Санкт-Петербургский государственный университет

Математическое обеспечение и администрирование информационных систем
Системное программирование

Лучинский Владимир Дмитриевич

Разработка системы распознавания городских объектов

Бакалаврская работа

Научный руководитель:
к.т.н., доцент Литвинов Ю. В.

Научный консультант:
руководитель проектов ООО «Системы компьютерного зрения»
Пенкрат Н. А.

Рецензент:
к. ф.-м.н., ст. преп. Салищев С. И.

Санкт-Петербург
2020

SAINT-PETERSBURG STATE UNIVERSITY

Software and Administration of Information Systems
Software Engineering

Vladimir Luchinskiy

System for city objects recognition

Bachelor's Thesis

Scientific supervisor:
C.Sc., assistant professor Yurii Litvinov

Scientific consultant:
Project manager at the «Computer Vision Systems» Limited
Nikolay Penkrat

Reviewer:
PhD, senior lecturer Sergey Salishev

Saint-Petersburg
2020

Оглавление

Введение **4**

1. Постановка задачи **7**

2. Обзор **8**

2.1. Augmented.City — система дополненной реальности в городских сценах 8

2.2. Система извлечения локальных ключевых точек DELF (DEep Local Features) 9

2.3. Глубокое метрическое обучение (Deep Metric Learning) . 11

2.4. Нейронная сеть для семантической сегментации DeepLabv3+ 12

2.5. Архитектуры свёрточных нейронных сетей 13

2.5.1. Остаточная нейронная сеть (Residual neural network, ResNet) 14

2.5.2. Нейронная сеть Inception 14

3. Предлагаемый нейросетевой подход **16**

3.1. Обзор подхода 16

3.2. Реализация предлагаемого подхода 18

3.2.1. Выявление требований и сценарии использования 18

3.2.2. Архитектура системы 20

4. Проведение экспериментов **23**

4.1. Формирование обучающей выборки 23

4.2. Обучение распознающей сети 24

4.3. Анализ результатов 25

Заключение **31**

Список литературы **33**

Введение

Давайте представим, что, гуляя по улицам чужого города, наткнувшись на ресторан, который раньше не видели, мы могли бы по его фотографии узнать отзывы о нём, блюда, которые в нём подают, и их стоимость. Это представляет собой задачу распознавания городских объектов (назовём её целевой), которую можно решить с помощью компьютерного зрения.

Компьютерное зрение — это область, посвящённая анализу содержимого изображений. В настоящее время методы компьютерного зрения основываются как на классических алгоритмах, так и на нейронных сетях и глубоком машинном обучении. Компьютерное зрение находит применение в различных сферах деятельности, например, в медицине для определения раковых опухолей [22] или в робототехнике [5].

В частности, в рамках компьютерного зрения интересующая нас целевая задача сводится к поиску изображения по содержанию (image retrieval), то есть когда мы ищем изображение, похожее на то, которое содержится в некоторой базе изображений.

Существует довольно много способов решения задачи image retrieval, однако все подходы можно объединить в две большие группы. Первая основывается на классических алгоритмах компьютерного зрения, использующих локальные ключевые точки (features). Например, дескрипторы ключевых точек в сочетании с bag-of-words [25] или геометрической верификацией [11].

Главным отличительным признаком другой группы подходов, которые на данный момент являются самыми эффективными, является использование глубоких свёрточных нейронных сетей (CNN), которые производят глобальные ключевые точки, сопоставляют изображению некоторый ключевой вектор (feature vector) [21]. Такие подходы различаются функциями ошибки, предобработкой изображений, применением уже обученных моделей и использованием различных слоев. Кроме того, предпринимались довольно удачные попытки комбинирования второго подхода с первым [15].

Готовые системы существуют, однако они либо ещё не выпущены в релиз (Google Visual Positioning System[1]), либо имеют недостаточно высокую точность (Augmented.City[2]), либо требуют достаточно больших затрат по времени (DELF [15]). Поэтому возникает необходимость в создании новой системы.

Однако независимо от того, какой используется подход, имеется ряд проблем, обусловленных спецификой задачи. Во-первых, здания могут быть похожи, и чем больше зданий, тем выше вероятность этого. Или, например, мы можем находиться в каком-нибудь новом жилом комплексе, где все дома выполнены в похожем архитектурном стиле. Во-вторых, изображения одного и того же объекта могут существенно различаться, если они сделаны с разных ракурсов, или, например, зимой или летом. В-третьих, часто в кадр могут кроме здания могут попасть прохожие или машины, способные помешать корректному распознаванию.

Применение лучшего существующего решения может оказаться недостаточным для достижения необходимого качества работы системы из-за названных выше проблем. Поэтому необходимо также использовать различные вспомогательные средства. И если в решении первой проблемы может помочь геолокация, в решении второй – применение различных техник для получения дополнительных данных из имеющихся (аугментация данных), то для снижения негативного влияния третьей – устоявшегося способа нет. В данной работе для этого была использована нейронная сеть для семантической сегментации, чтобы на основе результатов её работы вырезать из снимков двигающиеся объекты.

Таким образом, в работе предлагается нейросетевой подход для распознавания городских объектов. В нём для сопоставления изображений была обучена нейросеть с применением глубокого метрического обучения, а для снижения негативного влияния двигающихся объектов (пешеходов, автомобилей) была использована сеть для семантической

[1] `https://patents.google.com/patent/WO2014170758A3/en`, дата обращения: 27.05.2020

[2] `https://www.augmented.city/`, дата обращения: 27.05.2020

сегментации изображений. На основе этого подхода создана система, реализованная в виде библиотеки на языке C++ и консольного приложения, использующего её. Для сравнения её с существующими, а также для исследования влияния семантической сегментации на распознавание объектов были проведены эксперименты.

По их результатам применение сегментации в различных конфигурациях дало прирост точности от 0.5% до 1.5%, прирост mAP – от 0.01 до 0.03. В целом же полученная система на 38% превзошла по точности Augmented.City, основывающуюся на классических алгоритмах компьютерного зрения, и показала сопоставимый со state of the art подходом DELF результат, уступив по точности 5% и по mAP 0.016, но при этом превзойдя его по времени почти в 100 раз. Реализацию системы можно найти на сайте: `https://github.com/ucLh/city_obj_retrieval`.

1. Постановка задачи

Цель данной работы – улучшить качество поиска изображений по содержимому (image retrieval) при распознавании городских объектов для предоставления информации о них в интерактивном режиме. Для достижения этой цели были поставлены следующие задачи.

1. Изучить предметную область.
2. Разработать подход на основе нейронных сетей, который выявляет подобие изображений городских объектов.
3. Протестировать разработанный подход на городских сценах и проанализировать результаты.

2. Обзор

2.1. Augmented.City — система дополненной реальности в городских сценах

Система Augmented.City для решения задачи image retrieval применяет так называемый мешок визуальных слов (Bag of Visual Words, BOVW) [25], который широко применим для решения подобного рода задач.

Идея данного метода позаимствована из обработки естественного языка (Natural Language Processing, NLP), где для информационного поиска применяется мешок слов (Bag of Words, BOW) [25]. В Bag of Words подсчитывается частота появлений слов в тексте, на основе чего делается вывод о его ключевых словах. С изображениями применяется та же концепция, только в качестве слов выступают ключевые точки изображения (image features), например, углы или границы предметов.

Главная идея BOVW заключается в том, чтобы представить изображение в виде множества ключевых локальных точек и их дескрипторов — векторов, описывающих ключевую точку. Получить их можно с помощью таких алгоритмов детектирования, как SURF [3], SIFT [19], ORB [20] и т.п. Здесь следует заметить, что в Augmented.City применяется ORB, основанный на методе выделения ключевых точек FAST [23] и дескрипторе BRIEF [2]. В отличие от алгоритмов SURF и SIFT ORB распространяется свободно (MIT license), немного уступая им по точности, но превосходя их по скорости работы [20].

После получения дескрипторов производится их кластеризация с помощью алгоритма K-Means [14](где k – гиперпараметр). Центры каждого из кластеров изображения попадают в запасы слов (vocabularies), по которым потом получаются диаграммы частотности (frequency hystograms). Гистограммы изображений из базы данных и образуют мешок слов.

Получая на вход изображение-запрос, мы детектируем ключевые точки, извлекаем дескрипторы, кластеризируем их и строим гистограм-

мы такой же длины, как и в мешке слов. С помощью полученного мешка слов мы можем определить ближайшее изображение к запросу, например с помощью алгоритма k-nn (k nearest neighbours) [6].

