Учреждение образования
“Белорусский государственный университет
информатики и радиоэлектроники”
Кафедра информатики

Отчёт по проекту:
«Детектирование ключевых точек рук»

Выполнил: Фисько Дмитрий Владимирович
магистрант кафедры информатики
группа № 956241

Проверил: Стержанов Максим Валерьевич
Кандидат технических наук

Минск 2019

Содержание

1. Постановка задачи

Необходимо создать и обучить модель для поиска ключевых точек руки на фотографиях. Ключевые точки руки — это точки с заданным расположением на руке. Всего необходимо выделить 21 ключевую точку на руке. Расположение ключевых точек задано на рисунке 1.

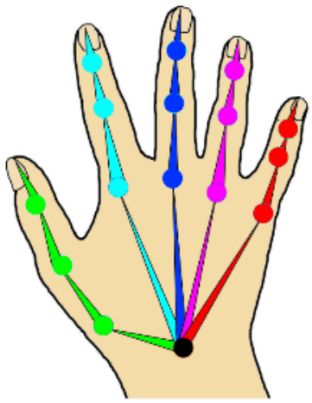


Рисунок 1. Расположение ключевых точке руки

Для обучения модели распознаванию ключевых точек рук, дан датасет GANerated Hands Dataset с примерами размеченных ключевых точек на руках. В нём содержится 100000 фотографий рук с соответствующими ключевыми точками. Датасет разделён в следующей пропорции: в обучающей выборке 90000 изображений, а в валидационной 10000. Примеры изображений из датасета GANerated Hands Dataset даны на рисунке 2.

Рисунок 2. Примеры изображений из датасета GANerated Hands Dataset

Необходимо по данному множеству изображений с размеченными ключевыми точками рук обучить модель для детектирования. Полученная модель по детектированию ключевых точек рук должна соответствовать требованию по стабильной работе на всей вариативности возможных положений рук. Критерием для утверждения обученной модели является метрика среднеквадратичного отклонения между ожидаемыми точками в валидационной выборке и фактическим результатом работы модели на уровне не больше 2.5e-3.

2. Процесс реализации задачи

Данные для обучения были взяты из датасета GANerated Hands Dataset. Каждый элемент данного датасета включает в себя файл с вырезанным изображение руки и соответствующей аннотацией ключевых точек.

Для достижения стабильности в детектировании ключевых точек рук, в качестве модели был выбран state-of-art подход для данного типа задач. Была выбрана нейронная сетевая архитектура Unet с преобразованием исходный ключевых точек в карты сегментации.

Чтобы появилась возможность использовать Unet в качестве модели для детекции ключевых точек. Каждая ключевая точка из исходного датасета приводилась к маске с точкой нормального распределения. Данная операция производилась с каждой точкой из аннотации. Используя данное преобразование, появлялась возможность превратить задачу по детекции точек в советующую задачу сегментации.

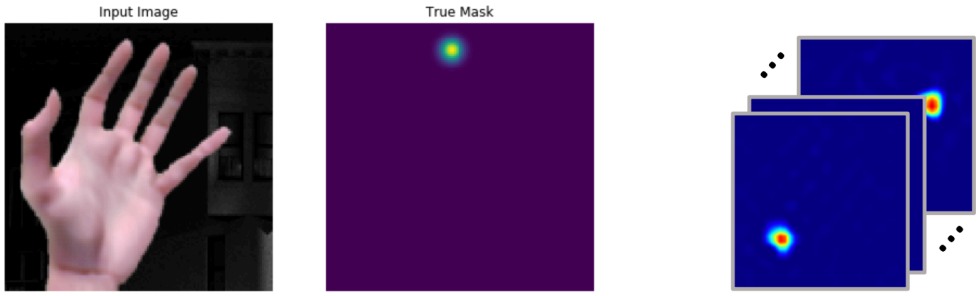


Рисунок 3. Преобразование задачи по детектированию в задачу сегментации

Для проектирования нейросетевой модели и её последующего обучения был использован фрэймворк TensorFlow 2.0 и модель Unet. Архитектура Unet состоит из 2 частей: Encoder и Decoder. В качестве Encoder части модели нейронной сети Unet был выбран MobileNet. В качестве Decoder нейроной сети были взяты слои сформированные через последовательную upscaling свёртку. Размер входного слоя был выбран 160x160. Потенциально данный выбор архитектуры позволит запускать нейронные сети на мобильном устройстве. Общий процесс реализации задачи изображён на рисунке 4.

