Инженерный вестник Дона, №4 (2017)
ivdon.ru/ru/magazine/archive/n4y2017/4433

Автоматическая сегментация спутниковых снимков на базе модифицированной свёрточной нейронной сети UNET

Р.А. Соловьев, Д.В. Тельпухов, А.Г. Кустов

Институт проблем проектирования в микроэлектронике РАН

Аннотация: В статье предложена методика для автоматической сегментации спутниковых снимков по нескольким классам, таким как здания, реки, дороги и.т.д. на базе свёрточных нейронных сетей. Программная реализация предложенной методики заняла второе место в конкурсе по сегментации спутниковых снимков на площадке Kaggle в задаче: Dstl Satellite Imagery Feature Detection. В статье изложено как именно следует готовить изображения для тренировки нейронной сети, в подробностях изложены принципы транировки. Предложена структура нейронной сети для сегментации. Сеть построена на базе UNET с дополнительными BatchNormalization и Dropout слоями, на базе двойных свёрточных блоков. Описана процедура кроссвалидации для оценки точности полученных моделей. Приведены описания алгоритмов для постпроцессинга и методика уточнения сегментации за счёт применения ансамбля из нескольких моделей. Предложена специализированная модель для нахождения на снимке объектов малого размера, таких как «автомобили» и «мотоциклы». Так же приведён обзор других методов, которые использовались для решения этой задачи, но не были включены в финальное решение. В экспериментальных результатах показано, что эффективность нейронных сетей в этой задаче крайне высока и можно автоматически подготовить разметку местности, похожую на разметку, сделанную вручную. И тем самым сэкономить средства, так как на ручную разметку на сегодняшний день тратятся существенные финансовые средства.
Ключевые слова: свёрточные нейронные сети, спутниковые снимки, сегментация изображений, машинное обучение, кроссвалидация, коэффициент Жаккара, сеть UNET, распознание изображений, компьютерное зрение, результаты соревнований.

Введение

В статье описана методика для автоматической сегментации спутниковых снимков. Ручная сегментация это очень долгий и дорогой процесс. На ручную обработку одного снимка с размером 3360x3360 пикселов тратится более 1000 британских фунтов [1, 2]. Поэтому автоматизация этой задачи является актуальной задачей хотя бы с экономической точки зрения. Для решения задач распознания образов в данный момент наибольшую эффективность показывают нейронные сети различных архитектур [3, 4]. Новая архитектура нейронной сети, созданная специально для задачи сегментации спутниковых снимков, предложена в данной статье.

Описанная в статье методика проверена на реальных данных, представленных в он-лайн соревновании от Kaggle и DSTL (Defence Science and Technology Laboratory) [5]. Финальное решение, на базе данной методики заняло 2-ое место в общем зачёте конкурса.

Для решения задачи использовался язык программирования Python в связке с популярными модулями для работы с нейронными сетями Keras и Theano. Расчёты велись на нескольких видеокартах от NVIDIA GTX 980Ti 8 GB + 2* GTX Titan 12 GB, которые работали почти круглосуточно в течение двух месяцев.

Основные подходы к решению задачи:

1) Для решения использовалась модифицированная нейронная сеть UNET [6], которая показала свою высокую эффективность в задачах сегментации биомедицинских изображений. Основные модификации структуры: увеличено число свёрточных слоёв. Добавлены слои BatchNormalization и Dropout (для уменьшения оверфиттинга).

2) В качестве loss-функции использовался Jaccuard Index [7]. Для поиска оптимального решения производился поиск по сетке оптимальных параметров (grid search). Выбирались модели с максимальным счетом по результатам валидации.

3) Всего требовалось разметить 10 различных типов объектов. Для каждого типа объектов была создана отдельная модель, которая отлаживалась независимо от остальных. Возможно, это приводило к некоторой потере информации о взаимодействии объектов, но эта проблема частично решалось финальным постпроцессингом (например, из полигонов машин, вычитались полигоны воды и.т.д.).

4) Каждая модель это на самом деле набор из K различных весов, которые были получены из кросс валидации. Для большей части классов использовался 5 KFold. Разбиение на тренировочную часть и

валидационную части проводилось по ID изображения (всего в тренировочном наборе было 25 больших изображений). Из-за малого размера тренировочной части требовалось разбить картинки наиболее оптимальным для обучения образом. Поэтому разбиение было выполнено руками один раз в самом начале по следующим принципам:

- Разбиение было разным для разных классов
- Классы были сильно несбалансированы и часть классов присутствовали только на некоторых картинках, поэтому картинки с присутствующим классом были равномерно распределены между всеми частями.

5) Для класса с маленькими объектами, например 10, использовался маленький вариант UNET, с входным размером 32x32 пикселя и отдельной Loss функцией с большим штрафом за False Positive, основано на Tverski index [8].

Анализ входных данных и метрики

В качестве входных данных были предоставлены два набора изображений: тренировочные (train) и тестовые (test). В тренировочном наборе 25 различных областей поверхности земли, снятых со спутника, в тестовом 429 поверхностей. Для каждой поверхности было предоставлено 20 различных изображений снятых в разных спектрах спутником World View 3 [9] (см. таблицу 1).

Таблица №1

Характеристики изображений спутника World View 3

Название на английском	Длина волны, нм	Разрешение, м	Динамический диапазон,	Разрешение файлов,

			бит/пиксел	пиксел
Panchromatic	450-800	0.31	11	~ 3396x3348
RGB (Red)	630-690			
RGB (Blue)	450-510			
RGB (Green)	510-580			
Coastal	400-450	1.24		~ 849x837
Blue	450-510			
Green	510-580			
Yellow	585-625			
Red	630-690			
RedEdge	705-745			
Near-IR1	770-895			
Near-IR2	860-1040			
SWIR-1	1195-1225	7.5	14	~136x134
SWIR-2	1550-1590			
SWIR-3	1640-1680			
SWIR-4	1710-1750			
SWIR-5	2145-2185			
SWIR-6	2185-2225			

SWIR-7	2235-2285			
SWIR-8	2295-2365			

Для тренировочных данных была также предоставлена готовая разметка по 10 классам. От участников соревнования требовалось автоматически получить разметку на тестовых данных. Список классов и их относительная площадь приведены в таблице 2.

Таблица №2

Список классов для разметки и максимальный размер полигона

Номер класса	Описание	Максимальный размер полигона для отдельной картинки, %
1	Строения	20.7
2	Заборы и другие структуры	2.9
3	Большие дороги	4.3
4	Тропинки	12.3
5	Деревья	24.9
6	Возделываемые поля, зерновые культуры, и.т.д.	93.6
7	Быстрая вода (реки, моря)	10.6

8	Медленная вода (лужи, высохшие русла)	1.4
9	Большие транспортные средства (грузовики, автобусы)	0.018
10	Автомобили и мотоциклы	0.156

Пример одной из тестовых картинок в RGB приведен на рис. 1. Полигоны для этой картинки изображены на рис. 2.

