Санкт-Петербургский государственный университет

Математическое обеспечение и администрирование информационных систем
Технологии программирования

Валентин Румянцев

Использование методов машинного обучения для семантической классификации дорожной обстановки

Бакалаврская работа

Научный руководитель:
к.ф.-м.н., ст. преп. СПбГУ Салищев С. И.

Рецензент:
к.т.н., доцент ГУАП Ронжин А. Л.

Санкт-Петербург
2018

SAINT-PETERSBURG STATE UNIVERSITY

Software and Administration of Information Systems
Technology in Programming

Valentin Rumyantsev

Application of machine learning methods for semantic classification of road conditions

Graduation Thesis

Scientific supervisor:
PhD, senior lecturer Sergey Salishev

Reviewer:
PhD, assoc. prof. Alexander Ronzhin

Saint-Petersburg
2018

Оглавление

Введение

Машинное обучение в наше время переживает новое рождение и постепенно входит во все отрасли человеческой деятельности, начиная с рекомендации рекламы, заканчивая управлением автоматическими заводами. В том числе сложные и важные задачи решаются методами компьютерного зрения:

- беспилотные автомобили используют камеры и радары для того, чтобы правильно взаимодействовать друг с другом.
- Системы ПВО обнаруживают стелс-самолеты с помощью высокоточных камер.
- вывески автоматически определяют пол проходящего мимо человека с целью показать таргетированную рекламу
- автоматически распознаются аномалии на рентгеновских снимках;

Все это и многое другое стало возможно в связи с революцией в области высокопроизводительных видеокарт и сверточных сетей([7]). Работа рассматривает конкретную область комьютерного зрения – семантическую сегментацию изображений для задачи понимания дорожной обстановки, подробнее об этом будет рассказано в следующих разделах, но позволю привести себе практический пример для чего это точно необходимо. Известно, что сейчас мы переживаем бум развития беспилотного транспорта, многие компании сейчас выпустили свои прототипы(самые известные из них – Google, Uber, Yandex), и очень важно для такой машины в реальном времени находить дорожные знаки, пешеходов, другие машины и прочие семантические классы, иначе далеко не уедешь.

Мы рассмотрим методы, начиная от представляющих большую историческую ценность, заканчивая теми, которые могут быть эффективно использованы на устройствах не таких мощных, как GPU-сервер на Amazon AWS.

1. Постановка задачи

Семантическая сегментация — это процесс разбиения изображения на составные части (сегменты) и одновременная классификация этих частей. Разбиение происходит таким образом, что при объединении всех частей вместе, получается оригинальное изображение.

Каждый сегмент изображения состоит из пикселей, пиксели одного сегмента имеют одну и ту же метку класса, которая показывает принадлежность данного сегмента к классу сегментов, объединенных какими-либо общими характеристиками, например, текстурой, цветом, яркостью, наличием повторяющихся элементарных объектов и т.п. Примеры классов сегментов: дорога, автомобиль, небо, пешеход, здание, животное, велосипед и т.п. Пример работы семантической сегментации представлен на рисунке 1.

Соответственно, задача семантической сегментации состоит в нахождении на изображении областей (сегментов), а также их классификация по заранее заданному набору классов. Не существует универсального метода решения данной задачи, поэтому чаще всего выбор метода основывается на предметной области, в которой эта задача ставится. Однако, в последнее десятилетие на первый план по исследованиям и частоте применения (особенно для сегментации дорожной обстановки) вышел метод, основанный на попиксельной сегментации.

Попиксельная сегментация состоит в отдельной классификации каждого пикселя изображения, а на основе этого уже происходит восстановление сегментов. Классификация каждого пикселя чаще всего происходит с учетом окружающих его пикселей, то есть не независимо. Способ сегментации зависит от конкретного набора методов, которые будут рассмотрены в следующих главах.

