Анализ состязательных атак на системы сегментации изображений

Е. А. Воробьев

***Аннотация*—В статье рассматриваются методы сегментации изображений и проблемы, связанные с состязательными атаками на эти системы. Необходимо обеспечить безопасность таких систем, поскольку сегментация широко применяется в различных задачах компьютерного зрения и может быть слабым местом в критических применениях. Представлен обзор различных типов сегментации, включая сегментацию образов, семантическую и паноптическую сегментации. Рассматриваются популярные архитектуры моделей сегментации, такие как FCN, U-Net, YOLO, Segment Anything и другие. В статье проводится анализ состязательных атак на системы сегментации изображений, включая как цифровые, так и физические атаки. Особое внимание уделяется методам и алгоритмам создания состязательных примеров. Целью работы является привлечение внимания исследовательского сообщества к проблеме безопасности систем сегментации, разработка новых, современных и более устойчивых к состязательным атакам моделей сегментации.**

***Ключевые слова*—состязательные атаки, сегментация изображений, свёрточные нейронные сети, энкодер-декодер модели**

I. Введение

Сегментация изображений является ключевой задачей в области компьютерного зрения, заключающейся в разделении изображения на сегменты. Эта задача находит широкое применение в различных отраслях, включая медицину, автономные транспортные средства, агропромышленность и т.д. Развитие глубокого обучения и нейронных сетей существенно повысило точность и эффективность моделей сегментации, что сделало их неотъемлемой частью современных приложений компьютерного зрения.

Однако, несмотря на впечатляющие достижения в области сегментации изображений, безопасность этих моделей остаётся под вопросом. Состязательные атаки, направленные на введение небольших, но целенаправленных искажений в исходные данные, могут значительно ухудшить точность сегментации и производительность модели. Такие атаки могут быть проведены злоумышленниками для нарушения работы систем сегментации, что представляет собой серьёзную угрозу для приложений, где точность и надёжность являются критически важными.

В данной статье проводится всесторонний анализ уязвимости систем сегментации к состязательным атакам. Рассматриваются цифровые и физические атаки, а также методы их реализации.

Статья получена 30 августа 2024

Егор Александрович Воробьев, МГУ им. М.В. Ломоносова, (email: vorobyov.egor@gmail.com).

II. Сегментация изображений

Сегментация изображений – задача компьютерного зрения, представляющая собой алгоритм разбиения входного цифрового изображения на сегменты. Под сегментом понимается область (набор пикселей), которая содержит какой-нибудь отдельный объект изображения. Такое разбиение на сегменты упрощает изображение для дальнейшей обработки и анализа. Изображение может содержать неинформативные области, например, задний фон. Поэтому удобнее анализировать сегментированное изображение, чем всё изображение целиком. В общем и целом сегментация это ключ к понимаю сцены изображения.

Алгоритм сегментации объединяет в отдельные наборы пиксели, которые имеют одинаковые атрибуты. К таким атрибутам можно отнести цвет, глубину, контраст, яркость пикселя и т.д. Например, благодаря контрастности, между собой можно различить тёмные и светлые участки (например, небо и фон). Таким образом составляется сегментированная маска изображения, отражающая его содержимое.

A. Алгоритмы сегментации изображений

Семантическая сегментация (semantic segmentation) – алгоритм разбиения цифрового изображения на сегменты, который соотносит каждый пиксель изображения с некоторым семантическим классом. Все пиксели изображения, относящиеся к одному определенному классу представляются одинаковым цветом: например, машины – красным, здания – жёлтым и т.д. (см. рис. 1).

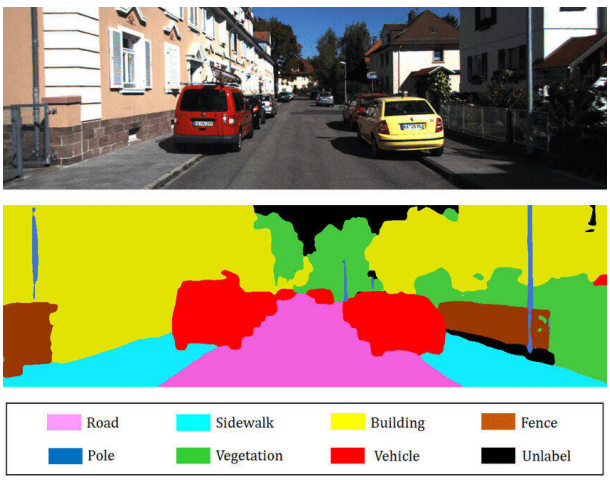


Рис. 1: Применение алгоритма семантической сегментации к цифровому изображению. Источник: https://towardsai.net/

Сегментация образов (instance segmentation) объединяет в себе алгоритм сегментации изображений и обнаружения объектов. В дальнейшем для обозначения сегментации образов будем использовать термин «инстанс сегментация». Целью алгоритма является сегментация изображения и выделение из него отдельных экземпляров каждого класса. В отличии от семантической сегментации, которая присваивает пикселям метки классов (например, все пиксели, принадлежащие к классу «человек», получают одну метку), алгоритм инстанс сегментации выделяет каждый объект данного класса отдельно. На сегментированном изображении объекты одинакового класса помечаются разными цветами (см. рис. 2).

Рис. 2: Алгоритм сегментации образов. Различные экземпляры одного объекта представлены разными цветами. Источник: [1]

Паноптическая сегментация (panoptic segmentation) – алгоритм сегментации, объединяющий в себе семантическую и инстанс сегментацию. Алгоритм позволяет получить сегментированное изображение с высоким уровнем детализации в результате чего достигается наиболее точное и полное понимание сцены. Каждому пикселю изображения на рисунке 3, сегментированного с помощью алгоритма паноптической сегментации, сопоставлена метка класса, а все экземпляры объектов сегментированы индивидуально.

Рис. 3: Паноптическая сегментация. Источник: [1]

B. 3D Point Cloud Segmentation

3D Point Cloud Segmentation [2] – это процесс разделения трёхмерного облака точек на значимые области и

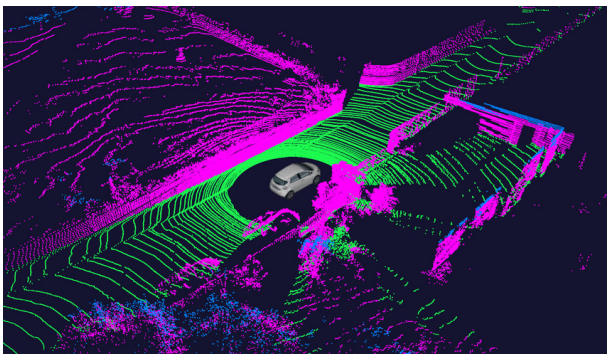


Рис. 4: Пример сегментации трёхмерного облака точек, полученного с помощью LiDAR. Источник: https://researchgate.net/

присвоения каждой точке определенной метки или класса (см. рис. 4). Этот алгоритм сегментации применяется в автономных системах, использующих LiDAR[1], таких как автономные транспортные средства, дроны, роботы и другие. Облако точек – это набор точек в трёхмерном пространстве, представляющие информацию о окружающей среде. Эти точки обычно получаются с помощью LiDAR или других 3D-сканирующих устройств.

C. Семантические классы

Разница между различными типами сегментации изображений заключается в том, каким образом они относят объекты к семантическим классам. Семантический класс – это класс, содержащий образы какого-нибудь объекта (например, машина, дерево, человек).

Семантические классы делятся на *things* и *stuff*. К первому относятся объекты, имеющие хорошо определенную форму (например, машина, человек). Ко второму относятся фоновые объекты (например, небо, трава). Большинство алгоритмов классификации и обнаружения сфокусированы на *things* классах, в то время как объектам класса *stuff* уделяется меньше внимания. Тем не менее объекты класса *stuff* играют важную роль, так как благодаря им мы можем получить важные аспекты изображения: тип сцены, контекст, физические атрибуты и геометрические свойства объектов [3].

Понимание контекста помогает распознать небольшие или необычные объекты. Например, можно определить, что металлический объект в небе – это, скорее всего, самолет, а металлический объект в воде – лодка [3]. Таким образом, чёткое разделение и применение семантических классов позволяет добиться детализированной сегментации изображений с сохранением его контекста и сцены.

D. Области применения

Сегментация изображений и компьютерное зрение стали важными инструментами во многих областях. Рассмотрим некоторые из областей применения алгоритмов сегментации.

Автономное вождение. Беспилотные автомобили имеют множество сенсоров, таких как камеры, радары и LiDAR, расположенных по периметру транспортного

[1]Light Detection and Ranging

средства. Для автономных транспортных средство важна корректность восприятия окружающей среды. Ключевую роль в восприятии играет LiDAR, который путём излучения света и замера времени его возвращения измеряет расстояние до объектов. LiDAR собирает собирает информацию об окружающей среде, извлекает из неё географические данные и предоставляет точное 3D представление окружающей среды. Таким образом, сенсоры автономного транспортного средства позволяют обнаруживать препятствия, разметку на дороге, дорожные знаки, свободные парковочные места и т.д. Затем к входным изображения применяется алгоритм сегментации, который классифицирует каждый пиксель изображения, планирует и корректирует поведение автомобиля. Обычно, в автономных системах применяется алгоритм сегментации трёхмерных облаков данных.

Распознавание медицинских снимков/изображений. Сегментация изображений также применяется в медицине и играет важную роль. Можно заблаговременно выявлять опухоли на медицинских снимках и моментально переходить к лечению. Например, благодаря тому, что семантическая сегментация даёт чёткое представление о форме объекта, можно определить является ли опухоль злокачественной или доброкачественно и начать заблаговременное лечение.

Сегментация спутниковых снимков. Применяется в картографии и управлении чрезвычайными ситуациями. Своевременная и точная геопространственная информация, получаемая с помощью спутникового дистанционного зондирования, играет ключевую роль в следующих применениях:

- Мониторинг и оценка последствий стихийных бедствий, таких как наводнения, землетрясения, цунами, оползни, сильные штормы и извержения вулканов [4]. На основе сегментированных спутниковых данных можно оперативно наносить на карту зоны бедствий.
- Отслеживание техногенных чрезвычайных ситуаций, включая промышленные аварии, пожары и их последствия для окружающей среды.
- Анализ антропогенного воздействия на ландшафты, такого как масштабное сельскохозяйственное освоение земель, вырубка лесов и незаконная добыча ресурсов. Например, правительство Перу использовало спутниковые снимки для обнаружения территорий, пострадавших от мелко- и крупномасштабного сельского хозяйства, вырубки лесов и добычи золота [5].
- Создание высокоточных карт местности.

Агропромышленность. Крупные агропромышленные предприятия активно используют современные технологии, в том числе – дроны, которые позволяют делать снимки полей и других сельскохозяйственных угодий с воздуха. С помощью сегментации фермеры подсчитывают и оценивают объёмы урожая сельхоз культур, выявляют сорняки и т.д.

III. Архитектура нейронных сетей

Глубокие нейроные сети – по истине мощный инструмент, активно используемый в компьютерном зрении для решения задач обнаружения, классификации, сегментации и т.д. В этой секции рассмотрим наиболее популярные и используемые нейросетевые архитектуры.

A. Свёрточные нейронные сети (CNNs)

Свёрточная нейронная сеть (Convolutional Neural Network, CNN) [6] – класс нейронных сетей, который специализируется на обработке изображений и видео. С помощью CNN решают задачу распознавания и классификации. Такие нейросети хорошо улавливают локальный контекст, когда информация в пространстве непрерывна, то есть её носители находятся рядом.

CNN состоит из нескольких слоёв (см. рис. 5). Основные элементы сети:

- Свёрточный слой (convolutional layer)
- Функция активации
- Пуллинг (pooling)
- Полносвязный слой (fully connected layer)

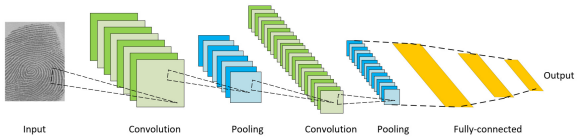


Рис. 5: Архитектура CNN. Свёртка и пулинг чередуются несколько раз, чтобы выделять всё более сложные признаки. Источник: [7]

1) Свёрточный слой: Ключевым компонентом CNN является свёрточный слой. Во время свёртки нейросеть удаляет лишние и оставляет информативные участки, которые помогут проанализировать изображение. Например, линии, края, текстуры и другие паттерны.

Для свёрточной нейронной сети изображение представляется в виде трёхмерного массива чисел (тензора) или массивов матриц. Такое представление изображения показано на рисунке 6.

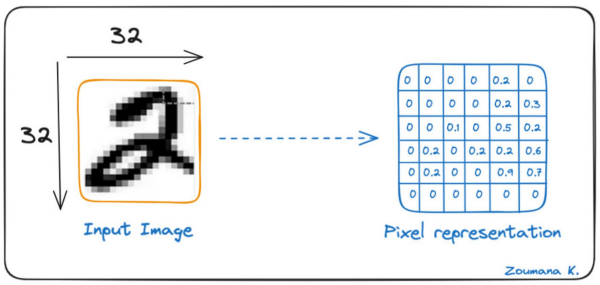


Рис. 6: Иллюстрация входного изображения и его пиксельного представления. Источник: [8]

Свёрточный слой применяет фильтры (ядра свёртки) к входному изображению или к выходным данным предыдущего слоя. Фильтр представляет собой небольшую матрицу весов, которая перемещается по изображению, выполняя поэлементное умножение и суммирование на каждом участке (см. рис. 7). Такое перемещение фильтра называется методом «скользящего окна» (sliding window). Таким образом каждый фильтр ищет на изображении определенные паттерны, такие как линии, кривые и т.д. При перемещении по изображению он создаёт

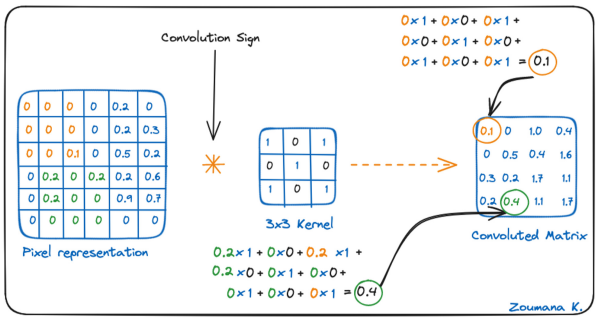


Рис. 7: Применение 3×3 фильтра к изображению с единичным шагом. Источник: [8]

новую сетку, которая подчеркивает места обнаружения этих паттернов.

Сеть может состоять из нескольких слоёв свёртки. Например, один фильтр может хорошо находить прямые линии, другой – кривые и так далее. Чем больше слоёв (фильтров) будет использоваться в CNN, тем мощнее будет ее архитектура, и, соответственно, тем сложнее паттерны она сможет обнаруживать.

В общем и целом, операцию свёртки можно описать следующим образом:

- Применить фильтр к изображения (начиная с верхнего левого угла до правого угла).
- Выполнить поэлементное умножение.
- Сложить произведения элементов.
- Результирующее значение соответствует первому значению (верхнему левому углу) в свёрнутой матрице.
- Переместить ядро вниз относительно размера скользящего окна (фильтра).
- Повторить шаги $1 - 5$ пока матрица изображения полностью не будет покрыта.

2) Функция активации: Функция активации применяется после каждой операции свёртки. В качестве таковой выбирается *ReLU* (Rectified Linear Unit). Она добавляет нелинейность, позволяя сети обучаться сложными зависимостями между признаками изображения.