Рис. 1. – Один из тренировочных снимков

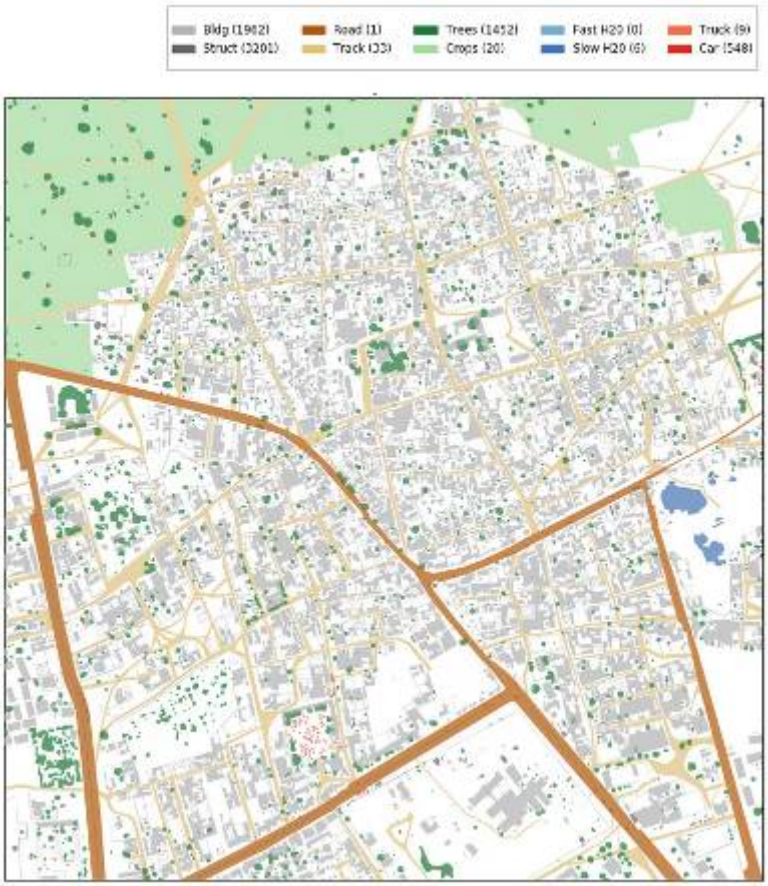

Рис. 2. – Полигональная разметка для тренировочного снимка

Для оценки решения использовалась метрика Jaccard Index, который задаётся формулой (1):

$$J = \frac{TP}{TP + FP + FN} = \frac{A \cap B}{A \cup B} = \frac{A \cap B}{A + B - A \cap B}, , \quad (1)$$

где TP - True positive, FP - False Positive, FN - False negative. Схематично формулу можно пояснить с помощью кругов Эйлера (см. рисунок 3)

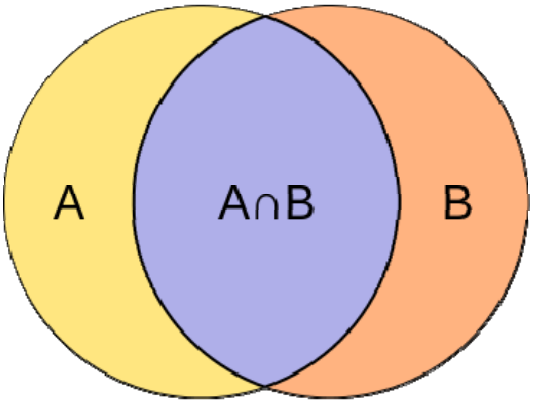

Рис. 3. – Пересечение реального полигона A, с предсказанным полигоном B

Из формулы (1) видно, что Jaccard Index изменяется на промежутке от 0 до 1. При этом в задаче метрика считалась одновременно для всех картинок. Плохое предсказание на одной картинке значительно влияло на финальную метрику. Поэтому требовалось избегать ложноположительных срабатываний.

Подготовка данных для обучения

Для каждой области снятой поверхности (определяется по IMAGE_ID) было предоставлено 20 различных изображений (см. таблицу 1). Наибольшее разрешение было у панхроматического снимка, но оно немного варьировалось между изображениями. Поэтому было принято решение привести все снимки к формату 3360 на 3360. Число 3360 было выбрано из следующих соображений:

1) В качестве базовой нейронной сети была выбрана UNET, которая была модифицирована под наши задачи с входной размерностью 20x224x224, а 3360 = 224*15.

2) Число 224 часто используется как сторона квадрата для большого числа известных свёрточных нейросетей, например VGG16 [10] и ResNet. Для этих сетей есть веса, подготовленные на большом наборе данных из ImageNet [11]. Таким образом, подготовленные данные можно будет использовать для экспериментов и с этими сетями.

Все доступные изображения для одной области были преобразованы к виду 3360x3360 и склеены в одну большую 3-мерную матрицу с размерами: 20x3360x3360. Полигоны также были спроецированы на изображение-маску размером 3360x3360. Значение 1 в маске означает наличие искомого объекта по координатам данного пиксела, 0 - соответственно отсутствие.

Тренировочные данные готовились отдельно для каждого из 10 классов сегментации (см. таблицу 2). Для подготовки тренировочного набора для отдельного класса эта 3-мерная матрица и маска были разделены на 225 непересекающихся частей с размерами 20x224x224 и 224x224 соответственно.

Таблица №3

Распределение классов по изображениям. Красным отмечены ячейки, если на картинке есть хотя бы один полигон для заданного класса. Число в ячейке - процент площади изображения занятой полигонами данного класса.

Class IMAGE_ID	1	2	3	4	5	6	7	8	9	10
6040_2_2	-	-	-	0.95	18.74	-	-	-	-	-
6120_2_2	20.66	2.04	4.25	8.65	4.43	10.29	-	0.28	0.007	0.156
6120_2_0	1.79	0.72	0.85	4.40	5.63	79.63	-	-	0.013	0.004
6090_2_0	-	0.03	-	0.40	10.11	28.25	-	0.31	-	0.001
6040_1_3	-	-	-	0.20	18.74	3.66	-	-	-	-
6040_1_0	-	-	-	1.44	8.01	-	-	-	-	-
6100_1_3	8.77	2.73	2.21	12.25	6.20	2.69	-	0.68	0.011	0.045

6010_4_2	-	-	-	1.95	12.34	-	-	-	-	-
6110_4_0	2.40	0.57	1.84	2.80	5.74	80.81	-	1.42	0.013	0.001
6140_3_1	5.20	1.43	3.42	2.52	5.88	57.42	-	0.46	0.004	0.035
6110_1_2	13.13	2.86	0.41	4.18	3.31	49.80	-	0.15	-	0.006
6100_2_3	8.22	1.41	1.21	9.60	7.53	-	-	0.06	0.014	0.066
6150_2_3	-	0.60	-	3.02	13.52	80.71	-	-	-	-
6160_2_1	-	-	-	2.80	10.27	-	-	-	-	-
6140_1_2	12.92	2.45	0.35	4.14	3.10	49.62	-	0.14	-	0.008
6110_3_1	4.55	1.25	3.63	2.82	5.41	57.64	-	0.55	0.018	0.025
6010_4_4	-	-	-	-	22.86	-	-	-	-	-
6170_2_4	-	0.001	-	2.50	7.78	49.56	-	0.008	-	-
6170_4_1	-	-	-	0.13	20.23	-	-	-	-	-
6170_0_4	-	0.001	-	0.19	24.90	-	-	0.01	-	-
6060_2_3	0.13	0.30	-	3.03	8.45	93.61	-	-	-	0.000
6070_2_3	1.55	0.30	0.81	-	16.05	-	10.63	0.05	-	0.005
6010_1_2	-	0.06	-	1.33	4.56	-	-	-	-	-
6040_4_4	-	-	-	1.89	2.91	-	-	-	-	-
6100_2_2	3.18	0.81	1.19	3.72	7.61	44.34	1.88	0.05	0.010	0.024

Данные были распределены неравномерно, что хорошо видно на таблице номер 3. Часть классов отсутствовала на большинстве картинок

(например, быстрая вода (7 класс)), а часть классов имело очень маленькую площадь полигонов относительно общей площади изображения. Поэтому тренировочные изображения готовились отдельно для каждого класса. После первого этапа, когда были добавлены все непересекающиеся изображения размером 20x224x224, в каждый класс были добавлены дополнительные изображения с ненулевой маской. Получены они были методами плавающего окна вокруг ненулевой маски. Пример отдельного тестового изображения (20 слоев показаны как отдельные изображения) для нейронной сети приведен на рис. 4, и маска для него на рис. 5.

