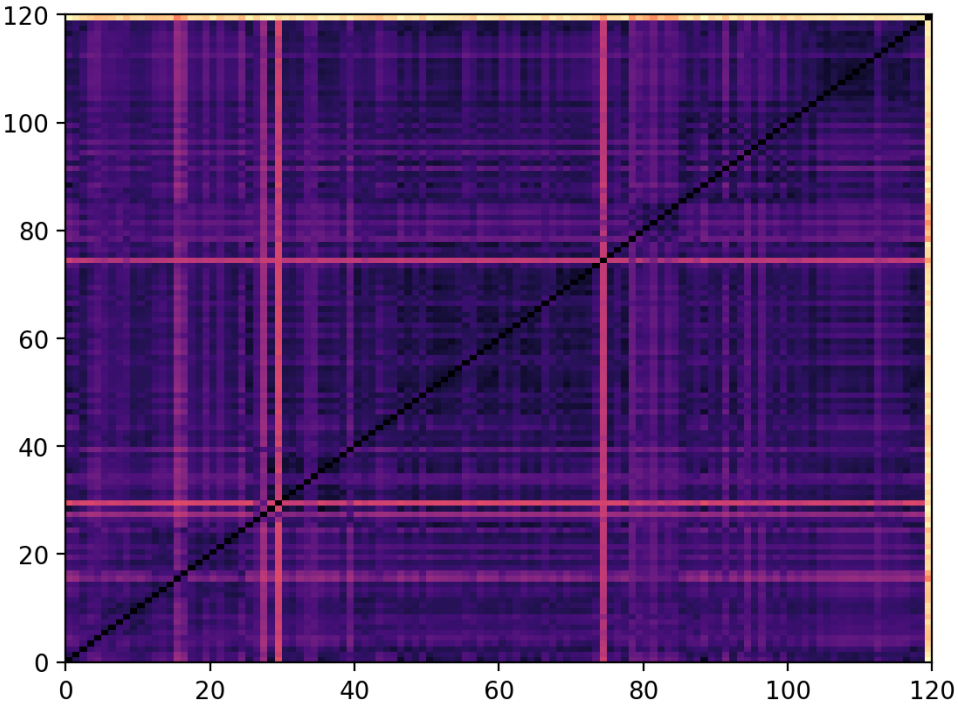



Рисунок 18 Особенности извлечённые с помощью MobileNetV2

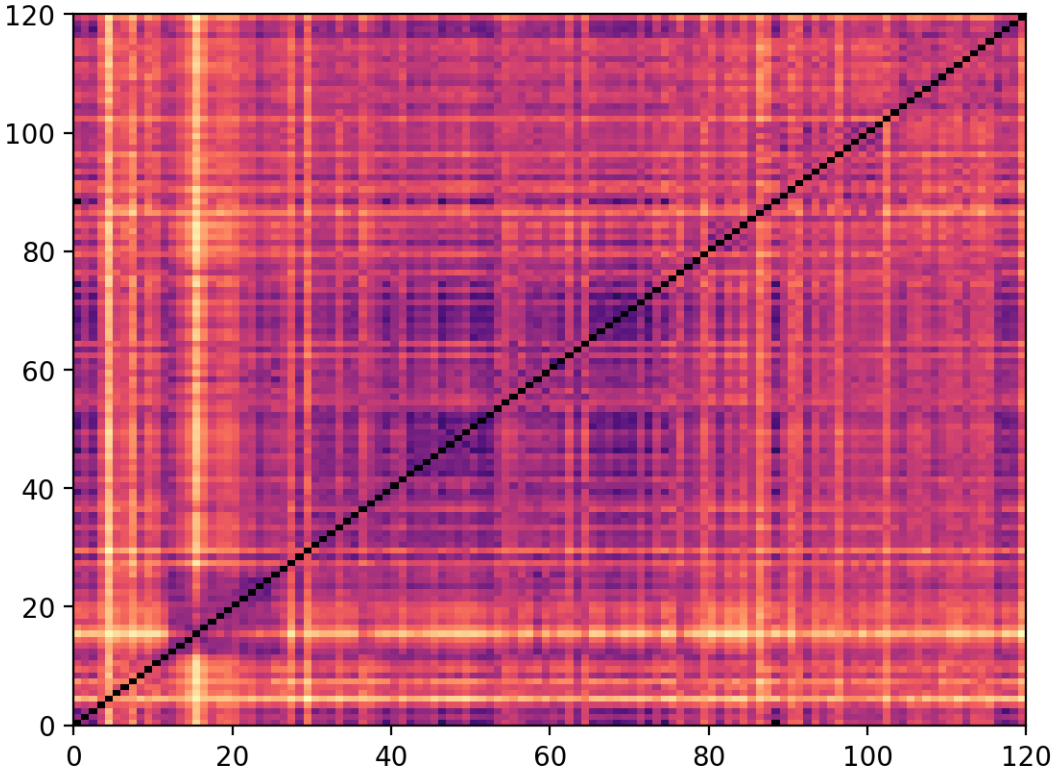



