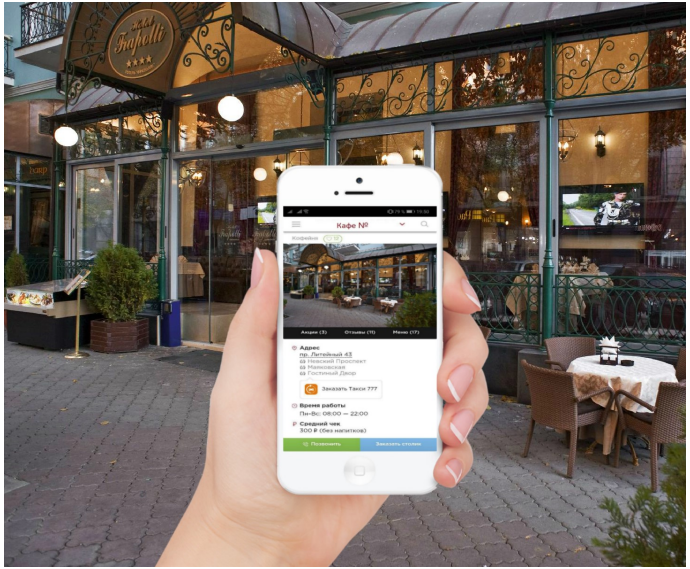
Разработка системы распознавания городских объектов

Лучинский Владимир Дмитриевич
Научный руководитель: к.т.н, доцент, Ю.В. Литвинов
Консультант: руководитель проектов
ООО «Системы Компьютерного Зрения», Н.А. Пенкрат
Рецензент: к. ф.-м.н, старший преподаватель, Салищев С.И.

18 июня 2020 г.

Здравствуйте, меня зовут Лучинский Владимир. Тема моей дипломной работы — разработка системы распознавания городских объектов. Тематика работы — глубокое обучение в компьютерном зрении.

Применение



У данной работы довольно много применений. Например, предоставление информации о различных заведениях жителям города и его туристам, а также уточнение своей позиции и соответствующая отрисовка вспомогательных направлений в форме дополненной реальности. Такой проект есть в компании ООО «СКЗ», которая выдала мне тему диплома.

Из чего состоит локализация

- Сбор данных и 3D реконструкция
- **Распознавание объекта по изображению (Image retrieval)**
- Локализация

В ней (в компании) это делается с помощью локализации. Сначала собираются данные, и проводится их 3D реконструкция. Далее, когда мы получаем изображение на вход, мы смотрим GPS координаты снимка, находим в нашей базе соответствующую 3D модель, с помощью локализации вычисляем позу этого снимка в 3D-модели, и что-то отрисовываем на экране.
Однако после GPS фильтрации у нас может остаться от 1 до 10к снимков, из которых нам надо выбирать. Поэтому после этого осуществляется также распознавание изображения с помощью классических алгоритмов компьютерного зрения, которое позволяет сузить число кандидатов до 100 - 200. Однако оно работает не очень хорошо. Почему?

Основные трудности

- Здания могут быть похожи
- Реконструкция и локализация происходит в разное время
  - Разные ракурсы изображений
  - Разные сезоны
- **Наличие в кадре прохожих, машин и др. двигающихся объектов**

Потому что мы здесь сталкиваемся с некоторыми проблемами, а именно:

1. Здания могут быть похожи.
2. Реконструкция и локализация проводится в разное время, поэтому данные в базе отличаются по ракурсу или сезону от тех, что поступают в виде пользовательских запросов.
3. В кадре могут быть двигающиеся объекты, способные помешать распознаванию.

Постановка задачи

Цель

Улучшить качество сопоставления снимков городских сцен для предоставления информации о них в интерактивном режиме.