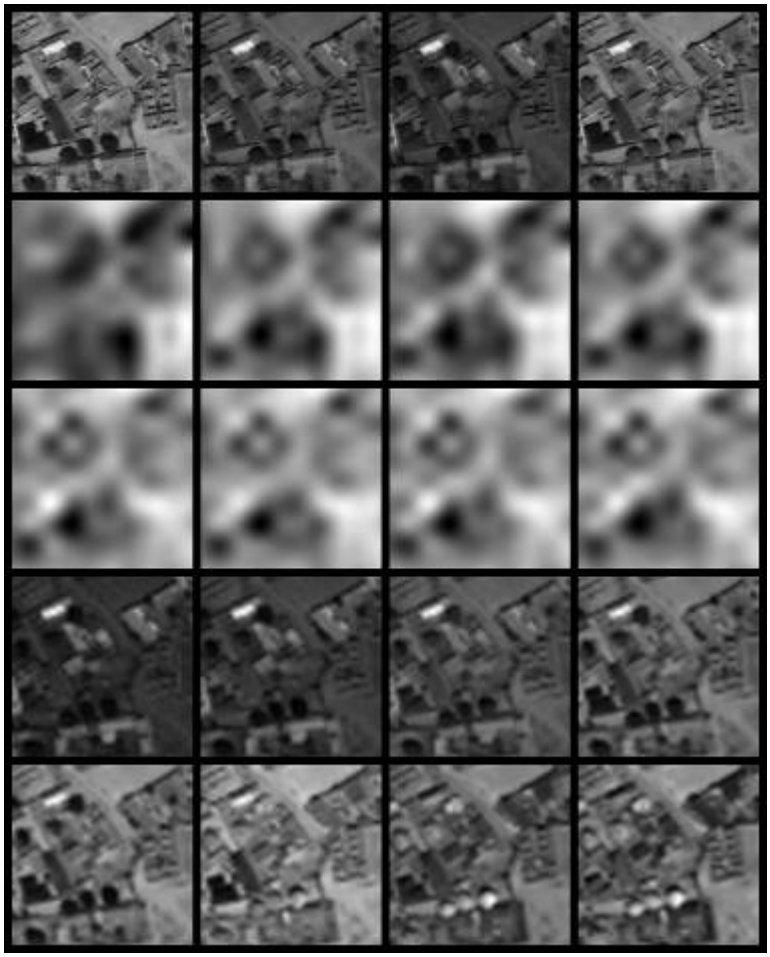

Рис. 4. – Тестовый пример для обучения нейронной сети размерности 20x224x224

Рис. 5. – Пример маски, полученной из полигонов для класса деревьев

В процессе анализа тренировочного набора был обнаружен ряд проблем, которые затрудняли обучение.

1) Классы быстрой и медленной воды не всегда хорошо отличимы друг от друга даже визуально. По быстрой воде также было предоставлено очень мало тестовых данных. В идеале эти классы следовало объединить. Часть участников обучали эти классы в рамках единой модели и затем разделяли их эвристическими методами.

2) Классы 9 и 10 были несколько искусственно разделены на большие и малые машины, дополнительно разметка по ним оставляла желать лучшего. В класс 10 также попали посторонние объекты, такие как мусорные баки.

3) Часть изображений в разных частотных диапазонах были немного смещены друг относительно друга.

Сеть UNET требует распределение яркости пикселей близкое к нормальному, поэтому требовалось посчитать арифметическое среднее и стандартное отклонение значений пикселов. Мы использовали один набор коэффициентов для всего тензора, а не 20 отдельных нормирующих коэффициентов для каждого слоя.

Мы не производили фильтрацию пикселов с краёв гистограммы для картинок. Судя по всему это приводит к потери части важной информации.

Подготовка модели

Сеть UNET в ходе экспериментов была доработана, путём увеличения глубины, добавлением BatchNormalization и Dropout уровней. Был разработан специализированный Double Convolution слой на базе которого была переписана сеть. Указанные модификации позволили сократить оверфиттинг и улучшить сходимость модели. UNET на базе этих моделей (мы назвали эту сеть ZF_UNET_224) приведена на рис. 6.

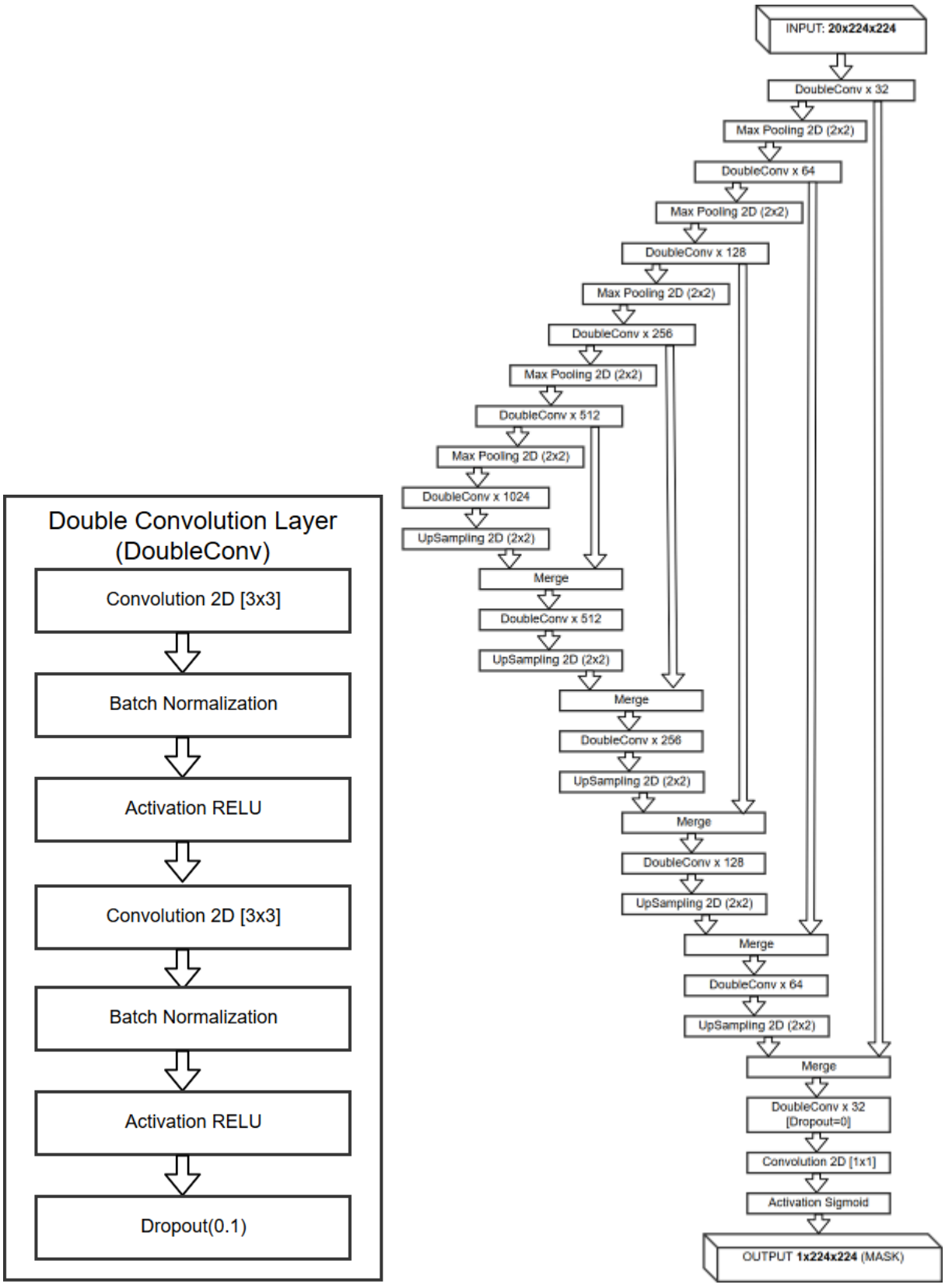

Рис. 6. – Структура модифицированного Double Convolution слоя и модифицированной сети ZF_UNET_224

В качестве Loss функции использовался немного видоизменный коэффициент Жаккара (см. формулу (1)) с дополнительными слагаемыми для того что бы сделать его дифференцируемым. Код на языке Python приведён ниже.

```
def jacard_coef(y_true, y_pred):
    y_true_f = K.flatten(y_true)
    y_pred_f = K.flatten(y_pred)
    intersection = K.sum(y_true_f * y_pred_f)
    return (intersection + 1.0) / (K.sum(y_true_f) + K.sum(y_pred_f) - intersection + 1.0)
```

Типовой процесс обучения показан на рис. 7, где по горизонтали число эпох обучения, по вертикали коэффициент Жаккарда. Синим на графике показан коэффициент Жаккарда для тренировочного набора, красным коэффициент Жаккарда для валидационных данных.