ReLU описывается выражением 1, то есть функция активации возвращает исходное значение, если оно положительно, и 0 в противном случае.

$$f(x) = \max(0, x) \tag{1}$$

3) Пуллинг: На этом слое извлекаются наиболее важные и информативные признаки из свёрнутой матрицы, а несущественные удаляются. Это делается путём применения некоторой операции агрегирования, уменьшающей размерность матрицы. При этом уменьшается размерность изображения, а следовательно и объём памяти, используемый при обучении сети. Также пуллинг способствует уменьшению переобучения сети.

В качестве функции агрегирования обычно выбирается max pooling, которая извлекает из матрицы максимальное значение признака. На рисунке 8 показан пример использования операции пуллинга с фильтром 2×2 к изображению размером $4 \times 4 \times 1$. Фильтр перемещается по матрице со сдвигом равным размерности фильтра.

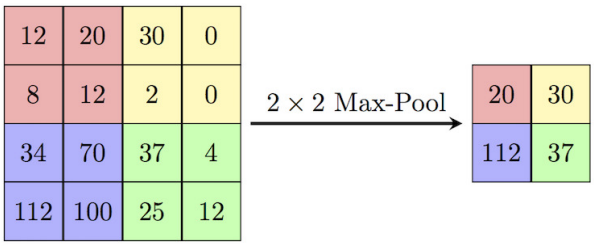


Рис. 8: Применение операция max pooling к изображению. Источник: computersciencewiki.org

4) Полносвязный слой: На последнем этапе полносвязный слой преобразует все полученные слои и связывает их между собой, образуя вектор. В случае задачи классификации или регрессии к полученному вектору применяется слой предсказания softmax, который используется для генерации значений вероятности для каждой из возможных выходных меток. Окончательная прогнозируемая метка имеет наибольшую оценку вероятности.

B. Encoder-Decoder модели

Энкодер-декодер [9, 10] – семейство моделей, которые учатся отображать точки данных из входной области в выходную область через двухэтапную сеть (см. рис. 9). Эта двухэтапная сеть состоит из двух модулей: энкодер и декодер. Энкодер переводит входные данные в более компактное представление, тем самым сжимая их. Обычно он представляет собой несколько свёрточных слоёв, которые постепенно понижают размер входных данных, извлекая значимые признаки. Декодер, в свою очередь, принимает сжатые данные и постепенно повышает размерность данных, тем самым восстанавливая исходные данные, или преобразует их в другой формат. Такие модели широко используются благодаря своей гибкости и способности эффективно обрабатывать сложные задачи преобразования и генерации данных.

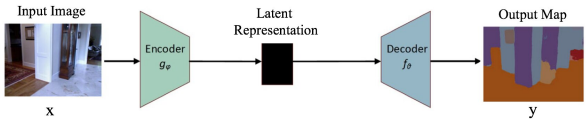


Рис. 9: Архитектура простой энкодер-декодер модели. Источник: [7]

IV. Датасеты

Для обучения и тестирования моделей компьютерного зрения, в том числе и сегментации, используются наборы 2D изображений (датасеты).

В некоторых случаях для обучения модели данных может быть недостаточно. Например, для обучения модели сегментации медицинских снимков данных может оказаться недостаточно, поскольку в основном датасеты медицинских снимков ограничены в количестве данных. В таком случае для повышения эффективности модели используют аугментацию данных. *Аугментация данных* – процесс искусственного увеличения объёма данных путём применения различных преобразований, таких как поворот, отражение, сдвиг, масштабирование, обрезка, изменение яркости и контрастности.

Таким образом, можно взять маленький датесет медицинских изображений, применить к нему аугментацию и обучить модель на новых данных. Тем самым эффективность такой модели будет выше, а переобучение – ниже. Для некоторых маленьких датасетов применение аугментации повысило производительность модели более чем на 20%.

A. Microsoft Common Objects in Context

Microsoft Common Object in Context (MS COCO) [11] – масштабный и широко используемый набор данных для задач компьютерного зрения, разработанный Microsoft. Он содержит более $330,000$ изображений с более чем $2,5$ миллионами аннотированных экземпляров объектов, относящихся к 80 категориям. Датасет предоставляет разнообразные аннотации, включая сегментацию объектов, ограничивающие рамки и текстовые описания на естественном языке. Особенно ценным для задач сегментации изображений является наличие детальных масок сегментации для каждого аннотированного объекта. Эти маски представляют собой попиксельную разметку, точно определяющую границы объектов на изображениях. Благодаря своему масштабу, разнообразию и качеству аннотаций, MS COCO стал стандартным эталоном для оценки и сравнения алгоритмов сегментации изображений в исследовательском сообществе.

B. Cityscapes

Cityscapes [12] создан для задач семантической и инстанс сегментации городских сцен. Он содержит $5,000$ изображений с детальными пиксельными аннотациями и более $20,000$ дополнительных слабосегментированных изображений. Датасет включает 30 классов объектов, охватывающих различные элементы городской инфраструктуры такие как дороги, здания, автомобили и пешеходы. Изображения городских сцен сняты в 50 разных городах Европы при различных погодных условиях и времени суток. Этот богатый и разнообразный датасет широко используется для разработки и оценки моделей, предназначенных для автономного вождения и анализа городской среды.

C. PASCAL Visual Object Classes

PASCAL Visual Object Classes (VOC) [13] один из популярных и широко используемых датасетов аннотированных изображений, используемый в задачах классификации и сегментации изображений, обнаружения и распознавания объектов. Датасет содержит 20 классов объектов, среди которых люди, животные, транспортные средства и бытовые предметы. Каждому изображения предоставляется точные аннотации: метки классов, ограничивающие прямоугольники и маски сегментации. Датасет PASCAL VOC состоит из 3 частей: $1,464$ изображения для обучения модели, $1,449$ изображений для тестирования и закрытый тестовый набор. Закрытый тестовый набор используется для оценки эффективности и точности предсказаний обученной модели.

D. PASCAL Context

PASCAL Context [14] – это расширение оригинального датасета PASCAL VOC. Он содержит более 400 классов (включая исходные 20 классов и сегментированные фоны оригинального датасета PASCAL VOC). Большинство классов объектов содержат небольшое количество изображений, поэтому для обучения модели обычно выбирается подмножество классов из 59 наиболее распространённых объектов. Для обучения и валидации используется $10,103$ изображения, а для тестирования $9,637$ изображений.

E. ADE20K

ADE20K [15] содержит более чем 27 тысяч изображений ($25,574$ для обучения и $2,000$ для валидации). Изображения полностью аннотированы объектами, охватывающими более 3000 категорий. Многие изображения также содержат части объектов и части частей. ADE20K используется в исследованиях для обучения и оценки моделей задач компьютерного зрения, в том числе и для задач сегментации изображений.

F. Segment Anything 1 Billion

Segment Anything 1 Billion (SA-1B) [16] – абсолютно новый датасет, представленный и разработанный Meta's FAIR[2] специально для их передовой и современной модели Segment Anything (SAM). На данный момент это самый крупный и масштабный датасет, состоящий из 11 разнообразных изображений высокого разрешения и $1,1$ миллиарда высококачественных масок сегментации. По сравнению с MS COCO, датасет SA-1B содержит приблизительно в 32 раза больше изображений и в 400 раз больше масок сегментации.

V. Архитектура моделей сегментации

В этой секции проведём обзор популярных моделей, решающих задачу как семантической, так и инстанс сегментации. Вкратце разберём архитектуру каждой модели и рассмотрим их конструктивные особенности.

A. Fully Convolutional Network

Один из подходов к решению задачи семантической сегментации был предложен в 2014 году. Его идея заключалась в использовании полностью свёрточных сетей [17] (Fully Convolutional Network, FCN) и обработке изображения по принципу энкодер/декодер (см. рис. 10). Энкодер понижает разрешение (downsampling) входного изображения, используя пошаговую свёртку, давая сжатое изображение. Декодер повышает разрешение изображения (upsampling), используя метод транспонированной свёртки для получения сегментированного вывода [18].

Так как модуль энкодера снижает разрешение входных данных в 32 раза, декодеру сложно создать детализированную маску сегментации и на выходе получается нечёткое сегментированное изображение с «размазанными» границами. Для решения этой проблемы в нейросетях применяется метод добавления «skip connections», которые представляют собой данные из предыдущих

[2]Fundamental AI Research

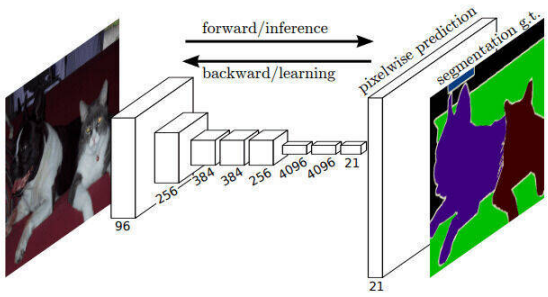


Рис. 10: FCN для задачи семантической сегментации изображений. Источник: [17]

слоёв. Таким образом, постепенно повышая разрешения изображения к нему будет добавляться и суммироваться информация (признаки) из более ранних слоёв. Этот процесс показан на рисунке 11.

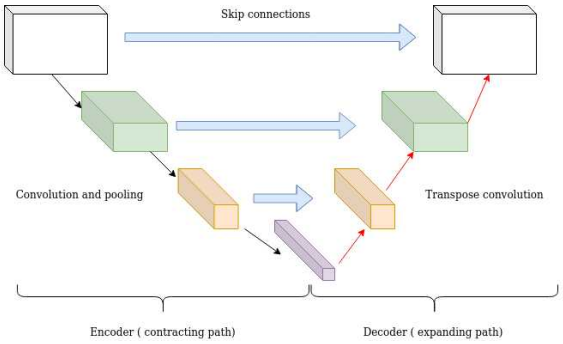


Рис. 11: Схема повышения дискретизации изображения с использованием «skip connections», позволяющая повысить детализацию сегментации. Источник: https://theaisummer.com/skip-connections/

B. U-Net: сегментация биомедицинских снимков

U-Net [19] – свёрточная нейронная сеть, используемая для решения задачи семантической сегментации биомедицинских изображений. На рисунке 12 показана архитектура сети U-Net, которая строится по принципу энкодер-декодер. Таким образом сеть состоит из двух основных частей: сжимающего (слева) и расширяющего (справа) путей.

Сжимающий путь (энкодер, понижение разрешения изображения) состоит из 4 повторяющихся блоков. Каждый блок состоит из двух свёрточных слоёв с 3×3 ядрами свёртки (фильтрами), сопровождающийся функцией активации ReLU для введения нелинейности. За каждым блоком следует операция 2×2 макс пуллинга (max pooling) с шагом 2 для понижения размерности изображения. На этапе макс пуллинга количество каналов признаков удваивается.

Расширяющий путь (декодер, повышение разрешения изображения) также состоит из 4 блоков (два 3×3 свёрточных слоя и ReLU), только каждому блоку предшествует операция повышения размерности изображения (декодер). В качестве декодера используется операция 2×2 свёртки вверх (up convolution), которая уменьшает количество каналов признаков вдвое. Затем следует операция конкатенации: к каждому уровню пути расширения добавляется соответствующий уровень из пути сжатия. Это позволяет сети восстанавливать пространственную информацию, которая может быть утеряна из-за обрезки краевых пикселей в каждой свёртке.

На последнем слое (выходной слой) используется 1×1 свёртка для сопоставления каждого 64-компонентного вектора признаков с желаемым количеством классов. Всего сеть имеет 23 свёрточных слоя.

На рисунке 12 также видно, что пути сжатия и расширения связаны между собой двумя 3×3 свёртками. На этом этапе слои имеют наименьшую размерность и наибольшее количество каналов признаков, что делает это узким место сети. Это место является так называемым горлышком (bottleneck).

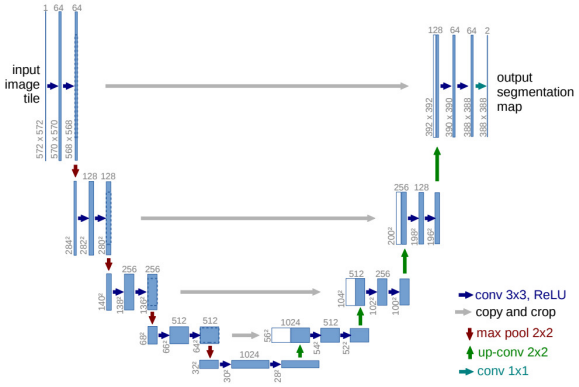


Рис. 12: Архитектура сети U-Net. (слева) – сокращение, (справа) – расширение. Источник: [19]

Сеть U-Net обладает высокой точностью сегментации, что делает её мощным и эффективным инструментом сегментации биомедицинских изображений. С помощью U-Net сегментируют медицинские снимки, например, мозга, заблаговременно выявляют опухоли и начинают лечение.

C. SegNet

SegNet [10] – свёрточная нейросеть семантической сегментации изображений, использующая в своей архитектуре принцип экнодер-декодер модели. На рисунке 13 показана архитектура сети. Она представляет собой энкодер, состоящий из 13 свёрточных слоёв, идентичных первым слоям сети VGG16 [20]. За свёрточными слоями следует операция макс пуллинга для уменьшения размерности изображения и выделения важных признаков. Ключевым отличием архитектуры сети SegNet является то, что на каждом этапе пуллинга в энкодере сохраняются индексы максимальных значений. В последующем эти индексы передаются в слой повышения разрешения декодера, что позволяет восстановить исходное пространственное разрешение изображения. На последнем этапе слой предсказания softmax вычисляет вероятности принадлежности каждого пикселя к одному из предопределенных классов. Такой подход позволяет эффективно использовать память, поскольку не нужно хранить промежуточные представления.

D. You Only Look Once

You Only Look Once (YOLO) [21] – популярная модель обнаружения объектов и инстанс сегментации изображений в режиме реального времени, разработанная

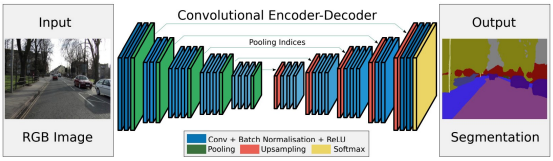


Рис. 13: Архитектура SegNet. Источник: [10]

в Университете Вашингтона. Появившись в 2016 году, YOLO быстро завоевала популярность благодаря своей высокой скорости и точности. За время существования архитектура модели претерпела множество модификаций и улучшений, которые значительно повысили производительность модели. Мы будем рассматривать последнюю модель YOLOv8 от Ultralytics [22].

Передовая и современная (state-of-the-art, SOTA) модель Ultralytics YOLOv8 [22] опирается на успех своих предшественников. Однако новая модель предлагает несколько ключевых нововведений, которые значительно расширяют возможности, повышают производительность, гибкость и эффективность модели. Модель поддерживает полный спектр задач компьютерного зрения, включая обнаружение, сегментацию, оценку позы, отслеживание и классификацию. YOLOv8 является одной из самых быстрых, точных и универсальных моделей, что позволяет пользователям использовать её в различных областях и приложениях.

Кроме того, универсальность модели подчёркивается тем, что имеется несколько вариаций модели, таких как, например, YOLOv8-tiny и YOLOv8x, которые различаются в размере и вычислительной сложности. Это позволяет пользователю выбрать модель, подходящую под его задачу и системные требования. YOLOv8 обучена на датасете MS-COCO, но при необходимости её можно обучить на любом другом датасете.