Рисунок 19 Особенности извлечённые с помощью VGG19

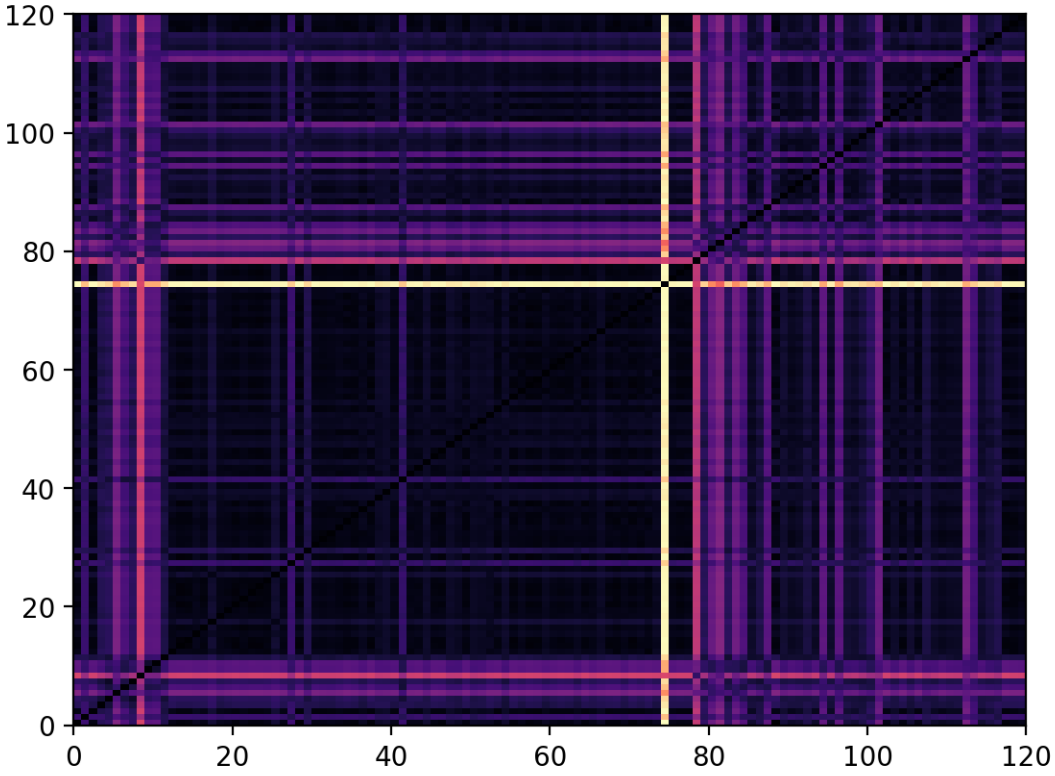



Рисунок 20 Особенности извлечённые с помощью XCeption

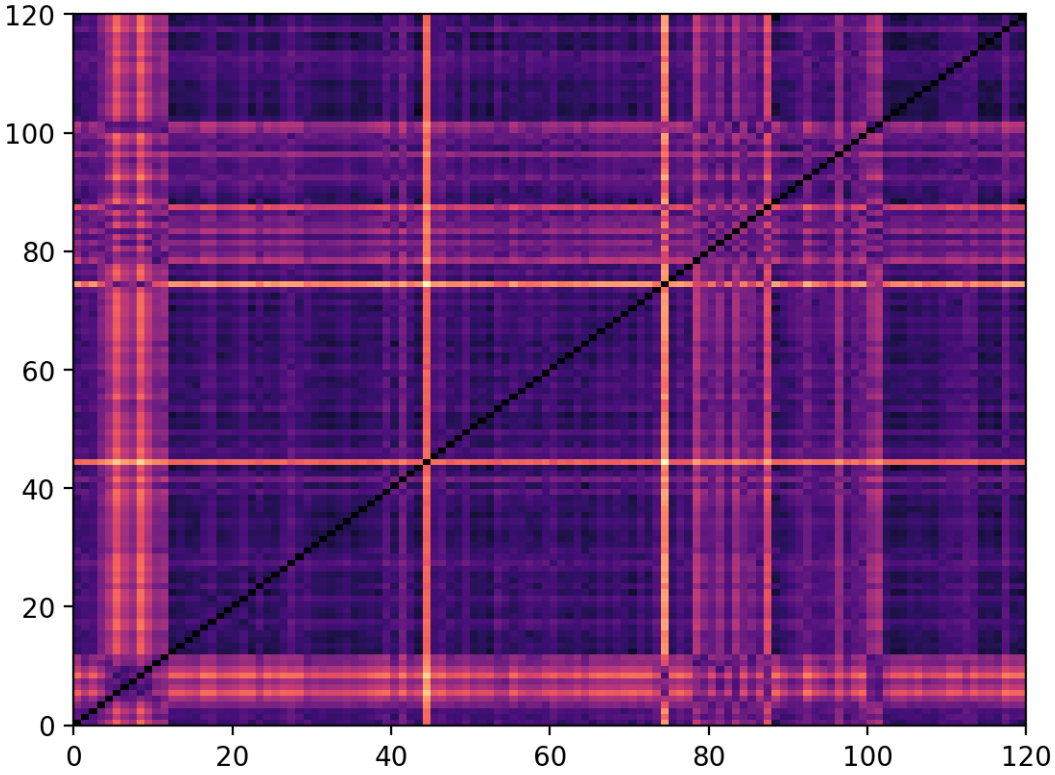



