МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра «Прикладная математика и информатика»
(наименование)

01.04.02 Прикладная математика и информатика
(код и наименование направления подготовки)

Математическое моделирование
(направленность (профиль))

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ)

на тему «Модификация архитектуры модели обнаружения объектов в режиме реального времени»

Обучающийся А.А. Дьяченко
(Инициалы Фамилия) (личная подпись)

Научный руководитель канд.пед. наук, доцент, О.М. Гущина
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2022

Содержание

Введение

Быстрые темпы цифровизации, производимые в различных сферах жизнедеятельности в течение последнего десятилетия порождают необходимость в осуществлении тщательного контроля качества выполняемых процессов, проведении непрерывного анализа различных явлений и автоматизации сбора данных без. Все это возможно благодаря современным системам компьютерного зрения, полностью перенимающих на себя функцию глаз человека, либо оказывающих дополнительную помощь в критически важных задачах. Применение систем компьютерного зрения в космической, научной и медицинской сферах, а также в сфере безопасности предъявляет высокие требования к точности и быстродействию разрабатываемых решений. Все это стимулирует непрерывные исследования в поиске новых подходов к задаче обнаружения объектов на изображении и попытки модификаций и переосмысления уже существующих и широко используемых методов.

Актуальность данной работы обуславливается ростом потребности в более точных детекторах объектов для приложений реального времени, поскольку большая часть исследований, нацеленных на создание быстрых детекторов приводит к значительному снижению точности обнаружения объектов.

Целью данной выпускной квалификационной работы является модификация одного из современных детекторов объектов на изображении для улучшения показателей точности и скорости обнаружения.

Объектом исследования в данной работе являются детекторы объектов, пригодные для работы в приложениях реального времени. Предметом исследования является влияние структурных особенностей различных детекторов объектов на показатели точности и скорости обнаружения.

Гипотеза исследования заключается в предположении, что производительность детекторов объектов на изображении во многом зависит

от сверточной архитектуры, входящей в его состав, поэтому правильный выбор и конструирование различных частей детектора, состоящих из сверточных нейронных сетей, позволят значительно повысить точность и скорость обнаружения.

Для достижения поставленной цели необходимо выполнить следующие задачи:

- провести обзор и оценку существующих подходов к обнаружению объектов на изображении;
- провести структурный и сравнительный анализы современных детекторов объектов на изображении, пригодных для работы в режиме реального времени, с целью определения наилучшего из них;
- выявить сильные и слабые стороны наилучшего по результатам сравнения детектора, предложить способы его улучшения и оценить производительность модифицированного детектора.

Научная новизна данного исследования состоит в использовании подхода оптимизации сверточных архитектур, входящих в состав детектора, для повышения его производительности. Кроме того, в данном исследовании предлагается новая сверточная нейронная сеть для извлечения признаков из изображения.

Практическая значимость исследования состоит в разработке сверточной архитектуры, которую можно применять для улучшения производительности большинства существующих детекторов объектов на изображении.

Положения, выносимые на защиту:

- в ходе исследования была разработана новая сверточная нейронная сеть для выделения признаков на изображении, используемая для улучшения работы детектора YOLOR;
- модифицированные детекторы объектов, полученные в результате перестроения экстрактора признаков и сети уточнения признаков

YOLOR, демонстрируют лучшие результаты как в точности, так и в скорости обнаружения.

Магистерская диссертация состоит из введения, анализа классов методов распознавания объектов на изображении, сравнительного анализа детекторов объектов на основе сверточных нейронных сетей и их компонентов, модификации архитектуры детектора объектов YOLOR, оценки производительности модифицированного детектора и заключения.

В первом разделе проводится обзор подходов к распознаванию объектов на изображении и анализ методов, реализующие данные подходы.

Во втором разделе проводится анализ основных принципов функционирования детекторов на основе сверточных нейронных сетей, сравнительный анализ сверточных архитектур, входящих в состав детекторов, обзор основных детекторов объектов на изображении и их сравнительный анализ.

В третьем разделе производится выявление сильных сторон детектора YOLOR, предлагаются способы улучшения и описывается математическая модель детектора. Также производится оценка точности и скорости модифицированного детектора.

Полученные в ходе выполнения данной работы модификации детектора объектов на изображении демонстрируют наилучшие показатели точности обнаружения по сравнению с оригинальным детектором, а также обладают достаточной скоростью для работы в приложениях реального времени.

1 Анализ классов методов распознавания объектов на изображении

1.1 Обзор подходов к распознаванию объектов на изображении

Впервые, о компьютерном зрении начали задумываться в 50-х годах 20 века. В 1955 году профессор Оливер Селфридж опубликовал статью «Глаза и уши для компьютера» [13], в которой автор выдвигал теорию о создании компьютера, который бы смог распознавать звуки и изображения.

Процесс обнаружения объектов на изображении состоит в том, чтобы распознать признаки и на основе них принять решение о наличии объекта на изображении и его принадлежности к какому-либо классу [27].

Можно выделить два основных класса методов обнаружения объектов на изображении: на основе машинного обучения и на основе глубокого обучения.

Машинное обучение является подразделом искусственного интеллекта – области, занимающейся разработкой систем, которые смогли бы справляться с задачами, решение которых считается исключительно прерогативой человека [28]. Данный термин был предложен в 1959 году специалистом в области информатики и искусственного интеллекта Артуром Сэмюэлем.

Глубокое обучение представляет из себя набор алгоритмов машинного обучения, которые моделируют абстракции высокого уровня в данных с использованием архитектур, состоящих из нескольких нелинейных преобразований [14]. Глубокое обучение является одной из форм машинного обучения и впервые как термин был введен в 1986 году ученым в области информатики Риной Дехтер.

Разграничивая данные понятия, под машинным обучением понимают алгоритмы, которые могут настраиваться на основе обучающих данных без привлечения человека, в то время как глубокое обучение – это множество уровней таких же алгоритмов, где каждый уровень обеспечивает различную

интерпретацию данных [19]. Говоря о глубоком обучении, подразумеваются искусственные нейронные сети [3].

Среди методов обнаружения объектов на основе машинного обучения можно выделить следующие:

– метод Виолы-Джонса,

– масштабно-инвариантное преобразование признаков (SIFT),

– гистограмма ориентированных градиентов (HOG).

Следующий класс методов обнаружения объектов, использующий глубокое обучение, основывается на использовании искусственных нейронных сетей, а именно одной из наиболее популярной архитектуры - сверточной нейронной сети [24].

Стоит отметить, что помимо упомянутых методов обнаружения объектов на изображении, есть и более простые, основанные на простых операциях с матрицами, такие как:

– сопоставление с шаблонами,

– отслеживание BLOB-объектов.

Однако данные методы не имеют широкого практического применения, поскольку очень чувствительны к свойствам объекта на изображении и используются исключительно в задачах определенной специфики [27].

Таким образом, были рассмотрены особенности двух основных классов методов обнаружения объектов на изображении: на основе машинного обучения и на основе глубокого обучения.

1.2 Анализ методов распознавания объектов на основе машинного обучения

В 1999 году ученым Дэвидом Лоу был опубликован алгоритмом выявления признаков для обнаружения объектов на изображениях под названием масштабно-инвариантное преобразование признаков (SIFT) [4]. Данный метод в процессе обнаружения объектов на изображении использует

базу ключевых точек, выделенных в процессе обучения. Во время анализа изображения выделяются ключевые признаки, которые сравниваются с признаками в базе, после чего на основе распознанных признаков делается вывод о присутствии объектов на изображении [7].

Преимуществом данного метода является инвариантность к ориентации и равномерному масштабированию изображения, однако он не лишен и недостатков, таких как медленная скорость обработки изображений и низкая эффективность на устройствах с малой мощностью [1].

Еще один алгоритм распознавания объектов в 2001 году был разработан исследователями в области компьютерного зрения Полом Виолой и Майклом Джонсом. Он был назван методом Виолы-Джонса и основывался на применении признаков Хаара, бустинга и каскадных классификаторов [7]. Признаки Хаара, с помощью которых происходит поиск нужных объектов представлены на рисунке 1.

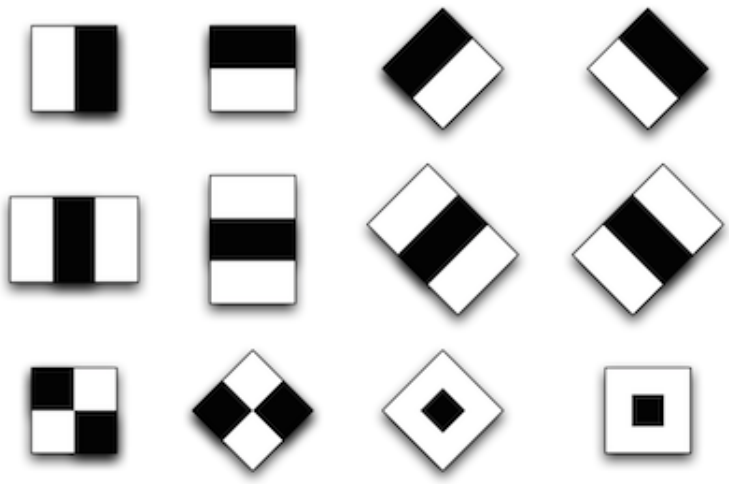


Рисунок 1 – Признаки Хаара

Для ускорения поиска схожих признаков, изображение представляется в интегральном формате. В результате применения данных признаковых шаблонов даже для самого маленького изображения может быть найдено огромное количество различных признаков, поэтому в методе Виолы-Джонса

было предложено использовать бустинг – алгоритм усиления классификаторов, который позволяет выбирать только те признаки, которые наилучшим образом описывают искомый объект [38]. Кроме того, те области изображения, которые не содержат признаков – вовсе отбрасываются.

Среди преимуществ метода Виолы-Джонса можно выделить высокую точностью и скоростью распознавания объектов, однако данный метод специализируется в основном на распознавании лиц и не подходит для обнаружения сложных объектов, имеющих различную форму. Кроме того, данный метод устойчив лишь к незначительным поворотам искомого объекта [7]. К примеру, если объект повернут более, чем на 30 градусов, то эффективность метода резко снижается.

В 2005 году французские исследователи в области компьютерных наук Навнит Далал и Билл Триггс предложили использовать гистограмму ориентированных градиентов (HOG) – дескрипторы особых точек для обнаружения объектов на изображении. Данный метод основывается на идее о том, что любой объект на изображении может быть описан с помощью направления краев и распределения градиентов эффективности [7]. Поэтому исследуемое изображение делится на несколько областей, для каждой из которых составляется гистограмма направлений градиента. Пример составления гистограмм направлений градиента на основе изображения представлен на рисунке 2.