Традиционные модели, такие как, например, рассмотренные выше, используют подход «скользящего окна» когда фильтр перемещается по изображению и извлекает из него определённые паттерны. Такой подход является вычислительно очень трудоёмким и медленным, поэтому с помощью него нельзя добиться быстроты работы модели. YOLO произвёл революцию в этой области, рассматривая обнаружение объектов как единую задачу регрессии. Вместо скользящих окон YOLO прогнозирует ограничивающие рамки и вероятности классов для объектов непосредственно из входного изображения за один прямой проход, что делает её значительно быстрее.

Архитектура модели YOLOv8 состоит из трёх основных компонентов: *backbone*, *neck* и *head*.

1) Backbone: Представляет собой свёрточную нейронную сеть, которая отвечает за извлечение признаков из входного изображения. В основе архитектуры YOLOv8 лежит модифицированная версия сети Darknet – сеть CSPDarknet53. Эта модификация включает в себя межэтапные частичные соединения, повышающие способность к обучению и эффективность модели.

2) Neck: Соединяет между собой backbone и head, а также отвечает за слияние признаков, полученных на разных этапах backbone. YOLOv8 использует сеть *Path Aggregation Network* (PANet), которая облегчает поток информации в различных пространственных разрешениях, позволяя модели эффективно захватывать масштабные объекты.

3) Head: Этот компонент отвечает за предсказание на основе признаков, извлеченных на предыдущих двух этапах. Head предсказывает координаты, ограничивающие прямоугольники, objectness score и вероятности классов для каждого поля привязки (anchor box), связанного с ячейкой сетки. В архитектуре используются поля привязки для эффективного предсказания объектов различных форм и размеров.

E. You Only Look At CoefficienTs

You Only Look At CoefficienTs (YOLACT) [23] еще одна модель инстанс сегментации объектов в реальном времени, представленная в 2019 году. На рисунке 14 представлена подробная схема архитектуры модели. YOLACT разделяет сложную задачу сегментации объектов на две более простые задачи и выполняет их параллельно:

- Генерация набора масок прототипов
- Предсказание коэффициентов масок для каждого экземпляра

На первом этапе (первая ветвь) модель использует FCN [17], которая создаёт на основе изображения набор масок прототипов. Вторая ветвь реализует дополнительный модуль обнаружения объектов для прогнозирования вектора коэффициентов масок для каждого якоря (anchor), который кодирует представление экземпляра в пространстве прототипа. Наконец, для каждого экземпляра, прошедшего через NMS[3], создаётся маска путём линейного объединения работы этих двух ветвей.

1) Генерация прототипов: Ветвь генерации прототипов (protonet) предсказывает набор из k масок прототипов для всего изображения. Protonet представляет из себя FCN, последний слой которой имеет k каналов (см. рис. 15) и соединён с признаками основного слоя сети (feature backbone). Каждому каналу слоя соответствует своя маска прототипа.

Использование Feature Pyramid Network (FPN) [26] обосновывается тем, что более глубокие слои признаков (в нашем случае $P3$, см рис. 14) позволяют получить более устойчивые маски. Кроме того, прототипы высокого разрешения порождают маски высокого качества и показывают хорошую производительность на мелких объектах. Затем, для повышения производительности архитектуры на мелких объектах, разрешение слоя уменьшается в 4 раза. На итоговой стадии генерации прототипов применяется ReLU для введения нелинейности в модель.

2) Коэффициенты масок: Для предсказания коэффициентов масок используется модуль, состоящий из трёх параллельных ветвей: первая предсказывает c вероятностей классов, вторая предсказывает 4 регрессора ограничивающих рамок и, наконец, третья предсказывает k коэффициентов масок. На выходе для каждого якоря получается $4 + c + k$ коэффициентов.

[3]**Non-Maximum Suppression** (NMS) – метод постобработки, используемый в задачах обнаружения объектов для устранения повторяющихся обнаружений и выбора наиболее релевантных обнаруженных объектов. Алгоритм выбирает один объект из множества пересекающихся между собой объектов и позволяет уменьшить количество ложных срабатываний и вычислительную сложность алгоритма обнаружения.

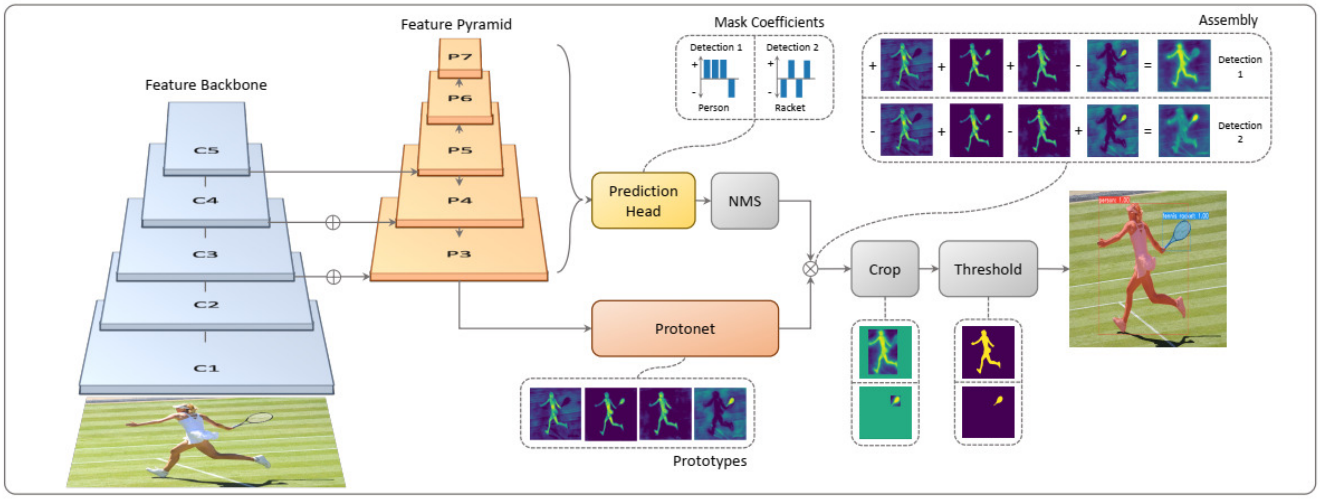


Рис. 14: Архитектура YOLACT. Основана на модифицированной RetinaNet [24] с использованием ResNet-101[25] и FPN [26]. Источник: [23]

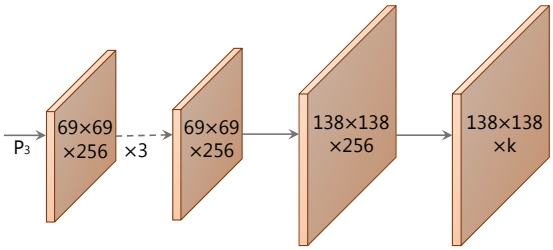


Рис. 15: Архитектура Protonet. Источник: [23]

Что касается нелинейности, то из финальных масок вычитаются прототипы, поэтому к k коэффициентам масок применяется функция активации tanh, что даёт более стабильные выходные данные без нелинейности.

3) Сборка масок: Финальная маска M для каждого объекта формируется путём линейной комбинации прототипов с использованием предсказанных коэффициентов масок. Для получения финальной маски применяется сигмоид (sigmoid):

$$M = \sigma(PC^T), \tag{2}$$

где P – матрица прототипа маски размера $h \times w \times k$, C – $n \times k$ матрица коэффициентов масок для n образов, прошедших этап NMS и оценку погрешности.

4) Backbone Detector: В качестве основны (backbone feature) для извлечения признаков взята RetinaNet [24] с некоторыми модификациями для ускорения. Используется сеть ResNet-101 [25] в сочетании с FPN [26]. Сеть обрабатывает изображения размером 550×550 пикселей. Подобно RetinaNet, FPN модифицируется таким образом, что больше не создаёт слой $P2$. Вместо этого FPN создает $P6$ и $P7$ в виде последовательных 3×3 свёрточных слоёв с шагом 2, начиная с $P5$ (не $C5$). На каждом таком слое размещается три якоря с соотношениями сторон $[1, 1/2, 2]$ на каждом. На слое $P3$ якорь имеет площадь 24 квадратных пикселя и каждый последующий слой имеет площадь в два раза больше предыдущего слоя. Таким образом получаем следующие масштабы слоёв от $P3$ до $P7$: $[24, 48, 96, 192, 384]$. Для prediction head, соединённой с каждым слоем P_i, используется одна 3×3 свёртка, общая для всех трёх ветвей, а затем каждая ветвь получает свою собственную 3×3 свёртку параллельно.

F. Segment Anything Model

Segment Anything Model (SAM) [16] – инновационная модель сегментации изображений, совсем недавно представленная компанией Meta's FAIR, но которая уже произвела революцию в области компьютерного зрения. SAM представляет абсолютно новый подход к анализу изображений, а в основе модели лежит хорошо продуманная архитектура. На рисунке 16 представлена иллюстрация пайплайна, состоящего из трёх компонентов: task, model и data. Все эти компоненты связаны между собой и в совокупности позволяют SAM выполнять сегментацию изображений в реальном времени с беспрецедентной гибкостью и точностью.

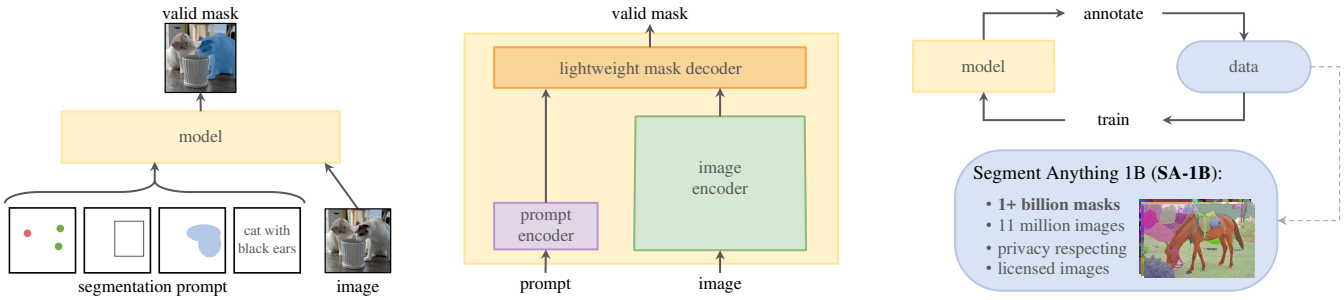
Task: Компонент, определяющий набор промптов, посредством которых пользователь может взаимодействовать с задачей сегментации. Учитывается множество реальных сценариев. Например, пользователь может загрузить в модель изображение и с помощью определенных промптов (выделение, текст и т.д.) определить область, в которой нужно произвести сегментацию.

Model: Использует энкодер изображений, энкодер промптов и облегчённый декодер для быстрого и точного создания масок сегментации.

Data: Представляет из себя движок сбора изображений для сегментации и, соответственно, сам датасет SA-1B, содержащий более 1 миллиарда масок сегментации, для обучения Segment Anything без обширного переобучения.

Мы рассмотрим только вторую компоненту, которая касается именно архитектуры модели сегментации, поскольку она представляет для нас наибольший интерес. Описание Task, определяющей взаимодействия между пользователем и моделью, выходят за рамки данной статьи. Датасет SA-1B, применяемый для обучения модели, частично был описан в секции IV.

На рисунке 17 показана архитектура модели Segment Anything, состоящая из трёх модулей, каждый из которых использует некоторые конструктивные решения:



(a) **Task**: promptable segmentation (b) **Model**: Segment Anything Model (**SAM**) (c) **Data**: data engine (top) & dataset (bottom)

Рис. 16: Основные компоненты пайплайна SAM: task, model и data. Источник: [16]

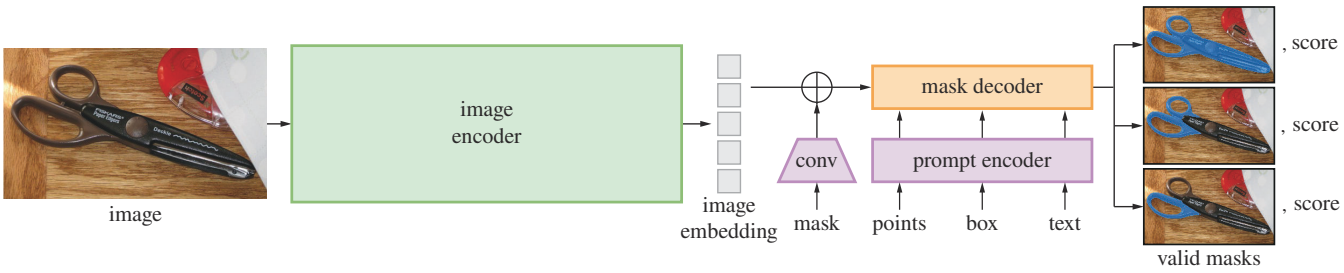


Рис. 17: Архитектура модели Segment Anything. Источник: [16]

- **Энкодер изображений:** Производит однократное встраивание (one-time embedding) входного изображения и извлекает его основные признаки, которые служат основой для последующей сегментации.
- **Энкодер промптов:** Модель использует гибкий энкодер промптов, который в режиме реального времени преобразует пользовательские промпты во встраиваемые векторы (embedding vectors). Основной целью энкодера является преобразование пользовательских промптов в интерпретируемый моделью формат.
- **Декодер масок:** Отвечает за предсказание масок сегментации для каждого объекта. Декодер объединяет информацию из встроенного изображения (image embedding) и встроенного промпта (prompt embedding) для создания точных масок, которые идентифицируют объект или область, заданную пользователем.

1) Энкодер изображений: Благодаря своей масштабируемости и мощным методам предварительного обучения, на первом этапе SAM использует минимально адаптированный для обработки входных данных высокого разрешения Vision Transformer (ViT) [27], предварительно обученный с помощью Masked Autoencoder (MAE) [28]. Энкодер запускается один раз для каждого изображения и может применяться до запроса модели.

2) Энкодер промптов: Предлагается два вида промптов, используемых SAM: разреженные (sparse) и плотные (dense). К разреженным промптам относятся точки, прямоугольники и текст, к плотным – маски. Точки и прямоугольники представлены позиционными кодировками, которые дополнены обученными внедрениями (learned embeddings) для каждого типа промптов. Текстовые промпты произвольной формы представляются с помощью готового текстового кодировщика из CLIP [29]. Плотные промпты, такие как маски, включаются в свёртки и суммируются поэлементно с встроенным изображением.

3) Декодер масок: Облегчённый декодер масок предсказывает маски сегментации на основе встраиваний из энкодера изображений и из энкодера промптов. Декодер эффективно отображает изображение и предлагает встраивания для создания масок сегментации. Он сопоставляет встроенное изображение, встроенные промпты и выходной токен с маской. Декодер масок использует модифицированный Transformer decoder block [30], сопровождающийся dynamic mask prediction head.

VI. Состязательные атаки на сегментацию

Эта секция представляет собой общий обзор состязательных атак на системы сегментации изображений. Рассмотрим таксономию состязательных атак, их оценку и методы повышения эффективности состязательных атак.

Состязательная атака – это процесс добавления к изображению состязательного возмущения, способного обмануть модель машинного обучения. В качестве состязательного возмущения можно взять, например, специально сгенерированный цифровой шум. Входное изображение вместе с состязательным возмущением называется состязательным примером. В случае сегментации целью злоумышленника является создание такого состязательного примера, который обманывал бы модель и приводил к неправильным маскам сегментации.