Задачи

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

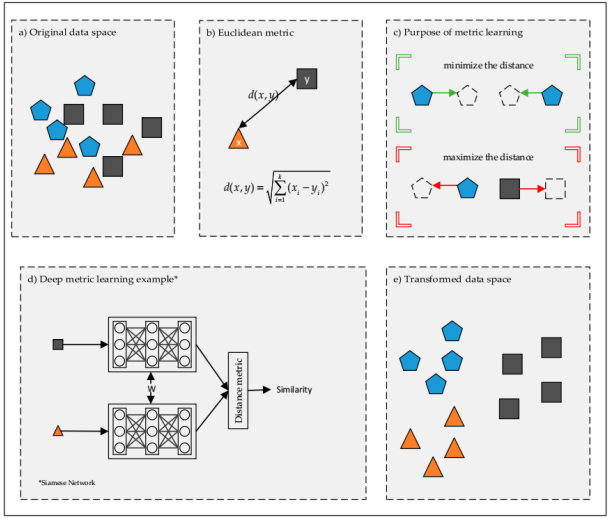
Мы хотим разобраться с этими недостатками и улучшить качество сопоставления снимков городских сцен. Для этого нам необходимо разработать соответствующий подход, протестировать его и проанализировать результаты.

Существующие подходы к распознаванию городских объектов

- Google Visual Positioning System
- **Augmented.City**
- DELF (deep local features)

Здесь надо упомянуть Google VPS, о которой, наверное, многие слышали, но новостей о её выходе пока не было. Augmented.City — это решение ООО «СКЗ», та система, которую хочется улучшить. Для поиска изображений по содержанию (image retrieval-а) она использует классические алгоритмы компьютерного зрения. И DELF — нейронная сеть от Google, готовая, однако при базе из нескольких тысяч изображений обрабатывает один запрос несколько минут, так как на этапе тестирования там используется времязатратный алгоритм RANSAC для сопоставления изображений. Поэтому в сложной системе, которая занимается не только распознаванием городских объектов, но и ещё, например, локализацией на его основе, её применение недопустимо, и было решено обучать свою сеть в надежде получить подход точнее, чем Augmented.City, и быстрее, чем DELF, но с сопоставимой ему точностью.

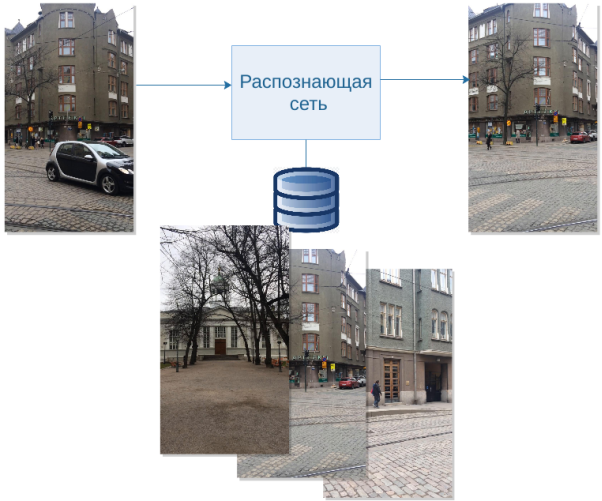
Распознающая сеть. Метрическое обучение.