Рисунок 2 – Пример составления гистограммы направлений градиента

Набор этих гистограмм называются дескрипторами, именно к ним и применяются классификаторы на конечном этапе распознавания объектов.

На сегодняшний день данный метод имеет множество различных модификаций, однако используется лишь для узкого круга задач, поскольку является неустойчивым к изменениям ориентации объектов и используется в основном для распознавания людей [7].

Таким образом, были описаны методы обнаружения объектов, основанные на машинном обучении. Все рассмотренные алгоритмы настраивают свои классификаторы, на основе множества обучающих примеров, другими словами, применяя метод обучения с учителем.

1.3 Анализ методов распознавания объектов на основе глубокого обучения

Еще одно направление обнаружения объектов на изображении, основанное на принципе работы мозга и зрительной коры человека, зародилось еще в 1958 году, когда психолог Фрэнк Розенблатт спроектировал упрощенную модель биологической нейронной сети головного мозга человека под названием персептрон [1]. Модель персептрона представлена на рисунке 3.

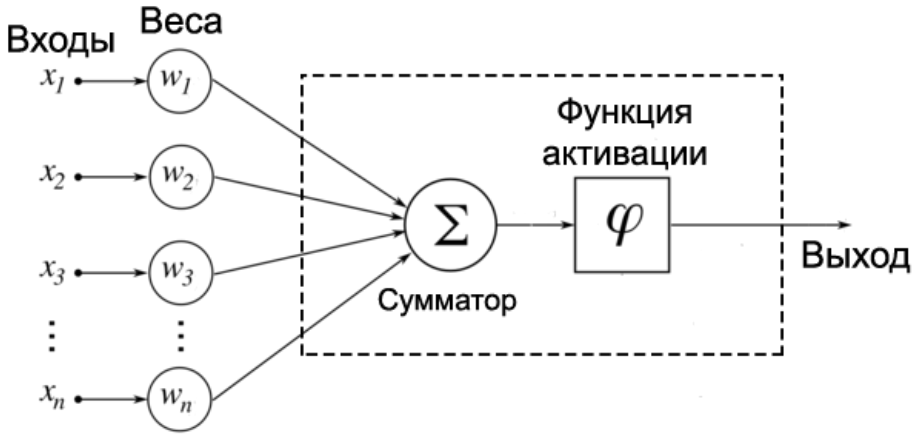


Рисунок 3 – Модель персептрона

Между входом или иначе входными нейронами, получающими сигнал, и сумматором находятся связи, каждая из которых имеет свой весовой коэффициент, определяющий значимость соответствующего входа для формирования выходного значения [2]. Сумматор вычисляет взвешенную сумму посредством сложения входных сигналов, помноженных на свои весовые коэффициенты. Данная сумма подается в функцию активации, которая производит нелинейное преобразование и определяет выходное значение нейрона [25]. Другими словами, она производит нормализацию проходящего через нее сигнала и представляет результат работы нейронной сети в нужном диапазоне, в зависимости от типа функции активации [2].

Именно на основе персептрона в 1960 году был построен первый нейрокомпьютер «Марк-1». Изначально данный компьютер предназначался для классификации визуальных образов и представлял собой электронно-механическую систему, в её основе лежали 400 управляемых фото-сенсоров, которые служили моделью сетчатки [3].

Позже для распознавания изображений пытались использовать более сложные архитектуры искусственных нейронных сетей, состоящих из множества нейронов, сгруппированных по слоям, однако проблема заключалась в том, что для распознавания даже небольшого изображения было необходимо произвести огромное количество вычислений, поскольку все нейроны в соседних слоях были связаны между собой [30]. Данная проблема была решена французским ученым в области информатики Яном Лекуном, который в 1988 году предложил специальную архитектуру искусственных нейронных сетей для распознавания образов [24]. Данная архитектура называется сверточной нейронной сетью (СНС) и является одной из технологий глубокого обучения.

Концепция СНС заимствована из принципа функционирования зрительной коры головного мозга, в которой отдельные участки клеток реагируют на возбуждение только в своих областях поля зрения, называемых рецептивными полями [30]. Восприятие различных образов влечет за собой

активацию конкретных групп нейронов зрительной коры [6]. Построенная на основе этих принципов архитектура СНС позволяет обнаруживать во входных данных различные признаки, начиная от очень простых и заканчивая более сложными.

Классическая модель СНС представлена на рисунке 4.

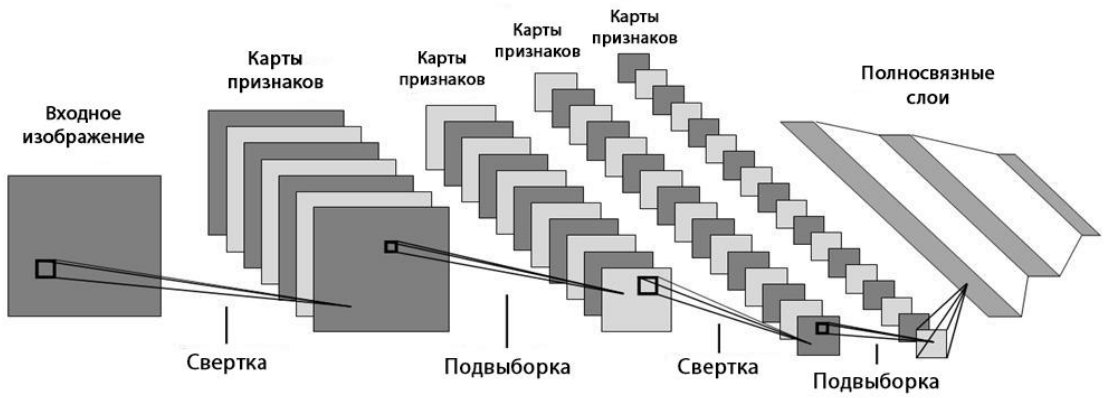


Рисунок 4 – Структура сверточных нейронных сетей

На данном рисунке представлена модель СНС, которая способна классифицировать объекты, находящиеся на изображении. В качестве ее основных частей выступают операции свертки, подвыборки и полностью связанные слои.

С помощью операции свертки конкретное изображение проходит через группу нейронов, каждый из которых отвечает за обнаружение определенного признака [3]. Иными словами, данная операция отражает реакцию отдельного нейрона на конкретную область поля зрения. Математически этот процесс можно описать как поэлементное умножение матрицы нейрона на подматрицы входного изображения аналогичного размера. Элементы каждой матрицы, полученной в результате умножения, суммируются и становятся значением новой матрицы, называющейся картой признаков [24]. Под матрицей изображения в данном случае подразумевается матрица значений интенсивности пикселей входного изображения. Чаще всего на вход СНС

поступает сразу 3 матрицы, каждая из которых отвечает за интенсивность конкретного цветового канала в соответствии с аддитивной моделью RGB. Под матрицей нейрона или иначе ядром свертки подразумевается матрица с весовыми коэффициентами, которые подлежат оптимизации в процессе обучения модели СНС [24]. Каждое значение карты признаков, полученной в результате выполнения операции свертки, отображает присутствие определенного признака в соответствующем месте исходного изображения. Операция свертки изображена на рисунке 5.

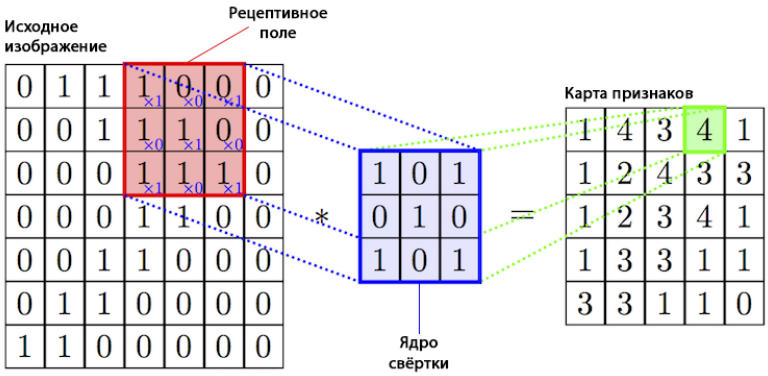


Рисунок 5 – Операция свертки

Представленный рисунок демонстрирует операцию свертки между одной картой признаков и одним ядром свертки. Если на вход сверточному слою поступает множество карт признаков, то ядро свертки применяется к каждому из них, а результирующие карты признаков складываются [6]. Для обнаружения нескольких признаков используют множество ядер свертки. Размер ядра свертки позволяет регулировать размер области анализируемого рецептивного поля и обычно составляет 3-7 пикселей [3]. В результате применения одного сверточного слоя с несколькими сверточными фильтрами к изображению получают набор карт признаков, являющихся входными данными для последующих слоев. Количество матриц, входящих в состав карт признаков называется глубиной [3].

Помимо размера ядра существуют и другие параметры свертки, такие как размер нулевого заполнения, шаг и межпиксельное заполнение. Оперируя данными параметрами, можно производить различные операций над картами признаков, например масштабирование или увеличение глубины [14]. Пример применения различных параметров свертки представлен на рисунке 6.