К достоинствам данного метода можно отнести то, что для его работы не требуется никаких дополнительных данных, в отличие от нейронных сетей, поэтому его можно довольно быстро начать применять. Однако по точности они уступают последним [21].

Кроме того, в своей работе Bag of Visual Words использует локальные ключевые точки, число которых может достигать сотен или тысяч, поэтому памяти расходует больше, чем нейросетевые подходы, использующие глобальные ключевые точки и устанавливающие взаимооднозначное соответствие между изображением и ключевой точкой.

2.2. Система извлечения локальных ключевых точек DELF (DEep Local Features)

DELF[15] — этот метод решает задачу image retrieval, комбинируя классический и нейросетевой подходы, то есть находит локальные ключевые точки, используя глубокое обучение, показывая state-of-the-art результат. DELF применяет глубокую свёрточную нейронную сеть, на последнем из слоев которой обучают attention модуль.

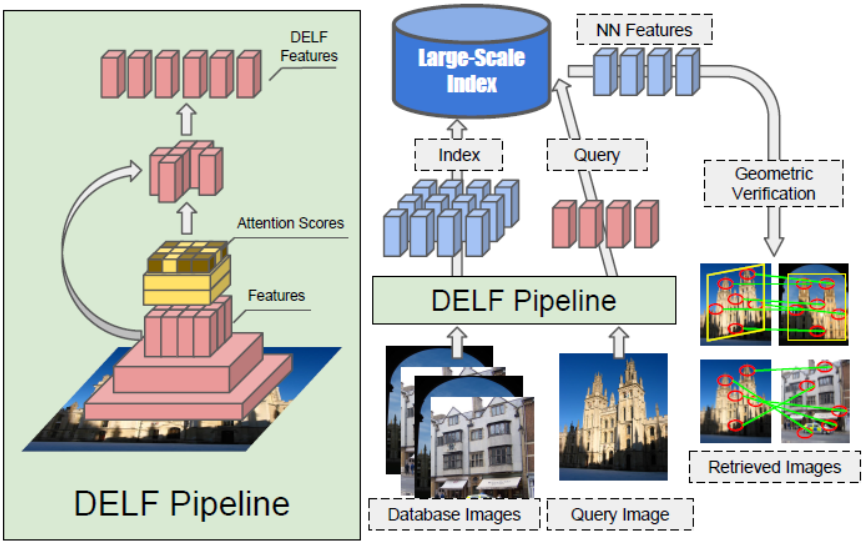

Рис. 1: Пайплайн DELF. Изображение взято из [15]

Сначала входное изображение пропускается через свёрточную сеть, и результат работы последнего свёрточного слоя принимают за множество локальных дескрипторов ключевых точек (dense local features). После из данного множества с помощью attention-модуля выбираются наиболее релевантные ключевые точки. Это позволяет улучшить точность и вычислительную эффективность системы. Затем происходит уменьшение размерности полученных точек применением метода главных компонент (PCA), и на этом работа метода DELF заканчивается.

Для того чтобы решить задачу image-retrieval с помощью DELF, применяется поиск ближайших соседей и геометрическая верификация с использованием алгоритма RANSAC [11].

Сначала извлекаются дескрипторы особых точек изображений некоторой базы данных, классы которой мы знаем и с которой планируется сопоставлять (match) изображение-запрос q.

После получения запроса, его пропускают через DELF, также получая дескрипторы. После этого в базе ищется k ближайших соседей для каждого дескриптора запроса (k — гиперпараметр). Будем считать, что получилось совпадение, если дескриптор изображения из базы попал в k соседей. Тогда, сосчитав количество таких совпадений для каждого элемента базы, мы извлекаем n (n — гиперпараметр) штук с максимальным количеством совпадений.

После этого к дескрипторам полученных изображений применяется алгоритм RANSAC, который осуществляет геометрическую верификацию, таким образом убирая шумы. Ближайшим к запросу признается то изображение t, у которого осталось наибольшее количество совпадений, и q присваивается класс t.

Главным недостатком данного подхода следует назвать большие затраты по времени на исполнение алгоритма RANSAC, запуск которого необходим для каждой пары изображений.

2.3. Глубокое метрическое обучение (Deep Metric Learning)

Целью метрического обучения является нахождение подходящей меры похожести между изображениями так, чтобы расстояния для изображений одного объекта были относительно малы, а для изображений разных объектов — относительно велики. Способность определять близость тех или иных изображений может позволить осуществить поиск изображений по содержимому (image retrieval), особенно на больших наборах данных.

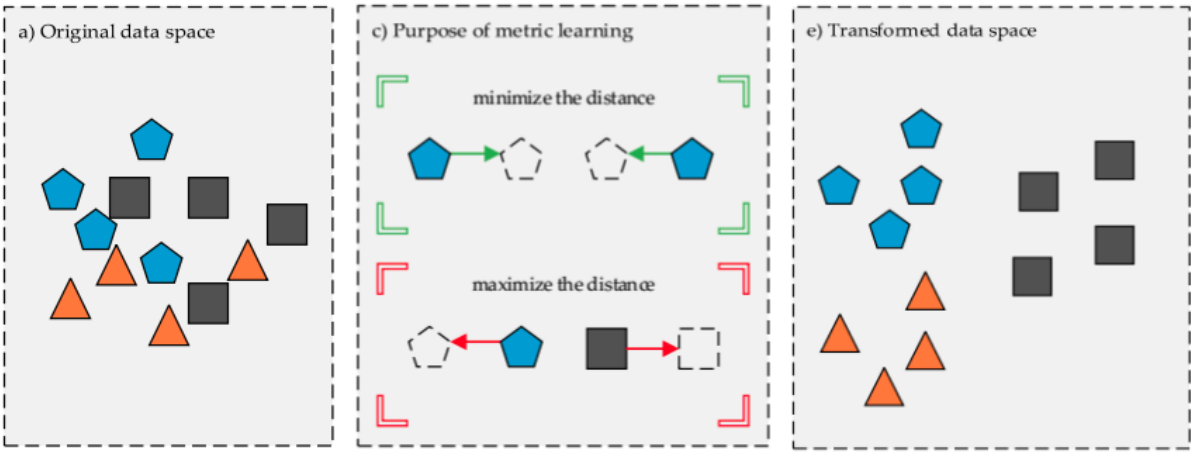

Рис. 2: Метрическое обучение. Изображение взято из [17]

Глубокое метрическое обучение использует для определения похожести объектов глубокие свёрточные нейронные сети. Для этого они сопоставляют каждому изображению n-мерный характеристический вектор (точку в n-мерном пространстве, назовём его характеристическим пространством). После этого для пары таких векторов вычисляется расстояние, например с помощью Евклидовой нормы, которое и считается мерой близости данной пары изображений.

Для того чтобы расстояния между точками в характеристическом пространстве были малы для представителей одного класса и велики для остальных классов, могут дополнительно применяться такие функции потери, как Center loss [9], Contrastive loss [13], Arcface [8].

Одним из главных преимуществ данного подхода является то, что обучив сеть на одних классах изображений, на этапе тестирования её

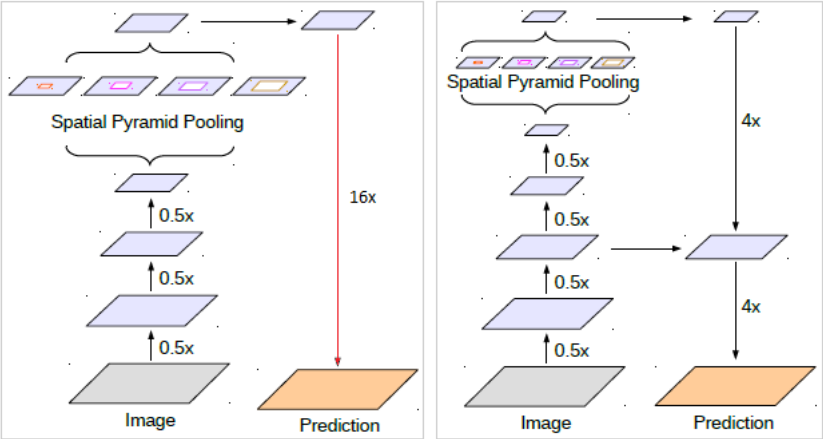

Рис. 3: Сравнение архитектур DeepLabv3 (слева) и DeepLabv3+ (справо). Изображение взято из [10]

можно использовать на совершенно других классах, просто сравнивая характеристические вектора изображений.