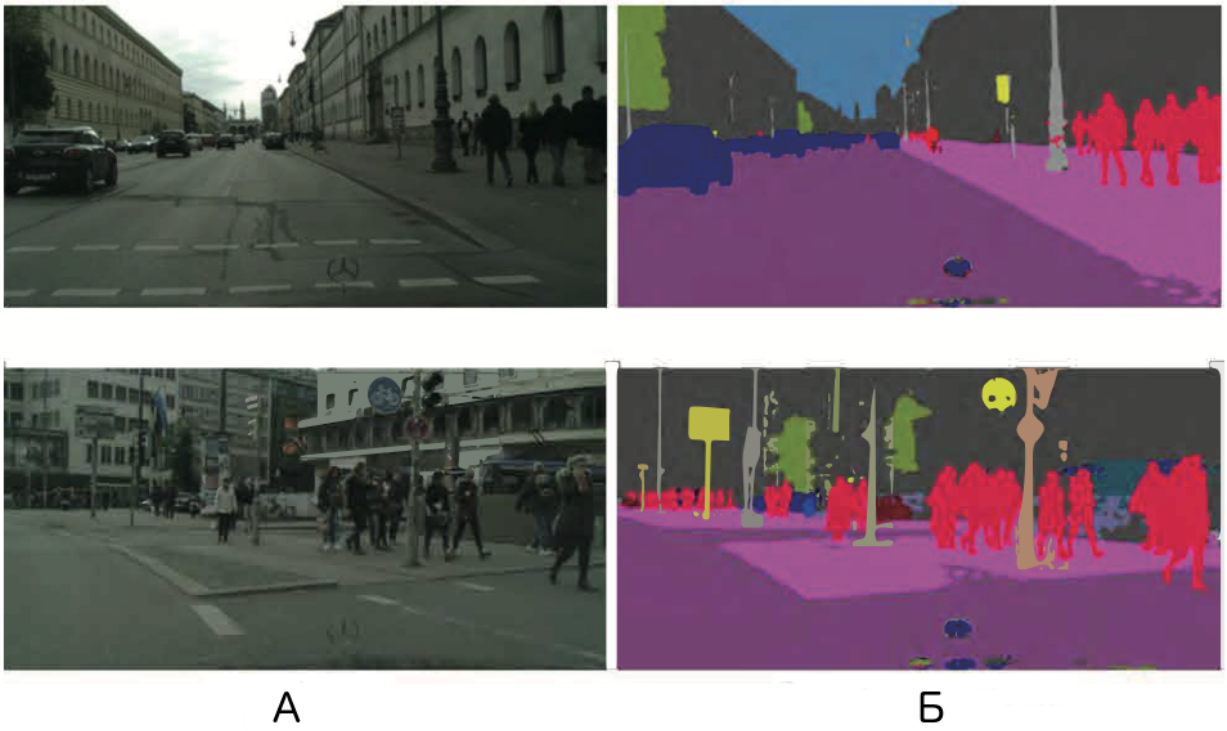

Рис. 1: Пример сегментации. (А) - оригинальная дорожная обстановка. (Б) - семантически размеченная дорожная обстановка

2. Обзор наборов данных

Cityscapes [4] – Этот набор данных состоит из 5000 аннотированных изображений, из которых 2975 доступны для обучения, 500 рекомендуются для валидации, а остальные 1525 рекомендованы для финального тестирования моделей. Считается лучшим открытым набором данных из-за высокого качества разметки и хорошей проработки дорожных сценариев. Содержит фотографии 50 городов, сделанных во все времена года, при различных погодных условиях.

Рис. 2: Пример образца из cityscapes [4]

CamVid [3] – набор данных для семантической сегментации, первый набор видеороликов с семантическими метками класса объектов, в комплекте с метаданными. База данных содержит наземные метки истинности, которые связывают каждый пиксель с одним из 32 семантических классов, таких как животное, ребенок, дорога, машина, дерево, столб и другие.

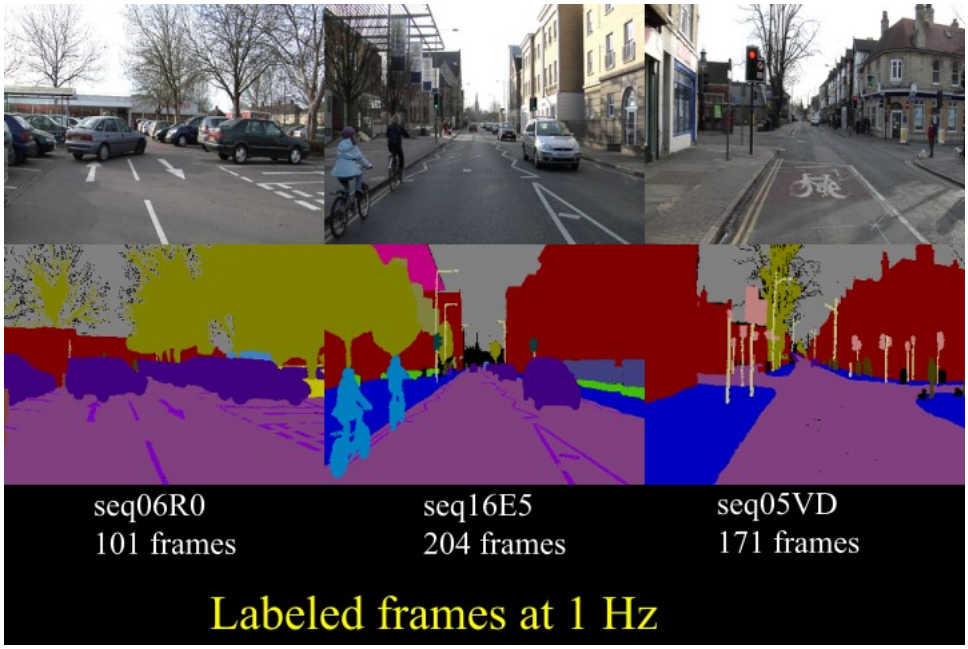

Рис. 3: Пример изображений и разметки из CamVid[3]

3. Метрики для оценки качества сегментации

В основном для оценки семантической сегментации используют нижеизложенные метрики.

Введем некоторые обозначения:

- t_i – число пикселей в классе i
- n_{ij} – число пикселей в классе i, каторые мы отнесли к классу j: Отсюда возникает несколько вариантов:
  - n_{ii} – число правильно классифицированных пикселей (true positives)
  - n_{ij} – число неправильно классифицированных как j пикселей (false positives)
  - n_{ji} – число пикселей неверно неотнесенных к классу j (false negatives)

Метрики

1. Попиксельная точность – метрика показывающая какой процент пикселей был правильно отнесен к своему семантическому классу.
2. средняя точность – $\frac{1}{k}\sum_i^k \frac{n_{ii}}{t_i}$, где k – количество классов
3. среднее Intersection Over Unin(IO) – $\frac{1}{k}\sum_i^k \frac{n_{ii}}{t_i - n_{ii} + \sum_j^k n_{ji}}$, где k – количество классов
4. взвешенный IU – $(\sum_i^k t_i)^{-1}\sum_i^k t_i \frac{n_{ii}}{t_i - n_{ii} + \sum_j^k n_{ji}}$, где k – количество классов

4. Обзор методов

4.1. Классические методы машинного обучения

Такие методы машинного обучения, как Случайный лес (Random Forest, RF) [9], Опорная машина векторов (Support vector machine, SVM) [13, 15], Логистическая регрессия (Logistic Regression, LR) [1] — являются классическими методами машинного обучения с учителем, входная размерность (количество входных признаков) которых фиксирована. Экспериментально удалось показать, что случайный лес лучше остальных справляется с задачей семантической попиксельной сегментации [12].