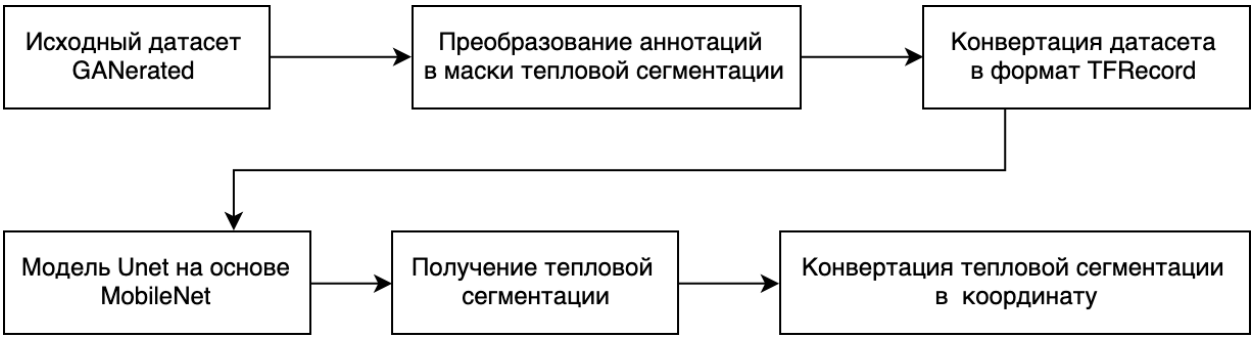
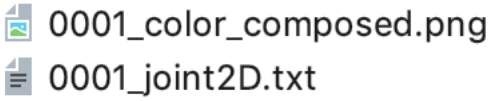


Рисунок 4. Общий процесс реализации задачи

3. Хранение данных

Датасет GANerated Hands хранится в виде обычной файловой структуры, в которой есть JPEG картинка, а рядом с картинкой с похожим названием располагаются аннотации для ключевых точек.



Формат файловой структуры:

1) 0001_color_composed.png – изображение руки
2) 0001_joint2D.txt – 42 подряд идущих числа, каждые 2 подряд идущих числа описывают координаты одной из 21 точки

Минусами данного способа хранения информации является его проблемы с производительностью. Каждое чтение файла изображения с диска приводит к операции декодирования картинки из JPEG, и перемещение курсора диска на отступ, с которым располагается картинка в файловой системе. Также нам нужно преобразовывать исходные аннотации в маску. Чтобы выполнить данные операции лишь единожды при создании обучающей выборки исходные данные было решено переконвертировать в бинарный формат называемый TFRecods. TFRecods – это специальный бинарный формат для хранения множества данных с определённой структурой. Формат хранения данных в файле TFRecods использовавшийся для исходной задачи описан в таблице 1.

Название атрибута	Описание атрибута	Размерность атрибута
image	Изображение из датасета	160x160x3
mask	Карта сегментации для пальца руки	160x160

Таблица 1. Описание формата хранения исходного датасета в формате TFRecords

Последовательное чтение данных для обучения нейронной сети из TFRecords не позволило скорости чтения данных стать узким место для производительности процесса обучения нейросетевой модели.

В связи с ограниченностью вычислительных ресурсов для обучения нейронной сети, был использованы не все элементы обучающей выборки, были взяты только первые 100 тысяч изображений из всего 700 тысяч доступных в датасете GANerated Hands Dataset.