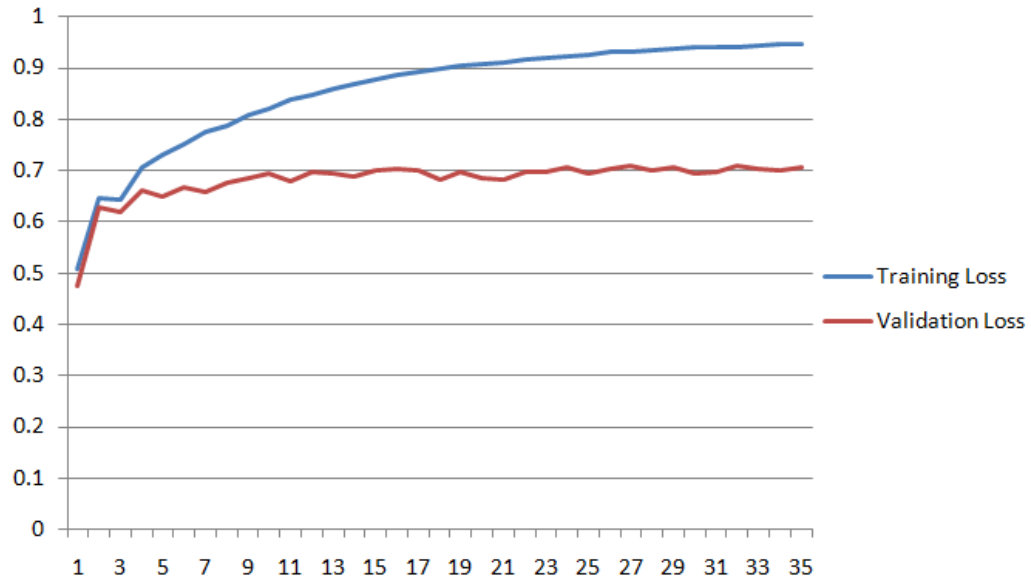

Рис. 7. – Процесс обучения нейронной сети

Обучение модели и валидация

Основной проблемой в процессе обучений была нестабильность. В некоторых случаях обучение “заходило в тупик”, достигая локального минимума, например, начиная предсказывать в качестве сегментации всю

картинку. Поэтому для нахождения оптимальных моделей использовался поиск по сетке (grid search).

В качестве узлов сетки использовались следующие параметры:

Optimizer (Adam, SGD); LR SGD: (0.05, 0.01, 0.001); LR Adam: (0.01, 0.001, 0.0001); Вращение (включено, выключено); тип модели: UNET 224x224, UNET с dropout 0.1, UNET с Batch Normalization; количество изображений на эпоху (Часть от ½ до 1)

Параметры выбирались на одной части из разбиения и затем использовались на оставшихся. После большого числа запусков стало понятно, что лучшие результаты получаются на оптимизаторе Adam, типе нейронной сети с BatchNormalization с включенными поворотами картинки. Настраивался только Learning Rate.

Процесс тренировки останавливался в случае если не было улучшений коэффициента Жаккарда в течении нескольких эпох (8-15). После получения отдельной модели из всего набора, она прогонялась на всех валидационных изображениях и рассчитывался общий индекс Жаккара. В качестве финальной модели выбиралась модель, имеющая максимальный общий индекс Жаккара по итогам Grid Search.

После получения моделей для всех 5 частей разбиения. Рассчитывался “полный тренировочный индекс Жаккара” на основе всех 25 доступных изображений. Для его расчёта использовалось полное пересечение всех полигонов по всем изображениям одновременно. Также по всем изображением считалась сумма площадей всех предсказанных полигонов и заданных полигонов. Полный тренировочный индекс Жаккара хорошо согласовывался с полученным результатом на общей Leaderboard таблице результатов Kaggle.

Поскольку вероятности для пикселей расположены равномерно на промежутке от 0 до 1, а нам требуются дискретные 0 и 1 для формирования полигонов, то на базе валидации искалось оптимальное значение THR по которому производить отсечку. Пример значения общего коэффициента Жаккара в зависимости от величины отсечки приведён на рис. 8, где по горизонтали значение THR, а по вертикали - значение общего коэффициента Жаккара на валидационных данных.

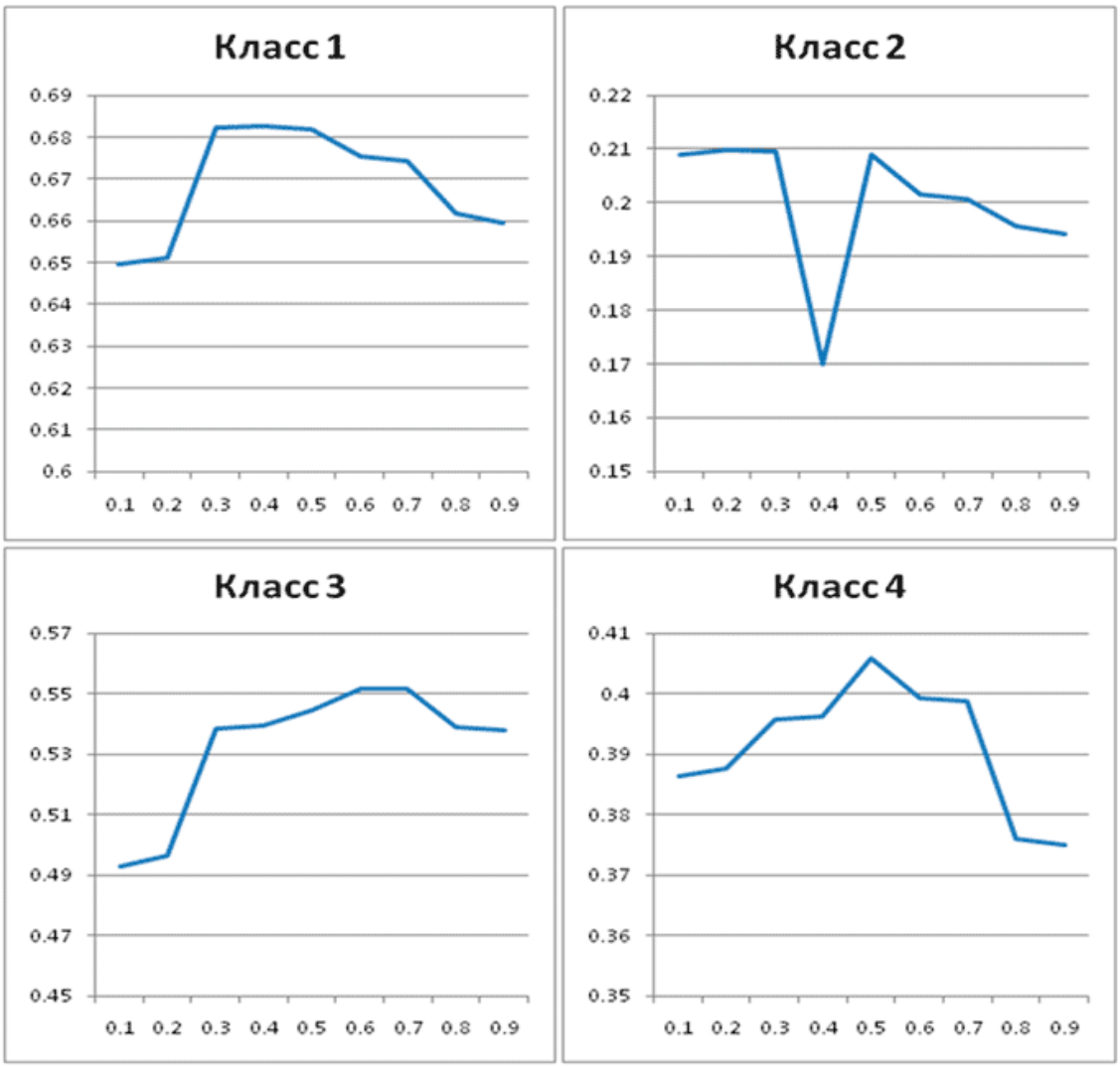

Рис. 8. – Выбор оптимального значения THR на примере классов 1-4

Обработка тестовых данных

Набор тестовых данных был намного больше тренировочных и состоял из 429 изображений. Каждое тестовое изображение обрабатывалось отдельно. Перед началом обработки, изображение по аналогии с тренировочными данными приводилось к виду матрицы 20x3360x3360 и нормализовалось.