Принцип работы классических методов для попиксельной сегментации состоит в следующем:

1. Зафиксировать количество ближайших соседей, признаки которых будут использоваться для классификации каждого пикселя. Назовем это сканирующим окном

2. Определить какие именно признаки пикселя будут использоваться, например значения RGB (мера красного, зеленого и синего цветов)

3. Составить обучающую выборку, где каждому пикселю (и его метке класса), для которого нужно предсказать класс будет соответствовать вектор признаков, состоящий из признаков соседей пикселя

4. Вектора признаков для пикселей, находящихся на границе изображения, необходимо дополнить до необходимой размерности с помощью заполняющего (padding) значения. Это необходимо в угоду ограничениям классических методов машинного обучения (количество входных признаков должно быть фиксировано)

5. Обучить одну или несколько моделей (в зависимости от подхода)

При увеличении сканирующего окна классификатор будет получать больше информации о соседях пикселя, однако все меньше различий для него будет между соседними пикселями. При это размер сканирующего окна при таком подходе можно варьировать от небольшого до размера всего изображения (в таком случае размеры всех изображений в распределении должны совпадать, либо должен быть зафиксирован конкретный размер и, если изображение меньше, то оно дополняется заполняющим символом, а если больше - обрезается)

Классические методы показывают заметно худшую точность семантической сегментации по сравнению с нейросетевыми подходами [14], а также крайне вычислительно неэффективны(из-за необходимости ходить по изображению окнами), поэтому в данной работе рассмотрены не будут.

4.2. Нейросетевые методы

4.2.1. Общее описание

В настоящее время методы, использующие сверточные нейронные сети являются state of the art во многих задачах компьютерного зрения и семантическая сегментация не исключение.

Превосходство достигается с одной стороны из-за принципиального свойства сверточной сети автоматически выделять признаки изображений(и она делает это намного лучше, чем фантазия специалистов по компьютерному зрению) – за счет этого достигается превосходство по качеству, с другой – из-за возросшей мощности комьютерных систем(в частности появление мощных GPU), что позволяет работать таким нейросетевым архитектурам, обучение и применение которых ранее считалось невозможным. Сейчас же, обучение таких моделей занимает разумное время, а применение возможно в реальном времени даже на мобильных устройствах.

Впервые значительное превосходство нейронных сетей в задачах компьютерного было показано в работе, посвященной победе в соревновании ImageNet.[7]

$$(I * K)_{xy} = \sum_{i=1}^{h} \sum_{j=1}^{w} K_{ij} \times I_{x+i-1,y+j-1}$$

Рис. 4: Оператор свертки

Очень важно, что нейронные сети позволяют отойти от крайней вычислительно-неэффективной процедуры скользящего окна в пользу архитектуры энкодера-декодера, работая со всей картинкой целиком, а не с ограниченной её областью.

4.2.2. Сверточная нейронная сеть

Преимущественно, сверточные нейронные сети [8] используются для обработки изображений. Существует весьма эффективный способ решения этой задачи, который обращает в свою пользу саму структуру изображения: предполагается, что пиксели, находящиеся близко друг к другу, теснее "взаимодействуют" при формировании интересующего нас признака, чем пиксели, расположенные в противоположных углах. Кроме того, если в процессе классификации изображения небольшая черта считается очень важной, не будет иметь значения, на каком участке изображения эта черта обнаружена. Введем понятие оператора свертки. Имея двумерное изображение I и небольшую матрицу K размерности $(h * w)$ (ядро свертки), построенная таким образом, что графически кодирует какой-либо признак, мы вычисляем свернутое изображение $I * K$, накладывая ядро на изображение всеми возможными способами и записывая сумму произведений элементов исходного изображения и ядра (рис. 4). На самом деле, точное определение предполагает, что матрица ядра будет транспонирована, но для задач машинного обучения не важно, выполнялась эта операция или нет.

На рис. 5 схематически изображена вышеуказанная формула.