4. Модель

Часть кода по построению и обучению нейронной сети было взято из примера TensorFlow Examples. Была проделана работа по модифицированию исходной модели из примера. Исходная модель была ориентирована на создание классифицирующей сегментационной маски, поэтому использовался softmax (формула 1) в функции потерь. В модифицированной версии изменилась целевая функция модели, теперь она должна создавать сегментирующую тепловую карту, поэтому функция потерь была изменена на MSE (формула 2).

$$\mathrm{Softmax}(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)}$$

Формула 1. Исходная softmax формула в функции потерь

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \tilde{y}_i)^2$$

Формула 2. Новая MSE формула

В качестве нейросетевой модели была выбрана Unet, с предобученной моделью MobileNet из пакета Keras в качестве Encoder для извлечения признаков. В качестве Decoder части была использована стандартная архитектура Unet с последовательным upscaling слоёв.

Итого получилась следующая архитектура нейронной сети:

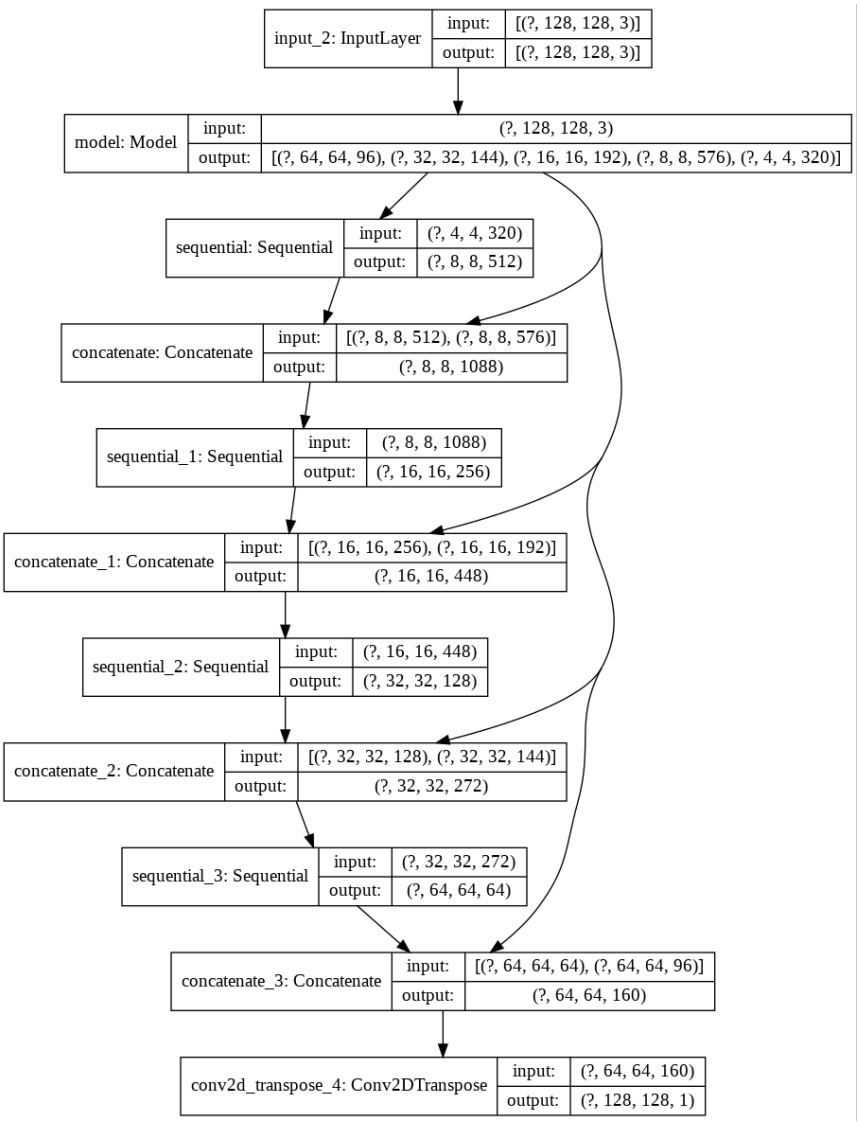


Рисунок 5. Подробное описание выбранной архитектуры на основе архитектуры Unet

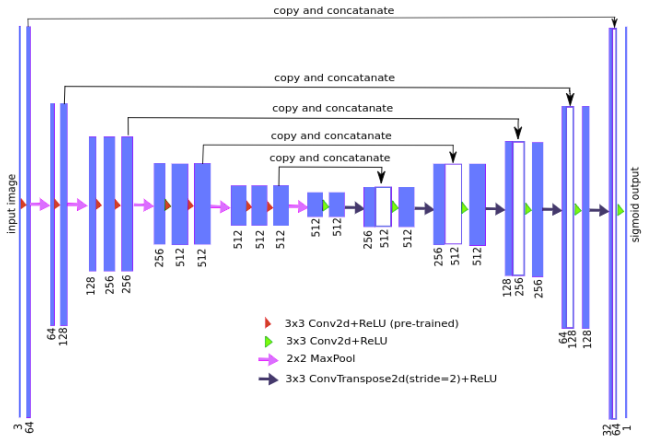


Рисунок 6. Обобщённое описание выбранной архитектуры Unet

Данная модель была в дальнейшем обучена на 90000 изображениях из обучающей выборки и провалидирована на 10000 изображениях из валидационной выборки. В качестве функции потерь использовался квадрат попиксельной разницы между исходной маской для обучения и результатом модели. Обучение производилось в течении 20 эпох.

Обучения и построения модели сегментации проводилось на основании фрэймворка Tensorflow. Для работы с датасетом, специальных вспомогательных библиотек помимо Tensorflow и встроенных пакетов для мультипроцессорной обработки в Python 3.5 не потребовалось. Общий pipeline обучения модели (Рисунок 7).