Базовым примером добавления состязательного возмущения в виде цифрового шума является картинка с пандой (см. рис. 18). Такой состязательный пример нарушает работу модели и заставляет её классифицировать панду как гиббона.

A. Таксономия состязательных атак

Для описания и классификации состязательных атак на модели машинного обучения выделяют следующие признаки:

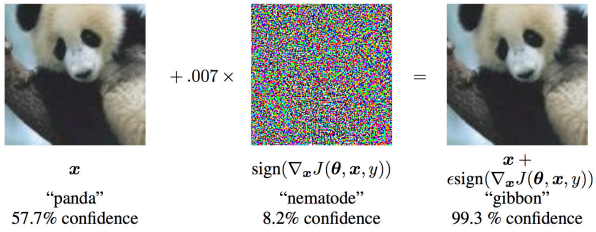


Рис. 18: Пример состязательной атаки с помощью цифрового шума. Слева – исходное изображение, по центру – специально подготовленный цифровой шум, справа – состязательный пример. Источник: https://pytorch.org/

- Знания атакующего
- Цель атаки
- Место применения атаки

При помощи описательного анализа атаки можно сделать предположение о том, насколько она будет эффективной и целесообразной.

1) Знания об атакуемой системе: Для описания состязательной атаки важно понимать какие знания были у злоумышленника об атакуемой системе. Таким образом, атаки можно разделить на три основные категории:

Атаки «белого ящика»: Атаки методом белого ящика представляют собой состязательные атаки, при которых злоумышленник обладает полными знаниями об атакуемой системе: обучающие и тестовые данные, архитектура и параметры модели. На основе имеющихся данных атакующий может создать копию целевой модели и тестировать на ней состязательные примеры.

Атаки «чёрного ящика»: Состязательные атаки, при которых злоумышленник не имеет представления об архитектуре и параметрах атакуемой модели, называются атаками чёрного ящика. В данном случае атакующий полагается на информацию, которую он может получить из выходных данных модели посредством отправки в неё запроса. Он посылает запросы в атакуемую модель и наблюдает за её ответами. Злоумышленник стремится построить модель, которая замещала бы модель-жертву. Используя замещающую модель, атакующий может затем создать состязательные примеры для атаки на классификатор модели-жертвы.

Атаки «серого ящика»: Сочетают в себе атаки белого и черного ящика. В этом случае атакующий имеет частичное представление о модели-жертве. Например, атакующий может иметь информацию об архитектуре модели или информацию о данных, на которых модель была обучена, но не иметь доступа к самим данным.

2) Цель атаки: В зависимости от целей атакующего можно выделить целевые и нецелевые атаки.

Целевые атаки: Целью атакующего является нарушение работы классификатора модели, при которой классификатор относит специально подготовленное входное изображение к заранее определенному целевому классу. Таким образом, атакующий может намеренно изменять поведение модели и относить, например, знак остановки к знаку ограничения скорости. Сказанное можно выразить через уравнение 3, где t – целевой класс [31].

$$f(x + \delta) = t \tag{3}$$

Нецелевые атаки: Нацелены на то, чтобы внести небольшие изменения в исходное изображение и нарушить работу классификатора модели. Цель таких атак состоит в том, чтобы просто нарушить правильную классификацию изображения, см. уравнение 4 [31].

$$f(x + \delta) \neq f(x) \tag{4}$$

3) Место применения атаки: В контексте ML пайплайна атаки могут быть разделены на два типа: атаки отравлением и backdoor атаки.

Атаки отравлением: Представляют собой атаки на этапе обучения и настройки модели. Они подразумевают изменение обучающего набора данных путём добавления в него вредоносных сэмплов. Целью таких атак является нарушение целостности модели и снижение её общей производительности на этапе обучения.

Backdoor атаки: На этапе обучения в ML модель внедряется что-то наподобие «чёрного входа» (backdoor). Он может представлять из себя, например, какие-нибудь триггеры или паттерны. Затем злоумышленник подготавливает специальные входные данные, посредством которых воздействует на эти триггеры и открывают доступ к недоступной другим пользователям функциональности модели. Таким образом, проводя атаку такого типа злоумышленник может изменять поведение модели.

Подобного рода атаки на модели могут привести к серьезным проблемам. В случае атак, которые применяются этапе ML пайплайна, нужно уделить особое внимание данным, на которых тренируется и обучается модель. Необходимо тщательно изучить данные на предмет наличия в них вредоносных сэмплов.

B. Оценка атак

При оценке атак на модели опираются на четыре ключевых аспекта, причём все они взаимосвязаны.

1) *Эффективность:* Атакующий должен продумать план атаки, чтобы извлечь из неё максимальную эффективность и нанести как можно больший вред модели.
2) *Устойчивость:* Устойчивость атаки является одним из ключевых факторов при оценке физических атак. Физические атаки проводятся в реальном мире, который постоянно меняется. Поэтому атакующий должен реализовать такое состязательное возмущение, которое будет устойчиво к переменам окружающей среды таким, как освещение, фон, погодные условия и т. д. Из устойчивости атаки следует её эффективность.
3) *Скрытность:* Не маловажную роль в эффективность атак играет скрытность. Атакующий должен позаботиться о том, чтобы состязательное возмущение было незаметным как для обычного человека, так и для жертвы. Способность сохранять скрытность гарантирует, что злоумышленник сможет осуществить атаку, не вызывая подозрений и не активируя какие-либо защитные механизмы модели.
4) *Переносимость (transferability) атак:* Способность состязательного возмущения воздействовать и нарушать работу не одной конкретной модели, а множества моделей с одинаковым эффектом.

C. Методы повышения устойчивости физических атак

Поскольку физические атаки проводятся в реальном мире, который вовсе не является постоянным, то встаёт вопрос о повышении устойчивости физических состязательных атак к изменениям окружающей среды.

Пусть, например, мы проводим атаку на модель распознавания дорожных знаков для автономного вождения. Допустим, что злоумышленник разместил состязательных дорожный знак где-то на стене или на столбе в реальном мире. В таком случае предсказание модели будет зависеть от того на каком расстоянии или под каким углом сенсоры обнаружили этот состязательный знак.

Transformation	Parameters	Remark
Rotation	$\pm 20°$	Camera Simulation
Cropping	$-0.7 \sim 1.0$	Photograph/Occlude Simulation
Affine	0.7	Perspective/Deformed Transforms
Scale	$[0.25, 1.25]$	Distance/Resize
Random Noise	± 0.1	Noise
Brightness	± 0.1	Illumination
Contrast	$[0.8, 1.2]$	Camera Parameters

Таблица I: Примеры трансформаций [31].

Для повышения надежности физических состязательных атак на системы компьютерного зрения используются следующие техники:

1) Expectation over Transformation (EOT): Используется для повышения устойчивости моделей к физическим атакам. Улучшение устойчивости (робастности) модели достигается имитацией потенциальных преобразований в реальном мире во время оптимизации атаки. В таблице I представлены основные преобразования. Проще говоря, на каждой итерации оптимизационного процесса к изображению добавляются рандомные искажения. EOT позволяет создавать изображения, устойчивые к подобным искажениям.

2) Spatial Transformer Layer (STL): Представляет собой компонент нейронных сетей, который динамически применяет пространственные преобразования (вращение, масштабирование, трансляция) к входному изображению или возмущению для обмана модели. Это позволяет управляемым образом модифицировать положение, ориентацию или масштаб возмущения, увеличивая шансы нарушить работу модели.

3) Total Variation Norm (TV Loss): Естественные изображения обычно имеют гладкие и последовательные участки с равномерными цветовыми переходами. Резкие перепады между соседними пикселями в возмущении могут быть плохо распознаны камерами из-за шумов оцифровки. TV Loss – дополнительный штраф в функции потерь, который нацелен на поддержание гладкости возмущения. Он вычисляется как сумма модулей разностей между соседними пикселями и минимизируется во время оптимизации возмущения. Для возмущения P он определяется как

$$L_{tv} = \sum_{i,j} \sqrt{(P_{i+1,j} - P_{i,j})^2 + (P_i, j+1) - P_{i,j}^2}$$

где i, j – координаты пикселя патча P.

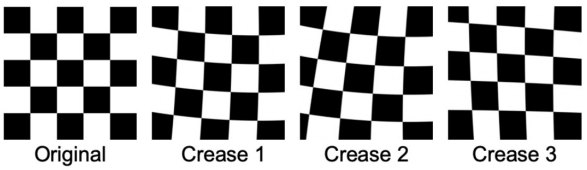
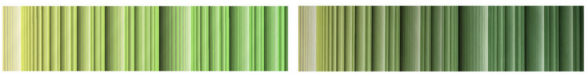


Рис. 19: Пример создания рандомных складок. Источник: [31]



(a) Digital Image (b) Printed Image

Рис. 20: Различие между цифровым (a) и распечатанным (b) изображением. Источник:[31]

4) Creases Transformation: Данный метод используется для повышения эффективности физических атак при помощи состязательной одежды. При движении человека одежда склона к появлению складок и, следовательно, состязательный патч на ней тоже искажается. Такие искажения состязательного патча могут исказить атаку и снизить её эффективность. Метод Creases Transformation моделирует эти складки (см. рис. 19) и добавляет их к состязательному патчу, тем самым повышая устойчивость атаки.

5) Non-Printability Score (NPS): Физические атаки зачастую представляют собой генерацию состязательного патча в цифровой среде и последующее его размещение в физическом пространстве. Во время печати патча цвета могут исказиться из-за ограниченных возможностей принтера. Пример искажения показан на рисунке 20. Искажения цветов патча могут существенно повлиять на качество атаки, сделав её малоэффективной или вовсе бесполезной. NPS – метрика для измерения расстояния между цветами оптимизированного возмущения и цветами, которые может воспроизвести типичный принтер. NPS минимизируется в процессе оптимизации, чтобы возмущение оставалось эффективным после печати. NPS определяется формулой

$$L_{nps} = \sum_{c_{perturb} \in p} ||c_{perturb} - c_{print}||$$

6) Randomly Transformed Patch: Для создания патчей, устойчивых к изменениям освещения и ракурса, применяется композиция следующих случайных преобразований: изменение яркости, контраста, вращение, сдвиг, наклон. Эти преобразования имитируют вариации, которые могут произойти в реальных условиях.

Таким образом, все эти техники направлены на повышение реалистичности, гладкости и устойчивость состязательных возмущений к различным преобразованиям и искажениям, характерным для физических атак.

D. Формы состязательных возмущений

Существует множество различных способов атаковать и обмануть модель сегментации. Выбор формы состязательного возмущения ограничивается лишь фантазией атакующего (см. рис. 21). В контексте задач классификации, обнаружения объектов и дорожных знаков, а

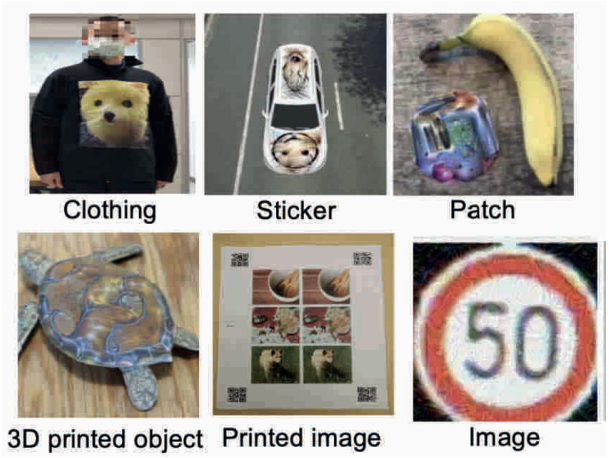


Рис. 21: Демонстрация различных форм физических состязательных атак. Источник: [31]

также сегментации можно выделить следующие формы состязательных возмущений:

Патчи – аккуратно сгенерированные возмущения, которые наносятся на физические поверхности, попадающие в поле зрение сенсоров модели. Правильно созданные и размещённые патчи приводят к нарушению работы целевой модели. Такие патчи не только приводят к неверным предсказаниям модели и нарушению её работы, но и в разы ухудшают её производительность.

Стикеры, наклейки – возмущения, которые размещаются (клеятся) на объектах. Такие возмущения могут содержать в себе специальные символы или паттерны, которые могут обмануть модель. В основном такие стикеры наносят на целевой объект модели, например, на автомобили для атаки на автономное вождение. Такие атаки эффективнее патчей поскольку стикеры можно разместить на значительной площади. Помимо этого стикер можно замаскировать под условия окружающей среды, что сделает атаку более устойчивой и незаметной.

Одежда – один из способов атаки методом маскирования. Данная атака подразумевает, что одежда человека (штаны, футболка, куртка, плащ и т. д.) содержит состязательные рисунки, паттерны. В первую очередь атака с помощью состязательной одежды нацелена на обман моделей обнаружения людей. С помощью состязательной одежды человек может скрыться от систем видеонаблюдения и остаться незамеченным. Вообще говоря, атаки посредством состязательной одежды применимы в контексте задачи распознавания людей или задачи подсчёта людей. Однако, мы можем рассматривать эту атаку в контексте сегментации изображений, поскольку нет разницы будет состязательный патч размещен на одежде человека или на каком-либо объекте.

Физические состязательные принты и объекты представляют собой атаку посредством размещение в физическом пространстве распечатанных картинок или 3D объектов, повторяющих какой-то реальный объект. Такие принты или объекты могут распознаваться моделью как реальные и привести к нарушению её работы.

VII. Цифровые состязательные атаки на системы сегментации

С повышением популярности и важности систем сегментации изображений возрастает и интерес к их уязвимости к состязательным атакам. Сначала рассмотрим цифровые состязательные атаки. Такие атаки представляют собой искусственно созданные изменения в данных, которые приводят к ошибочным выводам модели, сохраняя при этом незаметность для человеческого глаза. Цифровые атаки могут серьезно нарушить работу модели, приводя к ошибкам и ложной сегментации объектов, что может иметь катастрофические последствия, особенно в критических применениях.

Рассмотрим основные методы и подходы к реализации цифровых состязательных атак на системы семантической сегментации.

A. Dense Adversary Generation

Dense Adversary Generation [32] – алгоритм создания состязательных примеров для атак на системы семантической сегментации изображений и обнаружения объектов. Основной задачей алгоритма является генерация визуально незаметных для человека возмущений, способных обмануть модель и привести к ложному сегментированному выводу. В отличии от задачи классификации изображений, где достаточно обмануть классификатор на предсказывании одной цели, в случае сегментации или детекции требуется воздействовать на огромное множество отдельных целей.

Алгоритм генерации применим к системам сегментации и детектирования. В исследовании [32] отмечается, что состязательные возмущения могут передаваться между сетями с разными архитектурами и обучающими данными, в том числе и для разных задач распознавания.

Описание алгоритма. Пусть X – входное изображение, содержащее N целевых переменных $\tau = \{t_1, t_2, \ldots, t_N\}$. Каждая целевая переменная соответствует какому-нибудь классу $l_n \in \{1, 2, \ldots, C\}$, где C – количество меток в датасете, и пусть f – функция классификатора модели. Тогда $f(X, t_n)$ – выходной вектор (до нормализации) для каждого $t_n \in \tau$.