2.4. Нейронная сеть для семантической сегментации DeepLabv3+

Нейронная сеть DeepLabv3+[10] — это сеть, предназначенная для семантической сегментации, целью которой является присвоение некоторой семантической метки (“человек”, “машина”, “дорожный знак” и так далее) каждому пикселю изображения. Основным отличительным признаком данной нейросети является добавление простого, но эффективного декодера, который улучшает выделение границ меток.

В предыдущих версиях данной нейросети использовалось “наивное” декодирование, а именно билинейное растяжение в 16 раз (красная линия на рис 3). Однако такой способ декодирования не всегда мог успешно восстанавливать границы классов сегментированных объектов. В DeepLabv3+ результат работы кодирующей части сети (encoder) растягивается только в 4 раза, а затем конкатенируется с низкоуровневыми ключевыми точками (low-level features), которые перед этим пропускают через свёртки с ядром 1×1 для корреляции размерностей (см. рис 12). К полученному результату применяют свёртку 3×3, а затем растягивают ещё в 4 раза.

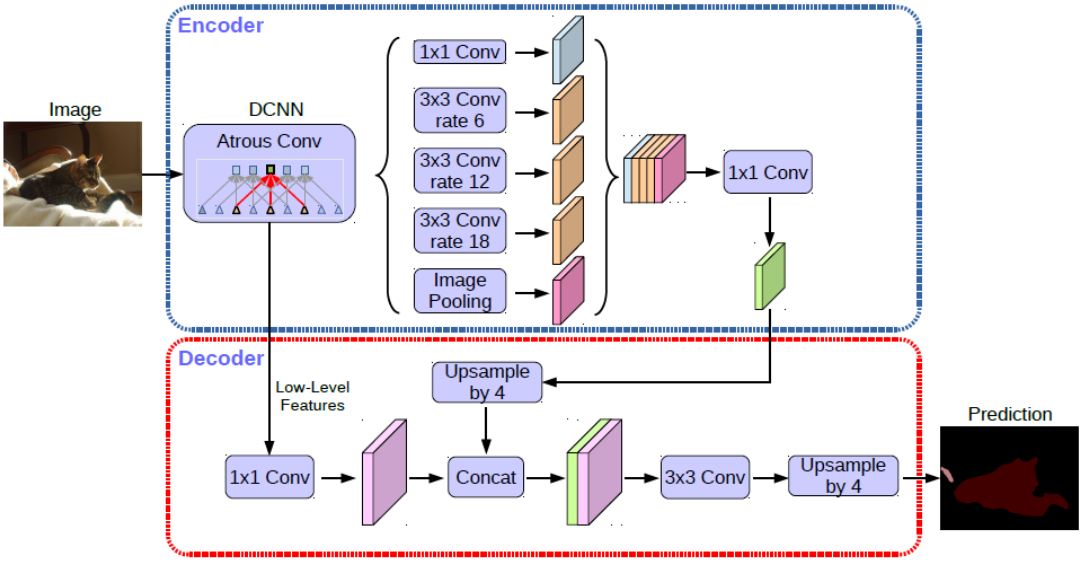

Рис. 4: Строение сети DeepLabv3+. Изображение взято из [10]

Также стоит заметить, что на текущий момент DeepLabv3+ считается state-of-the-art подходом.

2.5. Архитектуры свёрточных нейронных сетей

Рассмотрим две архитектуры, которые сейчас лежат в основе практически всех свёрточных сетей.

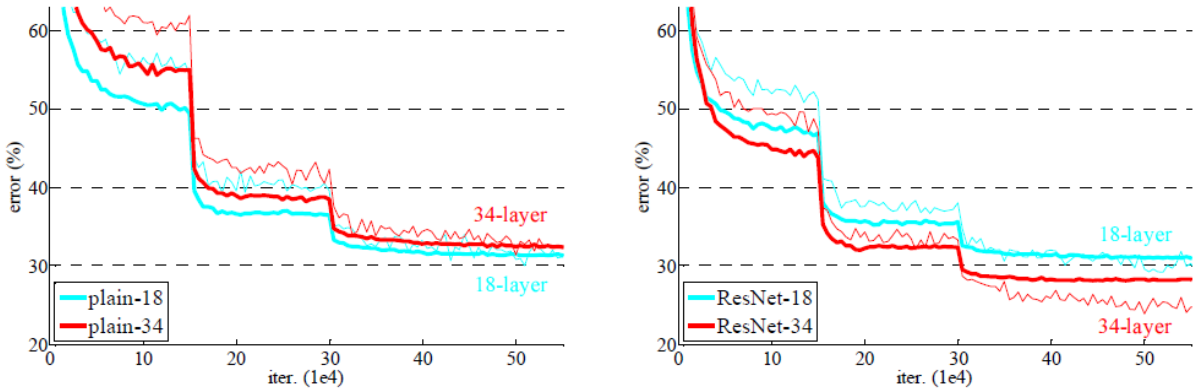

Рис. 5: Сравнение функции потерь сетей, использующих только последовательные соединения слоев с сетями ResNet архитектуры при обучении на ImageNet[16]. Толстые кривые обозначают потерю при валидации, тонкие – при обучении. Изображение взято из [7].

2.5.1. Остаточная нейронная сеть (Residual neural network, ResNet)

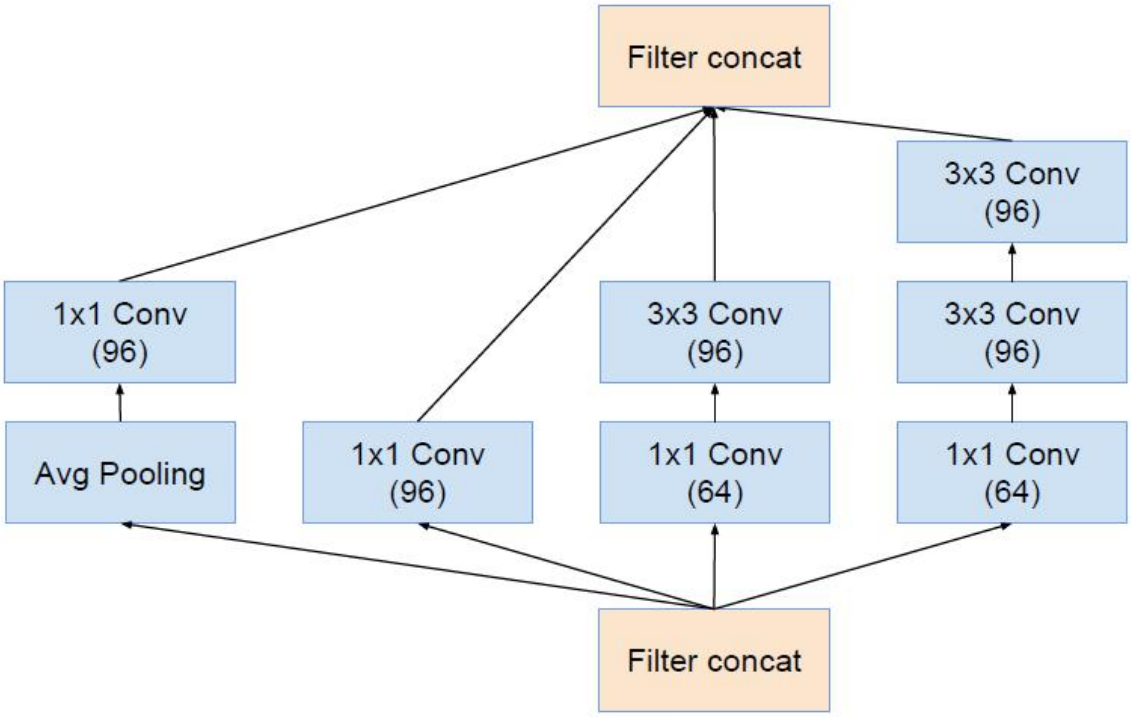

Рис. 6: Строение Inception модуля. Изображение взято из [26].

Остаточная нейронная сеть (Residual neural network)[7] – это свёрточная нейронная сеть, которая использует сквозные соединения (skip-connections), чтобы пропускать блоки свёрточных слоёв, соединённые напрямую, при этом не пропуская за раз больше одного блока. Таким образом, на вход блоку подаются как входные данные предыдущего блока, так и результат его работы. Это снижает угасание градиентов, проходящих через слои во время обучения, и даёт возможность увеличить количество слоев, что в свою очередь повышает выразительную мощность сети и её результаты (см. рис 5).