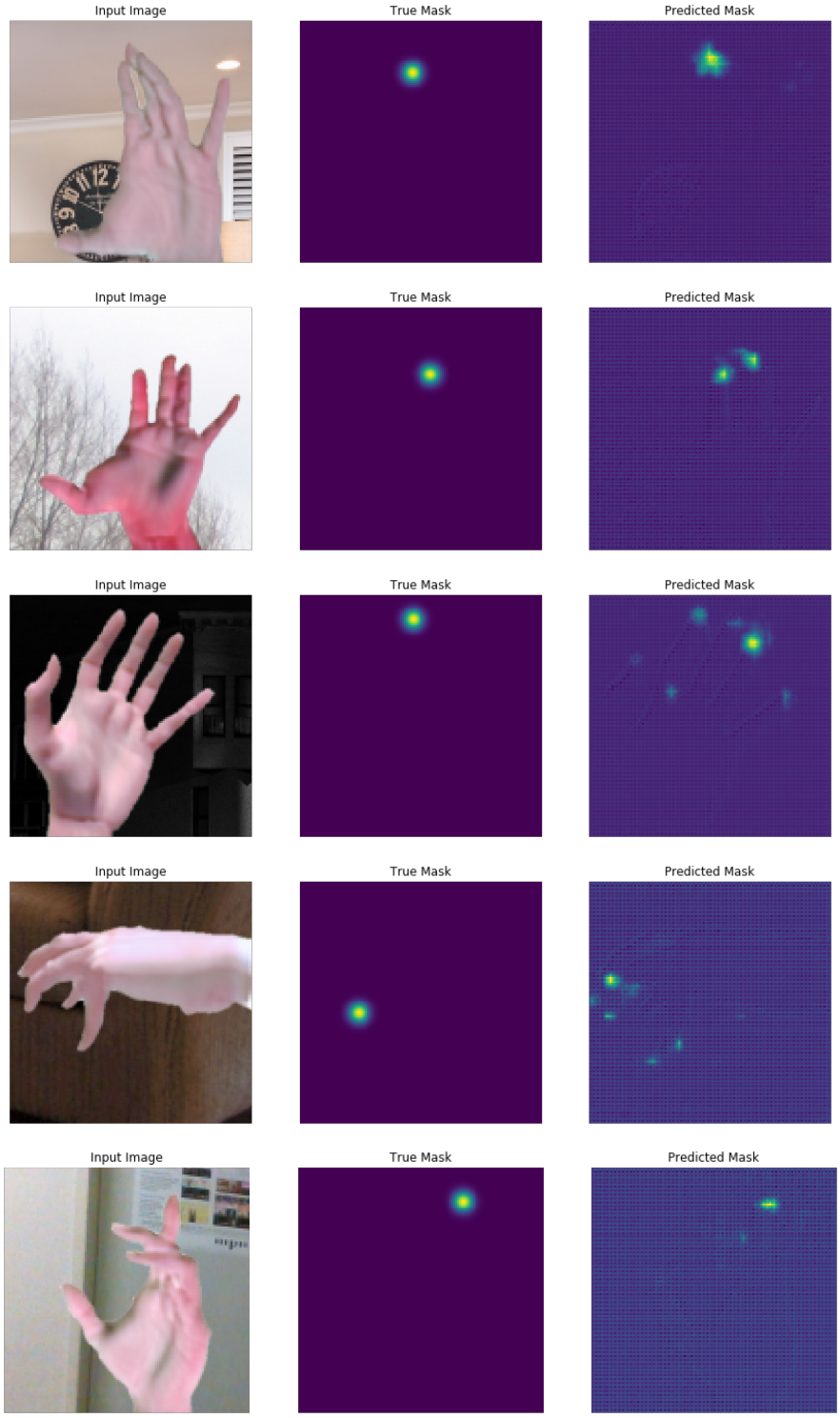
Рисунок 7. Общий pipeline обучения модели сегментации на основе Tensorflow.

5. Метрики качества

Процесс обучения нейронной сети изображён ниже, всего было сделано 20 интераций над датасетом из обучающей выборки в 90 тысяч изображений. Финальная метрика MSE (формула 2) на валидационой выборке была равна 1.1993e-04 для обучающей выборке и 0.0023 для тестовой.

```
Train for 695 steps, validate for 15 steps
Epoch 1/20
319s 458ms/step - loss: 2.8777e-04 - val_loss: 0.0022
Epoch 2/20
319s 459ms/step - loss: 2.4053e-04 - val_loss: 0.0022
Epoch 3/20
319s 459ms/step - loss: 2.1979e-04 - val_loss: 0.0021
Epoch 4/20
318s 458ms/step - loss: 1.9479e-04 - val_loss: 0.0023
Epoch 5/20
319s 459ms/step - loss: 1.7418e-04 - val_loss: 0.0025
Epoch 6/20
319s 459ms/step - loss: 1.6475e-04 - val_loss: 0.0023
Epoch 7/20
319s 459ms/step - loss: 1.5843e-04 - val_loss: 0.0022