Перед началом обработки изображения создавались два массива HEATMAP и COUNT оба размера 3360x3360, изначально заполненные нулями. Из тестового изображения последовательно извлекались изображения размером 20x224x224 со сдвигом D=112 пикселей (см. рис. 9). Для каждого изображения всеми моделями, подготовленными для данного класса делались предсказания разметки. Полученные вероятности прибавлялись к массиву HEATMAP по тем же координатам, из которых было получено изображение для анализа. К массиву COUNTS по тем же координатам соответственно прибавлялась единичка. По окончании расчета массив HEATMAP поэлементно делился на массив COUNTS. И попиксельно с использованием, полученного на этапе валидации значения THR приводился к 0 или 1. На базе полученного “двухцветного” 2D массива формировались финальные полигоны решения.

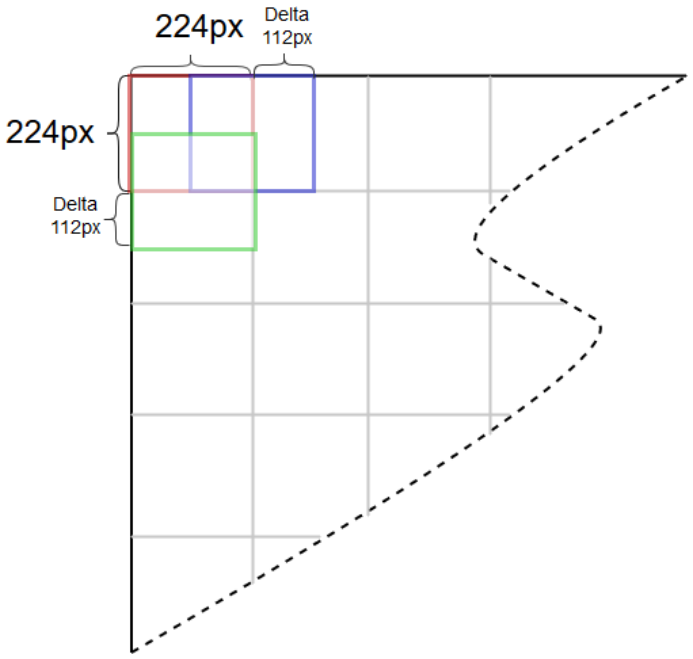

Рис. 9. – Подход на базе скользящего окна

Ниже предложены дополнительные методики увеличения точности предсказаний. Они вошли в наше финальное решение лишь частично из-за ограниченных доступных аппаратных ресурсов и ограниченное время соревнования. На обработку тестового набора для одного класса в среднем тратилось 8-10 часов.

1) Уменьшение шага для скользящего окна со 112 до 56 (и ниже) почти всегда приводит к улучшению метрики, но одновременно с этим увеличивает время расчета как $O(N^2)$. Например, для класса 4 на валидации счет вырос с 0.385714 до 0.403683 при уменьшении шага со 112 до 56 пикселей.

2) Точность распознавания на краях изображения в нейронной сети ниже. Поэтому были проведены эксперименты по сегментации используя только центральную часть UNET. Что дало для класса 5 на валидации:

Обычное значение: 0.507446

- центральная часть 160x160 пикселей из квадрата 224x224 с шагом 80 пикселей: 0.514557

- центральная часть 200x200 пикселей из квадрата 224x224 с шагом 100 пикселей: 0.512844

3) В некоторых случаях на валидации THR=0.5 не было самым оптимальным параметром. Поэтому имеет смысл использовать оптимальное значение. Например, для класса 6:

THR 0.1: 0.738943

THR 0.5: 0.757858

THR 0.9: 0.759850

На рис. 10 приведен пример получения сегментации для некоторого участка изображения для класса 5. Сначала сегментация, предсказанная каждой из 5 моделей. Затем общий HEATMAP и потом различная сегментация в зависимости от значения THR.

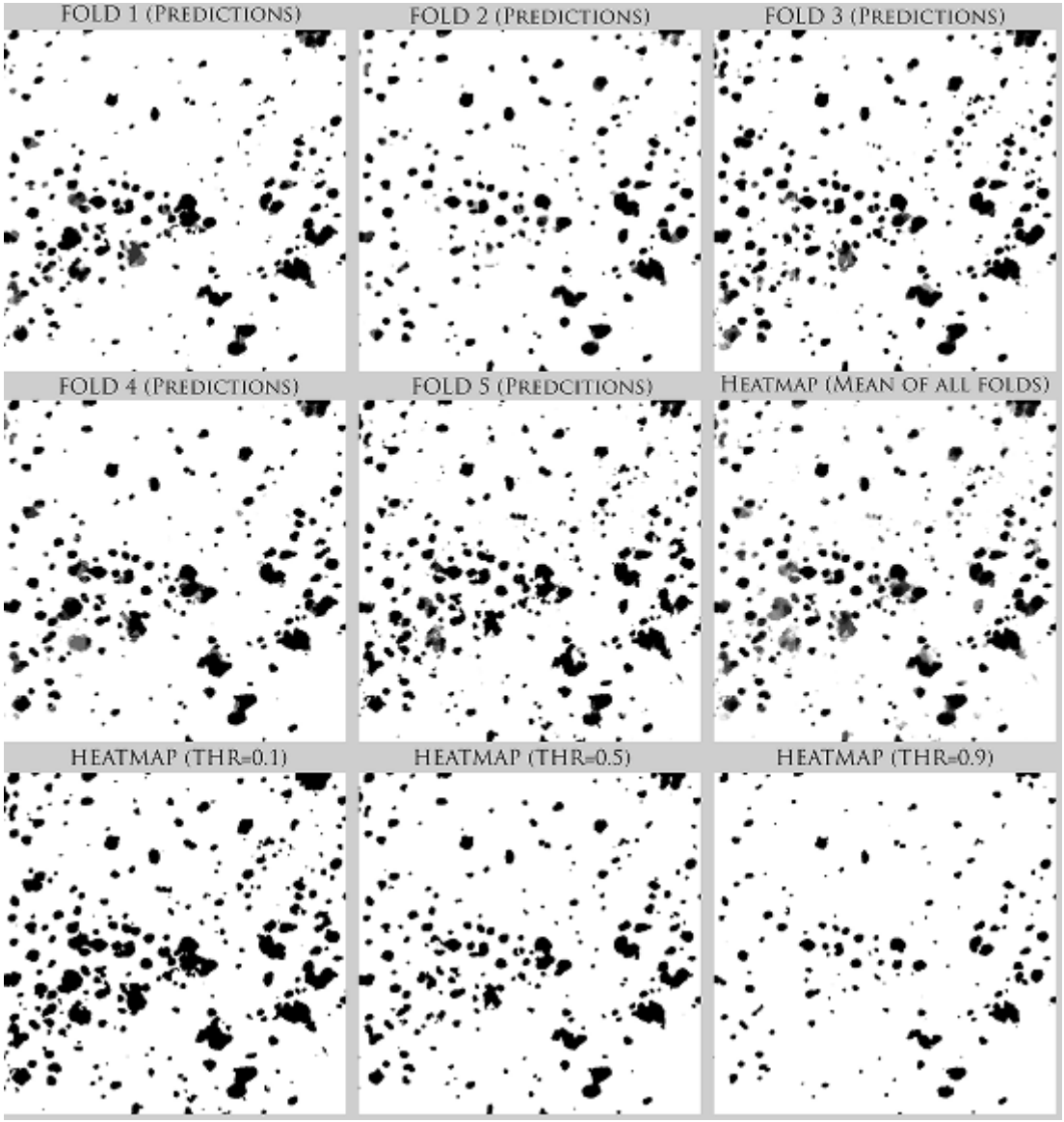

Рис. 10. – Пример сегментации для класса 5 (деревья)