2.5.2. Нейронная сеть Inception

Свёрточная нейронная сеть архитектуры Inception[12] — это нейросеть, которая состоит из особых Inception модулей (см рис. 6). Главная особенность такого модуля – это использование свёрток с ядром (kernel)

1x1 до свёрток с ядром 3x3, являющихся более “затратными”. Данный приём даёт возможность понизить количество необходимых для работы сети вычислений, а также уменьшить время её обучения, что позволяет ей быть довольно эффективной. Существует несколько версий данной нейронной сети, последней из которых является Inception-v4[26].

3. Предлагаемый нейросетевой подход

3.1. Обзор подхода

Для распознавания городских объектов было решено взять за основу глубокое метрическое обучение, так как оно хорошо показало себя в решении задачи image retrieval на данных разного характера [24][17]. Кроме того, для того чтобы попытаться снизить негативное влияние двигающихся объектов на точность распознавания, было принято решение использовать нейронную сеть для семантической сегментации, чтобы вырезать данные объекты из изображения.

Таким образом, структура подхода состоит из двух главных компонентов:

1. Распознающая свёрточная нейронная сеть.

2. Нейронная сеть для семантической сегментации.

Сеть для распознавания является глубокой свёрточной нейронной сетью. Во время тренировки на её последний свёрточный слой также добавляется слой регрессии, чтобы в результате получить скаляр, который представляет вероятность того, что входное изображение является примером того или иного городского объекта (принадлежит тому или другому классу), то есть решается задача классификации.

После тренировки она принимает на вход изображение, сопоставляет ему глобальную ключевую точку (n-мерный выход последнего слоя) и находит ближайший к нему по евклидовой норме n-мерный вектор (будем называть его характеристическим) из тех, что были получены из уже имеющихся в базе изображений. Будем считать, что расстояние между парой изображений d — это расстояние между соответствующими им векторами. Таким образом, можно получить городской объект s самый близкий к входному и присвоить ему класс s. Классом здесь считаются те изображения, которые мы решили ассоциировать с некоторым городским объектом.

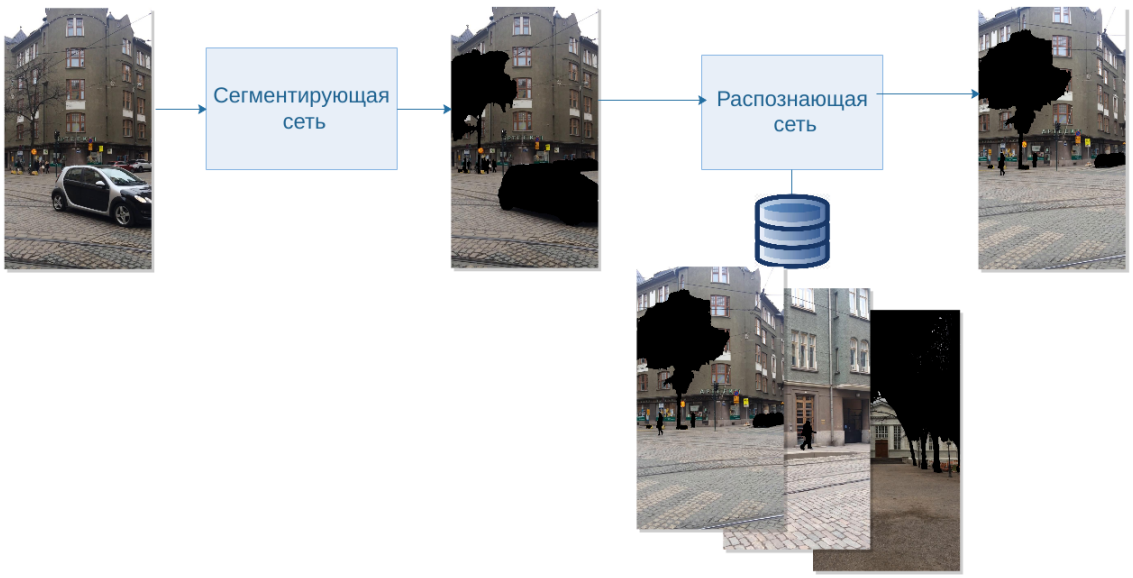

Рис. 7: Сегментирующий подход.

Семантическая сегментация изображения заключается в том, чтобы назначить каждому его пикселю определенный класс, например “здание”, “человек”, “небо” и так далее. Обладая данной информацией можно выделить интересующую часть изображения и отсечь всё остальное.

Таким образом, принцип работы предлагаемого подхода (см рис. 7) состоит в том, что сначала из каждого изображения базы вырезается та его часть, которой сегментационной сетью был присвоен класс “здание”. На этапе инференса (inference, процесс пропуска входных данных через уже обученную модель без изменения выученных параметров) изображение-запрос q, полученное на вход, аналогичным образом маскируется. После этого q сопоставляется изображение из базы с помощью распознающей сети.

Также возможны комбинированные варианты (см рис. 8), когда сеть для распознавания одновременно пытается сопоставлять маскированные запросы с маскированными и обычные с обычными. В таком случае выбор между ближайшим маскированным изображением и ближайшим немаскированным производится в пользу того, которое ближе к соответствующему запросу по расстоянию d.

Здесь необходимо заметить, что на этапе тестирования можно использовать классы городских объектов, которые не были представлены

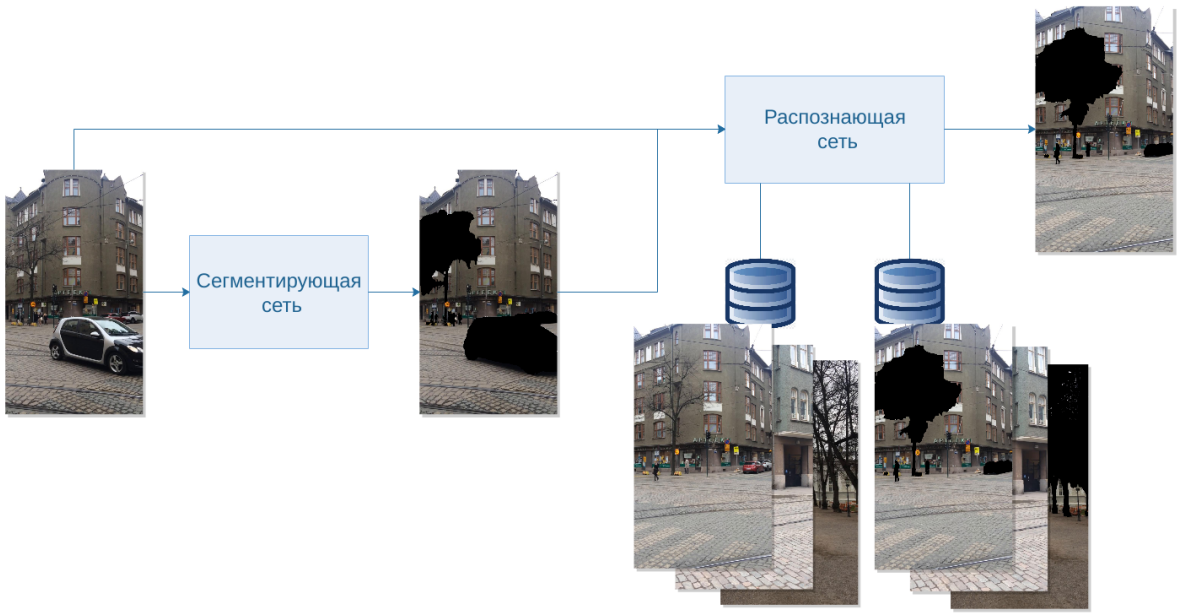

Рис. 8: Смешанный подход.

на этапе обучения распознающей сети, так как в основу подхода было положено метрическое обучение. Кроме того, это также позволяет постепенно расширять базу изображений без необходимости обучать сеть заново.