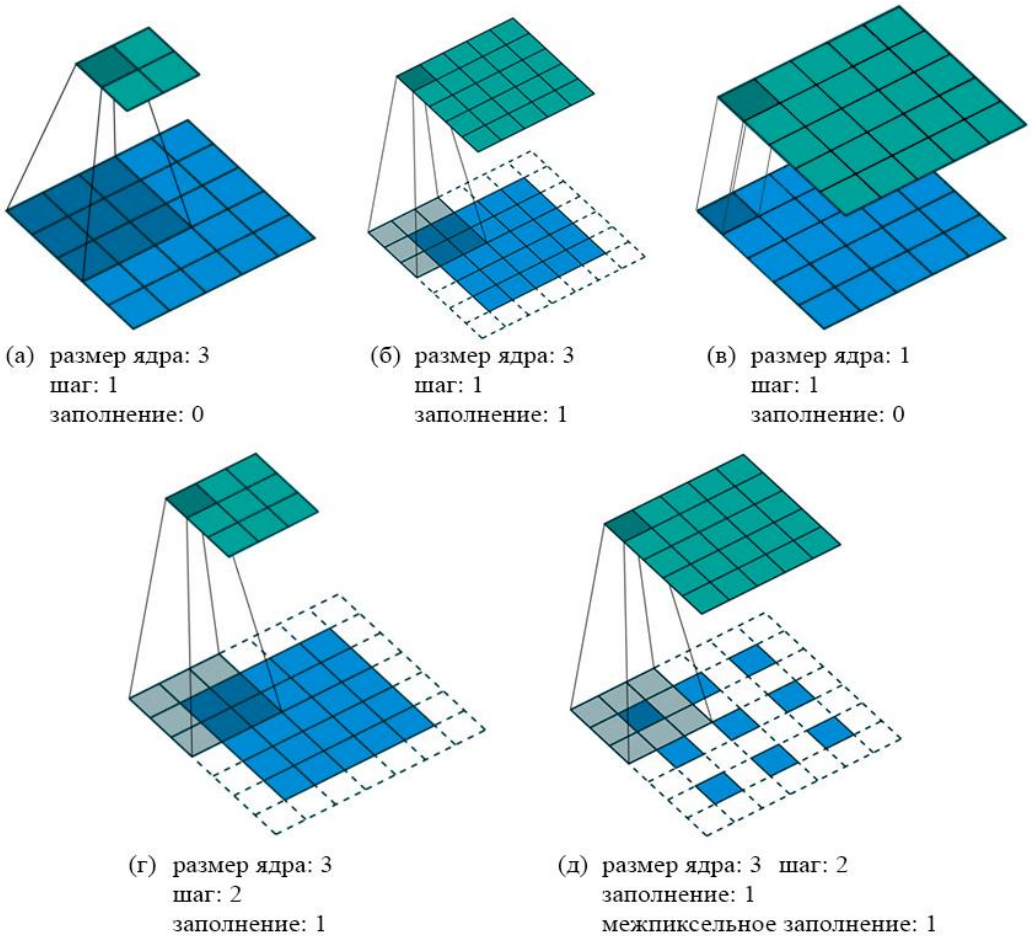


Рисунок 6 – Влияние параметров свертки на результат операции

Свертка (а) из рисунка 6 представляет из себя самый простой вариант операции свертки, аналогично представленной на рисунке 5.

Свертка (б) демонстрирует введение параметра нулевого заполнения, определяющего толщину рамки со значением пикселей равным 0. Данный параметр используется тогда, когда необходимо сохранить масштаб входных

карт признаков. Кроме того, нулевое заполнение позволяет сохранить информацию о граничных признаках [39].

Свертка (в) называется единичной или проекционным сверточным слоем и может масштабировать глубину карт признаков, следовательно, количество фильтров единичной свертки определяет количество выходных карт признаков. Кроме того, данная свертка производит канальное объединение, что в совокупности с уменьшением глубины позволяет сократить количество вычислений для последующих сверточных слоев, сохраняя при этом характерные особенности всех входных карт признаков [24]. Тот факт, что после каждой свертки, в том числе и единичной, карты признаков проходят через функцию активации, повышает пользу от ее использования благодаря внесению дополнительного нелинейное преобразование в модель [39]. Данный вид свертки используется практически во всех глубоких сверточных архитектурах.

Свертка (г) аналогична свертке (б), однако использует вдвое больший шаг, поэтому результирующая карта признаков имеет меньший размер. Операция свертки с увеличенным шагом в совокупности с нулевым заполнением во многих сверточных архитектурах заменяет операцию подвыборки для уменьшения масштаба карт признаков [22].

Свертка (д) называется деконвалюционным слоем и вводит параметр межпиксельного заполнения, который позволяет увеличивать масштаб карты признаков. Данный вид свертки зачастую используется вместо слоев повышения дискретизации, которые увеличивают изображение путем заполнения некоторой области новой карты признаков значением одного пикселя из исходной карты признаков. Деконвалюционный слой составляет основу множества архитектур пирамидальных сетей уточнения признаков [35].

Подобно полносвязным нейронным сетям, в СНС также присутствует смещение – изменяемый весовой коэффициент, значение которого прибавляется к каждому элементу результирующей матрицы [24].

Каждая операция свертки сопровождается функцией активации. Без данной функции любая архитектура нейронной сети представляла бы из себя обычную модель линейной регрессии, именно она позволяет выполнять нелинейное преобразование для входных данных [24]. Существует большая разновидность функций активации. Основные из них представлены на рисунке 7.

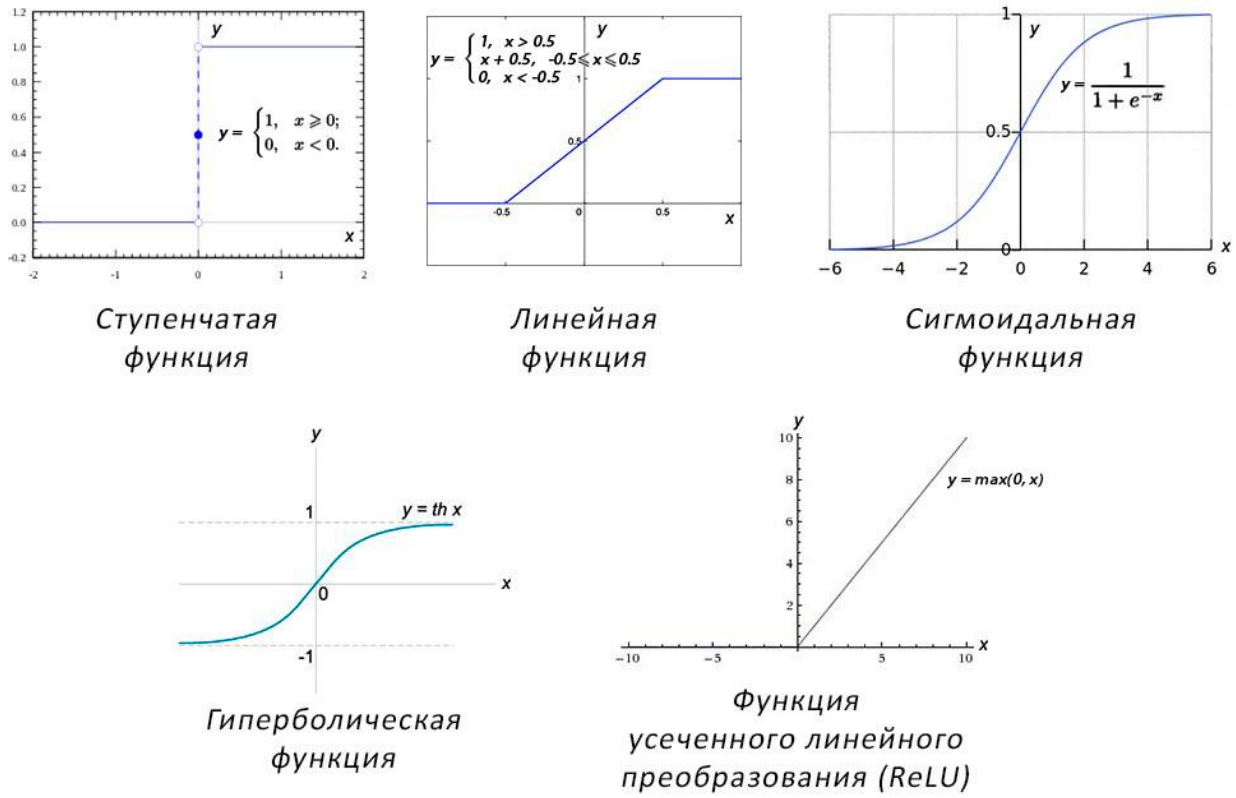
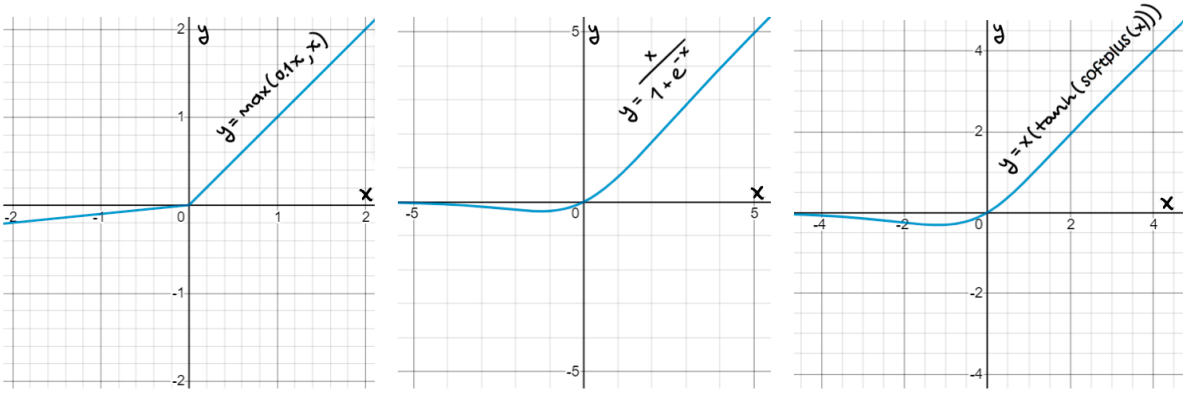


Рисунок 7 – Основные функции активации

Среди всех функций активации, представленных на рисунке 7, наиболее используемой, является функция активация ReLU, поскольку она является простой в реализации и обеспечивает быстрое вычисление производной, что необходимо в процессе оптимизации весовых коэффициентов. Главным недостатком данной функции является то, что для отрицательных или нулевых значений ReLU выдает нулевое значение, что приводит к прекращению обучения весовых коэффициентов, стоящих раньше нее [21]. Данную

проблему позволяют решить модифицированные функции активации на основе ReLU, такие как Leaky ReLU, SiLU/Swish или Mish [22]. Данные функции активации представлены на рисунке 8.



Leaky ReLU SiLU/Swish Mish

Рисунок 8 – Функции активации, основанные на ReLU

Помимо сверточных слоев в СНС часто используются слои объединения или иначе слои подвыборки. Данный слой располагается после сверточного слоя и уменьшает пространственную размерность карты признаков путем замены некоторой области пикселей одним пикселем. Необходимость применения данного слоя объясняется тем, что при распознавании образов важна лишь информация о наличии конкретных признаков на изображении, а не их местоположение [12]. Поэтому каждый слой подвыборки сжимает карты признаков до менее подробных, состоящих только из доминирующих признаков. Кроме того, при уменьшении размера изображения происходит уменьшение количества настраиваемых параметров СНС и снижение вычислительной нагрузки [23]. Используется два типа подвыборки: по среднему значению, в котором находится среднее значение интенсивности области пикселей, и по максимальному значению, в котором область пикселей заменяется одним пикселем из этой области с максимальной интенсивностью. Подвыборка по максимальному значению является самым распространенным

вариантом объединения за счет более четкого выделения ключевых признаков и подавления лишних шумов на изображении [34]. Каждый слой подвыборки определяется двумя параметрами: размером ядра и размером шага, которые аналогичны параметрам свертки. Обычно пространственная размерность карт признаков в СНС не уменьшается более чем вдвое за одну операцию, поэтому оба параметра чаще всего принимают значение 2 [12]. Пример выполнения операции подвыборки приведен на рисунке 9.