Цель алгоритма – найти минимальное возмущение r, которое при добавлении к X приведет к неверной сегментации большинства пикселей изображения. То есть необходимо выполнение условия

$$\forall n : \arg\max\{f(X + r, t_n) \neq l_n\}. \tag{5}$$

Для оптимизации состязательного возмущения на каждом шаге применятся алгоритм градиентного спуска. Функция потерь оптимизируется таким образом, чтобы увеличить оценку принадлежности каждого пикселя целевой переменной t_n к произвольно выбранному неверному классу и уменьшить оценку для истинного класса.

На рисунке 22 представлена математическая запись алгоритма. Положим изначально $X = X_0, r = 0$. На $m-$ой итерации алгоритма выполняются следующие действия:

1) Определяем множество τ_m
2) Вычисляем градиент суммарной функции потерь по τ_m относительно X_m

$\mathbf{X}_0 \leftarrow \mathbf{X}$, $\mathbf{r} \leftarrow \mathbf{0}$, $m \leftarrow 0$, $\mathcal{T}_0 \leftarrow \mathcal{T}$;
while $m < M_0$ **and** $\mathcal{T}_m \neq \varnothing$ **do**
  $\mathcal{T}_m = \{t_n \mid \arg\max_c \{f_c(\mathbf{X}_m, t_n)\} = l_n\}$;
  $\mathbf{r}_m \leftarrow \sum_{t_n \in \mathcal{T}_m} \left[\nabla_{\mathbf{X}_m} f_{l'_n}(\mathbf{X}_m, t_n) - \nabla_{\mathbf{X}_m} f_{l_n}(\mathbf{X}_m, t_n)\right]$;
  $\mathbf{r}'_m \leftarrow \frac{\gamma}{\|\mathbf{r}_m\|_\infty} \mathbf{r}_m$;
  $\mathbf{r} \leftarrow \mathbf{r} + \mathbf{r}'_m$;
  $\mathbf{X}_{m+1} \leftarrow \mathbf{X}_m + \mathbf{r}'_m$;
  $m \leftarrow m + 1$;
end
Return: $\mathbf{r}$

Рис. 22: Математическая интерпретация алгоритма Dense Adversary Generation [32].

3) Обновляем возмущение r_m на основе вычисленного градиента
4) Нормализация и добавление r_m к текущему возмущению r
5) Обновление изображение
6) Переход к 1 шагу

Алгоритм успешно генерирует состязательные примеры, значительно ухудшающие качество сегментации изображений популярными архитектурами FCN [17] на датасетах PASCAL VOC [13] и Cityscapes [12]. При этом добавляемые возмущения r имеют небольшую норму и визуально незаметны для человека.

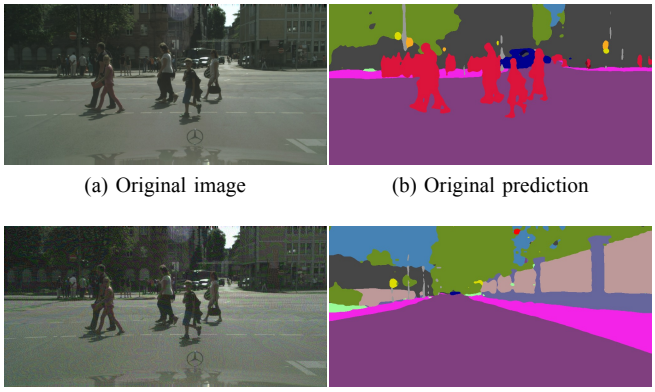
B. Универсальные состязательные возмущения

В исследовании [33] предлагается способ создания универсальных состязательных возмущений, способных обмануть систему семантической сегментации. В отличии от возмущений, зависящих от конкретного входного изображения, универсальные возмущения представляют собой шум, который, будучи добавленным к произвольному изображению, с высокой вероятностью заставит модель сегментации предсказать желаемый целевой выходной результат. Предлагается два способа генерации возмущений: статический и динамический.

Атакам производится методом чёрного ящика, то есть атакующий ничего не знает о y_{true} (ground-truth target) и не может выбрать её в качестве y_{pred}. Вместо этого он может использовать $y_{pred} = f_\theta(x)$ в качестве основы для генерации. Предполагается, что атакующий имеет доступ к функции классификатора модели $f_\theta(x)$. Эксперименты проводились на модели FCN-8s [17], в основе которой лежит сеть VGG16 [20].

Static target segmentation: В этом методе злоумышленник определяет фиксированную целевую сегментацию, такую как предсказание системы на некотором начальном шаге t_0, в качестве цели для всех последующих шагов: $y_t^{target} = y_{t_0}^{pred} \quad \forall t > t_0$. Этот метод можно применять, например, в системах видеонаблюдения, где камера является статичной. Таким образом, атакующий может подготовить фиксированную сегментированную цель, атаковать камеру и скрыть какую-нибудь подозрительную активность в момент времени t. Как видно на рисунке 23, с помощью возмущения мы подменяем сегментированное изображение на совершенно другое, не соответствующее оригинальному предсказанию.

Dynamic target segmentation: В этом случае камера или сенсор системы уже не является статичными и мы



(a) Original image (b) Original prediction

(c) Adversarial example (d) Adversarial prediction

Рис. 23: Пример static target segmentation. Источник: [33]

не можем подменить изображение, так как атаку сразу же можно будет заметить. Напротив, целью этого метода является сохранение неизменной сегментации сети, за исключением удаления определенных целевых классов. Пусть o – класс объектов, которые хочет скрыть атакующий (цель атаки), и пусть $\mathcal{I}_o = \{(i,j) \mid f_\theta(x_{ij}) = o\}$ – множество пикселей предсказанных как класс o на исходном изображении. $\mathcal{I}_{bg} = \mathcal{I} \setminus \mathcal{I}_o$,– множество «фоновых» пикселей. Обозначим $\mathbf{y}_{ij}^{\text{target}} = \mathbf{y}_{ij}^{\text{pred}} \ \forall (i,j) \in \mathcal{I}_{bg}$, и $\mathbf{y}_{ij}^{\text{target}} = \mathbf{y}_{i'j'}^{\text{pred}} \ \forall (i,j) \in \mathcal{I}_o$, где

$$i', j' = \arg\min_{i',j' \in \mathcal{I}_{bg}} (i'-i)^2 + (j'-j)^2.$$

Таким образом, согласно последнему соотношению, для фоновых пикселей целевая сегментация совпадает с исходной y^{pred}, а «дырки» от удалённых объектов класса o заполняются ближайшими фоновыми пикселями. На рисунке 24 проиллюстрирована работа этого метода.

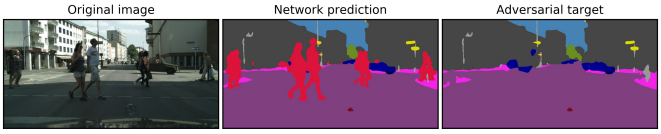
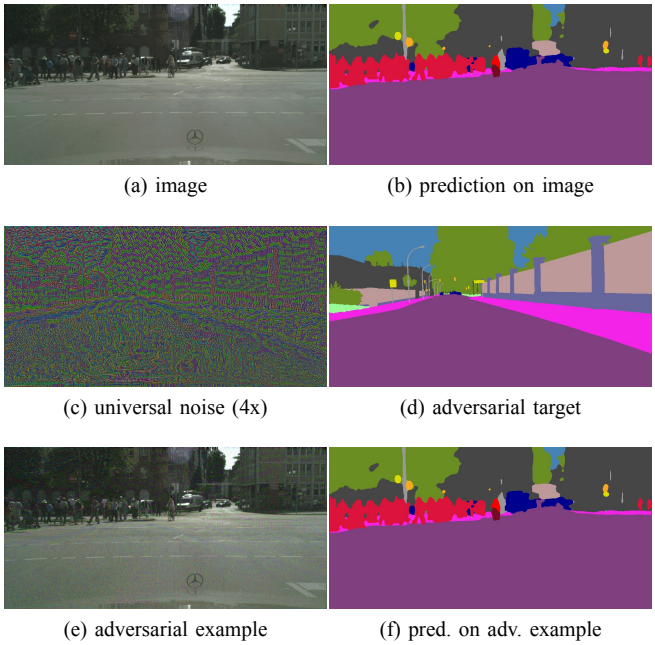


Рис. 24: Иллюстрация dynamic target segmentation, генерирующая сегментацию для сокрытия пешеходов. Источник: [33]

Введём ограничивающий оператор $\text{clip}_\varepsilon\{\xi\}$, который эквивалентен неравенству $|\xi_{ij}| \leq \varepsilon$. Предлагается следующий алгоритм генерации универсального возмущения Ξ:

1) Определить набор из m изображений $\mathcal{D}^{\text{train}} = \{(\mathbf{x}^{(k)}, \mathbf{y}^{\text{target},k})\}_{k=1}^m$, где $\mathbf{y}^{\text{target},k}$ сгенерировано одним из описанных выше способов.
2) Инициализировать универсальное возмущение $\Xi^{(0)} = \mathbf{0}$.
3) $\Xi^{(n+1)} = \text{clip}_\varepsilon\left\{\Xi^{(n)} - \alpha\,\text{sgn}(\nabla^{\mathcal{D}}(\Xi))\right\}$, где $\nabla^{\mathcal{D}}(\Xi) = \frac{1}{m}\sum_{k=1}^{m} \nabla_x J_{\text{ss}}^{\omega}(f_\theta(\mathbf{x}^{(k)} + \Xi), \mathbf{y}^{\text{target},k})$ – усредненный по всем тренировочным данным градиент функции потерь.
4) Регуляризация: $\nabla^{\mathcal{D}}(\hat{\Box}) = \frac{1}{mRS}\sum_{r=1}^{R}\sum_{s=1}^{S}\sum_{k=1}^{m} \nabla_x J_{\text{ss}}^{\omega}(f_\theta(\mathbf{x}_{[r,s]}^{(k)} + \hat{\Xi}), \mathbf{y}_{[r,s]}^{\text{target},k})$, где

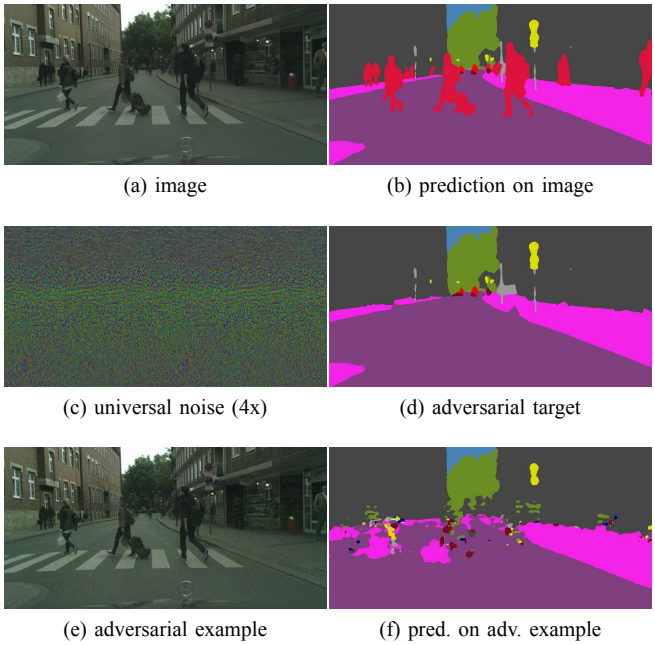


(a) image (b) prediction on image

(c) universal noise (4x) (d) adversarial target

(e) adversarial example (f) pred. on adv. example

Рис. 25: Влияние универсальных состязательных возмущения для **статических** целей ($\varepsilon = 10$): **(a)** Исходное изображение Cityscapes. **(b)** Предсказание сети над исходным изображением. **(c)** Универсальное состязательное возмущение (усиленное в 4 раза). **(d)** Состязательная цель, которую хотим получить на выходе. **(e)** Состязательный пример для (a). **(f)** Предсказание сети для (e). Источник: [33]



(a) image (b) prediction on image

(c) universal noise (4x) (d) adversarial target

(e) adversarial example (f) pred. on adv. example

Рис. 26: Влияние универсальных состязательных возмущения для **динамических** целей ($\varepsilon = 10$): **(a)** Исходное изображение Cityscapes. **(b)** Предсказание сети над исходным изображением. **(c)** Универсальное состязательное возмущение (усиленное в 4 раза). **(d)** Состязательная цель, которую хотим получить на выходе. **(e)** Состязательный пример для (a). **(f)** Предсказание сети для (e). Источник: [33]

R, S – количество «плиток» в каждом измерении и $[r, s] = \{i, j \mid [rh \leq i < (r+1)h] \wedge [sw \leq j < (s+1)w]\}$.

5) Итерации продолжаются в течение заданного числа шагов n.

На четвёртом шаге алгоритма мы используем регуляризацию, так как в этом месте потенциально может возникнуть проблема переобучения на тренировочных данных, что приведёт к уменьшению обобщающей способности Ξ на неизвестных входных данных. Это вполне очевидное поведение поскольку Ξ имеет ту же размерность [33]. Именно поэтому используется подход, при котором оптимизируется «протовозмущение» $\hat{\Xi}$ меньшего размера $h \times w$. Таким образом, регуляризация посредством периодического размножения меньшего протовозмущения $\hat{\Xi}$ вместо оптимизации всего высокоразмерного Ξ целиком снижает размерность оптимизационной задачи, тем самым уменьшая риск переобучения. Формула на четвёртом шаге – это градиент, усреднённый по обучающим данным и всем плиткам.

Для оценки воздействия универсальных возмущения на системы семантической сегментации был проведён ряд экспериментов. Сегментация изображений выполнялось с помощью FCN-8s + VGG16. На рисунках 25 и 26 показано влияние универсальных состязательных возмущений на статические и динамические цели соответственно.

Качество генерируемых состязательных возмущений зависит от размера m тренировочного набора. В силу того, что описанный метод не требует истинных меток (ground-truth labels), для генерации можно использовать любой большой неразмеченный набор данных [33].

Универсальные возмущения способны обобщаться на другие датасеты и архитектуры в качестве нецелевой атаки, вызывая общее искажение выходов модели, но не в качестве целевой атаки.

C. Fast Gradient Sign Method

Помимо рассмотренных выше существуют и другие методы создания состязательных примеров. Одним из них является метод FGSM и его разновидности.

Fast Gradient Sign Method (FGSM) [34] создаёт состязательные примеры путём увеличения функции потерь (обычно используется cross-entropy) сети на входном изображении x:

$$\mathbf{x}^{\text{adv}} = x + \epsilon \cdot \text{sgn}(\nabla_{\mathbf{x}} L(f(x; \theta), \mathbf{y}))$$

Здесь и далее L – функция потерь между предсказанием модели и целью y. FGSM – одношаговая нецелевая атака, приближенно минимизирующая норму возмущения l_∞, ограниченную параметром ϵ.