3.2. Реализация предлагаемого подхода

3.2.1. Выявление требований и сценарии использования

При проектировании системы распознавания городских объектов к ней были выработаны следующие требования:

- возможность встраивания в существующую систему дополненной реальности в городских сценах Augmented.City, в которой сопоставление снимков городских объектов является промежуточной задачей,
- возможность сопоставления входного изображения заданной базе изображений с помощью заданной нейронной сети,
- возможность осуществить семантическую сегментацию изображения с помощью заданной нейронной сети,

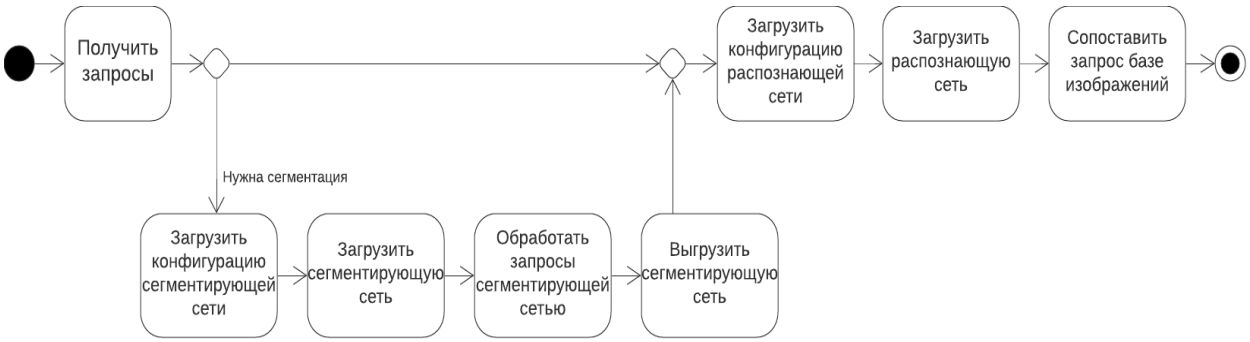

Рис. 9: Диаграмма активностей для основного сценария работы с библиотекой

- возможность маскирования изображения на основе семантической сегментации,
- отсутствие привязанности к определённым нейронным сетям.

На основе выявленных требований было решено разрабатывать систему в виде библиотеки. Для её реализации был выбран язык C++, так как подсистема поиска изображений по содержанию (image retrieval) в Augmented.City разработана на C++. Кроме того, многие библиотеки компьютерного зрения, самая популярная из которых OpenCV[3], разработаны на этом языке, что должно упростить потенциальную интеграцию разрабатываемой библиотеки с ними. Также для данного языка существует API фреймворка TensorFlow [27], который позволяет работать с нейронными сетями, поэтому данный фреймворк было решено использовать в реализации библиотеки.

Основным сценарием использования является предоставление списка изображений ближайших к входному из заданной базы. В соответствии с этим были выделены следующие шаги, которые пользователю необходимо проделать:

- сконфигурировать распознающую нейронную сеть,
- сконфигурировать при необходимости сегментирующую нейронную сеть,

[3] `https://github.com/itseez/opencv`, дата обращения: 29.05.2020

- указать путь до изображений-запросов,
- указать, надо ли маскировать запросы сегментирующей сетью,
- запустить сопоставление запросов с базой изображений и получить список ближайших.

Этим действиям соответствует диаграмма активностей на рисунке 9. Стоит добавить, что библиотека перед запуском сопоставления анализирует базу изображений и, если туда были добавлены новые данные, обновляет соответствующие ей характеристические векторы.

3.2.2. Архитектура системы

На рисунке 10 изображена диаграмма классов реализованной библиотеки.

Класс TensorFlowWrapperCore отвечает за настройку сторонней библиотеки TensorFlow для инференса (inference) нейронных сетей. Он загружает сеть, которая представляется в виде вычислительного графа, и запускает на нём вычисления. В этом вычислении на вход ожидается тензор, представленный объектом класса Tensor из библиотеки TensorFlow, на выход также подаётся тензор. Преобразованием изображений в тензоры также занимается класс TensorFlowWrapperCore.

Класс TensorFlowEmbeddings является наследником класса TensorFlowWrapperCore. Данный класс совершает преобразование тензора, получившегося в результате инференса изображения распознающей сетью, в массив вещественных чисел, который представляет собой характеристический вектор изображения.

Похожую роль играет класс TensorFlowSegmentator, который также является потомком класса TensorFlowWrapperCore, только тензор, получающийся после сегментации изображения соответствующей нейронной сетью, он преобразует в индексированную или цветную маску, которая отражает результаты сегментации.

Интерфейс IInferenceHandler и наследующие его интерфейсы IEmbeddingsInferenceHandler и ISegmentationInferenceHandler служат

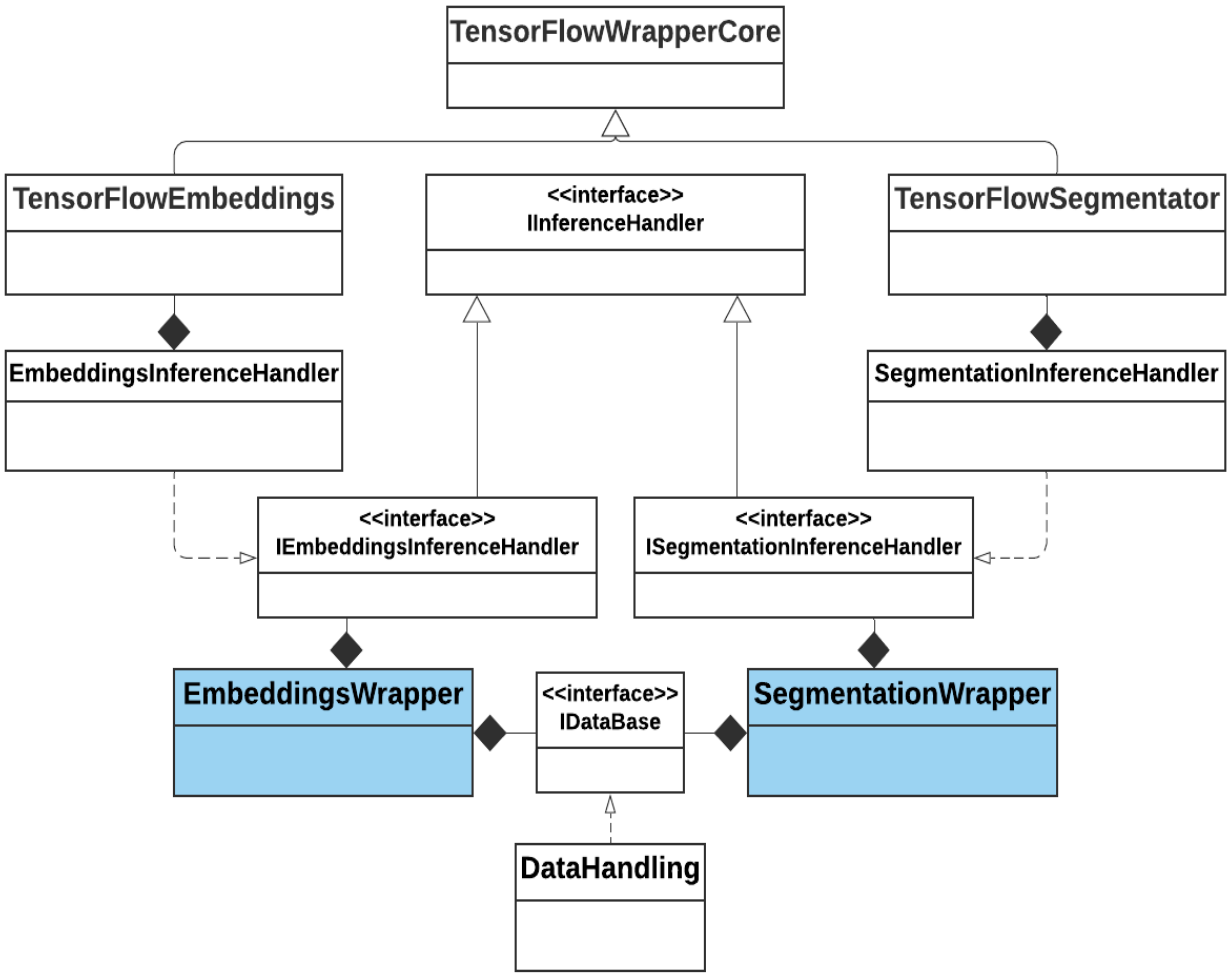

Рис. 10: Архитектура библиотеки

для того, чтобы обеспечивать взаимодействие с классами TensorFlowEmbeddings и TensorFlowSegmentator соответственно. Их адаптерами являются классы EmbeddingsInferenceHandler и SegmentationInferenceHandler, которые реализуют названные выше интерфейсы.

Класс EmbeddingsWrapper является фасадом, который оборачивает в себе работу с распознающей нейронной сетью и необходимую для этого работу с данными. Делает он это с помощью соответствующих интерфейсов IEmbeddingsInferenceHandler и IDataBase. Именно с данным классом предлагается работать пользователям библиотеки, также как и с классом SegmentationWrapper, который также является фасадом, только для сегментации.

Класс DataHandling реализует интерфейс IDataBase и отвечает за

работу с данными. Он осуществляет загрузку конфигурационных файлов нейронных сетей, логирование ошибочных предсказаний распознающей сети, загрузку карты цветов для семантической сегментации. Кроме того, данный класс также реализует запись полученных распознающей сетью характеристических векторов изображений в файл и чтение их оттуда.