https://www.researchgate.net/publication/335314481_Deep_Metric_Learning_A_Survey

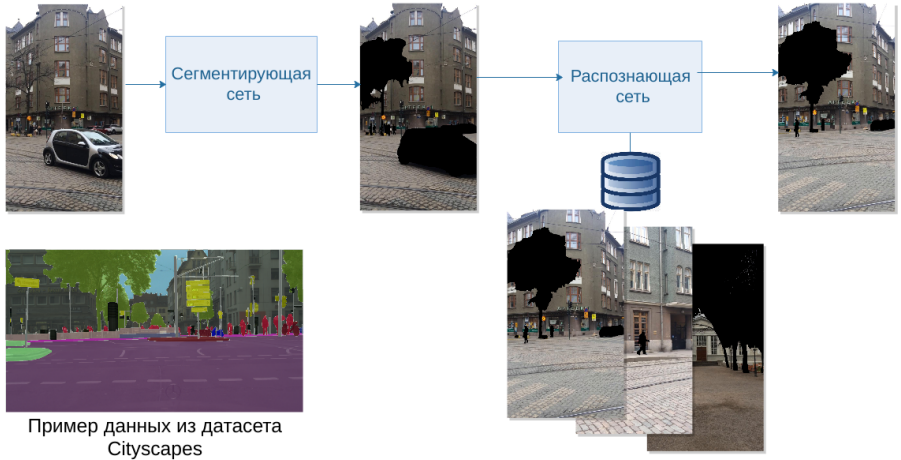
Для обучения сети за основу был взят подход, использующий метрическое обучение, так как он достаточно хорошо себя показал в решении задачи image retrieval в различных областях, например, в распознавании лиц или тигров. Кроме того, он позволяет на этапе тестирования использовать классы городских объектов, которых в обучении не было. После пропуска объекта через сеть мы получаем вектор (или embedding вектор).

Предлагаемые подходы: прямой



Полученный вектор мы затем можем сравнить просто по Евклидовой метрике с соответствующими векторами изображений в базе и выдать отсортированный список изображений, ближайших ко входному.

Предлагаемые подходы: сегментирующий

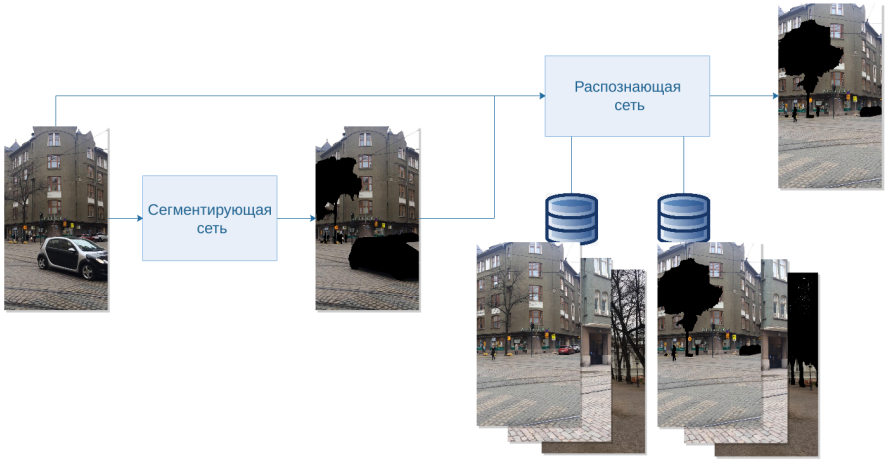


Пример данных из датасета Cityscapes

https://www.cityscapes-dataset.com

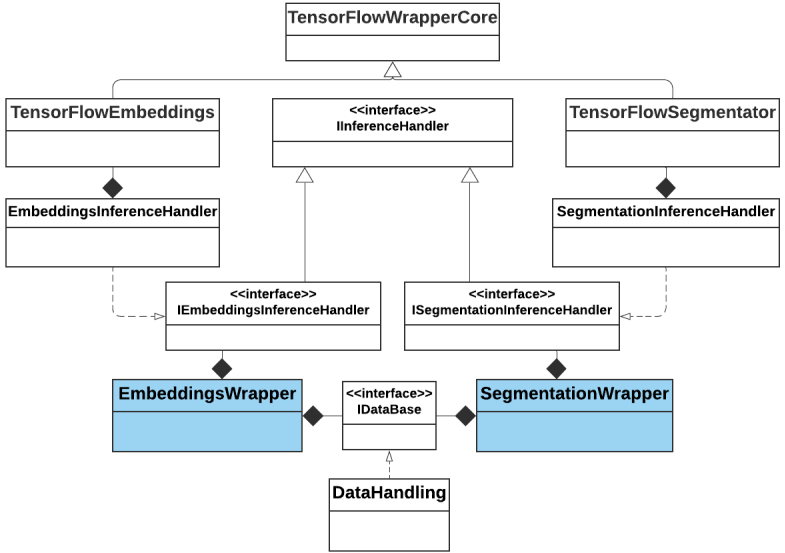
Чтобы уменьшить негативное влияние двигающихся объектов на сопоставление снимков, было решено использовать готовую state-of-the-art нейронную сеть deeplabv3+ для семантической сегментации, обученную на датасете Cityscapes, потому что в нём есть все нужные классы.