Ансамбли моделей

Ансамбли моделей с разной структурой являются мощным методом для увеличения точности разметки. Наиболее очевидным способом делать ансамбли – это использовать HEATMAPS полученные в одной и другой модели. Попиксельно складываем HEATMAPS и делим пополам, тем самым находим арифметическое среднее. Если HEATMAP отсутствуют по той или иной причине, то ансамбли можно делать напрямую на полигонах. Для этого использовались методы UNION и INTERSECTION из модуля shapely. В

команде были разные подходы к решению, поэтому по некоторым классам за счёт использования мы получили неплохой прирост точности:

Класс 6 Решение 1: 0.08149 + Решение 2: 0.08103 (Intersection) Счёт: 0.08179

Класс 8 Решение 1: 0.03890 + Решение 2: 0.05322 (Intersection) Счёт: 0.06194

Класс 9 Решение 1: 0.02113 + Решение 2: 0.02713 (Union) Счёт: 0.03254

Ансамбль в случае наличия трёх различных наборов полигонов для класса можно провести по схеме из формулы:

$$Ensemble = (A \cap B) \cup (A \cap C) \cup (B \cap C), \quad (2)$$

По формуле (2) получаем ансамбль, представленный на рис. 11.

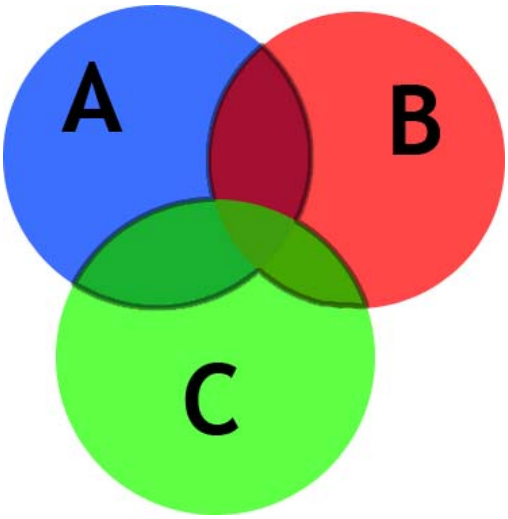

Рис. 11. – Ансамбль из трёх наборов полигонов. Чёрным контуром отмечена выбранная часть.

Постобработка

Анализ взаимоотношений между классами на тренировочном наборе показал, что после получения окончательного решения по всем 10 классам, его можно дополнительно улучшить за счёт вычитания полигонов, а также за счёт исключения полигонов с нестандартными (слишком большими или слишком малыми) размерами для данного класса.

Из таблицы 4 видно, что например грузовики (класс 9), чаще всего стоят на дорогах (39% пересечения полигонов). То же самое касается

маленьких машин (класс 10), которые можно встретить на асфальтовых и грунтовых дорогах. Деревья (класс 5) часто стоят в полях.

С другой стороны, часть классов вовсе не имеют пересечений полигонов, особенно это касается воды (классы 7 и 8). Также вода быстрая и медленная не пересекается между собой.

Таблица №4

Пересечение площадей между полигонами классов в процентах

Номер класса	1	2	3	4	5	6	7	8	9	10
1	-	0.0216	0.0004	0.0146	1.0940	0.1160	**0.0000**	**0.0000**	**0.0000**	0.0001
2	0.1016	-	0.4616	0.2071	1.9008	**21.258**	0.0204	0.0058	**0.0000**	**0.0000**
3	0.0019	0.4026	-	0.0094	0.5990	0.0069	0.0521	**0.0000**	0.1833	0.2785
4	0.0160	0.0486	0.0025	-	4.2427	1.1998	0.0000	0.0003	0.0102	0.0613
5	0.3555	0.1318	0.0476	1.2524	-	**16.791**	0.0371	0.0114	0.0000	0.0010
6	0.0139	0.5441	0.0002	0.1307	6.1976	-	0.0227	3.2785	0.1424	1.8230
7	0.0004	0.0291	0.0853	**0.0000**	0.7649	0.0227	-	**0.0000**	**0.0000**	**0.0000**
8	0.0014	0.0245	**0.0000**	0.0066	0.6943	3.2785	**0.0000**	-	**0.0000**	**0.0000**
9	0.0092	**0.0000**	**39.787**	**8.2283**	0.1517	0.1424	**0.0000**	**0.0000**	-	**0.0000**
10	0.0340	0.0008	**14.761**	**12.062**	0.7282	1.8230	**0.0000**	**0.0000**	**0.0000**	-

В нашем решении мы использовали следующие подходы:

- Удаляли слишком большие полигоны для данного класса (классы 8, 9 и 10)

- Вычитали из полигонов для машин, предсказанные полигоны для воды
- Эвристически разделяли типы воды между собой

Классы с маленькими объектами (грузовики, машины, мотоциклы)

Для классов с маленькими объектами общее решение работало не очень хорошо. Основными источниками проблем были: малые размеры объектов, небольшое их количество в тренировочном наборе, плохая тренировочная разметка, частичные сдвиги слоёв изображений полученных на разных частотных полосах между собой.

Для разметки этого класса была создана отдельная модель со входом 32x32 пикселя. Были подготовлены входные данные с большим числом изменений - повороты объектов на произвольный угол, сдвиги относительно центра. Для уменьшения числа ложноположительных срабатываний использовался индекс Тверски с большим штрафом за False Positive. Формула (3):

$$S(X,Y)=\frac{|X\cap Y|}{|X\cap Y|+\alpha|X-Y|+\beta|Y-X|}, \quad (2)$$

где X – предсказанная разметка а Y – реальная разметка.

Другие подходы к решению задачи сегментации

1) В самом начале соревнования мы пробовали подход с применением библиотеки XGBoost [12] для классификации областей 10x10. Всё изображение разбивалось на области 10x10 пикселей и модель предсказывала к какому из 10 классов принадлежит данный участок. Этот подход позволил подняться выше базового решения. Вероятно, подготовка отдельных моделей для каждого класса позволила улучшить решение еще больше, но мы ушли в сторону использования нейронной сети UNET. По окончании соревновании на форуме один из участников рапортовал, что этот метод (но на одиночных пикселях) позволил ему подняться очень высоко в общем рейтинге [13].

2) Мы использовали предобученную сеть VGG16 [14] для локализации редких классов (машины и грузовики). Сеть предсказывает наличие или отсутствие представителей указанного класса в заданной области по RGB каналам. Далее эти предсказания используются как ансамбль с предсказаниями UNET сети. Пример приведен на рис. 12.

Рис. 12. – Предсказания нейронной сети VGG16 по наличию автомобилей в заданной области

3) Мы также переделали предобученную сеть VGG16 для задач сегментации. Основная идея это поменять последний слой сети, что бы он предсказывал место на картинке 224x224 где данный класс есть в наличии. Для последнего слоя заменялась функция активации с softmax на sigmoid. Всего использовалось 16*16 = 256 нейронов, которые предсказывали для

области 14x14 пикселей наличие или отсутствие класса. Полученный результат был хуже чем при использовании UNET.

4) В то время пока задачи сегментации выполнялись без использования нейронных сетей, было разработано большое число разнообразных индексов, которые помогали делать сегментацию изображений автоматическими методами (например, NDWI) [15]. Один из таких индексов (CCCI) отлично работает для водных поверхностей, как было показано одним из участников соревнования [12]. Часть наиболее эффективных индексов рассчитанных для изображений можно добавлять как дополнительные плоскости для обучения нейронных сетей. То есть, например, для воды можно добавить индекс CCCI как 21 плоскость наряду с остальными изображениями, полученными на разных частотах. У нас не было времени протестировать много различных индексов в момент проведения соревнования.

5) Методы Conditional random field (CRF) [16] могут быть использованы для уточнения полигонов, полученных из предсказаний нейронных сетей. Эти полигоны зачастую имеют рваные края в отличие от разметки сделанной человеком. И методы CRF могут исправить эту проблему, тем самым увеличив точность.