Epoch 8/20
319s 459ms/step - loss: 1.5231e-04 - val_loss: 0.0023
Epoch 9/20
319s 458ms/step - loss: 1.5660e-04 - val_loss: 0.0025
Epoch 10/20
318s 458ms/step - loss: 1.4389e-04 - val_loss: 0.0025
Epoch 11/20
319s 458ms/step - loss: 1.3622e-04 - val_loss: 0.0021
Epoch 12/20
319s 458ms/step - loss: 1.3599e-04 - val_loss: 0.0021
Epoch 13/20
319s 458ms/step - loss: 1.3422e-04 - val_loss: 0.0021
 Epoch 14/20
319s 460ms/step - loss: 1.3552e-04 - val_loss: 0.0021
Epoch 15/20
319s 459ms/step - loss: 1.2018e-04 - val_loss: 0.0022
Epoch 16/20
319s 459ms/step - loss: 1.2932e-04 - val_loss: 0.0022
Epoch 17/20
317s 457ms/step - loss: 1.2129e-04 - val_loss: 0.0023
Epoch 18/20
318s 458ms/step - loss: 1.2134e-04 - val_loss: 0.0021
Epoch 19/20
318s 458ms/step - loss: 1.1965e-04 - val_loss: 0.0021
Epoch 20/20
319s 458ms/step - loss: 1.1993e-04 - val_loss: 0.0023
```