Рисунок 9 – Пример выполнения подвыборки по максимальному значению

Для выявления сложных признаков на изображении, СНС содержат большое количество слоев. Таким образом, в результате поочередного применения сверточных слоев и слоев объединения к поступающим на вход картам признаков, их глубина и масштаб постоянно изменяются в зависимости от числа используемых ядер свертки и параметров ядер объединения соответственно.

В современных СНС операцию подвыборки часто заменяют операцией свертки с увеличенным шагом и с использованием нулевого заполнения, как было представлено сверткой (г) на рисунке 6. Это обосновывается тем, что использование свертки позволяет изучить определенные свойства изображения, которые могут быть утрачены при использовании подвыборки

[22]. Стоит отметить, что данный подход увеличивает количество обучаемых параметров.

В зависимости от решаемой задачи, СНС могут иметь разную архитектуру [7], во многих СНС для классификации объектов на изображении используются полносвязные слои, представленные на рисунке 4. Для этого последний полносвязный слой содержит столько нейронов, сколько классов объектов необходимо различать на изображении.

Для того, чтобы сверточная нейронная сеть правильно извлекала признаки из изображения и корректно классифицировала объекты на их основе, необходимо оптимизировать все ее весовые коэффициенты таким образом, чтобы ошибка была минимальной. Ошибка находится с помощью функции потерь, определяемой типом решаемой задачи, и позволяющей оценить на сколько качественно сделано предсказание сети для конкретных входных данных [36]. После нахождения общей ошибки сети используется метод обратного распространения ошибки. Суть данного метода заключается в нахождении частных дифференциалов по каждому весовому коэффициенту, что позволяет определить вклад каждого из этих весов в общую ошибку [38]. Вначале вычисляется ошибка выходного слоя и ее значение передается на слой, стоящий перед выходным. Данная операция повторяется для всех слоев, начиная от последнего и заканчивая первым, вычисляя производную потерь по весовым коэффициентам этих слоев. Найденная производная позволяет определить в каком направлении необходимо регулировать веса для уменьшения общих потерь. Данная операция производится на заранее размеченных данных до тех пор, пока ошибка не перестанет уменьшаться. В лучшем случае это будет означать нахождение глобального минимума функции потерь [14].

Несмотря на большое количество преимуществ, СНС также, как и другие методы распознавания объектов на изображении, не лишены недостатков. Неправильно спроектированная модель или плохой набор

обучающих данных может привести к переобучению модели и непригодности применения на реальных данных [3].

Таким образом, был проведен обзор особенностей одного из наиболее используемых методов распознавания объектов на основе глубокого обучения - сверточных нейронных сетей.

Вывод по разделу 1

Итак, в данном разделе были рассмотрены основные классы методов обнаружения объектов на изображении: на основе машинного обучения и глубокого обучения. Методы, основанные на машинном обучении, применяются в основном для узкого круга задач. Основными недостатками данных методов является низкая скорость распознавания и плохая обобщающая способность. Несмотря на это, данные методы широко использовались до развития вычислительных мощностей и теории искусственных нейронных сетей.

С развитием глубокого обучения стало возможным создание сложных технологических архитектур нейронных сетей, обладающих отличной производительностью и позволяющих решать широкий спектр задач. Современные сверточные нейронные сети представляют из себя именно такие архитектуры и являются наиболее подходящим методом решения задач распознавания объектов на изображении, хотя и не лишены проблем, связанных с тонкостями проектирования архитектуры сети и её обучения.

2 Сравнительный анализ детекторов объектов на основе сверточных нейронных сетей и их компонентов

2.1 Анализ основных принципов функционирования детекторов объектов на основе сверточных нейронных сетей

На сегодняшний день, наиболее распространенными детекторами объектов на изображении, основанными на СНС, являются: семейство R-CNN, R-FCN, YOLO, SSD, RetinaNet, RefineDet, YOLOR. Несмотря на различие архитектур данных детекторов, включая глубину сети и особенности обучения, они все работают по схожим принципам.

Все детекторы объектов на основе СНС разделяются на двухступенчатые и одноступенчатые [8].

Двухступенчатые детекторы производят обнаружение объекта в два прогона изображения через сеть: первый раз для определения потенциальных зон на изображении, в которых могут содержаться объекты и второй раз для классификации объектов, расположенных в этих зонах.

Первый этап основывается на алгоритмах выборочного поиска, методе EdgeBoxes или сети предположения регионов [32]. Данный этап производит регрессию координат ограничивающих рамок объектов и предсказывает оценку объектности для каждой предсказанной рамки, которая показывает на сколько хорошо ограничивающая рамка покрывает реальный объект [5].

Второй этап целиком состоит из групп сверточных и полносвязных слоев. Сверточные слои выделяют ключевые признаки изображения, а полносвязные слои на основе полученных признаков выполняют классификацию объектов, находящихся в ограничивающих рамках, предложенных на первом этапе [32]. Для этого находится распределение вероятностей по всем классам, которые должен классифицировать данный детектор и отбирается класс с самой большой вероятностью.

Первой частью любого одноступенчатого детектора объектов является прямая сверточная архитектура для выделения признаков, называемая экстрактором признаков. Качество и скорость выделения признаков напрямую зависит от архитектуры этой части сети. Чаще всего используются заранее обученные для классификации архитектуры, что позволяет быстрее обучать оставшиеся части детектора [8].

Вторая часть детектора, называемая шеей, присутствует исключительно в одноступенчатых детекторах и предназначена для уточнения и масштабирования признаков, полученных на предыдущем этапе [35]. Данная часть сети основывается на архитектуре сети пирамид признаков (FPN) или ее модификациях [8, 9, 20].

Последняя логическая часть одноступенчатого детектора производит обнаружение объекта с помощью нескольких групп сверточных слоев [12]. Первая группа сверточных слоев производит регрессию значений координат, определяющих ограничивающие рамки объектов на изображении, для их корректного выделения. Вторая группа слоев выполняет классификацию объектов, находящихся в этих ограничивающих рамках. Некоторые детекторы имеют третью группу сверточных слоев, предсказывающих оценку объектности, аналогичную используемой в двухступенчатых детекторах [5, 32]. Описанная логическая часть архитектуры часто называют головой детектора.

Оба типа рассмотренных детекторов объектов основываются на методе регрессии ограничивающей рамки, который позволяет ограничивать объекты на изображении [12]. Для этого все карты признаков, на которых происходит обнаружение объектов, покрываются группами ограничивающих прямоугольников разного масштаба и с разным соотношением сторон, называемыми якорями или якорными блоками [16] и представленными на рисунке 10.

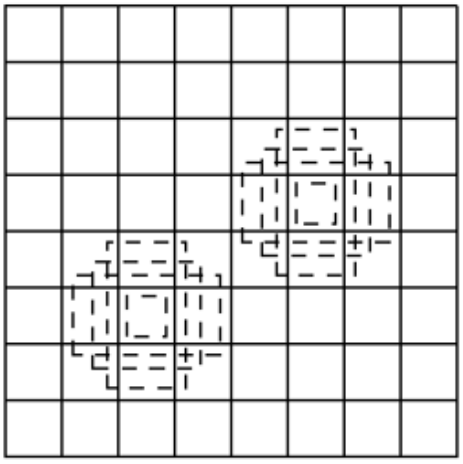


Рисунок 10 – Пример якорных блоков, используемых при обнаружении объектов

Именно для каждого из этих блоков производится предсказание смещений для координат ограничивающих рамок, классов объектов и оценки объектности.

Для определения достоверности положения ограничивающей рамки и нахождения оценки объектности используется метрика объединения по пересечению (Intersection over Union), которая позволяет рассчитать индекс пересечения двух областей [26]. Данная метрика определяется формулой (1).

$$IoU = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}, \quad (1)$$

где $|A \cap B|$ – площадь пересечения ограничивающих рамок A и B;

$|A \cup B|$ – площадь объединения ограничивающих рамок A и B.

В результате работы детекторов создается множество пересекающихся ограничивающих рамок для одних и тех же объектов, что мешает визуальному восприятию результата. Для удаления лишних рамок используется алгоритм подавления не-максимумов (non-maximum suppression) [26].

Подход данного алгоритма состоит в следующем:

– из множества всех ограничивающих рамок выбирается только одна рамка T, с наибольшей уверенностью в том, что данный объект принадлежит к конкретному классу;
– данная ограничивающая рамка рисуется на изображении и удаляется из множества всех рамок;
– находится индекс пересечения рамки T со всеми остальными рамками, рассчитываемый по формуле (1);
– те рамки, у которыми индекс пересечения с рамкой T превосходит пороговое значение, удаляются из множества всех рамок.

Данные действия повторяются до тех пор, пока множество рамок не останется пустым. В результате этого, на выходном изображении будут присутствовать только самые достоверные ограничивающие прямоугольники с наибольшей уверенностью в том, что объект внутри ограничивающей рамки принадлежит к конкретному классу [40].

Таким образом были рассмотрены основные принципы двухступенчатых и одноступенчатых методов обнаружения объектов на изображении, основанных на СНС.

2.2 Сравнительный анализ сверточных архитектур, входящих в состав детекторов объектов

Наибольшее влияние на результаты работы детектора объектов оказывают именно сверточные архитектуры, входящие в его состав. Как было сказано в предыдущем подразделе, основными сверточными архитектурами, выступающими в роли составных частей детектора являются экстрактор признаков, шея и голова детектора [26], две первые из которых имеют множество различных реализаций. Представители этих частей детектора используют различные типы слоев, их количество и расположение, а также разные конфигурации сверточных фильтров для достижения одной и той же цели. Исходя из этого, важно определить наиболее оптимальные архитектуры,

способные выступить в качестве строительных блоков точного, но в то же время быстрого детектора объектов [20].

Среди существующих экстракторов признаков наибольшей популярностью пользуются архитектуры VGG, ResNet, Darknet, DenseNet, DPN.