Оператор свертки представляет из себя рабочую единицу сверточного слоя сверточной нейронной сети. Сам слой состоит из набора таких ядер, которые настраивают свои веса во время обучения и каждый из них обучается на определенный тип паттернов исходного распре-

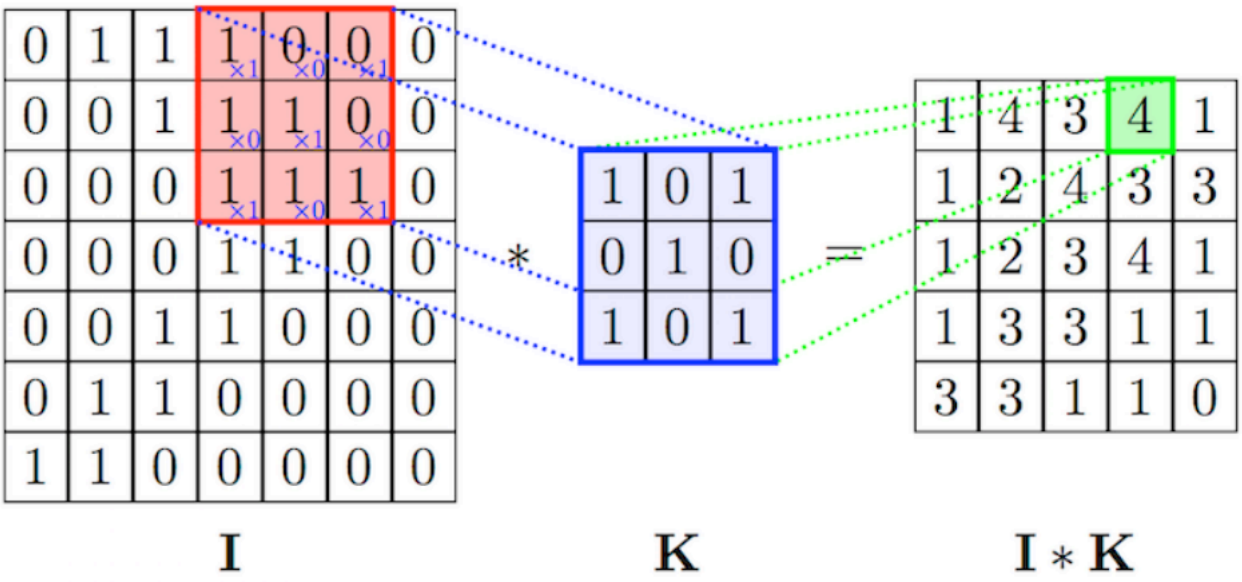

Рис. 5: Операция свертки

деления. Кроме сверточного слоя, важную роль играют субдискретизирующие слои (пример на рис. 6), цель которых - уменьшить размерность поступающего на вход слоя изображения. Причем происходит это достаточно просто - если изображение уменьшается в два раза по каждой размерности, то последовательно из квадрата (4 пикселя) выбирается максимальный, а остальные отбрасываются. Таким образом размер изображения уменьшается в два раза по длине и ширине, а большинство контента остается, лишь какая-то его часть теряется из-за принесенного шума. Соответственно, существует и обратная операция, увеличивающая размерность изображения (такие слои часто называют Upsampling(в русскоязычной литературе апсэмплинг)).

Само собой, после извлечения признаков из изображения благодаря картам признаков, необходимо найденные признаки сопоставить, например, с метками классов (в случае задачи классификации), и обучить модель выделять именно те признаки, которые являются значимыми в рамках решения определенной задачи, в рамках контекста. Для этого на конце сверточной нейронной сети располагаются обычные полносвязные слои (как в многослойном персептроне), благодаря этим слоям сеть объединяет найденные признаки в совместные абстракции и находит зависимости в данных.