FGSM II [35] – одношаговая целевая атака, которая присваивает состязательному примеру наименее вероятный класс y_t:

$$\mathbf{x}^{\text{adv}} = \mathbf{x} - \epsilon \cdot \text{sgn}(\nabla_{\mathbf{x}} L(f(x; \theta), \mathbf{y_t}))$$

Iterative FGSM [35] – расширение одношагового метода FGSM путём итеративного применения, увеличивающего шанс обмануть исходную модель.

$$\mathbf{x}_0^{\text{adv}} = x$$
$$\mathbf{x}_{t+1}^{\text{adv}} = \text{clip}_\epsilon(\mathbf{x}_t^{\text{adv}} + \alpha \cdot \text{sgn}(\nabla_{\mathbf{x}_t^{\text{adv}}} L(f(\mathbf{x}_t^{\text{adv}}; \theta), y)))$$

Iterative FGSM II [35] – более сильная версия FGSM II, которая на каждой итерации назначает наименее вероятный целевой класс y_{ll} для состязательного примера на данной итерации.

$$\mathbf{x}_{t+1}^{\text{adv}} = \text{clip}_\epsilon(\mathbf{x}_t^{\text{adv}} - \alpha \cdot \text{sgn}(\nabla_{\mathbf{x}_t^{\text{adv}}} L(f(\mathbf{x}_t^{\text{adv}}; \theta), y_{ll})))$$

Все вышеупомянутые атаки были предложены в контексте классификации изображений, но они были адаптированы к задачам семантической сегментации и обнаружения объектов [36].

D. Атака инстанс сегментации на примере YOLACT

В исследовании [37] авторы предлагают улучшенный метод проекционного градиентного спуска (Projected Gradient Descent, PGD) для генерации состязательных возмущений. С помощью данного метода можно сгенерировать состязательный шум, которым можно атаковать модели инстанс сегментации. В настоящем исследовании эксперименты проводились на модели YOLACT [23] с различными сетями в основе (ResNet-101 [25], Darknet-53 [38], ResNet-50 [25]), обученной на датасетах MS-COCO и Rail dataset.

Данная атака предполагает, что атакующий имеет полный доступ к архитектуре атакуемой системы, и относится к цифровым состязательным атакам на системы сегментации. На рисунке 28 показан процесс атаки.

Постановка задачи: Определим функцию потерь как сумму трёх компонент: потеря для регрессии границ прямоугольников L_{box}, потери классификации L_{cls} и потери для предсказания масок сегментации объектов L_{mask}:

$$\begin{aligned} L = w_1 L_{box}(x+\eta, \theta, l, g) + \\ w_2 L_{cls}(x+\eta, \theta, c) + \\ w_3 L_{mask}(x+\eta, \theta, m), \end{aligned} \tag{6}$$

где x – исходное изображение, η – возмущение, θ – параметры модели, l – предсказание ограничивающей рамки, g – истинные границы объектов из обучающей разметки, c – предсказанные классы объектов, m – истинные маски сегментации объектов. Весовые коэффициенты w_1, w_2, w_3 задают относительную важность каждого вида потерь. В работе использовались значения: $w_1 = 1.5$, $w_2 = 1$, $w_3 = 6.125$.

Улучшенный PGD, используемый для создания возмущения, математически можно выразить следующим образом

$$x_\eta^t = \frac{\alpha}{T} \times \text{sgn}(\nabla_x L^t(\theta, x, y_{gt})) \tag{7}$$

$$x_{adv}^t = \text{Proj}_\varepsilon(x_{ori} + x_\eta^t) \tag{8}$$

где $\nabla_x L^t(\theta, x, y)$ – градиент функции потерь L по входу x, рассчитанный на текущих весах θ модели и истинных метках y, α – размер шага, T – количество итераций, x_{ori} – исходное изображение, x_{adv}^t – состязательное изображение , $\text{Proj}_\varepsilon()$ – функция обрезки (clip) значений пикселей, выходящих за пределы ε-окрестности исходного изображения x_{ori}.

Таким образом, на каждой итерации t метод улучшенного PGD:

1) Вычисляет возмущение в направлении градиента функции потерь
2) Нормализует его
3) Применяет к исходному изображению
4) Обрезает диапазон пикселей

Этот процесс повторяется T итераций.

Целью атаки является максимизация общей функции потерь L путём нахождения состязательного возмущения η для входа x, которое бы максимально нарушило работу модели YOLACT при предсказании рамок, классов и масок объектов.

VIII. Физические состязательные атаки на семантическую сегментацию

Рассмотренные выше цифровые атаки в общем случае представляют собой один и тот же процесс. Для входного изображения генерируется и добавляется специальный цифровой шум. Затем полученное изображение подаётся на вход модели, вычисляется функция потерь и производится оптимизация состязательного примера. Этот процесс происходит до тех пор, пока состязательный пример не начнёт значительно нарушать работу модели.

Физические атаки проводятся посредством размещения какого-нибудь, например, стикера или патча (см. секцию VI-D) в физическом пространстве. В исследовании [31] рассматривают атаки на основе патчей на системы семантической сегментации изображений. Такие атаки являются широко распространёнными в силу эффективности и простоты реализации.

A. SS Attack

В [39] предлагается новый подход к созданию состязательных возмущений, воздействующих на системы семантической сегментации изображений – SS Attack. Данная атака была представлена и применима к задачам автономного вождения и автономным транспортным средствам.

Сначала создаётся произвольный патч, который затем тестируется на предварительно обученной модели. Для оптимизации эффективности атаки применяется pixel-wise cross-entropy loss, которая оптимизирует патчи на предварительно обученной модели ICNet [31]. После оптимизации подготовленный патч печатается на плакате размером 2×1 метра и размещается в физическом пространстве. На рисунке 29 можем наблюдать разницу между атакой с использованием рандомного и оптимизированного патча. Во втором случае атака была наиболее эффективной и подготовленный патч значительно лучше справился с нарушением работы классификатора.

Для тестирования атаки был задействован CARLA Simulator[4], который позволил протестировать атаку с различными патчами в различных сценах.

Произвести такую атаку в реальных условиях не так просто, практически невозможно. SS Attack относится

[4]CARLA Simulator – open-source симулятор для исследований в области автономного вождения.

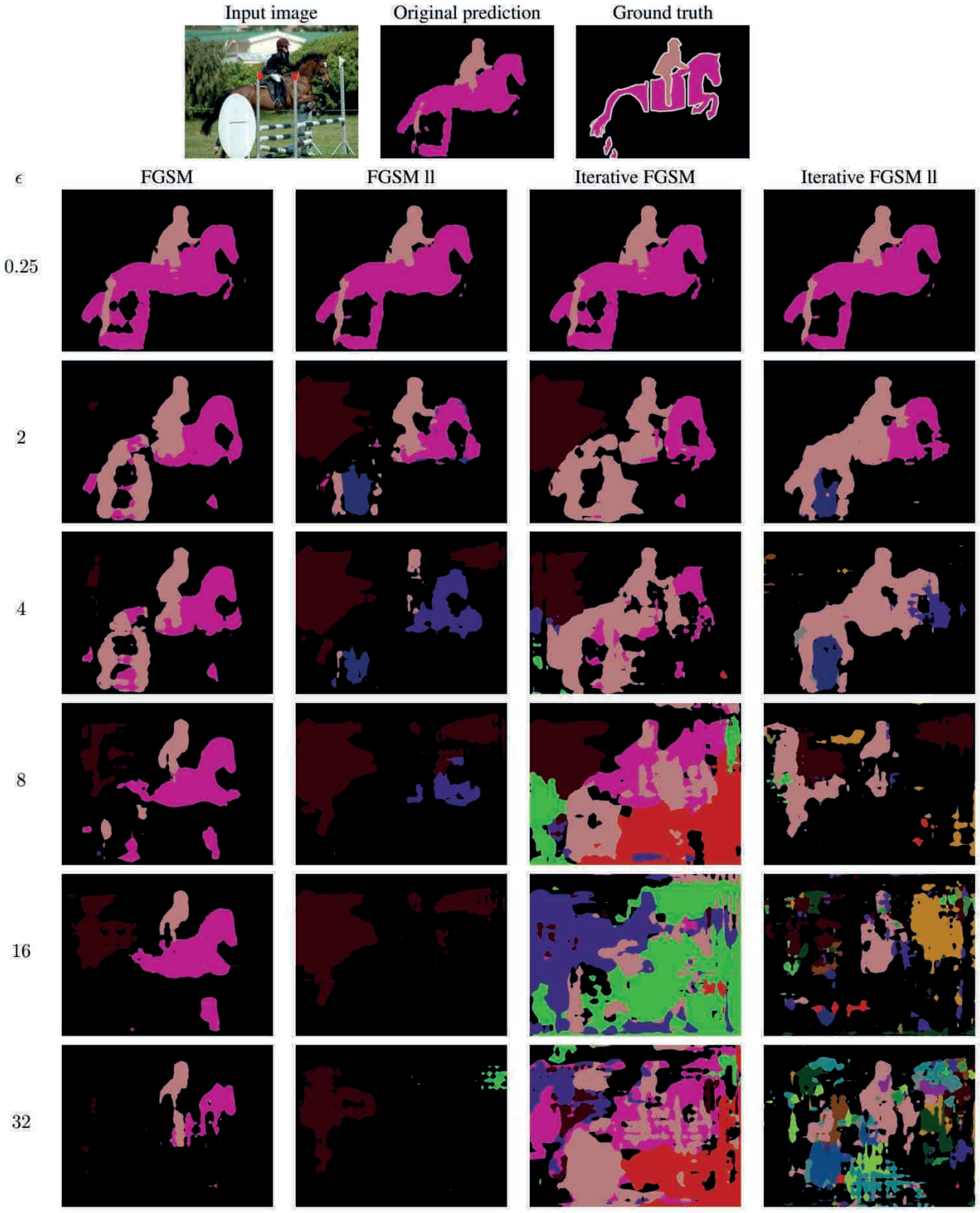


Рис. 27: Сравнение различных состязательных атак на сеть Deeplab v2 Multiscale ASPP. Итерационные атаки (последние две колонки), как и ожидалось, наиболее эффективные, чем одношаговые атаки (первые две колонки). Источник: [36]

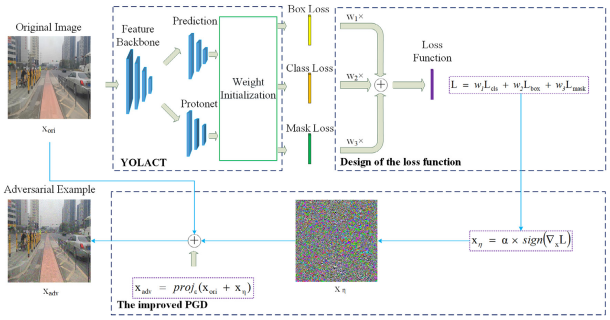


Рис. 28: Атака на инстанс сегментацию YOLACT. Источник: [37]

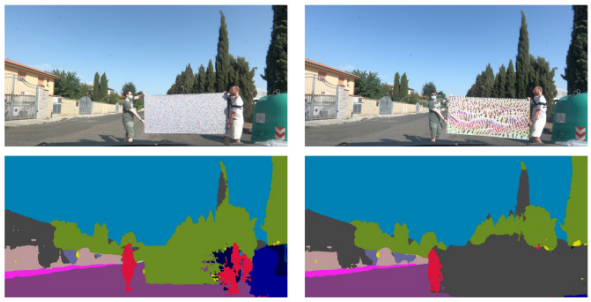


Рис. 29: Применения SS Attack в реальном мире: (слева) пример атаки с рандомным патчем, (справа) атака на основе оптимизированного патча. Источник: [39]

к атакам «белого ящика» и у атакующего должно быть полное представление об атакуемой системе, её архитектуре и данных. В реальных же условиях атакующий не может знать о том, как работает модель и какие данные использует. Поэтому задача создания эффективного состязательного патча становится менее реальной.

На рисунке 29 показано влияние оптимизированных патчей на модель BiSeNet [40]. Ненадёжные (non-robust) патчи (без EOT) обеспечивают более высокую эффективность атаки по сравнению с патчами, оптимизированными с помощью EOT. Это связано с тем, что процесс оптимизации упрощается, если не учитывать рандомизированные преобразования. Однако важно отметить, что эти патчи невозможно перенести в реальный мир и они неустойчивы даже к простым преобразованиям [39].

B. Атака на LiDAR

В исследовании [41] авторы одними из первых рассматривают и изучают физические атаки на системы автономного транспорта, использующие сегментацию трёхмерных облаков точек, полученных с помощью LiDAR [2]. Предлагается новый фреймворк для проведения состязательных атак на автономные системы путём размещения в физическом пространстве простых объектов (например, картон или дорожные знаки). Эксперименты в различных реальных сценариях показали, что проводимые атаки достигают успеха в более чем 90% случаев.

1) Сценарий атаки: Рассматривается два сценария атаки на системы автономного вождения:

- *Атака скрытия автомобиля/препятствия*. Злоумышленник может разместить объекты, отражающие лазерные лучи LiDAR, вокруг автомобиля на парковки или автомобиля припаркованного на обочине. Как показано на рисунке 31, эти объекты могут нарушить работу модели-жертвы, скрывая припаркованный автомобиль от системы восприятия. Такая атака может привести к столкновению, если автономное транспортное средство не заметит препятствие на своем пути.
- *Атака изменения поверхности дороги*. В этом случае атакующий размещает специальные объекты на дороге или обочине. На рисунке 32 видно, что эти объекты могут нарушить работу системы восприятия и, например, заменить участок дороги растительностью. Это может привести к резкому торможению или изменению направления движения, что может стать причиной аварии.

Кроме того, в исследовании рассматриваются и оцениваются атаки, проводимые методом как методом чёрного ящика, так и белого.

2) Способ проведения атаки: Атака на системы семантической сегментации LiDAR в автономных транспортных средствах проводится следующим образом: атакующий собирает оригинальные облака точек, соответствующие сцене, которую видит транспортное средство-жертва. Затем создаются физические объекты (состязательные примеры), которые могут быть размещены в реальном мире так, чтобы изменить восприятие автономной системы. Эти объекты располагаются в заранее определённых местах, чтобы гарантировать, что они будут замечены системой независимо от направления движения. Цель атаки может варьироваться от скрытия препятствий, таких как припаркованные автомобили, до изменения восприятия поверхности дороги, что может привести к неожиданному поведению, такому как резкое торможение или изменение курса. Успешность атаки оценивается на основе способности ввести в заблуждение модель сегментации и достичь целевого эффекта. На рисунке 33 показан реальный пример атаки.

C. IPatch

В статье [42] автор представляет абсолютно другой метод проведения состязательных атак на системы семантической сегментации. IPatch – это удалённый состязательный патч (remote adversarial patch, RAP), с помощью которого можно локально манипулировать системой сегментации и её выводом.

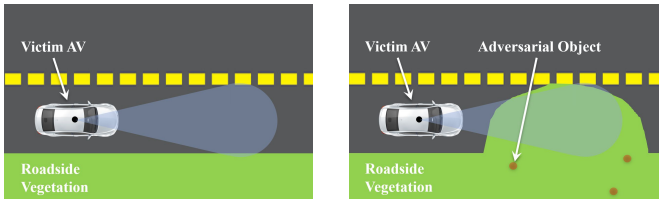
Суть атаки заключается в том, что мы можем выделить какую-то область на изображении и в совершенно другом месте изображения добавить RAP, который будет воздействовать на эту область и нарушать работу классификатора. На рисунке 34 показан пример такой атаки: на изображении выделена область дороги (красный квадрат), а в правом нижнем углу размещён RAP. При сегментации такого изображения на выходе мы получим неверно сегментированное изображение. Теперь дорога сегментируется и классифицируется как тротуар.

На изображении 35 показано, как работает IPatch. Метод позволяет точно изменять определенную область изображения в соответствии с желаемым шаблоном.