Архитектура VGG является очень простой, но весьма эффективной за счет подхода наращивания сверточных блоков одинаковой формы.

Архитектура ResNet подобно VGG использует стратегию использования блоков с одинаковой структурой, добавляя к ним остаточные соединения, что позволяет решить проблему исчезновения градиента при наращивании глубины сети [18].

Архитектуры ResNeXt и Res2Net повторяют блочную структуру ResNet, но изменяют структуру блоков. В архитектуре сети ResNeXt вместо последовательных сверток в блоке используются независимые стеки сверточных слоев, объединяющие результаты. Данная архитектура увеличивает точность, без изменения сложности и количества параметров [29]. В Res2Net строение блока представляет из себя иерархические остаточные соединения. Данная архитектура позволяет более детально изменять рецептивные поля, захватывая локальные и глобальные признаки [31].

Модель Darknet объединяет блочность и простоту VGG и остаточные соединения ResNet [5]. В архитектуре DenseNet внутри каждого из чередующихся блоков карты признаков, получаемые с каждого слоя, поступают на вход всем остальным слоям. Для их объединения применяется конкатенация. В результате, получается извлекать более разнообразные карты функций.

Архитектура DPN объединяет ResNet с DenseNet, что позволяет обеспечить возможность повторного использования функций и возможность более обширного исследования признаков [41].

Как видно из анализа различных экстракторов признаков, особенностью современных СНС является добавление в архитектуру различных связей с предыдущими слоями, а именно остаточных соединений (аналогично ResNet) и межэтапных частичных соединений (аналогично DenseNet). В обоих случаях ширина и высота объединяемых карт признаков должны совпадать [29, 31].

При объединении карт признаков с разных слоев через остаточные соединения происходит их поэлементное сложение. То есть при сложении двух карты признаков размерностью 32×32×512 получается точно такая же карта признаков 32×32×512. Данная операция способствует лучшему распространению ошибки на более ранние слои в процессе обучения [18].

При использовании межэтапных частичных соединений, производится конкатенация карт признаков с разных слоев [10]. Иначе говоря, конкатенации двух карт признаков размерностью 32×32×512 даст карту признаков 32×32×1024. Обычно эти соединения используются, когда только часть карт признаков проходит через группу слоев, а затем объединяется путем склеивания с другой частью. Это позволяет уменьшать вычислительную сложность сверточных архитектур, немного повышая точность [31].

В таблице 1 представлены характеристики рассмотренных экстракторов признаков: количество слоев, количество параметров, количество выполняемых операций с плавающей запятой (BFLOPs) и точность классификации на наборе данных ImageNet.

От количества обучаемых параметров зависит скорость обучения сети, то есть, чем их больше – тем дольше будет обучаться СНС [14]. Точно также зачастую от количества операций с плавающей запятой зависит скорость работы данной сети, чем этих операций меньше – тем меньше времени требуется для обработки одного изображения.

Таблица 1 - Значения основных характеристик популярных экстракторов признаков

Наименование модели	Количество слоев	Количество параметров, млн.	BFLOPs, млрд.	Точность (Топ-1)
VGG	16	138	15.3	74.4
	19	144	19.6	74.5
ResNet	50	23	3.8	75.8
	101	42	7.6	78.1
	152	58	11.3	78.6
ResNeXt	50	23	3.8	77.4
	101	42	7.6	78.8
	152	58	11.3	79.0
Res2Net	50	23	3.8	77.9
	101	42	7.6	79.2
	152	58	11.3	79.7
Darknet	19	50	7.29	72.9
	53	40	18.7	76.7
DenseNet	121	7.2	5.6	74.9
	169	12.8	6.7	76.2
	201	20	8.6	77.4
	264	33	11.5	77.8
DPN	92	37.8	6.5	78.7
	98	61.7	11.7	79.8

Так как основными оценочными критериями СНС являются скорость и точность классификации, сравнение данных показателей может помочь определить производительность модели. С этой целью была построена точечная диаграмма, представленная на рисунке 11.

Из приведенной диаграммы видно, что наиболее производительными экстракторами признаков являются архитектуры, использующие остаточные соединения и множественные комбинации объединения карт признаков. Среди таких архитектур DPN, Res2Net и ResNeXt.

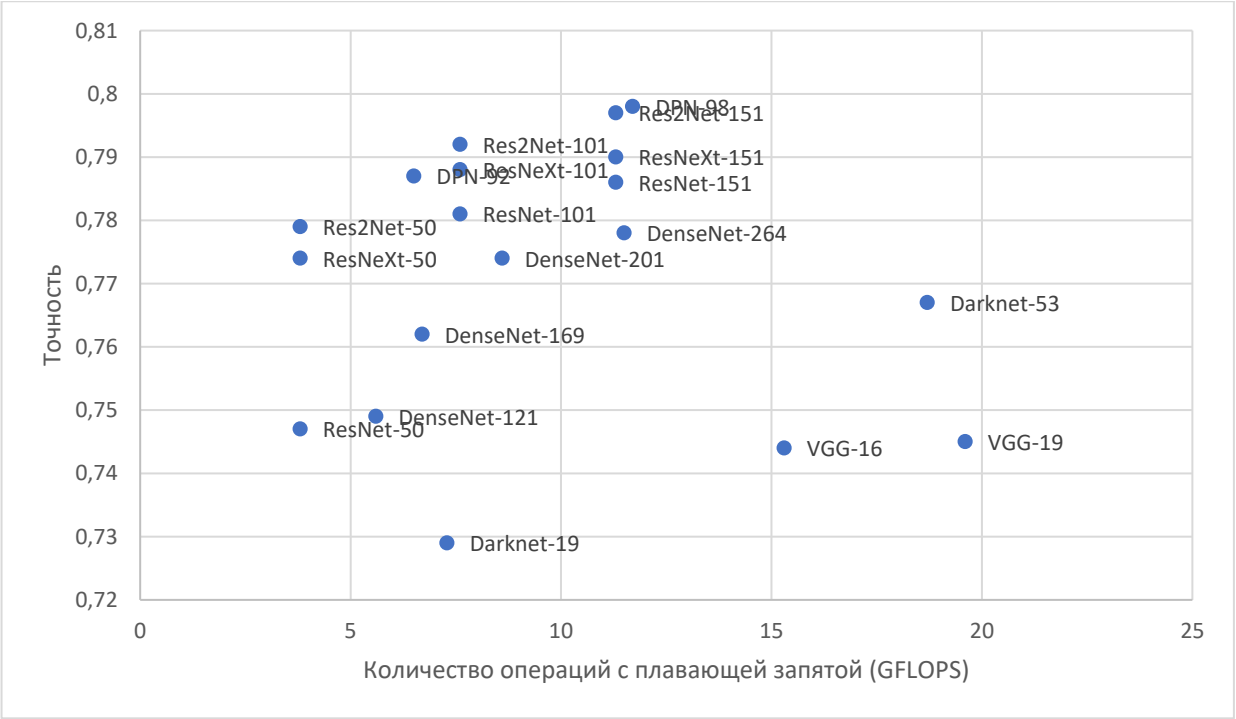


Рисунок 11 – Сравнение основных показателей экстракторов признаков

После этапа извлечения признаков следует их уточнение [26]. Самыми популярными шеями детекторов являются FPN, PAN, NAS-FPN, ASFF, BiFPN и TPN.

FPN (Feature Pyramid Networks) объединяет семантически сильные признаки низкого разрешения с семантически слабыми признаками высокого разрешения посредством нисходящего пути и боковых связей [35].

PAN (Path Aggregation Network) сокращает информационный путь и усиливает пирамиду признаков точными сигналами локализации с низких уровней посредством добавления к FPN восходящего расширения [8].

NAS-FPN (Neural Architecture Search - FPN) позволяет сети самостоятельно конструировать свою архитектуру для нахождения оптимальных соединений в заданном пространстве поиска.

ASFF (Adaptively Spatial Feature Fusion) позволяет решать проблему несогласованности объединяемых данных с помощью идентичного изменения масштаба и адаптивного объединения.

BiFPN (Bi-directional Feature Pyramid Network) обеспечивает более высокоуровневое объединение признаков за счет использования взвешенной двунаправленной пирамидальной сети признаков [20].

TPN (Trident Pyramid Networks) производит объединение признаков с помощью обработок на основе связи подобно BiFPN и самообработок, напоминающих структуру блока с остаточными соединениями [9].

Для сравнения производительности данных архитектур, было оценено влияние их внедрения на точность классификации и количество обучаемых параметров модели ResNet-50. Результаты представлены в таблице 2.

Таблица 2 - Влияние архитектур уточнения признаков на результат работы сети

Наименование архитектуры уточнения признаков	Увеличение параметров, млн.	Повышение точности в процентах
FPN	8	8
PAN	15	11
NAS-FPN	37.3	10
ASFF	13	10.3
BiFPN	11.7	12.6
TPN	14.1	11.9

Поскольку в задаче обнаружения объектов наиболее важным является именно показатель точности, для наглядности влияния каждой сети уточнения признаков на конечную точность модели ResNet-50 была построена линейчатая диаграмма, представленная на рисунке 12.

Из полученных результатов следует, что в качестве наиболее оптимальной шей детектора лучше всего подойдут архитектуры BiFPN и TPN. Оба этих подхода используют дополнительное восходящее расширение сети и подобие остаточных соединений, для объединения признаков одинаковой размерности [9, 20].