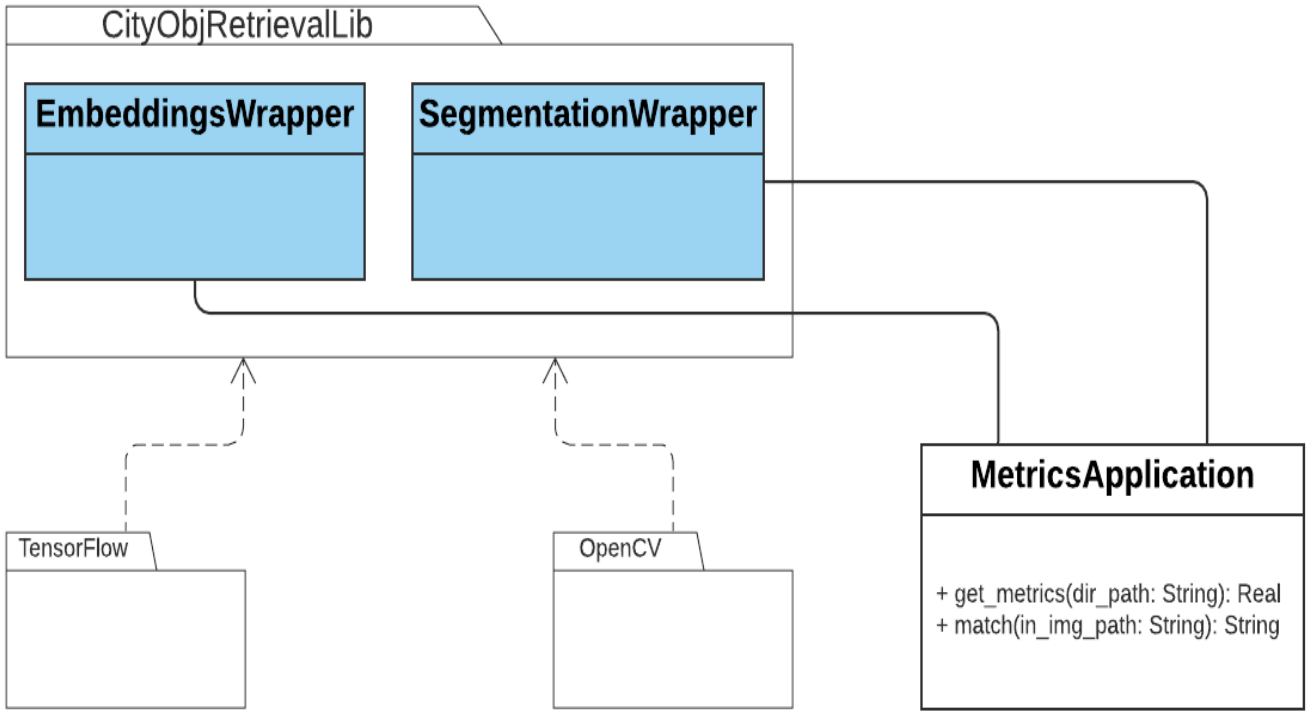

Рис. 11: Архитектура прототипа

Помимо библиотеки также было реализовано консольное приложение (см. рис. 11), которое принимает на вход путь к папке с изображениями-запросами и режим работы, сопоставляет их с базой изображений и при этом считает метрику. Для этого оно подключает два заголовочных файла библиотеки, один из которых содержит функциональность для работы с разпознающей сетью, а другой – с сегментирующей. Стоит также заметить, что библиотека не зависит от данного приложения и может быть использована отдельно.

4. Проведение экспериментов

4.1. Формирование обучающей выборки

Для обучения распознающей сети был вручную собран набор данных (dataset, обучающая и тестовая выборки вместе), состоящий из фотографий различных городских объектов Санкт-Петербурга, а также дополнен изображениями, собранными с помощью поисковой системы Google. Объём обучающей выборки составил примерно 41 400 изображений, распределённых среди 690 классов.

Рис. 12: Примеры изображений из обучающей выборки, результаты их сегментации и маскирования.

Следует заметить, что данная выборка была расширена с помощью

различных техник, позволяющих получить новые данные из имеющихся, таких как поворот на угол, отражения, дисторсия, изменение масштаба и т.д. Однако самой интересной из применённых техник надо признать расширение зимними изображениями, полученными с помощью генеративной нейронной сети, польза которой для обучения была показана в [30].

4.2. Обучение распознающей сети

При выборе распознающей сети, которая будет учиться сопоставлять изображению ключевой вектор (feature vector), была выбрана глубокая свёрточная сеть Inception-Resnet-v1, которая является сочетанием сетей Inception и Resnet, так как она уже успешно применялась в различных задачах распознавания, в частности в распознавании лиц [24]. В качестве функции потери было решено сделать выбор в пользу категориальной кросс-энтропии (softmax), из-за её способности сходиться на относительно малых набоах данных. Также решено использовать центральную функцию ошибки (center loss)[9], побуждающую изучаемые свойства быть хорошо разделимыми и входящую в число самых эффективных функций потери для распознавания лиц[29]. Итоговая функция ошибки представляет собой сумму softmax и center loss, причём последняя входит в сумму с коэффициентом α, который является гиперпараметром.

Для обучения распознающей сети был выбран фреймворк глубокого обучения TensorFlow[27]. В качестве кандидата также рассматривался фреймворк Pytorch[1], однако было решено отдать предпочтение TensorFlow. Главной причиной стало то, что в разработанной библиотеке для работы с нейронными сетями используется TensorFlow. Кроме того, следует сказать, что в данном фреймворке присутствуют все необходимые в данной работе нейронные сетей, а также у него существует ширкое сообщество программистов.

При тренировке распознающей сети применялся оптимизатор Adam[18], в котором для ускорения схождения сети сначала исполь-

зовался достаточно большой коэффициент скорости обучения (learning rate), значение которого было равно 0.005, и делилось пополам всякий раз, когда функция ошибки переставала значительно изменяться. Кроме того, стоит отметить, что для тренировки сети была применена центральная функция ошибки (center loss)[9] с коэффициентом α равным 0.7, который был выбран, исходя из результатов авторов подхода.

В качестве сети для семантической сегментации изображений было решено взять нейросеть DeepLabv3+, обученную на наборе данных Cityscapes[4], который состоит из попиксельно размеченных на классы городских сцен. Этот выбор обоснован тем, что на данный момент она является state-of-the-art моделью[10].

Стоит также отметить, что с помощью нейронной сети DeepLabv3+ была получена альтернативная версия обучающей выборки, где из каждого изображения были вырезаны двигающиеся объекты, а именно пешеходы, деревья и автомобили. Обучение распознающей сети на данной выборке также было произведено.

4.3. Анализ результатов

Тестирование проводилось на классах, не присутствовавших в обучении распознающей сети. Для этого было собрано 19243 фотографии (202 класса), из которых 2786, в которых был представлен 61 класс, использовались в качестве запросов. Из оставшихся снимков было сформировано две базы изображений. Одна из них содержала 2015 снимков из тех же классов, что и в запросах. Вторая содержала все те же кадры, что и первая, а также 141 класс из 14 442 изображений. Другие модели также тестировались на данной выборке.

Кроме того, для проведения экспериментов с помощью семантической сегментации были получены альтернативные версии тестовой выборки с вырезанными прохожими и автомобилями. Также из тестовых изображений были вырезаны деревья, так как они в разные времена года выглядят по-разному и могут помешать распознаванию городского объекта. Чтобы оценить, как изменится эффективность распознавания

при вырезании их из изображения, было решено рассмотреть две сегментированные версии тестовой выборки, как с вырезанными деревьями, так и без.

Так как распознающая сеть решает задачу image retrieval, то результатом её работы является отсортированный список изображений, которые "по мнению" сети являются ближайшими к входному, от самого похожего к наименее похожему. Все метрики в экспериментах вычисляются на основе данного списка. Чтобы сократить время их подсчёта рассматривается список только из 50 ближайших, а изображения, не вошедшие в это число, автоматически считаются непохожими на входное. Данный параметр был выбран экспериментально, при его увеличении значения метрик изменялись незначительно.

При проведении экспериментов измерялась точность (accuracy), которая считается, как отношение количества правильно распознанных изображений к общему количеству запросов. Точность измерялась в двух версиях: топ-1 и топ-4. В первом случае распознавание считается правильным, если первое изображение в списке похожих принадлежит тому же классу, что и входное. Напомню, что классом здесь считаются те изображения, которые мы решили ассоциировать с некоторым городским объектом. Во втором случае распознавание верно, если класс входного изображения присутствует среди четырёх ближайших классов в списке.