В итоге были получены следующие результаты для распознавания указательного пальца:



Как видно из результатов, обученная модель не всегда хорошо справляется со сложными тестовыми случаями. Данная проблема могла быть решена увеличением количества данных в тестовой выборке.

6. Результаты:

Была успешно спроектирована и обучена модель для детектирования одной ключевой точки на изображении руки. Была достигнута метрика качества MSE равная 0.0021 на валидационной выборке, позволяющая утвердить обученную модель исходя из критерия качества равного 0.003, установленного в разделе постановки задачи. Данных результатов удалось достичь, используя 90000 изображений для обучающей выборки и 10000 изображений для валидационной.

Исходная задача по детектированию ключевых точек руки была перефоормулирована в задачу сегментации, что позволило получить хорошие показатели на целевых метриках качества. Детектирование всех точек реализовано не было, но в ходе реализации проекта был получен опыт, позволяющий легко обобщить предоставленное решение на большее количество ключевых точек.

В процессе реализации проекта был получен опыт подготовки и очистке большого объёма данных. Из исходного датасета были убраны вхождения данных, которые были не реалистичными, тем самым улучшив сосредоточение модели на выделении наиболее важных признаков. Размер обработанного датасета составлял 30Gb. Из-за большого объёмы данных, в ходе реализации алгоритмов по конвертации, отдельное внимание уделялось алгоритмам параллельной обработки данных. Предобработка датасета осуществлялась с эффективным использованием ресурсных мощностей. При этом в ходе обучения модели использовалась аугментация данных. В режиме реального времени, происходило видоизменение элементов датасета, не дающее нейронной сети переобучиться на данных для обучения.

Также был получен опыт проектирования модели и её обучения. Была изучена нейросетевая архитектура Unet. В исходной архитектуру модели Unet была добавлена предобученная модель MobileNet v2 для выделения признаков. В ходе реализации проекта был получен опыт использования фрэймворка Tensorflow 2.0 совместно с библиотекой Keras.

7. Список использованных источников

[1] Mpi-inf.mpg.de: [Электронный ресурс]. – Электронные данные. – Режим доступа: https://handtracker.mpi-inf.mpg.de/projects/GANeratedHands/GANeratedDataset.htm. – Дата доступа: 20.11.2019.
[2] Wikipedia.org: [Электронный ресурс]. – Электронные данные. – Режим доступа: https://en.wikipedia.org/wiki/U-Net. – Дата доступа: 23.11.2019
[3] CS229.stanford.edu: [Электронный ресурс]. – Электронные данные. – Режим доступа: http://cs229.stanford.edu/proj2014/Rahul%20Makhijani,%20Saleh%20Samaneh,%20Megh%20Mehta,%20Collaborative%20Filtering%20Recommender%20Systems.pdf. – Дата доступа: 23.11.2019
[4] Tensorflow.org: [Электронный ресурс]. – Электронные данные. – Режим доступа: https://www.tensorflow.org/tutorials/images/segmentation. – Дата доступа: 25.11.2019
[5] Github.com: [Электронный ресурс]. – Электронные данные. – Режим доступа: https://imgaug.readthedocs.io/en/latest/. – Дата доступа: 25.11.201
[6] Dlcourse.ai: [Электронный ресурс]. – Электронные данные. – Режим доступа: https://dlcourse.ai/ 6. – Дата доступа: 26.11.2019