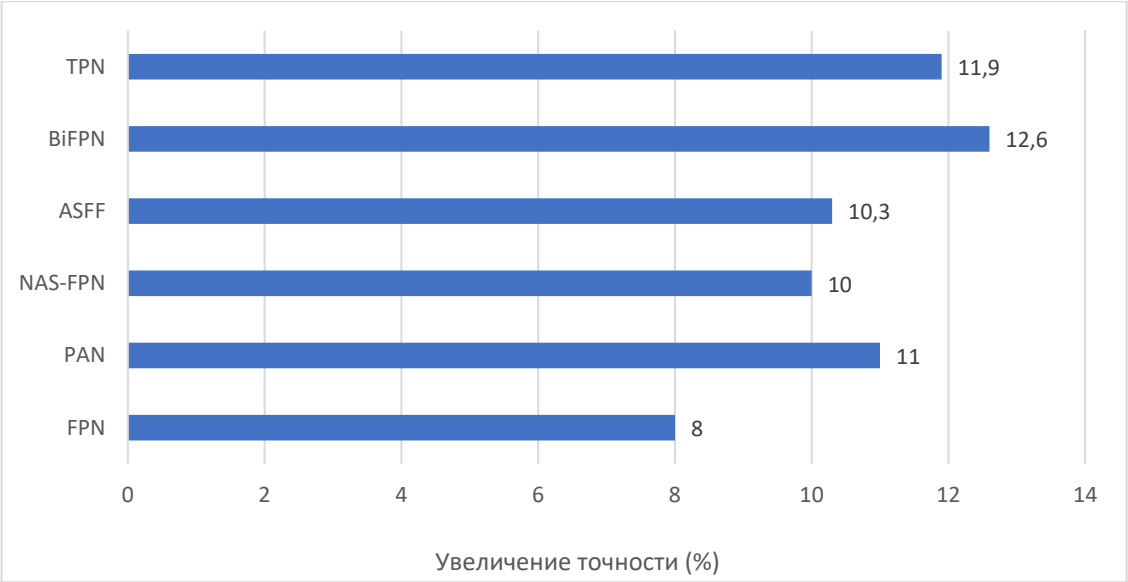


Рисунок 12 – Влияние сетей уточнения признаков на повышение точности модели ResNet-50

Таким образом, был проведен сравнительный анализ сверточных архитектур, входящих в состав детекторов объектов на изображении. Рассмотренные экстракторы признаков и сети уточнения признаков позволяют получать в совокупности мощные сверточные архитектуры для извлечения высокоуровневых карт признаков из изображения. Помимо разнообразного комбинирования сверточных архитектур, многие детекторы объектов добиваются значительного повышения показателей точности и скорости обнаружения именно благодаря дополнительным особенностям в их реализации.

2.3 Обзор современных детекторов объектов на изображении на основе сверточных нейронных сетей

2.3.1 Обзор детектора Faster R-CNN

Семейство детекторов объектов R-CNN (Faster Region Based Convolutional Neural Network) включает в себя следующие реализации: R-CNN, Fast R-CNN, Faster R-CNN [32]. Последний из них является результатом

улучшения двух предыдущих. Структурная модель детектора Faster R-CNN приведена на рисунке 13.

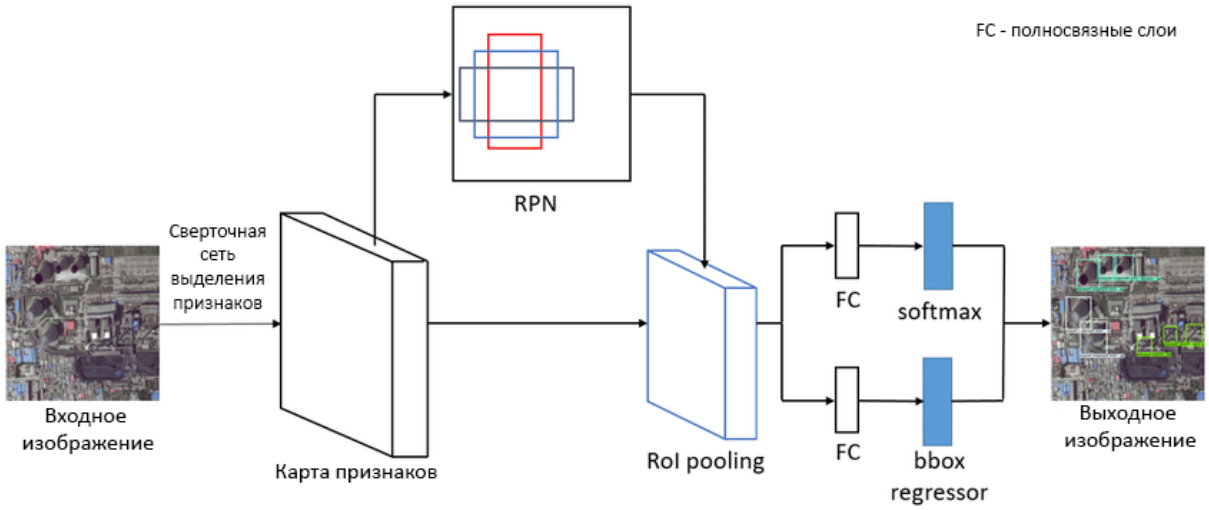


Рисунок 13 – Структурная модель детектора Faster R-CNN

В отличие от R-CNN и Fast R-CNN, в которых на этапе предложения регионов используются алгоритм выборочного поиска и алгоритм EdgeBoxes, в Faster R-CNN используется сеть предположений региона (RPN) [40]. Данная сеть анализирует исходное изображение с помощью собственных классификатора и регрессора, которые определяют потенциальные зоны, способные содержать искомый объект намного быстрее, чем остальные методы предложения регионов [32]. Принцип работы сети предположения регионов представлен на рисунке 14.

Для выделения признаков в данном детекторе зачастую используются неглубокие сети VGG-16 или ResNet-18. После выделения претендентов на роль объекта, координаты области претендента вместе с картами признаков изображения передаются на слой объединения областей интереса (Region of Interest pooling), который преобразует все неоднородные предположения в фиксированную форму с помощью операции max-pooling [32].

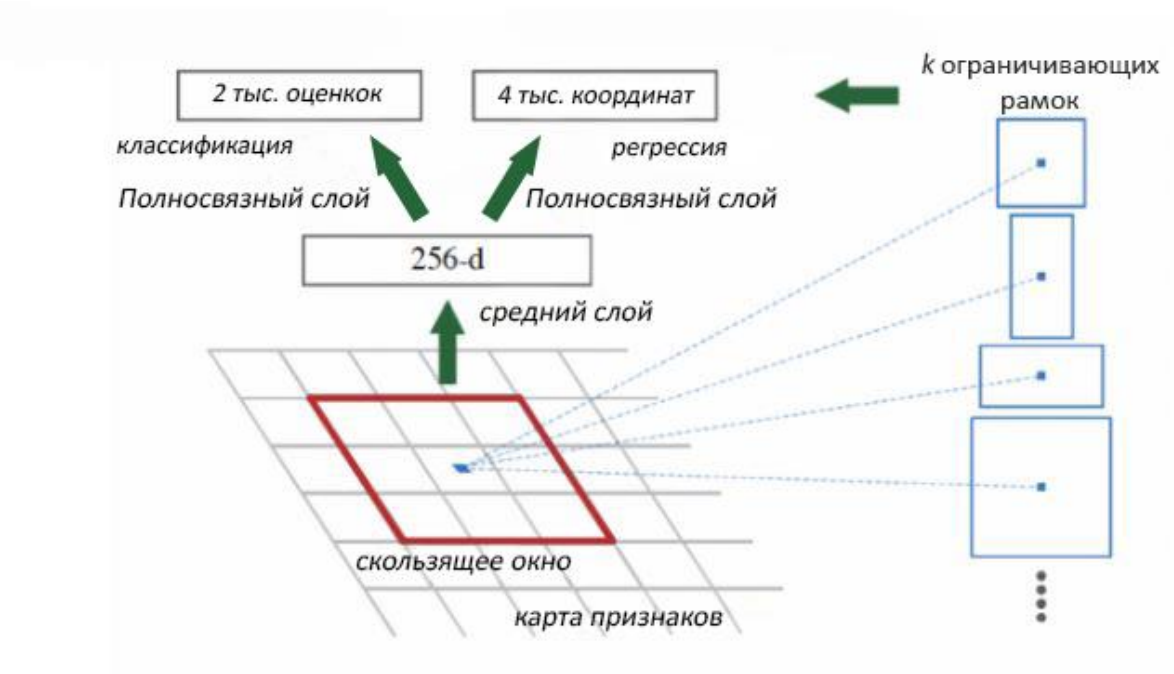


Рисунок 14 – Принцип работы сети предложения регионов (RPN)

Результатом этой операции является набор карт признаков фиксированного размера, соответствующих предложенным областям интереса. Более подробно операция объединения областей интереса представлена на рисунке 15.

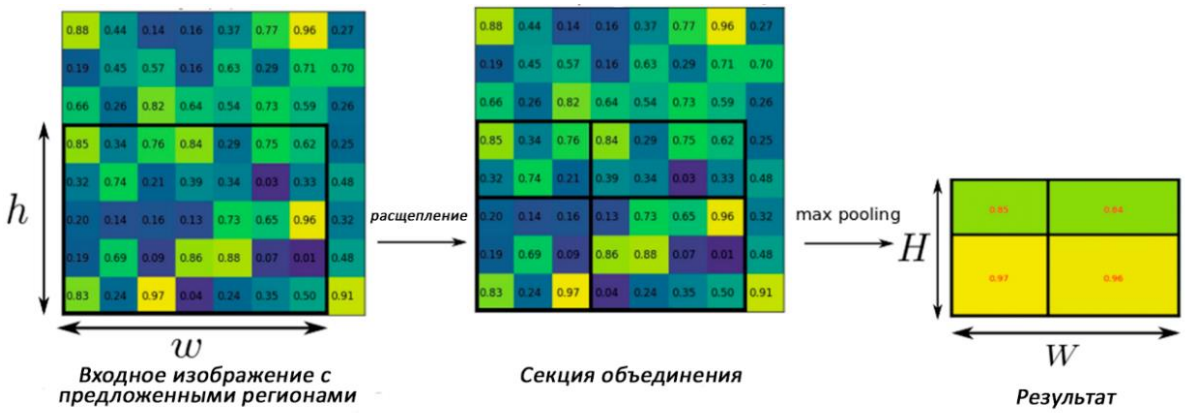


Рисунок 15 – Операция объединения областей интереса

Полученные после данного слоя карты признаков параллельно передаются на два полностью связанных слоя, первый из которых предсказывает класс объекта с помощью функции softmax, а второй уточняет координаты ограничивающего прямоугольника с помощью регрессии ограничивающей рамки [32].

Преимущества детектора Faster R-CNN:

– высокая точность обнаружения объектов,

– наилучший двухступенчатый детектор из семейства R-CNN.

Недостатком детектора Faster R-CNN является низкая скорость обнаружения, что не позволяет применять его в приложениях реального времени.

2.3.2 Обзор детектора R-FCN

Детектор R-FCN (Region-based Fully Convolutional Network) также является двухступенчатым и по логике работы схож с методом Faster R-CNN за исключением использования позиционно-зависимых карт оценок [16]. Различие между двумя данными подходами можно увидеть на структурной схеме R-FCN, представленной на рисунке 16.