Экспериментальные результаты

Результаты работы алгоритма проверялись на Leaderboard (доске лидеров) системы Kaggle. Лучшие результаты для полученных моделей по классам приведены в таблице 7. На базе изображений, для которых участникам неизвестна реальная разметка система автоматически подсчитывает коэффициент Жаккара.

Таблица №5

Коэффициент Жаккара сегментации для различных классов на публичном и приватном наборе изображений

Класс	Локальный рейтинг на валидации (если есть)	Публичный рейтинг, 85 изображений	Приватный рейтинг, 344 изображения
1 (Здания)	**0.68305**	**0.7832**	**0.6201**
2 (Структуры)	**0.20900**	**0.1932**	**0.2226**
3 (Дороги)	**0.54451**	**0.7744**	**0.4320**
4 (Треки)	**0.40597**	**0.3865**	**0.4214**
5 (Деревья)	**0.55900**	**0.4931**	**0.6063**
6 (Поля)	**0.75785**	**0.8103**	**0.8135**
7 (Быстрая вода)	**0.56378**	**0.9681**	**0.8996**
8 (Медленная вода)	-	**0.4708**	**0.2077**
9 (Большие машины)	-	**0.2113**	**0.0757**
10 (Автомобили и мото)	**0.07003**	**0.0321**	**0.0439**

Далее приведён пример сегментации на одном из тренировочных изображений, которое использовалось для валидации моделей. На рис. 13 - оригинальное изображение, построенное из 3 каналов (RGB), на рис. 14 - реальная разметка, подготовленная человеком, а на рис. 15 - разметка, полученная нашим набором моделей.

Рис. 13. – Одно из тренировочных изображений

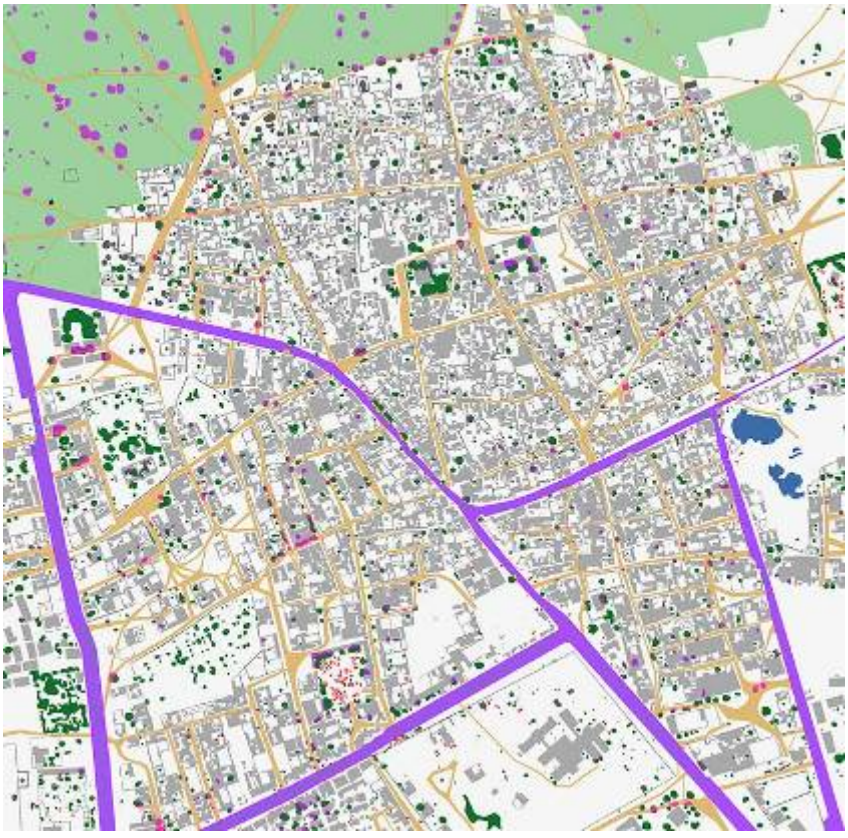

Рис. 14. – Разметка, подготовленная авторами набора данных

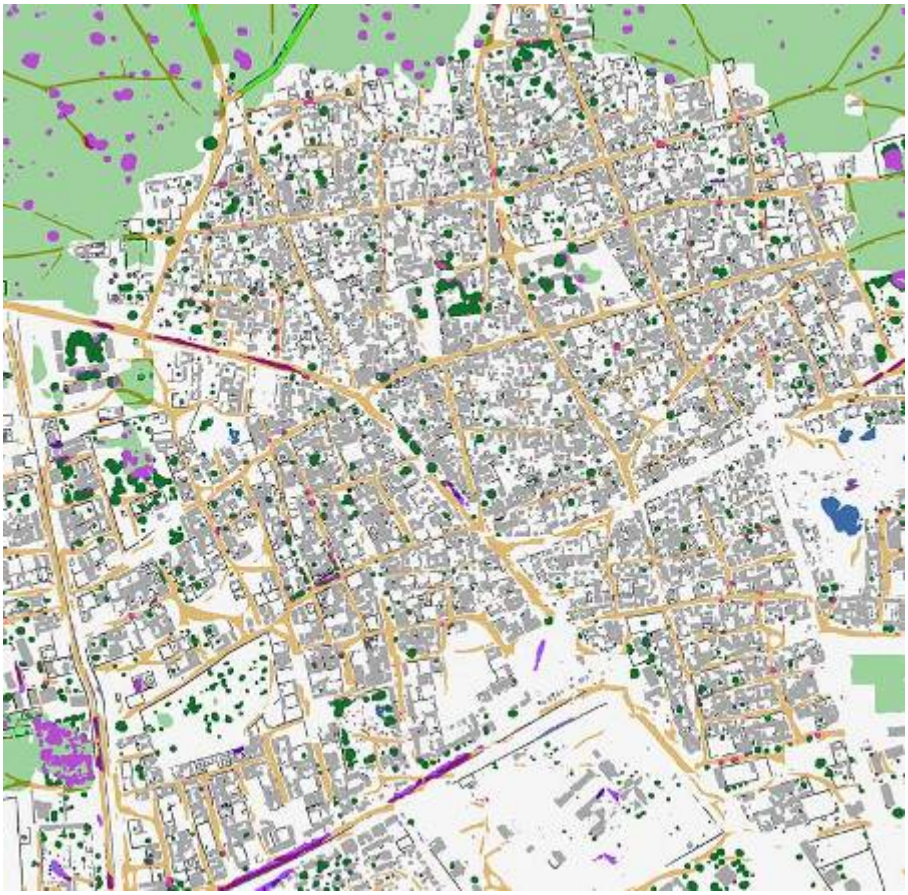

Рис. 15. – Разметка, предсказанная нашей моделью

На рис. 16 и 17 показано предсказание на одном из тестовых снимков (для которых нет тренировочной разметки). На рис. 18-20 приведены фрагменты сегментации в масштабе 1 к 1, соответственно оригинал картинки, тренировочная разметка и разметка, полученная нашей моделью.