Помимо этого было принято измерить метрику mAP (mean average precision) [28], которая часто используется при оценке качества решения задач image retrieval. Существуют разные варианты данной метрики, и различия, как правило, заключаются в том, учитывается количество изображений в базе или порядок их появления в списке ближайших. В данной работе было решено выбрать версию, представленную в статье [28] и учитывающую порядок изображений в списке похожих: чем дальше правильные классы в списке, тем меньше значение метрики. Такой выбор обусловлен несколькими причинами. Во-первых, в одном классе может быть много изображений, сделанных с разных ракурсов, и система, успешно выдав первые несколько изображений из нужного класса,

получит низкий показатель метрики, не выдав остальную сотню, которая была сделана с других ракурсов и, возможно, сильно отличается от запроса. Во-вторых, так как в базе изображений могут быть визуально похожие изображения из разных классов, важно оценить, как хорошо распознающая сеть их отличает.

Вычисляется mAP по следующей формуле:

$$mAP = \frac{1}{N} \sum_{j=1}^{N} \frac{1}{max(P, 1)} \sum_{i=1}^{k} \frac{rel(i)}{i} \tag{1}$$

Здесь:

- k – это количество изображений в списке ближайших (в нашем случае оно равно 50);
- P – количество изображений из нужного класса в списке;
- N – общее количество запросов;
- $rel(i)$ – индикаторная функция, которая равна 1, если i-е изображение в списке; принадлежит тому же классу, что и запрос, и 0 – в противном случае.

В приведённых ниже результатах экспериментов используются следующие обозначения:

- q – запросы;
- q_s – запросы с вырезанными автомобилями и пешеходами;
- q_{st} – запросы с вырезанными автомобилями, пешеходами и деревьями;
- s – база изображений;
- s_s – база изображений с вырезанными автомобилями и пешеходами;

№	Модель	Запросы	База	Точность топ-1	Точность топ-4	mAP
1	DELF	q	s	84.82%	94.79%	0.756
2	DELF(100)	q	s	43.13%	50.63%	0.342
3	net	q	s	78.50%	88.80%	0.727
4	net	q_{st}	s_{st}	78.93%	88.98%	0.736
5	net	q_s	s_s	78.97%	88.48%	0.736
6	seg	q_{st}	s_{st}	76.70%	89.23%	0.726

Таблица 1: Результаты, полученные с помощью сегментации изображений.

- s_{st} – база изображений с вырезанными автомобилями, пешеходами и деревьями;
- s' – расширенная база изображений;
- s'_{st} – расширенная база изображений с вырезанными автомобилями, пешеходами и деревьями;
- net – распознающая сеть;
- seg – распознающая сеть, обученная на изображениях с вырезанными двигающимися объектами.

Из таблицы 1 можно увидеть, что применение сегментации дало увеличение по точности топ-1 примерно на 0.5%, по точности топ-4 – увеличение составило всего 0,1% на данных с вырезанными деревьями, а mAP возросла почти на 0.01. На данных, где вырезались только автомобили и пешеходы, точность топ-4 снизилась относительно экспериментов 3 и 4, поэтому для дальнейших экспериментов решено использовать версию с вырезанными деревьями.

Результаты модели, обучавшейся на маскированных данных, оказались по точности топ-1 на 1.8% хуже модели, полученной путём тренировки на обычных данных, но по топ-4 превзошли результаты не только эксперимента 3, но и 4, 5, поэтому однозначный вывод о её состоятельности сделать трудно.

В DELF при решении сопоставлении изображений используется алгоритмы k-nn и RANSAC. Сначала все изображения сортируются по близости к запросу с помощью алгоритма k-nn, а после этого происходит их пересортировка с помощью алгоритма геометрической верификации RANSAC. Количество ближайших соседей, которые подвергаются верификации, является гиперпараметром. В эксперименте 1, верификации подвергаются все изображения базы. В эксперименте 2 была предпринята попытка снизить количество таких изображений до 100, однако это повлекло за собой заметное ухудшение результатов.

Про алгоритм RANSAC следует заметить, что его выполнение довольно затратно по времени, и на процессоре Intel(R) Core(TM) i5-6300HQ CPU @ 2.30GHz при оперативной памяти 8 Гб время работы данного алгоритма составляет не менее 0.2 секунды. В целом, при использовании видеокарты NVIDIA GeForce GTX 950M с памятью 4 Гб среднее время обработки запроса подходом DELF в эксперименте 1 составляет 431.2 секунды. Для сравнения, полученный в данной работе подход с применением сегментации при тех же технических ресурсах тратит на это в среднем 4.352 секунды.

Таким образом, для эффективной работы DELF с базами изображений, состоящих из тысяч снимков, может потребоваться довольно много ресурсов, что может создать трудности в сложных системах, которые занимаются не только распознаванием городских объектов, но и ещё, например, локализацией на его основе.

№	Модель	Запросы	База	Точность топ-1	Точность топ-4	mAP
7	net	q_{st}	s, s_{st}	79.47%	89.45%	0.738
8	net	q, q_{st}	s, s_{st}	79.33%	89.09%	0.740
9	net+seg	q, q_{st}	s, s_{st}	78.07%	89.23%	0.736

Таблица 2: Результаты смешанных подходов.

Далее были проведены замеры результатов комбинированных подходов. В эксперименте 7 маскированные запросы сопоставлялись как с маскированными изображениями, так и с обычными, в 8 и 9 экспериментах обычные сопоставлялись с обычными, а маскированные – с

маскированными. Процесс сопоставления подробно описан в разделе 3.1.

Результаты комбинированных подходов оказались лучше, чем результаты, показанные в экспериментах 3-6. Причём по точности лучший результат был показан в 7 эксперименте, а по mAP – в 8, когда сеть попарно сравнивала обычные снимки и снимки с вырезанными двигающимися объектами. Также стоит заметить, что отставание от DELF по mAP снизилось до 0.016, а по точности до 5%.

№	Модель	Запросы	База	Точность топ-1	Точность топ-4	mAP
10	Aug.City	q	s'	34.01%	57.69%	–
11	net	q	s'	70.64%	79.54%	0.612
12	net	q_{st}	s'_{st}	72.04%	80.69%	0.643
13	net	q_{st}	s', s'_{st}	61.49%	77.53%	0.586
14	net	q, q_{st}	s', s'_{st}	66.30%	78.00%	0.605
15	net+seg	q, q_{st}	s', s'_{st}	64.14%	77.49%	0.599

Таблица 3: Результаты, полученные на расширенном наборе данных.

После этого были проведены замеры на увеличенной тестовой выборке s'. Запросы при этом остались без изменений, таким образом в новых изображениях не содержалось классов из запросов. Это было сделано для того, чтобы проверить, как сеть сопоставляет снимки на большем количестве данных. Подход DELF на расширенном наборе не тестировался в силу его медленности.

На расширенных данных эффективность комбинированных подходов значительно снизилась, но, как видно из экспериментов 11 и 12, маскирование изображений дало улучшение по точности топ-1 в 1.4% и по mAP – в 0.3. Также надо заметить, что результаты системы Augmented.City были предоставлены её авторами, которые не измеряли свой подход по метрике mAP.

Заключение

В ходе выполнения данной работы были получены следующие результаты.

Результаты

1. Проведён обзор предметной области. В ходе данного обзора рассмотрены:
   - способы решения задачи распознавания городских объектов в системах Augmented.City и DELF, а также общий подход к решению задачи image retrieval с помощью глубокого метрического обучения;
   - нейронная сеть для семантической сегментации DeepLabv3+;
   - архитектуры нейронных сетей ResNet и Inception.
2. Разработан подход выявления подобия городских объектов, использующий распознающую сеть архитектуры Inception-ResNet и сегментирующую сеть архитектуры DeepLabv3+. Распознающая сеть основывается на подходе глубокого метрического обучения. Данный подход реализован в виде библиотеки на языке C++, которая поддерживает:
   - распознавание городских объектов;
   - осуществление семантической сегментации изображений;
   - маскирование элементов изображения с помощью семантической сегментации.
3. Вручную собраны обучающая и тестовая выборки, содержащие городские сцены Санкт-Петербурга. Проведено обучение распознающей сети. Проведены эксперименты, по результатам которых улучшение относительно существующей системы Augmented.City по точности топ-1 составило 38%, по точности топ-4 – 23%. Результаты подхода DELF оказались выше по метрикам точности на 5%

и по mAP – на 0.016, однако время работы данного подхода на собранной тестовой выборке почти в 100 выше, чем у полученного в данной работе.