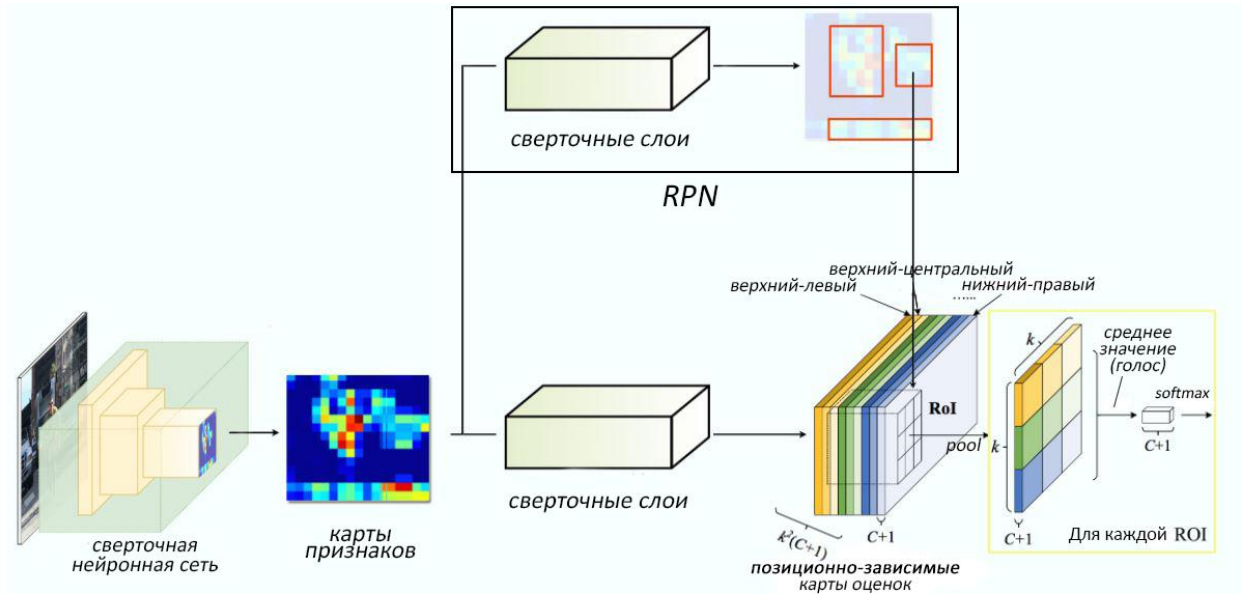


Рисунок 16 – Структурная модель детектора R-FCN

Как видно из рисунка 16, левая часть детектора R-FCN до операции объединения областей интереса (RoI) повторяет архитектуру сети Faster R-CNN, однако после этапа RPN для каждого класса объектов выполняется k^2 операций свертки, где k – это ширина (высота) карты оценок. Каждая ячейка в позиционно-зависимой карте оценок представляет вероятность того, что соответствующий ей регион в области интереса соответствует части объекта, расположенного в данной области [16]. Более подробно операция получения позиционно-зависимой карты оценок представлена на рисунке 17.

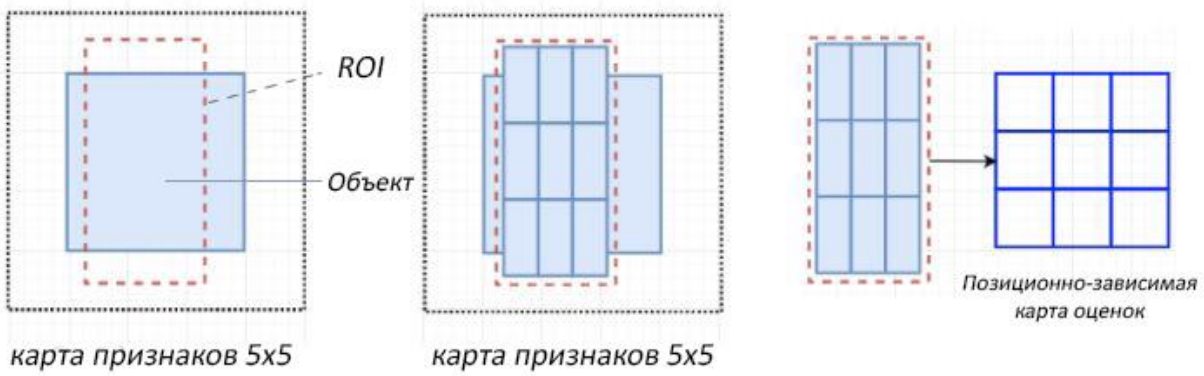


Рисунок 17 – Получение позиционно-зависимой карты оценок

Синий прямоугольник представляет собой объект, расположенный на карте признаков размером 5x5, в то время как пунктирной линией обозначена область интереса. Данная область интереса делится на несколько частей, после чего рассчитываются вероятности совпадения частей области интересов с частями изображения, а результат помещается в позиционно-зависимую карту оценок, являющуюся матрицей. Далее находится среднее значение по данной матрице, называемое голосом. В результате применения данной операции для каждого класса рассчитывается одномерный вектор голосов, который передается в функцию активации softmax для определения метки класса объекта [16].

Для уточнения местоположения объекта на изображении выполняется регрессия ограничивающего прямоугольника, не зависящая от класса. После сверточного слоя k×k×(C+1) добавляется новый сверточный слой 4×k×k. Далее выполняется позиционно зависимое объединение (RoI pooling), для каждой области интереса создается вектор размером 4×k×k [16]. После выполнения среднего голосования данный вектор преобразуется в значения координат tx, ty, tw и th, описывающих положение и размер ограничивающей рамки [12].

В качестве сети извлечения признаков данный детектор использует архитектуру ResNet-101.

Преимуществом детектора R-FCN является сохранение точности обнаружения при увеличении скорости работы по сравнению с Faster R-CNN за счет отсутствия полносвязных слоев.

Недостатком детектора R-FCN является то, что он все еще недостаточно быстрый для работы в режиме реального времени за счет использования этапа предложения регионов.

2.3.3 Обзор детектора YOLOv4

Детекторы объектов на изображении из семейства YOLO (You Only Look Once) являются одноступенчатыми и в отличие от двухступенчатых методов, изображение, поступающее на вход, анализируется только один раз за счет исключения этапа предложения регионов [5]. Детектор YOLO считается самым первым одноэтапным детектором объектов. К данному семейству относят YOLO, YOLOv2, YOLOv3 и YOLOv4. Все следующие за YOLO версии не изменяют базовую концепцию оригинального детектора, а лишь дополняют или изменяют некоторые структурные части с целью улучшения его скорости и точности [5]. Самой лучшей на сегодняшний день является версия YOLOv4, первая из которая обладает высокими показателями скорости и точности обнаружения. Полная структурная модель детектора YOLOv4 представлена на рисунке 18.

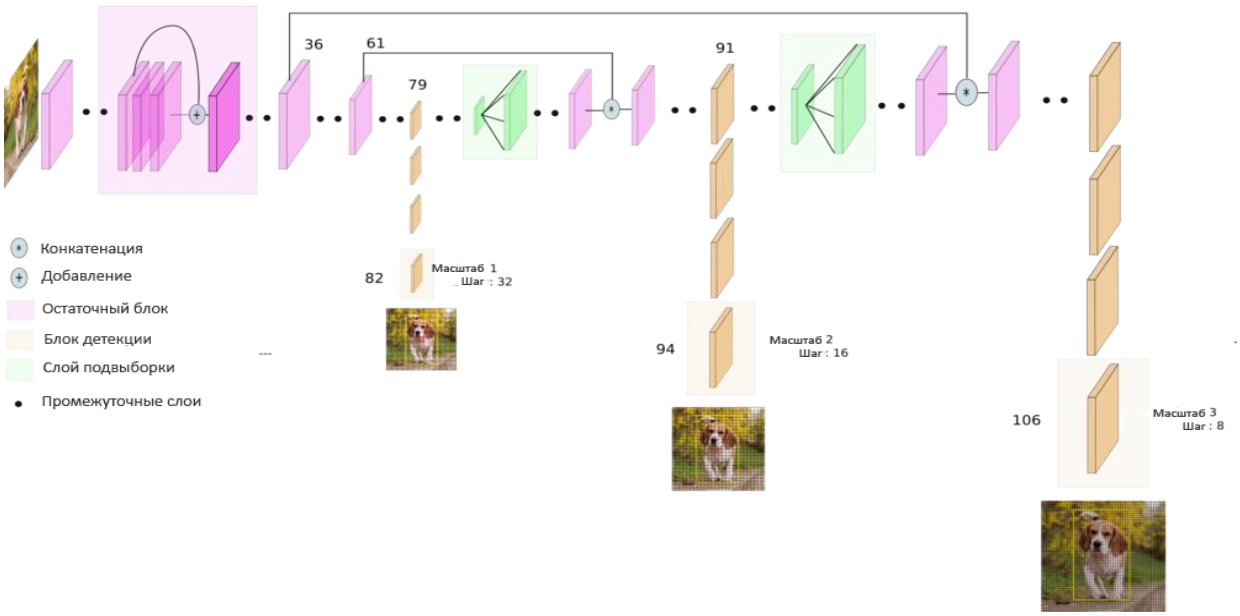


Рисунок 18 – Структурная модель детектора YOLOv4

В качестве экстрактора признаков в данной архитектуре используется CSPDarknet53. Приписка CSP (Cross-Stage-Partial-connections) говорит о том, что в сверточной нейронной сети Darknet53 используются межступенчатые частичные соединения [10]. Это означает, что входные данные перед поступлением в блок сверточных слоев каждый раз разделяются на 2 части, первая из которых проходит через данный блок, а вторая становится частью ввода для следующего переходного блока. Данная модификация позволяет снизить вычислительную сложность за счет разделения входных данных на две части [10].

Структура YOLOv4 также может быть представлена в виде сети пирамид признаков (Feature Pyramid Networks), содержащую восходящий и нисходящий пути [35]. Первая пирамида представляет из себя СНС для выделения признаков, где посредством выполнения операций свертки и подвыборки (max pool) уменьшается пространственная размерность карт признаков. Вторая пирамида, представляющая собой сеть уточнения признаков, увеличивает пространственную размерность карт признаков и производит их конкатенацию (перемножение) с картами признаков

соответствующего масштаба из экстрактора признаков [35]. Данная структура представлена на рисунке 19.