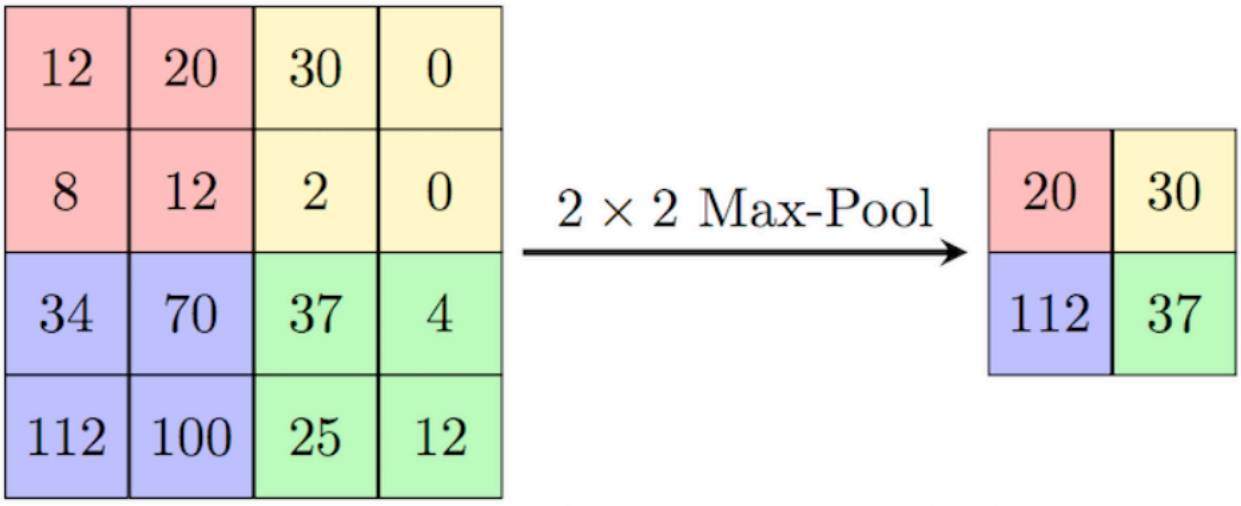

Рис. 6: Субдискретизирующий (maxpooling) слой

4.2.3. SegNet

Модель SegNet [2] является типичным автокодировщиком, основанным на сверточной нейронной сети. Схема этой модели представлена на рис. 7. Сеть состоит из блоков, в каждом блоке присутствуют свертки и пулинги (субдискретизирующие слои) или слои, повышающие дискретизацию (апсэмплинг слои), а также активационные слои ReLU [10] и слои нормализации батч-нормализации(BatchNorm) [6]. Архитектура полностью симметрична, кроме слоя мягкого максимума(Softmax) на конце декодера. Этот слой преобразует каждый пиксель выходной матрицы в целое число, показывающее класс данного пикселя. Главное отличие SegNet от обычного сверточного автокодировщика в том, что его апсэмплинг слои декодера информационно соединены с соответствующими пулинг слоями энкодера. То есть, апсэмплинг слои сети не обучаются, а получают необходимую информацию о том, как повысить размерносить и как восстановить сжатую (утерянную) топологию от соответствующих пулинг слоев, которые хранят индексы активированных пикселей (пикселей с наибольшим значением в окне).

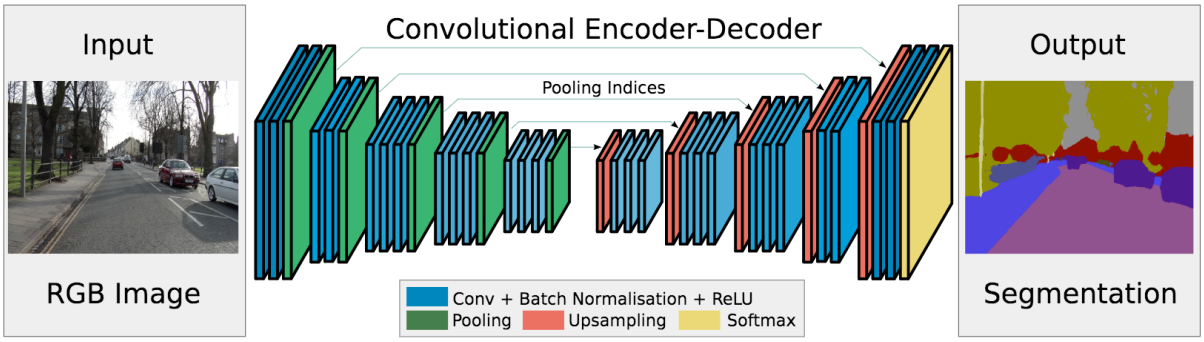

Рис. 7: Архитектура SegNet [2]

4.2.4. UNet

Unet является одним из ярких представителей сегментационных энкодер-декодер моделей, несмотря на то, что впервые он заявлен как сеть для сегментации биологических снимков [11], он хорошо зарекомендовал себя как бэйзлайн при решении практических задач и соревнований сегментации.

Архитектура сети показана на рис.2 и содержит две части, сужающуюся(энкодер) и расширяющуюся(декодер). Первая представляет из себя типичную архитектуру сверточную классификационной сети. Состоит из повторяющихся применений двух сверток 3x3, за которыми следуют активация (ReLU) и операция макс-пулинга 2x2 с шагом 2. На каждом шаге повышаем количество каналов вдвое.

Расширяющаяся часть состоит из шагов так называемой обратной свертки(деконволюции) 2x2, уменьшающей количество каналов, затем конкатенация с соответствующим образом обрезанным карту признаков от соответствующей части(видно на схеме) сужающейся части и две свертки 3x3 и активация(Relu). Обрезка необходима из-за потери пограничных пикселей в каждой свертке. (эта часть очень похожа на соответствующую в SegNet, но на самом деле это похоже только по картинке, а механика процесса совершенно другая, там, в SegNet эта связь обозначает, что мы используем индексы из пуллинг слоя для повышения размерности картинки, а здесь пробрасываем выходы слоя не только на следующий латентный слой, но и пропуская несколько

уровней, таким образом модель использует на этапе восстановления маски давно забытые признаки, что хорошо сказывается на результатах сегментации)

На последнем уровне свертка 1x1 используется для сопоставления каждому 64- компонентного вектору признаков класса.

В общей сложности сеть имеет 23 сверточных слоя.

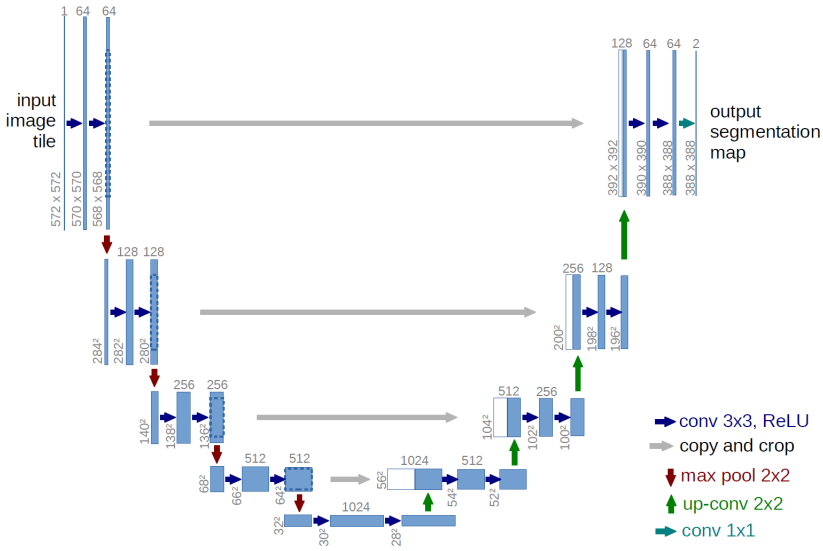

Рис. 8: Архитектура U-net (пример для 32x32 пикселей в самом низком разрешении) [11]