Рисунок 21 Особенности извлечённые с помощью Dense

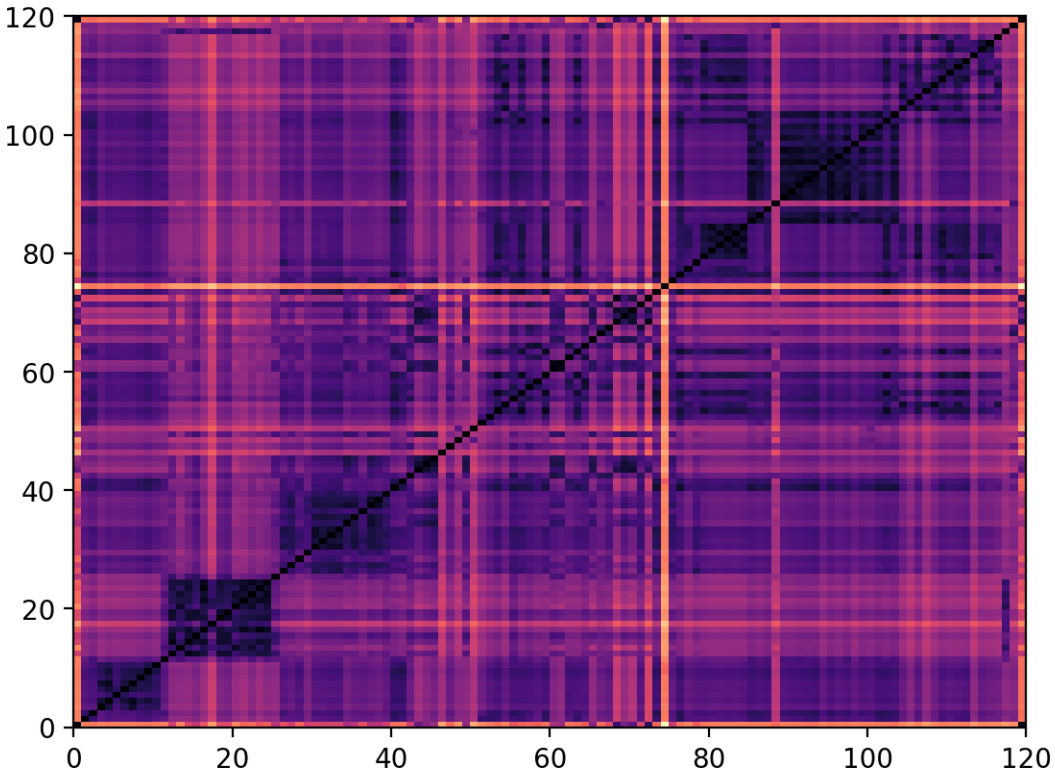



Рисунок 22 Особенности извлечённые с помощью HSV гисмтограмм