Рис. 16. – Оригинал снимка из тестового набора (в RGB)

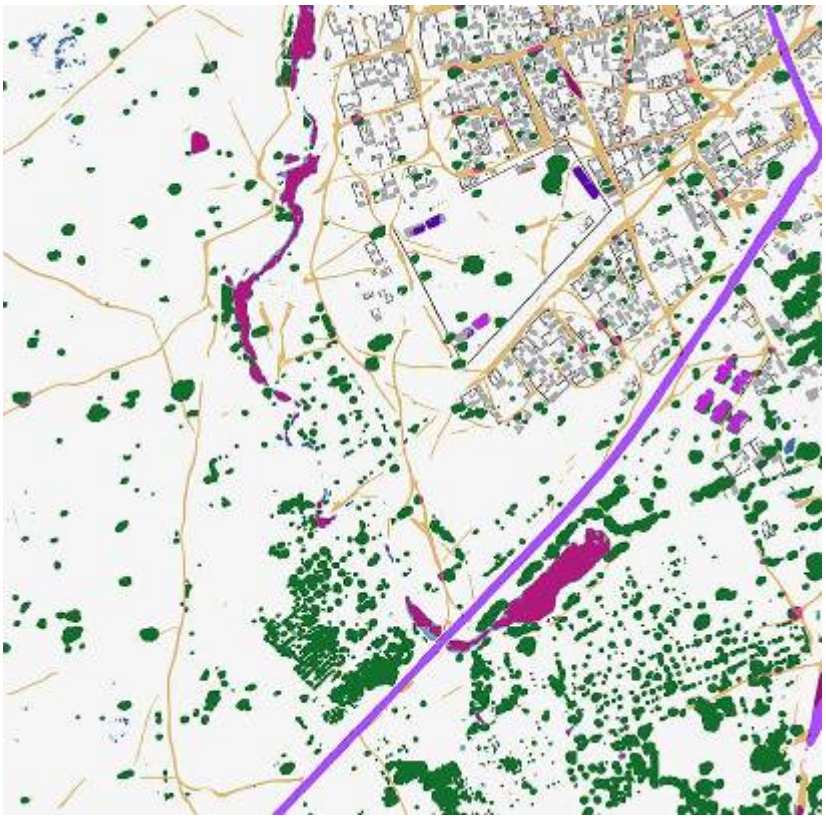

Рис. 17. – Разметка, полученная для тестового снимка с помощью нашей модели

Рис. 18. – Часть тренировочной картинки в масштабе 1 к 1

Рис. 19. – Разметка, предоставленная вместе с тренировочными данными

Рис. 20. – Предсказанная разметка в масштабе 1 к 1

Более подробно процесс обучения нейронной сети можно посмотреть на видео [17], которое мы подготовили по лог файлам обучения (к сожалению, в видео использовались не самые последние версии моделей и их точность несколько ниже). Во втором видео проведено детальное сравнение сегментации по разным классам [18]. Сеть ZF_UNET_224 для библиотеки Keras опубликована под свободной лицензией на GitHUB [19].

Заключение

Нейронные сети показали свою высокую эффективность при решении задачи автоматической сегментации спутниковых изображений. Даже в условиях ограниченного времени (2 месяца) удалось подготовить модель, которая автоматически создаёт разметку местности, похожую на разметку сделанную вручную. Несомненно, при расширении тестового набора и доработке структуры нейросети можно увеличить качество предсказаний до

гораздо более высокого уровня. Работа выполнена при поддержке гранта «РФФИ 17-07-00409».

Литература

1. United Kingdom - Salisbury: Research and experimental development services // URL: publictenders.net/node/3556044/.

2. Dstl Satellite Imagery Feature Detection. Private Leaderboard Final! // URL: kaggle.com/c/dstl-satellite-imagery-feature-detection/discussion/30135/.

3. Сайфеддин Д., Булгаков А. Г., Круглова Т. Н. Нейросетевая система отслеживания местоположения динамического агента на базе квадрокопетра //Инженерный вестник Дона, 2014, №.1 URL: ivdon.ru/magazine/archive/n1y2014/2293/.

4. Плуготаренко Н. К., Варнавский А. Н. Применение нейронных сетей для построения модели прогнозирования состояния городской воздушной среды //Инженерный вестник Дона, 2012, №4-2 URL: ivdon.ru/magazine/archive/n4p2y2012/1351/.

5. Dstl Satellite Imagery Feature Detection // URL: kaggle.com/c/dstl-satellite-imagery-feature-detection/.

6. Ronneberger O., Fischer P., Brox T. U-Net: Convolutional Networks for Biomedical Image Segmentation // Arxiv.Org. 2015. URL: arxiv.org/abs/1505.04597/.

7. Real R., Vargas J. M. The probabilistic basis of Jaccard's index of similarity //Systematic biology. – 1996. – V. 45. – №. 3. – pp. 380-385.

8. Tolias Y. A., Panas S. M., Tsoukalas L. H. Generalized fuzzy indices for similarity matching //Fuzzy Sets and Systems. – 2001. – V. 120. – №. 2. – pp. 255-270.

9. WorldView-3 Satellite Sensor (0.31m) // URL: satimagingcorp.com/satellite-sensors/worldview-3/.

10. Simonyan K., Zisserman A. Very Deep Convolutional Networks for Large-Scale Image Recognition // Arxiv.Org. 2014. URL: arxiv.org/abs/1409.1556/.

11. Deng J. et al. Imagenet: A large-scale hierarchical image database //Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on. – IEEE, 2009. – pp. 248-255.

12. Chen T., Guestrin C. Xgboost: A scalable tree boosting system //Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. – ACM, 2016. – pp. 785-794.

13. Dstl Satellite Imagery Feature Detection. Pixel-based method // URL: kaggle.com/c/dstl-satellite-imagery-feature-detection/discussion/29802

14. List of available Indices. // URL: indexdatabase.de/db/i.php/.

15. Osin V. Reflectance Index for Water Way // URL: kaggle.com/resolut/dstl-satellite-imagery-feature-detection/waterway-0-095-lb/.

16. Wang Y., Loe K. F., Wu J. K. A dynamic conditional random field model for foreground and shadow segmentation //IEEE transactions on pattern analysis and machine intelligence. – 2006. – V. 28. – №. 2. – pp. 279-289.

17. DSTL competition. Learning process visualization // URL: youtube.com/watch?v=OfGsiPyx94I/.

18. DSTL Competition. Examples of automatic neural net segmentation // URL: youtube.com/watch?v=rpp7ZhGb1IQ/.

19. ZF_UNET_224 Pretrained Model // URL: github.com/ZFTurbo/ZF_UNET_224_Pretrained_Model/.

References

1. United Kingdom - Salisbury: Research and experimental development services. URL: publictenders.net/node/3556044/.

2. Dstl Satellite Imagery Feature Detection. Private Leaderboard Final! URL: kaggle.com/c/dstl-satellite-imagery-feature-detection/discussion/30135/.

3. Sayfeddin D., Bulgakov A. G., Kruglova T. N. Inženernyj vestnik Dona (Rus), 2014, №1. URL: ivdon.ru/magazine/archive/n4p2y2012/1351/.

4. Plugotarenko N. K., Varnavskiy A. N. Inženernyj vestnik Dona (Rus), 2012, №.4-2. URL: ivdon.ru/magazine/archive/n4p2y2012/1351/.

5. Dstl Satellite Imagery Feature Detection. URL: kaggle.com/c/dstl-satellite-imagery-feature-detection/.

6. Ronneberger O., Fischer P., Brox T. U-Net: Convolutional Networks for Biomedical Image Segmentation. Arxiv.Org. 2015. URL: arxiv.org/abs/1505.04597/.

7. Real R., Vargas J. M. Systematic biology. 1996, V. 45, №3, pp. 380-385.

8. Tolias Y. A., Panas S. M., Tsoukalas L. H. Fuzzy Sets and Systems. 2001, V. 120, №2, pp. 255-270.

9. WorldView-3 Satellite Sensor (0.31m). URL: satimagingcorp.com/satellite-sensors/worldview-3/.

10. Simonyan K., Zisserman Arxiv.Org. 2014. URL: arxiv.org/abs/1409.1556/.

11. Deng J. et al. Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on. IEEE, 2009, pp. 248-255.

12. Chen T., Guestrin C. Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining. ACM, 2016, pp. 785-794.

13. Dstl Satellite Imagery Feature Detection. Pixel-based method. URL: kaggle.com/c/dstl-satellite-imagery-feature-detection/discussion/29802/.

14. List of available Indices. URL: indexdatabase.de/db/i.php/.

15. Osin V. Reflectance Index for Water Way. URL: kaggle.com/resolut/dstl-satellite-imagery-feature-detection/waterway-0-095-lb/.

16. Wang Y., Loe K. F., Wu J. K. IEEE transactions on pattern analysis and machine intelligence. 2006, V. 28, №2, pp. 279-289.

17. DSTL competition. Learning process visualization.URL: youtube.com/watch?v=OfGsiPyx94I/.

18. DSTL Competition. Examples of automatic neural net segmentation. URL: youtube.com/watch?v=rpp7ZhGb1IQ/.

19. ZF_UNET_224 Pretrained Model. URL: github.com/ZFTurbo/ZF_UNET_224_Pretrained_Model/.