4.2.5. Enet

Enet [5], в отличии от вышеизложенных моделей, состоит из блоков с нетривиальной структурой, называемой bottleneck(бутылочное горлышко, в русскоязычной литературе часто используют транскрипцию – боттлнек)

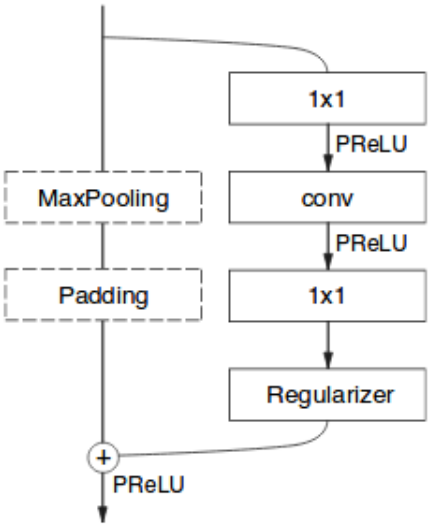

Рис. 9: Архитектура bottleneck(бутылочного горлышка) [5]

Bottleneck состоит из основной ветви с макс-пулингом и паддингом и вспомогательной(со свертками), затем результаты ветвей сливаются и проходят через активацию PRelu. Каждый блок состоит из трех сверточных слоев:

- 1×1 – проекция, уменьшающая размерность
- Основная свертка 3x3(обозначается conv)
- 1x1 – расширение

между всеми свертками стоит активационный PRelu слой.

В качестве Regularizer берут Dropout с параметром p=0.1.

В таблице приведена архитектура enet, где стадии модели отделены горизонтальными линиями, первые 3 – это энкодер, остальные – декодер. Можно заметить, что в отличии от SegNet и Unet, декодер гораздо

Name	Type	Output size
initial		$16 \times 256 \times 256$
bottleneck1.0	downsampling	$64 \times 128 \times 128$
$4\times$ bottleneck1.x		$64 \times 128 \times 128$
bottleneck2.0	downsampling	$128 \times 64 \times 64$
bottleneck2.1		$128 \times 64 \times 64$
bottleneck2.2	dilated 2	$128 \times 64 \times 64$
bottleneck2.3	asymmetric 5	$128 \times 64 \times 64$
bottleneck2.4	dilated 4	$128 \times 64 \times 64$
bottleneck2.5		$128 \times 64 \times 64$
bottleneck2.6	dilated 8	$128 \times 64 \times 64$
bottleneck2.7	asymmetric 5	$128 \times 64 \times 64$
bottleneck2.8	dilated 16	$128 \times 64 \times 64$
Repeat section 2, without bottleneck2.0		
bottleneck4.0	upsampling	$64 \times 128 \times 128$
bottleneck4.1		$64 \times 128 \times 128$
bottleneck4.2		$64 \times 128 \times 128$
bottleneck5.0	upsampling	$16 \times 256 \times 256$
bottleneck5.1		$16 \times 256 \times 256$
fullconv		$C \times 512 \times 512$

Рис. 10: архитектура enet [5]

меньше энкодера, так как считается, что задача выделения признаков, построения латентного пространства гораздо более сложная, чем восстановить маску из хороших признаков.

Несмотря на большую глубину enet, он содержит в 10-тки раз меньше параметров, чем Unet и SegNet, что хорошо сказывается на скорости работы и делает эту модель применимой для сегментации на мобильных устройствах в реальном времени.

5. Эксперименты и результаты

В качестве основных экспериментов работы было проведено сравнение современных методов, описанных в главе 4.2 (Нейросетевые методы). Модели SegNet, UNet и Enet были обучены и протестированы на наборе данных Cityscapes, используя идентичные разбиения на тренировочную и тестовую подвыборки. В качестве результатов эксперимента в данной главе представлены основные метрики моделей и сравнительные примеры их работы на тестовых объектах набора данных Cityscapes.

5.1. Результаты

В данной главе представлены результаты проведенных экспериментов. Все модели тренировались на одной и той же подвыборке данных Cityscapes, размер всех изображений в тренировочной и тестовой части составлял 1024x2048 пикселей.

Предварительно все библиотеки и модели были проверены на работоспособность на ноутбуке без видеокарты с процессором Intel i7-4700HQ 2.2Ghz (это делалось для того, чтобы не тратить лишнее время на платном ресурсе). Все вычисления производились с использованием инстанса p2.xlarge, предоставляемого облачным сервисом Amazon AWS. В общей сложности на все эксперименты понадобилось около 5-ти суток(130 часов) непрерывных вычислений (во время простоя инстанс отключался). Характеристики p2.xlarge следующие:

1. 4 ядра процессора Intel i7-5700HQ 2.7Ghz
2. 61Gb оперативной памяти
3. 1 Видеокарта NVIDIA Tesla K80 с 12Gb видеопамяти

На 5.1 изображены примеры сегментированных изображений с помощью моделей SegNet, Unet и Enet. На данном рисунке можно заметить, что результаты данных моделей похожи, однако Enet все же