Предлагаемые подходы: смешанный



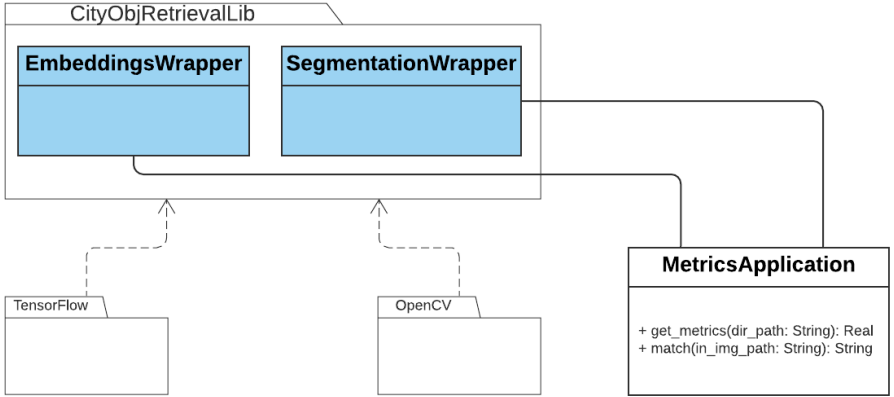
Также было решено попробовать смешанный подход, когда мы сопоставляем сегментированные изображения с сегментированными, и обычные снимки с обычными.

Архитектура библиотеки



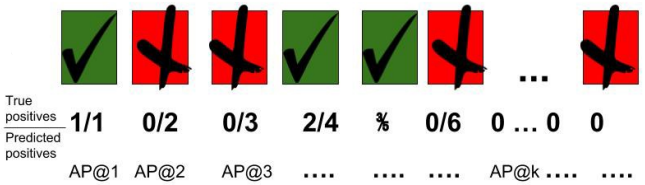
Подход был реализован в виде библиотеки на языке C++, так как исходный проект тоже на C++. Пользователю доступны классы EmbeddingsWrapper и SegmentationWrapper (выделены синим цветом), которые оборачивают в себе работу с нейронной сетью и необходимую для этого работу с данными.

Архитектура прототипа



Для работы с нейронными сетями библиотека использует фреймворк для обучения нейронных сетей TensorFlow, у которого есть API на C++. Также было реализовано консольное приложение, которое сопоставляет поданное на вход изображение заданной базе и считает метрику.

Метрики



Overall AP = ⅓ (1/1 + 0/2 + 0/3 + 2/4 + ⅖ + 0 … + 0) = 0.7

$$AP@k = \frac{1}{max(GTP, 1)} \sum_{i=1}^{k} \frac{rel(i)}{i} \quad (1)$$

$$mAP = \frac{1}{N} \sum_{j=1}^{N} AP_j \quad (2)$$

https://towardsdatascience.com/breaking-down-mean-average-precision-map-ae462f623a52

Напомню, что получив изображение на вход, на выходе мы выдаем отсортированный список ближайших к нему из нашей базы. Было решено посмотреть результаты на метрике качества ранжирования mean average precision (или map), так как она довольно часто применяется в задачах image retrieval и позволяет учитывать порядок изображений в списке ближайших, что важно оценить при сопоставлении изображений.

Здесь галочками обозначены объекты из того же класса, что и входное (класс — те изображения, которые мы решили ассоциировать с некоторым городским объектом, зданием). Крестиками — из других. Мы считаем average precision (ap) по формуле 1. Суммируем все правильные угадывания, делённые на их место в списке, и делим на общее количество изображений из нужного класса в списке. Дальше вычисляем map просто беря среднее.