Возможные пути улучшения и развития предложенного подхода

Одним из возможных путей улучшения данной работы является применение алгоритмов кластеризации для сокращения времени сопоставления запроса базе. Можно попробовать кластеризовать характеристические вектора изображений, принадлежащих одному классу, сопоставить каждому кластеру некий центр, например взять среднее всех векторов данного кластера. Тогда при сопоставлении характеристического вектора запроса базе его можно будет сравнивать не со всеми изображениями, а только с центрами. Выгода по времени очевидна, однако для этого необходимо постараться хорошо выделить центры, иначе эффективность сопоставления может снизиться в разы.

Кроме того, можно несколько расширить задачу, например, распознавать также и рестораны, магазины и кафе внутри торговых центров, то есть работать не только с уличными снимками. Данная задача особенно интересна если учесть то, что в торговом центре фильтрация данных с помощью GPS даст нам изображения практически всех заведений этого центра, и сопоставление можно будет осуществить практически только на основе изображений.

Список литературы

[1] Automatic differentiation in PyTorch / Adam Paszke, Sam Gross, Soumith Chintala et al. // NIPS-W. — 2017. — URL: https://openreview.net/pdf?id=BJJsrmfCZ (online; accessed: 29.05.2020).

[2] BRIEF: Binary Robust Independent Elementary Features / Michael Calonder, Vincent Lepetit, Christoph Strecha, Pascal Fua. — Vol. 6314. — 2010. — 09. — P. 778–792.

[3] Bay Herbert, Tuytelaars Tinne, Van Gool Luc. SURF: Speeded up robust features. — Vol. 3951. — 2006. — 07. — P. 404–417.

[4] The Cityscapes Dataset for Semantic Urban Scene Understanding / Marius Cordts, Mohamed Omran, Sebastian Ramos et al. // CoRR. — 2016. — Vol. abs/1604.01685. — URL: http://arxiv.org/abs/1604.01685 (online; accessed: 29.05.2020).

[5] Conde Bento Luis, Mendonca Duarte. Computer Vision and Kinematic Sensing in Robotics : Ph. D. thesis / Luis Conde Bento, Duarte Mendonca. — 2001. — 06. — URL: https://www.researchgate.net/publication/237833320_Computer_Vision_and_Kinematic_Sensing_in_Robotics (online; accessed: 29.05.2020).

[6] Cunningham Padraig, Delany Sarah. k-Nearest neighbour classifiers // Mult Classif Syst. — 2007. — 04. — URL: https://www.researchgate.net/publication/228686398_k-Nearest_neighbour_classifiers (online; accessed: 29.05.2020).

[7] Deep Residual Learning for Image Recognition / Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun. — Vol. abs/1512.03385. — 2015. — URL: http://arxiv.org/abs/1512.03385 (online; accessed: 29.05.2020).

[8] Deng Jiankang, Guo Jia, Zafeiriou Stefanos. ArcFace: Additive Angular Margin Loss for Deep Face Recognition // CoRR. — 2018. —

Vol. abs/1801.07698. — URL: http://arxiv.org/abs/1801.07698 (online; accessed: 29.05.2020).

[9] A Discriminative Feature Learning Approach for Deep Face Recognition / Yandong Wen, Kaipeng Zhang, Zhifeng Li, Yu Qiao. — Vol. 9911. — 2016. — 10. — P. 499–515.

[10] Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation / Liang-Chieh Chen, Yukun Zhu, George Papandreou et al. // CoRR. — 2018. — Vol. abs/1802.02611. — URL: http://arxiv.org/abs/1802.02611 (online; accessed: 29.05.2020).

[11] Fischler Martin A., Bolles Robert C. Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography // Commun. ACM. — 1981. — Vol. 24. — P. 381–395.

[12] Going Deeper with Convolutions / Christian Szegedy, Wei Liu, Yangqing Jia et al. // CoRR. — 2014. — Vol. abs/1409.4842. — URL: http://arxiv.org/abs/1409.4842 (online; accessed: 29.05.2020).

[13] Hadsell R., Chopra S., LeCun Y. Dimensionality Reduction by Learning an Invariant Mapping // 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06). — Vol. 2. — 2006. — P. 1735–1742.

[14] Hartigan J. A., Wong M. A. Algorithm AS 136: A K-Means Clustering Algorithm // Journal of the Royal Statistical Society. Series C (Applied Statistics). — 1979. — Vol. 28, no. 1. — P. 100–108. — URL: http://www.jstor.org/stable/2346830 (online; accessed: 29.05.2020).

[15] Image Retrieval with Deep Local Features and Attention-based Keypoints / Hyeonwoo Noh, Andre Araujo, Jack Sim, Bohyung Han // CoRR. — 2016. — Vol. abs/1612.06321. — URL: http://arxiv.org/abs/1612.06321 (online; accessed: 29.05.2020).

[16] ImageNet Large Scale Visual Recognition Challenge / Olga Russakovsky, Jia Deng, Hao Su et al. // International Journal of Computer Vision (IJCV). — 2015. — Vol. 115, no. 3. — P. 211–252.

[17] Kaya Mahmut, Bilge H.s. Deep Metric Learning: A Survey // Symmetry. — 2019. — 08. — Vol. 11. — P. 1066.

[18] Kingma Diederik, Ba Jimmy. Adam: A Method for Stochastic Optimization // International Conference on Learning Representations. — 2014. — 12. — URL: https://arxiv.org/abs/1412.6980 (online; accessed: 29.05.2020).

[19] Lowe David. Object Recognition from Local Scale-Invariant Features // Proceedings of the IEEE International Conference on Computer Vision. — 2001. — 01. — Vol. 2. — URL: https://www.researchgate.net/publication/2373439_Object_Recognition_from_Local_Scale-Invariant_Features (online; accessed: 29.05.2020).

[20] ORB: an efficient alternative to SIFT or SURF / Ethan Rublee, Vincent Rabaud, Kurt Konolige, Gary Bradski. — 2011. — 11. — P. 2564–2571.

[21] Revisiting Oxford and Paris: Large-Scale Image Retrieval Benchmarking / Filip Radenović, Ahmet Iscen, Giorgos Tolias et al. // 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. — 2018. — P. 5706–5715.

[22] Rokad Brij, Nagarajan Sureshkumar. Skin Cancer Recognition using Deep Residual Network // CoRR. — 2019. — Vol. abs/1905.08610. — URL: http://arxiv.org/abs/1905.08610 (online; accessed: 29.05.2020).

[23] Rosten Edward, Drummond Tom. Machine Learning for High-Speed Corner Detection // Computer Vision – ECCV 2006 / Ed. by Aleš Leonardis, Horst Bischof, Axel Pinz. — Berlin, Heidelberg : Springer Berlin Heidelberg, 2006. — P. 430–443.

[24] Schroff Florian, Kalenichenko Dmitry, Philbin James. FaceNet: A Unified Embedding for Face Recognition and Clustering // CoRR. — 2015. — Vol. abs/1503.03832. — URL: http://arxiv.org/abs/1503.03832 (online; accessed: 29.05.2020).

[25] Sivic, Zisserman. Video Google: a text retrieval approach to object matching in videos // Proceedings Ninth IEEE International Conference on Computer Vision. — 2003. — Oct. — P. 1470–1477 vol.2.

[26] Szegedy Christian, Ioffe Sergey, Vanhoucke Vincent. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning // CoRR. — 2016. — Vol. abs/1602.07261. — URL: http://arxiv.org/abs/1602.07261 (online; accessed: 29.05.2020).

[27] Abadi Martín, Agarwal Ashish, Barham Paul et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. — 2015. — Software available from tensorflow.org. URL: http://tensorflow.org/ (online; accessed: 29.05.2020).

[28] Turpin Andrew, Scholer Falk. User performance versus precision measures for simple search tasks // Noriko Kando, Wessel Kraaij and Arjen P. de Vries (editors), Proceedings of the ACM SIGIR International Conference on Research and Development in Information Retrieval. — 2006. — P. 11–18.

[29] Wang Mei, Deng Weihong. Deep Face Recognition: A Survey // CoRR. — 2018. — Vol. abs/1804.06655. — URL: http://arxiv.org/abs/1804.06655 (online; accessed: 29.05.2020).

[30] Лучинский В.Д. Распознавание городских объектов при различных погодных условиях. — 2019. — URL: http://se.math.spbu.ru/SE/YearlyProjects/spring-2019/344/344-Luchinskiy-report.pdf (online; accessed: 29.05.2020).