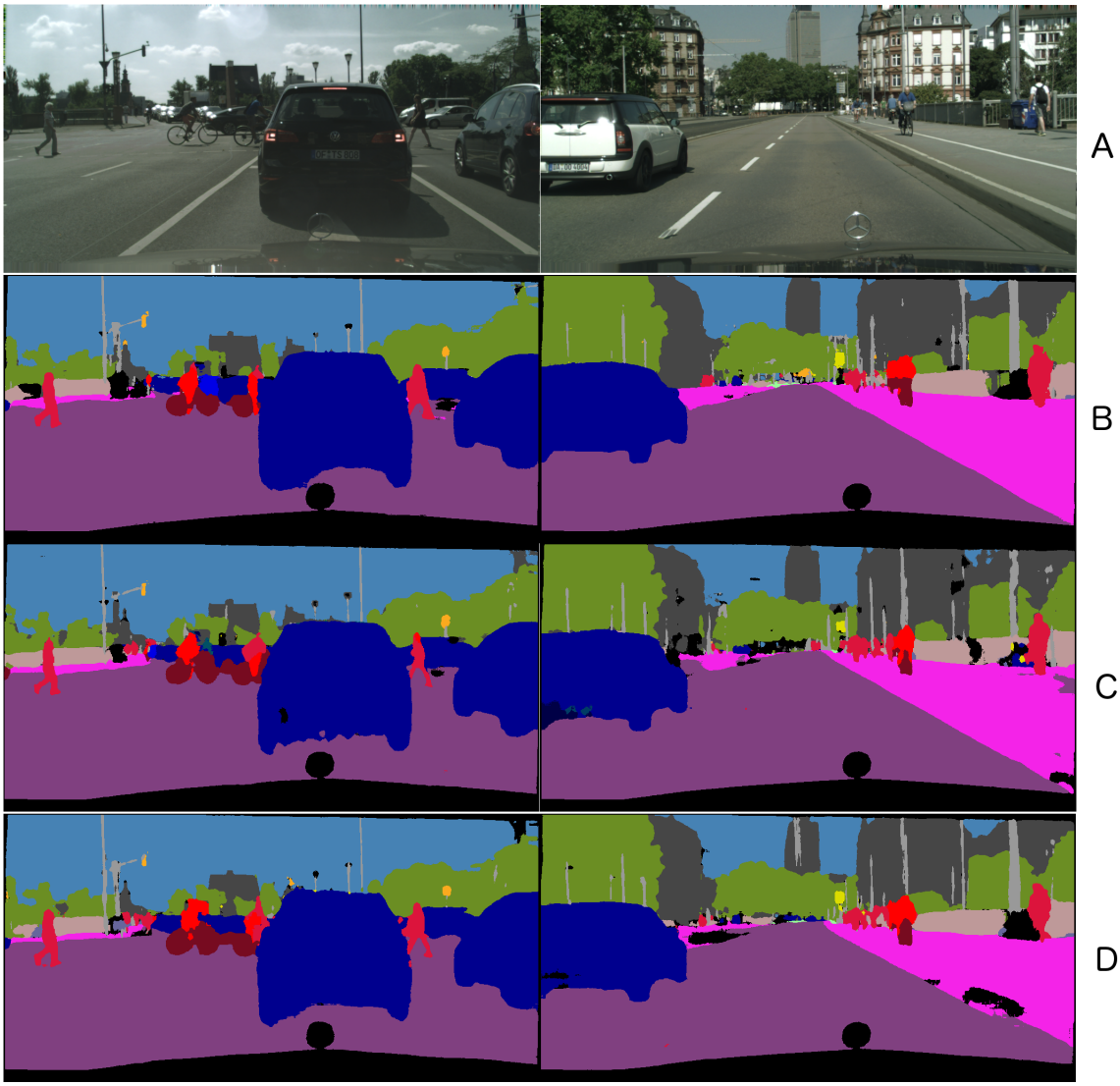

Рис. 11: Пример сегментации. (A) - Оригинальное изображение. (B) - ENet. (C) - UNet. (D) - SegNet

визуально лучше сегментирует Cityscapes и карта сегментации, полученная с помощью этого алгоритма выглядит более четкой и содержит меньше шумов и случайных пикселей, которые, например, можно заметить на результатах модели SegNet, полученной на тех же примерах. Худшие результаты SegNet можно объяснить тем, что его архитектуру из-за большого количества параметров сложнее обучать, а также потому, что SegNet архитектура была придумана раньше и по сути почти ничем не отличается от обычного сверточного автокодировщика.

Как и было указано ранее, Unet и Enet, напротив, имеют фундаментальные улучшения(Bottleneck архитектура и skip connections). Экспе-

риментально показано, что эти улучшения повышают производительность моделей. Unet отличается от Segnet наличием skip connections, типом архитектуры, при котором связи идут не только к латентному слою от предыдущего, но и насквозь сети, пропуская несколько уровней.

Что касаемо Enet, то за счет bottleneck архитектура получается очень глубокой, но при этом содержащей, относительно других моделей, небольшое количество параметров, что позволяет не только улучшить качество классификации, но и запускать его на мобильных устройствах, что будет показано после проведения экспериментов и сравения моделей по скорости работы.

В таблице 1 представлены метрики на тестовом подмножестве (а также точность на тренировочном подмножестве). В таблице представлены метрика точности классификации, метрика среднего IOU (Intersection Over Unit, метрика более подробно была описана в главе 3), а также IOU на трех классах: человек, дорога и автомобиль (основные классы при классификации дорожной обстановки), остальные метрики были опущены, т.к. показывают примерно те же самые зависимости. Из полученных метрик так же можно судить, что модели близки по точности и принципам работы, так как основаны на схожих архитектурах, однако из трех моделей можно выделить ENet, который побил остальные модели практически на всех классах.