IPatch выбирает конкретную область изображения, обозначенную как m, и задает целевой шаблон t, под который нужно подогнать эту область. Стратегически

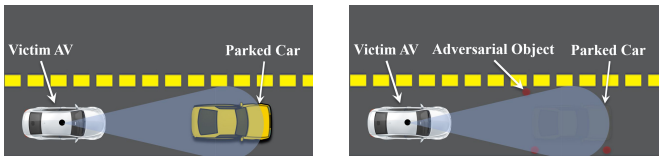
(a) original image (b) no patch (c) a random patch (d) an EOT-based patch (e) non-robust patch

Рис. 30: Атака на семантическую сегментацию BiSeNet [40] на примере изображения из валидационного набора Cityscapes. Источник: [39]



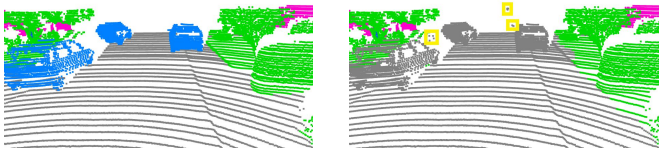
(a) До атаки (b) После атаки

Рис. 31: Скрытие автомобиля/препятствия с помощью специально расположенных вокруг припаркованного автомобиля объектов. Источник: [41]



(a) До атаки (b) После атаки

Рис. 32: Атака скрытием участка дороги. Источник: [41]



(a) До атаки (b) После атаки

Рис. 33: Пример атаки на систему восприятия автономного транспортного средства. К синим точкам относятся автомобили, серым – дорога, зелёным – растительность. Источник: [41]

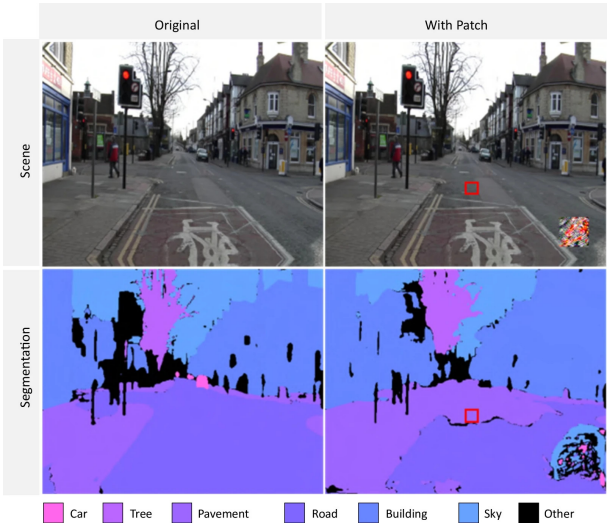


Рис. 34: Пример использования IPatch атаки на модель сегментации. В этом примере патч был сгенерирован специально для изменения заранее предопределённой области. Источник [42]

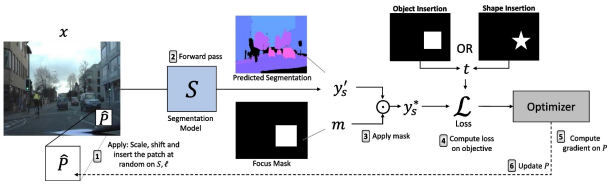


Рис. 35: Обзор тренировочного фреймворка IPatch для создания RAP. Для простоты показан только один образец x, хотя на практике используются пакеты изображений. Источник: [42]

выбирая размер и положение области, а также согласуя ее с шаблоном t, IPatch изменяет вероятности значений пикселей внутри области таким образом, чтобы распознанный в этой области объект соответствовал желаемому классу.

Ключевая особенность IPatch – его эффективность и направленность оптимизации. Метод концентрируется только на выбранной области изображения, избегая лишних вычислений для посторонних семантических классов.

Область m выбирается следующим образом: сначала определяются координаты центра $L = (i, j)$, затем вокруг этой точки выделяется квадратная или круглая область радиуса m. Именно эта область в дальнейшем будет подвержена манипуляциям.

В зависимости от задачи (вставка объекта, создание пользовательской формы и т.д.) целевой шаблон t корректируется соответствующим образом. Для генерации патча P, который оптимизирован под заданную целевую функцию, используется метод EOT со специальными семантическими масками.

Как и SS Attack, IPatch также относится к атакам «белого ящика». Как было показано выше, при построении состязательного патча используется предварительно обученная модель, на которой проверяется и оптимизируется патч. Поэтому атакующему необходимо иметь представление об архитектуре модели, чтобы построить состязательное возмущение.

IX. Дополнение к состязательным атакам

Несмотря на то, что задача сегментации изображений активно изучается, исследователи разрабатывают новые и усовершенствуют уже существующие модели, эта об-

ласть всё ещё остаётся малоисследованной. Наиболее важными для нас являются исследования в области физических состязательных атак, поскольку модели сегментации используются во многих задач, таких как, например, автономное вождение. В некоторых случаях атака на сегментацию может привести к серьёзным последствиям в критических применениях. Безопасность моделей сегментации имеет важное значение, поэтому необходимо иметь представление о том, как модель будет себя вести в нестандартных ситуациях.

Архитектура моделей обнаружения объектов и сегментации изображений имеет значительное количество сходств. В своей основе они используют свёрточные слои, которые используются для выделения важных признаков из изображения. Только в случае задачи обнаружения мы классифицируем объект целиком, а в случае задачи сегментации мы классифицируем каждый пиксель изображения по отдельности и соотносим его с нужным классом.

В секции VI-B было отмечено, что при оценке состязательных атак также учитывается их способность к переносимости. Поэтому в этой секции мы дополнительно рассмотрим несколько физических состязательных атак, применяемых к задаче, отличной от сегментации. Будем рассматривать только атаки на модели детекции объектов. В сижу схожести задач сегментации и обнаружения, атаку на обнаружение можно адаптировать и применить к сегментации.

Атаки на модели классификации объектов, изображений классифицируют только один объект. На деле в реальном мире у нас практически не будет ситуаций где встречается только один объект, их всегда будет несколько, а то и целое множество. Но стоит отметить, что некоторые из атак, которые будут рассмотрены, были изначально разработаны и применимы к задачам классификации, но были адаптированы под задачи обнаружения.

A. Атака на YOLOv3 object detection

В исследовании [43] авторы предлагают метод генерации состязательного возмущения, способный обмануть модель обнаружения объектов YOLOv3 [38]. Создаётся такой состязательный патч, который попадая в поле зрение камеры модели, нарушает её работу и перестаёт обнаруживать другие окружающие объекты. Правильно созданный патч может подавить практически все объекты на изображении, даже если патч не перекрывает их.

Для реализации алгоритма используется метод проекционного градиентного спуска (Projected Gradient Descent) и техника EOT, оптимизирующие функцию потерь для подавления (сокрытия) объектов для обнаружения.

На рисунке 36 показано применение состязательного патча в физическом пространстве для обмана YOLOv3.

B. Атака на аэрофотосъёмку

В секции II-D был рассмотрен один из примеров применения сегментации – сегментация снимков, сделанных с воздуха. В [44] рассматривается способ скрытия объектов на снимках, сделанных с воздуха (или с крыши

(a) До атаки (b) После атаки

Рис. 36: Применение состязательного патча для атаки на модель YOLOv3 в физическом пространстве. Источник: [43]

высокого здания), и предлагается метод создания состязательного патча. Одним из применений такой атаки является, например, сокрытие военной техники или стратегических объектов. Предполагается, что атакующий имеет полное представление об архитектуре модели.

Предлагается два способа размещения состязательного патча: на крыши автомобиля (Type ON) и вокруг него (Type OFF) (см. рис. 37a и 37b соответственно). Геометрические параметры патча:

- Type ON: прямоугольный патч, который располагается прямо на крыше автомобиля.
- Type OFF: три прямоугольные полосы, которые располагаются вне и вокруг автомобиля, образую форму «П».

В качестве базовой модели для экспериментов была выбрана модель YOLOv3 [38], предварительно обученная на датасете MS-COCO и дообученая на датасете COWC[5].

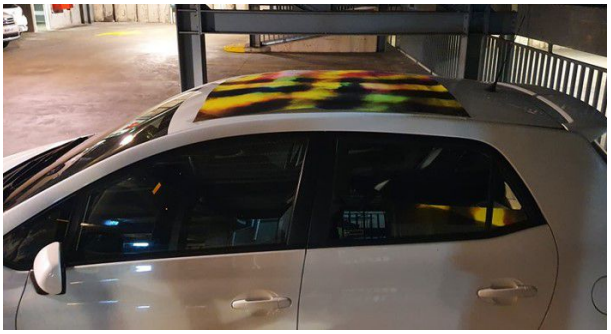
Рассматривается две зоны атаки: side street и car park. *Side street* – снимок улицы, сделанный с высоты 10 этажа высокого здания (40 метров), *car park* – снимок парковки, сделанный с высоты 60 метров при помощи дрона.

Вкратце процесс создания состязательного патча можно описать следующим образом:

1) Инициализируется случайный цифровой патч типа ON или OFF.
2) Этот патч встраивается в тренировочное изображение в область, где на изображении находится автомобиль.
3) К встроенному патчу применяются различные преобразования (геометрические - масштаб, поворот; цветовые - яркость, контраст, шум; и имитация погодных условий), чтобы смоделировать его реальный вид при съемке.
4) Модифицированные изображения с патчем подаются на вход детектору автомобилей YOLOv3.
5) Вычисляется функция потерь как комбинация максимального objectness score на этих изображениях, NPS и TV Loss патча.

Этот процесс итерационно продолжается до тех пор, пока в результате оптимизации не получится патч, который при наложении на реальные изображения максимально снизит уверенность детектора в присутствии автомобиля.

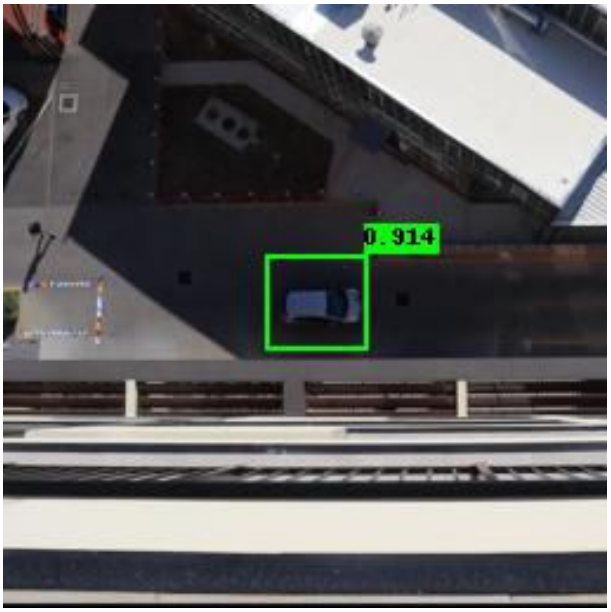
[5]Cars Overhead with Context (COWC) – датасет, содержащий $25,384$ цветных аннотированных изображений (256×256 пикселей) автомобилей, сделанных с воздуха.



(a) Патч на крыше автомобиля.

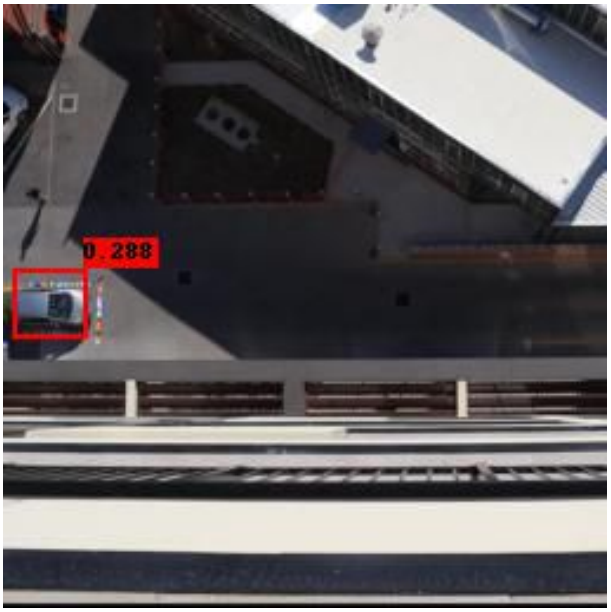
(b) Патч вокруг автомобиля.

(c) Автомобили на парковке без состязательного патча (objectness score ≥ 0.9)



(d) Автомобиль, стоящий вдоль улицы, без состязательного патча (objectness score $= 0.914$).

(e) На крышах двух автомобилях из (c) размещен состязательный патч. Их objectness score снизился до 0.315 и 0.396.



(f) Автомобиль из (d) окружён состяза состязательным патчем. Его objectness score снизился до 0.288.

Рис. 37: Физическая состязательная атака на систему обнаружения объектов (в нашем случае – автомобилей) YOLOv3 для аэроснимков. Источник: [44]

Рис. 38: Футболки с нанесенными на них состязательными изображениями. Источник: [31]

На рисунках 37c и 37d показаны чистые изображения (без патчей), а на рисунках 37e и 37f показаны состязательные изображения с патчем на автомобиле и вне соответственно.

C. Атака состязательной одеждой (Adversarial Knit)

В современном мире мы окружены различными технологиями наблюдения и отслеживания. В наших реалиях невозможно выйти из дома и не попасть в поле зрения хотя бы одной камеры видеонаблюдения. Камеры видеонаблюдения присутствуют практически везде – на улицах, в парках, торговых центрах, ресторанах, аэропортах, вокзалах и т. д. Их устанавливают государственные органы и частные компании для мониторинга общественного порядка и безопасности. Во многих местах используются технологии распознавания лиц, например, в Московском метро или аэропортах.

Вся эта постоянная слежка вызывает у некоторых опасения по поводу их конфиденциальности. В связи с этим был предложен проект CAPABLE [45], объединяющий в себе инженеров и дизайнеров одежды. Их задачей является разработка и создание такой одежды, которая смогла бы скрыть вас от постоянного видеонаблюдения в общественных местах.

Таким образом была разработана специальная состязательная одежда, которая отлично справляется со своей задачей. Были проведены эксперименты, которые показали, что такая одежда вводит в заблуждение модель YOLO и не позволяет ей идентифицировать человека.

При создании состязательной одежды необходимо учитывать то, что при ходьбе она будет деформироваться/искажаться на теле человека, из-за чего может значительно снизиться эффективность и устойчивость атаки к искажениям. Для решения этой задачи используется метод *Creases Transformation* (см. секцию VI-C), которая учитывает все возможные искажения на этапе моделирования возмущения.

X. Заключение

В данной статье был проведён всесторонний анализ уязвимости систем сегментации изображений к состязательным атакам. Были рассмотрены различные алгоритмы сегментации, архитектуры и методы проведения состязательных атак. Показано влияние состязательных примеров на различные модели сегментации. Анализ показал, что даже небольшие искажения в исходных данных могут привести к значительным ошибкам в работе систем сегментации, что представляет серьёзную угрозу для их применения в критически важных областях.

Библиография

[1] N. Shiledarbaxi, *Semantic vs Instance vs Panoptic: Which Image Segmentation Technique To Choose*, Available: https://analyticsindiamag.com/ai-mysteries/semantic-vs-instance-vs-panoptic-which-image-segmentation-technique-to-choose/, [Дата обращения: 04.07.2024].

[2] A. Nguyen и B. Le, «3D point cloud segmentation: A survey,» в *2013 6th IEEE conference on robotics, automation and mechatronics (RAM)*, IEEE, 2013, с. 225—230.

[3] H. Caesar, J. Uijlings и V. Ferrari, «Coco-stuff: Thing and stuff classes in context,» в *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, с. 1209—1218.