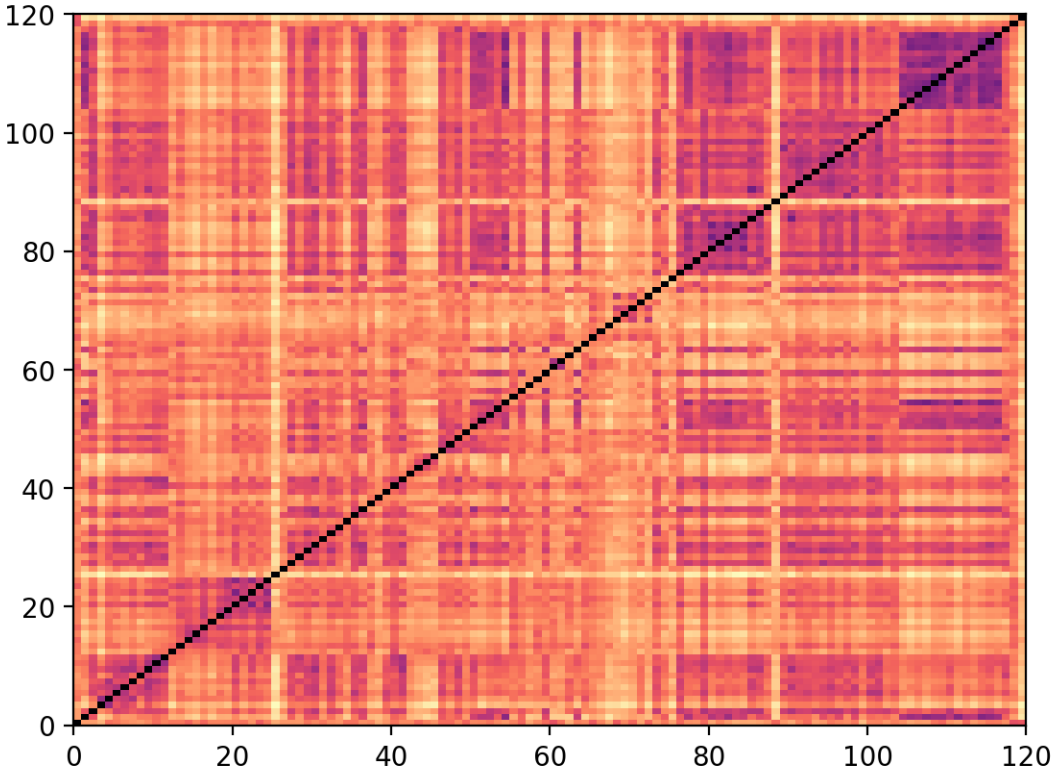



Рисунок 23 Особенности извлечённые с помощью ResNetV2

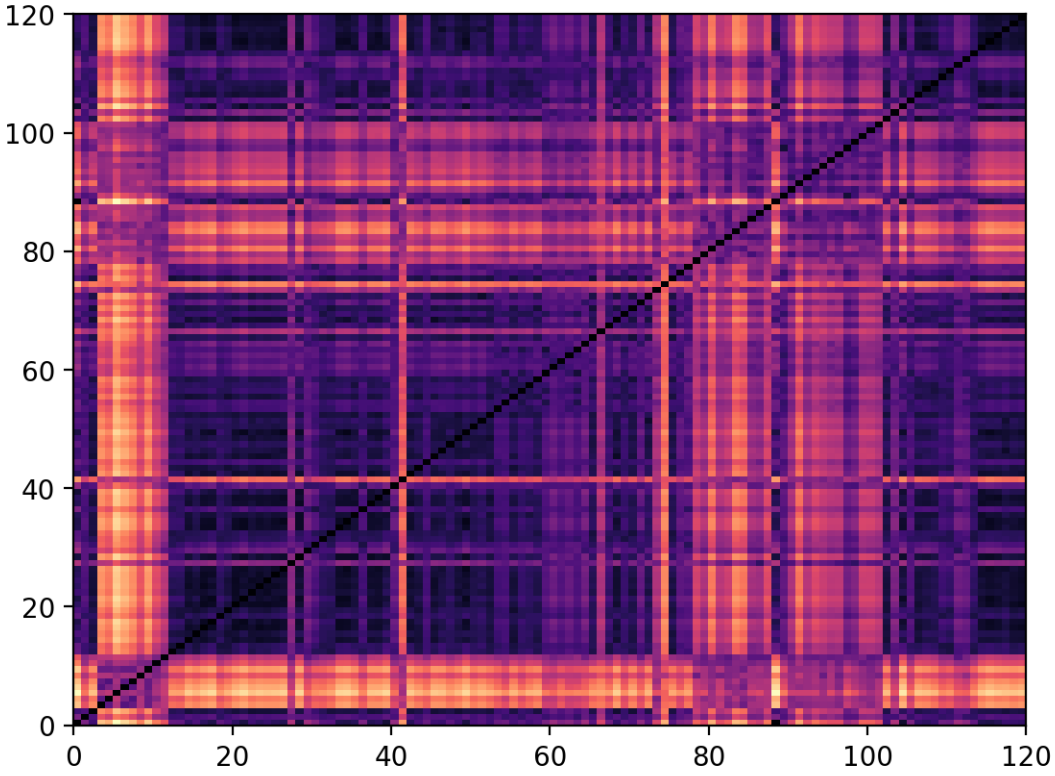