В таблице 2 указано время обучения и валидации в секундах на одном батче, содержащем 4 изображения. Батч - это подмножество объектов, которые обрабатываются нейронной сетью прежде чем подсчитать градиент и обновить параметры нейронной сети. Из таблицы видно, что модели схожи по скорости обучения, однако более слабые с точки зрения архитектуры модели обучаются быстрее. Однако, из-за особенностей ENet архитектуры и малого количества параметров скорость её валидации в 8-9 раз, а скорость тренировки в 2-3 раза быстрее, чем у других моделей. Засчет этого ENet действительно можно использовать для семантической сегментации в реальном времени на лету.

Таблица 1: Метрики моделей

Модель	**Точность на валидации**	**Средний IoU**	**IoU Человек**	**IoU Дорога**	**IoU Авто**
UNet	0.883	0.585	0.664	0.934	0.89
ENet	0.905	0.603	0.655	0.94	0.89
SegNet	0.881	0.575	0.648	0.925	0.881

Таблица 2: Время обучения и валидации (в секундах). В одном батче 4 изображения

Модель	**Батч тренировки**	**Батч валидации**
UNet	6.3	0.1
ENet	2.1	0.01
SegNet	4.7	0.08

Заключение

В заключение хотелось бы отметить, что тема, которой я проникся, оказалась крайне интересной и актуальной. Был проведён обзор и сравнительный анализ нескольких современных ”state of art” алгоритма, выявленны ключевые особенности, которые приводят один метод к успеху по сравнению с остальными. Экспериментально было показало, что Enet подходит для задачи сегментации дорожной обстановки лучше других моделей с которыми мы сравнивали, с точки зрения как показателей метрик, так и производительности по времени. И, в отличии от Unet и SegNet, может быть использовано для сегментации как на мобильном устройстве с небольшим количеством памяти, так и в системах реального времени, где важна скорость работы.

Список литературы

[1] Agresti Alan. Logistic regression. — Wiley Online Library, 2002.

[2] Badrinarayanan Vijay, Kendall Alex, Cipolla Roberto. Segnet: A deep convolutional encoder-decoder architecture for image segmentation // IEEE transactions on pattern analysis and machine intelligence. — 2017. — Vol. 39, no. 12. — P. 2481–2495.

[3] Brostow Gabriel J., Fauqueur Julien, Cipolla Roberto. Semantic Object Classes in Video: A High-Definition Ground Truth Database // Pattern Recognition Letters. — 2008. — Vol. xx, no. x. — P. xx–xx.

[4] The Cityscapes Dataset for Semantic Urban Scene Understanding / Marius Cordts, Mohamed Omran, Sebastian Ramos et al. // Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). — 2016.

[5] ENet: A Deep Neural Network Architecture for Real-Time Semantic Segmentation / Adam Paszke, Abhishek Chaurasia, Sangpil Kim, Eugenio Culurciello // CoRR. — 2016. — Vol. abs/1606.02147. — 1606.02147.

[6] Ioffe Sergey, Szegedy Christian. Batch normalization: Accelerating deep network training by reducing internal covariate shift // arXiv preprint arXiv:1502.03167. — 2015.

[7] Krizhevsky Alex, Sutskever Ilya, Hinton Geoffrey E. ImageNet Classification with Deep Convolutional Neural Networks // Advances in Neural Information Processing Systems 25 / Ed. by F. Pereira, C. J. C. Burges, L. Bottou, K. Q. Weinberger. — Curran Associates, Inc., 2012. — P. 1097–1105. — URL: `http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-netw` `pdf`.

[8] Krizhevsky Alex, Sutskever Ilya, Hinton Geoffrey E. Imagenet

classification with deep convolutional neural networks // Advances in neural information processing systems. — 2012. — P. 1097–1105.

[9] Liaw Andy, Wiener Matthew et al. Classification and regression by randomForest // R news. — 2002. — Vol. 2, no. 3. — P. 18–22.

[10] Nair Vinod, Hinton Geoffrey E. Rectified linear units improve restricted boltzmann machines // Proceedings of the 27th international conference on machine learning (ICML-10). — 2010. — P. 807–814.

[11] Ronneberger Olaf, Fischer Philipp, Brox Thomas. U-Net: Convolutional Networks for Biomedical Image Segmentation // CoRR. — 2015. — Vol. abs/1505.04597. — 1505.04597.

[12] Shotton Jamie, Johnson Matthew, Cipolla Roberto. Semantic texton forests for image categorization and segmentation // Computer vision and pattern recognition, 2008. CVPR 2008. IEEE Conference on / IEEE. — 2008. — P. 1–8.

[13] Support vector machines / Marti A. Hearst, Susan T Dumais, Edgar Osuna et al. // IEEE Intelligent Systems and their applications. — 1998. — Vol. 13, no. 4. — P. 18–28.

[14] Thoma Martin. A survey of semantic segmentation // arXiv preprint arXiv:1602.06541. — 2016.

[15] Vapnik Vladimir N. The Support Vector method // Artificial Neural Networks — ICANN'97 / Ed. by Wulfram Gerstner, Alain Germond, Martin Hasler, Jean-Daniel Nicoud. — Berlin, Heidelberg : Springer Berlin Heidelberg, 1997. — P. 261–271.