[4] A. Adegun, N. Akande, R. Ogundokun и E. Asani, «Image segmentation and classification of large scale satellite imagery for land use: a review of the state of the arts,» *Int. J. Civ. Eng. Technol*, т. 9, № 11, с. 1534—1541, 2018.

[5] *Satellite imagery is helping governments to combat deforestation*, Available: https://analyticsindiamag.com/ai-mysteries/semantic-vs-instance-vs-panoptic-which-image-segmentation-technique-to-choose/, [Дата обращения: 04.07.2024].

[6] K. O'Shea и R. Nash, *An Introduction to Convolutional Neural Networks*, 2015. url: https://arxiv.org/abs/1511.08458.

[7] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz и D. Terzopoulos, «Image segmentation using deep learning: A survey,» *IEEE transactions on pattern analysis and machine intelligence*, т. 44, № 7, с. 3523—3542, 2021.

[8] Z. Keita, *An Introduction to Convolutional Neural Networks (CNNs)*, Available: https://www.datacamp.com/tutorial/introduction-to-convolutional-neural-networks-cnns, [Дата обращения: 18.07.2024].

[9] I. Goodfellow, Y. Bengio и A. Courville, *Deep learning*. MIT press, 2016.

[10] V. Badrinarayanan, A. Kendall и R. Cipolla, «Segnet: A deep convolutional encoder-decoder architecture for image segmentation,» *IEEE transactions on pattern analysis and machine intelligence*, т. 39, № 12, с. 2481—2495, 2017.

[11] T.-Y. Lin и др., «Microsoft COCO: Common objects in context,» в *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, Springer, 2014, с. 740—755.

[12] M. Cordts и др., «The cityscapes dataset for semantic urban scene understanding,» в *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, с. 3213—3223.

[13] M. Everingham, L. Van Gool, C. K. Williams, J. Winn и A. Zisserman, «The pascal visual object classes (voc) challenge,» *International journal of computer vision*, т. 88, с. 303—338, 2010.

[14] R. Mottaghi и др., «The Role of Context for Object Detection and Semantic Segmentation in the Wild,» в *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[15] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso и A. Torralba, «Scene parsing through ade20k dataset,» в *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, с. 633—641.

[16] A. Kirillov и др., «Segment anything,» в *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, с. 4015—4026.

[17] J. Long, E. Shelhamer и T. Darrell, «Fully convolutional networks for semantic segmentation,» в *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, с. 3431—3440.

[18] H. Kumar, *Quick intro to semantic segmentation: FCN, U-Net and DeepLab*, Available: https://kharshit.github.io/blog/2019/08/09/quick-intro-to-semantic-segmentation, [Дата обращения: 12.06.2024].

[19] O. Ronneberger, P. Fischer и T. Brox, «U-net: Convolutional networks for biomedical image segmentation,» в *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, Springer, 2015, с. 234—241.

[20] K. Simonyan и A. Zisserman, «Very deep convolutional networks for large-scale image recognition,» *arXiv preprint arXiv:1409.1556*, 2014.

[21] J. Redmon, S. Divvala, R. Girshick и A. Farhadi, «You only look once: Unified, real-time object detection,» в *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, с. 779—788.

[22] G. Jocher, A. Chaurasia и J. Qiu, *Ultralytics YOLO*, вер. 8.0.0, янв. 2023. url: https : / / github . com / ultralytics/ultralytics.

[23] D. Bolya, C. Zhou, F. Xiao и Y. J. Lee, «Yolact: Real-time instance segmentation,» в *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, с. 9157—9166.

[24] T.-Y. Lin, P. Goyal, R. Girshick, K. He и P. Dollár, «Focal loss for dense object detection,» в *Proceedings of the IEEE international conference on computer vision*, 2017, с. 2980—2988.

[25] K. He, X. Zhang, S. Ren и J. Sun, «Deep residual learning for image recognition,» в *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, с. 770—778.

[26] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan и S. Belongie, «Feature pyramid networks for object detection,» в *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, с. 2117—2125.

[27] A. Dosovitskiy и др., «An image is worth 16x16 words: Transformers for image recognition at scale,» *arXiv preprint arXiv:2010.11929*, 2020.

[28] K. He, X. Chen, S. Xie, Y. Li, P. Dollár и R. Girshick, «Masked autoencoders are scalable vision learners,» в *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, с. 16 000—16 009.

[29] A. Radford и др., «Learning transferable visual models from natural language supervision,» в *International conference on machine learning*, PMLR, 2021, с. 8748—8763.

[30] A. Vaswani и др., «Attention is all you need,» *Advances in neural information processing systems*, т. 30, 2017.

[31] A. Guesmi, M. A. Hanif, B. Ouni и M. Shafique, «Physical adversarial attacks for camera-based smart systems: Current trends, categorization, applications, research challenges, and future outlook,» *IEEE Access*, 2023.

[32] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie и A. Yuille, «Adversarial examples for semantic segmentation and object detection,» в *Proceedings of the IEEE international conference on computer vision*, 2017, с. 1369—1378.

[33] J. Hendrik Metzen, M. Chaithanya Kumar, T. Brox и V. Fischer, «Universal adversarial perturbations against semantic image segmentation,» в *Proceedings of the IEEE international conference on computer vision*, 2017, с. 2755—2764.

[34] I. J. Goodfellow, J. Shlens и C. Szegedy, «Explaining and harnessing adversarial examples,» *arXiv preprint arXiv:1412.6572*, 2014.

[35] A. Kurakin, I. Goodfellow и S. Bengio, «Adversarial machine learning at scale,» *arXiv preprint arXiv:1611.01236*, 2016.

[36] A. Arnab, O. Miksik и P. H. Torr, «On the robustness of semantic segmentation models to adversarial attacks,» в *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, с. 888—897.

[37] Z. Zhang, S. Huang, X. Liu, B. Zhang и D. Dong, «Adversarial attacks on YOLACT instance segmentation,» *Computers & Security*, т. 116, с. 102 682, 2022.

[38] J. Redmon и A. Farhadi, «Yolov3: An incremental improvement,» *arXiv preprint arXiv:1804.02767*, 2018.

[39] F. Nesti, G. Rossolini, S. Nair, A. Biondi и G. Buttazzo, «Evaluating the robustness of semantic segmentation for autonomous driving against real-world adversarial patch attacks,» в *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, с. 2280—2289.

[40] C. Yu, C. Gao, J. Wang, G. Yu, C. Shen и N. Sang, «Bisenet v2: Bilateral network with guided aggregation for real-time semantic segmentation,» *International journal of computer vision*, т. 129, с. 3051—3068, 2021.

[41] Y. Zhu, C. Miao, F. Hajiaghajani, M. Huai, L. Su и C. Qiao, «Adversarial attacks against lidar semantic segmentation in autonomous driving,» в *Proceedings of the 19th ACM conference on embedded networked sensor systems*, 2021, с. 329—342.

[42] Y. Mirsky, «IPatch: a remote adversarial patch,» *Cybersecurity*, т. 6, № 1, с. 18, 2023.

[43] M. Lee и Z. Kolter, «On physical adversarial patches for object detection. arXiv,» *arXiv preprint arXiv:1906.11897*, 2019.

[44] A. Du и др., «Physical adversarial attacks on an aerial imagery object detector,» в *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, с. 1796—1806.

[45] R. Didero и G. M. Conti, «CAPABLE: Engineering, Textile, and Fashion Collaboration, for Citizens' Awareness and Privacy Protection,» *Human Factors for Apparel and Textile Engineering*, т. 32, с. 39, 2022.

Analysis of adversarial attacks on image segmentation systems

Egor Vorobyev

***Abstract*—The article discusses image segmentation methods and the problems associated with adversarial attacks on these systems. It is necessary to ensure the security of such systems, since segmentation is widely used in various computer vision tasks and can be a weak point in critical applications. An overview of different types of segmentation is presented, including image segmentation, semantic segmentation, and panoptic segmentation. Popular architectures of segmentation models are considered, such as FCN, U-Net, YOLO, Segment Anything and others. The article analyzes adversarial attacks on image segmentation systems, including both digital and physical attacks. Special attention is paid to methods and algorithms for creating adversarial examples. The aim of the work is to attract the attention of the research community to the problem of security of segmentation systems, to develop new, state-of-the-art and more robust to adversarial attacks segmentation models.**

***Keywords*—adversarial attacks, convolutional neural networks, encoder-decoder models, image segmentation**

References

[1] A. Adegun, N. Akande, R. Ogundokun, and E. Asani, «Image segmentation and classification of large scale satellite imagery for land use: A review of the state of the arts», *Int. J. Civ. Eng. Technol*, vol. 9, no. 11, pp. 1534–1541, 2018.

[2] A. Arnab, O. Miksik, and P. H. Torr, «On the robustness of semantic segmentation models to adversarial attacks», in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 888–897.

[3] V. Badrinarayanan, A. Kendall, and R. Cipolla, «Segnet: A deep convolutional encoder-decoder architecture for image segmentation», *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[4] D. Bolya, C. Zhou, F. Xiao, and Y. J. Lee, «Yolact: Real-time instance segmentation», in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 9157–9166.

[5] H. Caesar, J. Uijlings, and V. Ferrari, «Coco-stuff: Thing and stuff classes in context», in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 1209–1218.

[6] M. Cordts *et al.*, «The cityscapes dataset for semantic urban scene understanding», in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3213–3223.

[7] Z. Keita, *An introduction to convolutional neural networks (cnns)*, Available: https://www.datacamp.com/tutorial/introduction-to-convolutional-neural-networks-cnns, [Accessed July 18, 2024].

[8] R. Didero and G. M. Conti, «Capable: Engineering, textile, and fashion collaboration, for citizens' awareness and privacy protection», *Human Factors for Apparel and Textile Engineering*, vol. 32, p. 39, 2022.

[9] A. Dosovitskiy *et al.*, «An image is worth 16x16 words: Transformers for image recognition at scale», *arXiv preprint arXiv:2010.11929*, 2020.

[10] A. Du *et al.*, «Physical adversarial attacks on an aerial imagery object detector», in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 1796–1806.

[11] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, «The pascal visual object classes (voc) challenge», *International journal of computer vision*, vol. 88, pp. 303–338, 2010.

[12] I. J. Goodfellow, J. Shlens, and C. Szegedy, «Explaining and harnessing adversarial examples», *arXiv preprint arXiv:1412.6572*, 2014.

[13] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[14] A. Guesmi, M. A. Hanif, B. Ouni, and M. Shafique, «Physical adversarial attacks for camera-based smart systems: Current trends, categorization, applications, research challenges, and future outlook», *IEEE Access*, 2023.

[15] K. He, X. Zhang, S. Ren, and J. Sun, «Deep residual learning for image recognition», in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[16] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, «Masked autoencoders are scalable vision learners», in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 16 000–16 009.

[17] J. Hendrik Metzen, M. Chaithanya Kumar, T. Brox, and V. Fischer, «Universal adversarial perturbations against semantic image segmentation», in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2755–2764.

[18] G. Jocher, A. Chaurasia, and J. Qiu, *Ultralytics YOLO*, version 8.0.0, Jan. 2023. [Online]. Available: https://github.com/ultralytics/ultralytics.

[19] H. Kumar, *Quick intro to semantic segmentation: Fcn, u-net and deeplab*, Available: https://kharshit.github.io/blog/2019/08/09/quick-intro-to-semantic-segmentation, [Accessed June 12, 2024].

[20] A. Kirillov *et al.*, «Segment anything», in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 4015–4026.

[21] A. Kurakin, I. Goodfellow, and S. Bengio, «Adversarial machine learning at scale», *arXiv preprint arXiv:1611.01236*, 2016.

[22] M. Lee and Z. Kolter, «On physical adversarial patches for object detection. arxiv», *arXiv preprint arXiv:1906.11897*, 2019.

[23] T.-Y. Lin *et al.*, «Microsoft coco: Common objects in context», in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, Springer, 2014, pp. 740–755.

[24] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, «Feature pyramid networks for object detection», in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125.

[25] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, «Focal loss for dense object detection», in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2980–2988.

[26] J. Long, E. Shelhamer, and T. Darrell, «Fully convolutional networks for semantic segmentation», in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[27] S. Minaee, Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz, and D. Terzopoulos, «Image segmentation using deep learning: A survey», *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 7, pp. 3523–3542, 2021.

[28] Y. Mirsky, «Ipatch: A remote adversarial patch», *Cybersecurity*, vol. 6, no. 1, p. 18, 2023.

[29] R. Mottaghi *et al.*, «The role of context for object detection and semantic segmentation in the wild», in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

[30] F. Nesti, G. Rossolini, S. Nair, A. Biondi, and G. Buttazzo, «Evaluating the robustness of semantic segmentation for autonomous driving against real-world adversarial patch attacks», in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 2280–2289.

[31] K. O'Shea and R. Nash, *An introduction to convolutional neural networks*, 2015. [Online]. Available: https://arxiv.org/abs/1511.08458.

[32] A. Radford *et al.*, «Learning transferable visual models from natural language supervision», in *International conference on machine learning*, PMLR, 2021, pp. 8748–8763.

[33] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, «You only look once: Unified, real-time object detection», in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[34] J. Redmon and A. Farhadi, «Yolov3: An incremental improvement», *arXiv preprint arXiv:1804.02767*, 2018.

[35] O. Ronneberger, P. Fischer, and T. Brox, «U-net: Convolutional networks for biomedical image segmentation», in *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, Springer, 2015, pp. 234–241.

[36] N. Shiledarbaxi, *Semantic vs instance vs panoptic: Which image segmentation technique to choose*, Available: https://analyticsindiamag.com/ai-mysteries/semantic-vs-instance-vs-panoptic-which-image-segmentation-technique-to-choose/, [Accessed July 4, 2024].

[37] K. Simonyan and A. Zisserman, «Very deep convolutional networks for large-scale image recognition», *arXiv preprint arXiv:1409.1556*, 2014.

[38] A. Vaswani *et al.*, «Attention is all you need», *Advances in neural information processing systems*, vol. 30, 2017.

[39] *Satellite imagery is helping governments to combat deforestation*, Available: https://analyticsindiamag.com/ai-mysteries/semantic-vs-instance-vs-panoptic-which-image-segmentation-technique-to-choose/, [Accessed July 4, 2024].

[40] C. Xie, J. Wang, Z. Zhang, Y. Zhou, L. Xie, and A. Yuille, «Adversarial examples for semantic segmentation and object detection», in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 1369–1378.

[41] Z. Zhang, S. Huang, X. Liu, B. Zhang, and D. Dong, «Adversarial attacks on yolact instance segmentation», *Computers & Security*, vol. 116, p. 102 682, 2022.

[42] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, «Scene parsing through ade20k dataset», in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 633–641.

[43] C. Yu, C. Gao, J. Wang, G. Yu, C. Shen, and N. Sang, «Bisenet v2: Bilateral network with guided aggregation for real-time semantic segmentation», *International journal of computer vision*, vol. 129, pp. 3051–3068, 2021.

[44] Y. Zhu, C. Miao, F. Hajiaghajani, M. Huai, L. Su, and C. Qiao, «Adversarial attacks against lidar semantic segmentation in autonomous driving», in *Proceedings of the 19th ACM conference on embedded networked sensor systems*, 2021, pp. 329–342.

[45] A. Nguyen and B. Le, «3d point cloud segmentation: A survey», in *2013 6th IEEE conference on robotics, automation and mechatronics (RAM)*, IEEE, 2013, pp. 225–230.