Рисунок 24 Особенности извлечённые с помощью InceptionV3

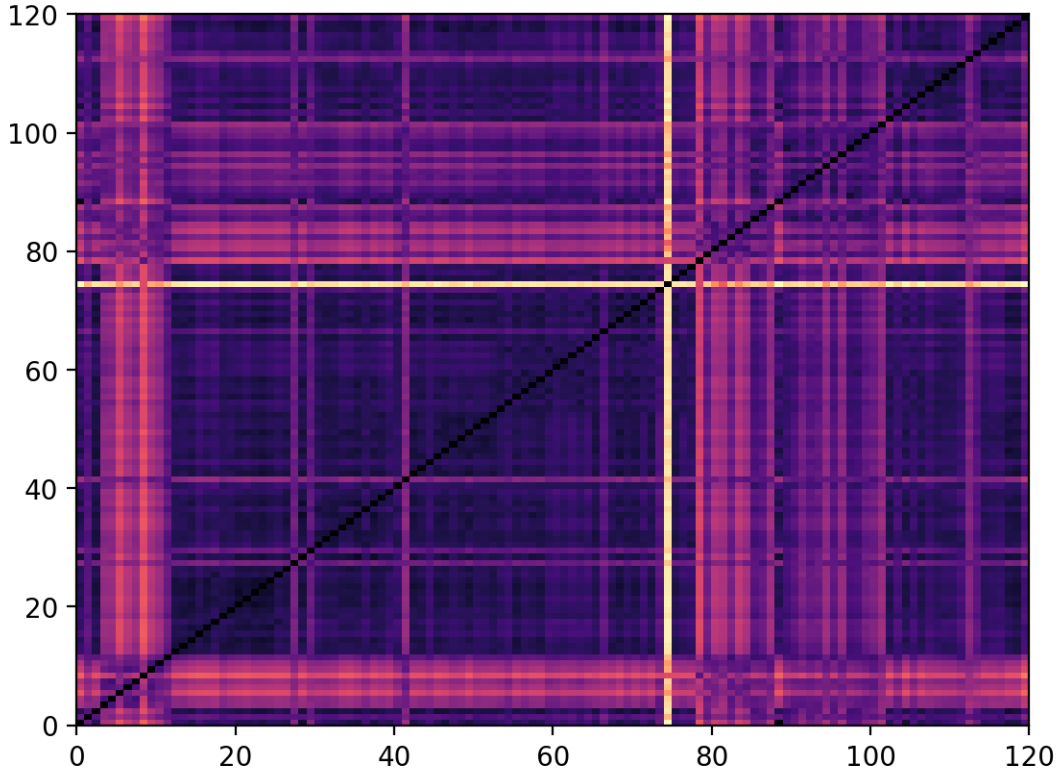



Рисунок 25 Усреднённые особенности всех способов извлечения