Результаты экспериментов

Подход	Acc. top-1	Acc. top-4	mAP
DELF	84.82%±0.05%	94.79%±0.05%	0.756±0.001
Прямой подход	78.50%	88.80%	0.727
Сегмент. подход	78.93%	88.98%	0.736
Смеш. подход	79.33%	89.09%	0.740

Таблица: Результаты предлагаемых подходов

Подход	Acc. top-1	Acc. top-4	mAP
Augmented.City	34.01%	57.69%	–
Прямой подход	70.64%	79.54%	0.612
Сегмент. подход	72.04%	80.69%	0.643
Смеш. подход	66.30%	78.00%	0.605

Таблица: Результаты предлагаемых подходов на расширенном наборе данных

Кроме того также оценивается точность, которая вычисляется просто как отношение правильно определённых классов ко всем. В топ-1 надо, чтобы первый предсказанный класс оказался правильным, в топ-4, чтобы он содержался среди первых 4-х.
Для проведения экспериментов и обучения сети был собран датасет, состоящий из городских сцен Санкт-Петербурга (не конкретно мной, но компанией СКЗ). Классы тестовых данных не участвовали в обучении. То есть сеть таких объектов в принципе не видела. Использовалось две базы данных. В одной все те же классы, что и в запросах, в другой помимо этого дополнительно содержался ещё 141 класс. Видим, что результаты полученных подходов выше, чем у Augmented.City, но по точности примерно на 5% ниже, чем у DELF, результаты которого здесь приведены в качестве ориентира.
Также надо заметить, что Результаты Augmented.City предоставлены авторами системы, которые не измеряли свой подход по метрике mAP. Подход DELF на расширенном наборе не тестировался в силу его медленности. Доверительный интервал приведён только для DELF, так как результат работы полученных подходов детерминированный.

Сравнение времени работы подходов

Подход	5-й перцентиль, с.	медиана, с.	95-й перцентиль, с.
DELF	240	450	560
Прямой подход	0.55	0.62	0.81
Сегмент. подход	4.1	4.2	4.5
Смеш. подход	4.7	4.8	5.3

Таблица: Сравнение времени работы предлагаемых подходов с DELF

Подход	Разрешение 972x2000	Разрешение 1500x2000
Прямой подход	0.55±0.01	0.80±0.01
Сегмент. подход	4.2±0.1	4.5±0.1
Смеш. подход	4.7±0.1	5.3±0.1

Таблица: Сравнение времени работы предлагаемых подходов на разных разрешениях, в сек.

- Процессор — Intel(R) Core(TM) i5-6300HQ CPU @ 2.30GHz
- Видеокарта — NVIDIA GeForce GTX 950M (4 Гб)
- Оперативная память — 8 Гб

По времени обработка одного запроса в подходе DELF может занимать от 240 до 560 секунд в зависимости от того, сколько раз запустится RANSAC для сопоставления изображений. Если он, например, запускается 800 раз, то сопоставление длится 240 секунд, если 2000 — то 560. Чаще всего он запускается в районе 1400 раз и тогда время составляет 453 секунды. Видно, что полученные подходы значительно быстрее, чем DELF. При этом сегментационный быстрее прямого, а сегментационный быстрее смешанного. Во второй таблице приведено среднее и стандартное отклонение.

Результаты

- Разработан подход распознавания городских объектов на основе нейронных сетей, реализованный в виде библиотеки на языке C++[1]
- Полученный подход протестирован на собранном вручную датасете, состоящем из городских сцен Санкт-Петербурга
- Улучшение относительно 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 его работы оказалось выше, чем у полученного в данной работе

[1] https://github.com/ucLh/city_obj_retrieval

Разработан подход распознавания городских объектов виде библиотеки позволяющей работать с нейронными сетями на C++. Проведено тестирование на городских сценах Санкт-Петербурга, по результатам которого можно сделать вывод о том, что полученный подход значительно превосходит Augmented.City по точности, уступает DELF по точности, но превосходит по времени работы.