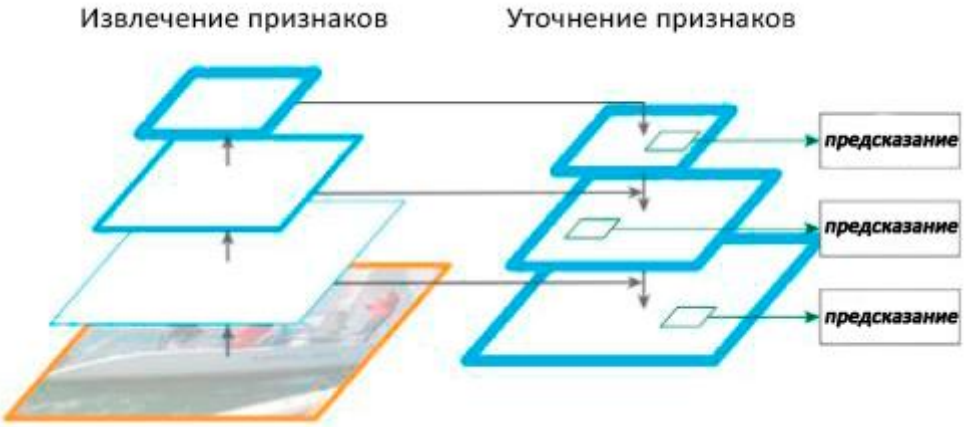


Рисунок 19 – Сеть пирамид признаков, используемая в YOLOv4

Данная структура обосновывается тем, что верхние слои сверточной нейронной сети содержат в себе большое количество семантической информации, но имеют слишком маленькое разрешение, а нижние слои – наоборот. Именно поэтому для того, чтобы избежать потери важной информации в результате обобщения и сжатия, производится изменение размерности вышележащих карт признаков до размера нижележащих и их конкатенация [35]. Это помогает улучшить обнаружение небольших объектов. На полученных, путем конкатенации, картах признаков, производится обнаружение объектов [5]. Функциональной пирамидальной сети соответствуют слои с 36 по 106.

Отличие функциональной пирамидальной сети в YOLOv4 от оригинальной FPN состоит в том, что в данной модели происходит именно конкатенация карт признаков одинаковой размерности, а не их объединение[5]. Кроме того, перед передачей карты признаков в детектор, несколько раз производится операция сверти с единичным ядром, что способствует уточнению признаков, извлеченных на предыдущих этапах.

Данная модель имеет 3 головы детектора для обнаружения объектов в трех различных масштабах, уменьшая изображение в 32, 16 и 8 раз соответственно. Еще одной особенностью YOLOv4 является то, что в отличии от остальных детекторов масштабы и соотношения сторон для якорных блоков, которыми покрываются карты признаков в процессе обнаружения объектов, рассчитывается с помощью алгоритма классификации k-NN на основе анализа обучающей выборки [5]. Каждая из голов детектора содержит три группы сверточных слоев: для регрессии смещений координат ограничивающей рамки, классификации объекта, находящегося в ней и определения объектности этой же рамки [39]. Последним этапом с помощью алгоритма подавления не-максимумов удаляются лишние ограничивающие рамки.

Преимуществом детектора YOLO является высокая скорость и точность обнаружения объектов.

Одним известным недостатком ранних детекторов YOLO является плохая способность обнаружения мелких объектов, однако авторы YOLOv3 утверждают, что данная проблема сведена к минимуму в их реализациях. В более поздних версиях данная проблема не наблюдалась.

2.3.4 **Обзор детектора SSD**

Детектор объектов SSD (Single Shot Multibox Detector) появился в результате исправления недостатков первых двух версий YOLO, поэтому их принцип работы слегка похож [27]. В качестве СНС для выделения признаков в данной модели используется VGG16. Подобно YOLO, детектор SSD производит обнаружение объектов в разных масштабах, однако в SSD сеть уточнения признаков заменена несколькими сверточными блоками, в которых отсутствует объединение карт признаков с различных этапов [37]. Структурная модель SSD представляет из себя глубокую сверточную нейронную сеть, постепенно уменьшающую пространственную размерность карт признаков. Данная структурная модель представлена на рисунке 20.

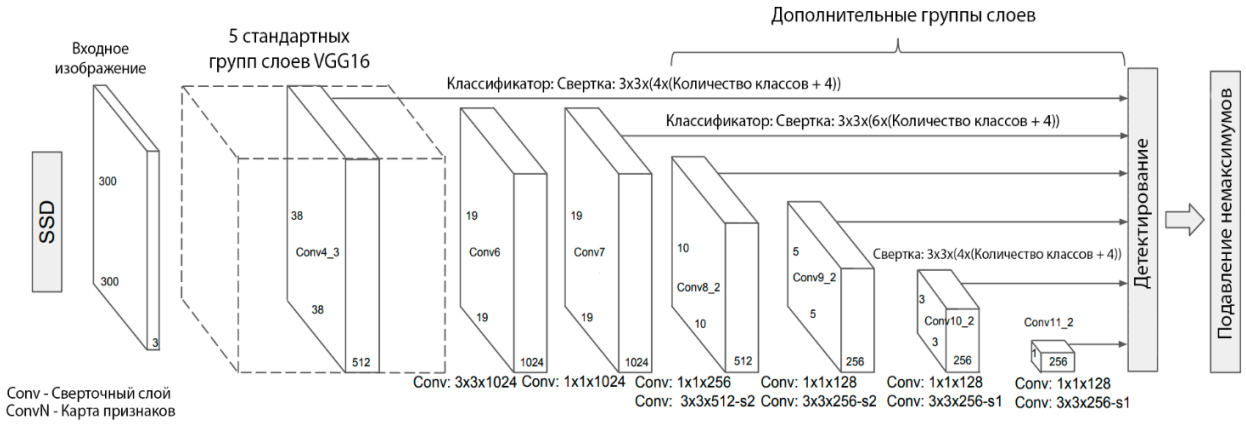


Рисунок 20 – Структурная модель детектора SSD

Как видно из рисунка, для обнаружения объектов используется всего 6 масштабов карт признаков: 38×38 и 19×19, получаемые с четвертой группы слоев оригинальной сети VGG16 и седьмого полносвязного слоя, преобразованного в сверточный слой, а также 10×10, 5×5, 3×3 и 1×1, получаемые с дополнительных групп сверточных слоев. На первых слоях происходит обнаружение мелких объектов, а на последующих – более крупных [37]. Аналогично YOLOv4, полученные с различных сверточных слоев карты признаков передаются в детектор и покрываются набором якорных блоков различного масштаба и с разным соотношением сторон для обнаружения объектов [12], однако в SSD размеры этих прямоугольников не подгоняются с помощью алгоритма классификации, а являются фиксированными. Голова детектора объектов детектора SSD состоит из двух групп сверточных слоев (для классификации и регрессии), с помощью которых для каждой рамки предсказывается распределение вероятности по меткам классам и смещения по 4 координатам ограничивающей рамки объекта [37]. Таким образом делается порядка 8732 детекций. Голова детектора SSD представлена на рисунке 21.

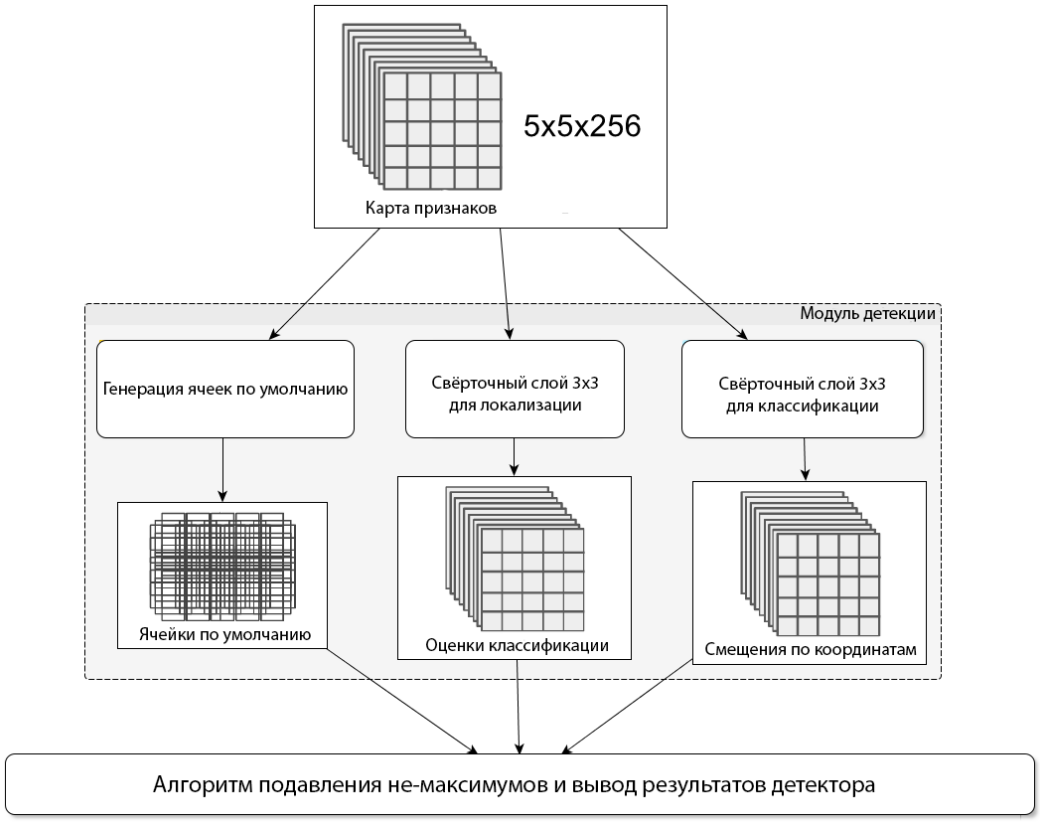


Рисунок 21 – Голова детектора SSD

Для того чтобы удалить дублирующие ограничивающие рамки и оставить только лучшие, так же, как и в YOLOv4 используется алгоритм подавление не-максимумов [37].

Преимуществами детектора SSD является быстрая обучаемость. Кроме того, он достигает хорошего баланса между точностью и быстродействием.

В качестве недостатка данного детектора отмечено плохое качество обнаружения мелких объектов.

2.3.5 Обзор детектора Retina-Net

Детектор Retina-Net, вдохновленный YOLOv3 и SSD, попытался объединить в себе конструктивные особенности этих детекторов для получения более точного инструмента обнаружения объектов [36]. Структурно Retina-Net сильно напоминает YOLOv4, однако у них есть множество различий на уровне конфигураций сверточных блоков, используемых функций активаций и функций ошибки. Кроме всего прочего,

Retina-Net использует сеть извлечения признаков ResNet-101. Структурная модель детектора Retina-Net представлена на рисунке 22.

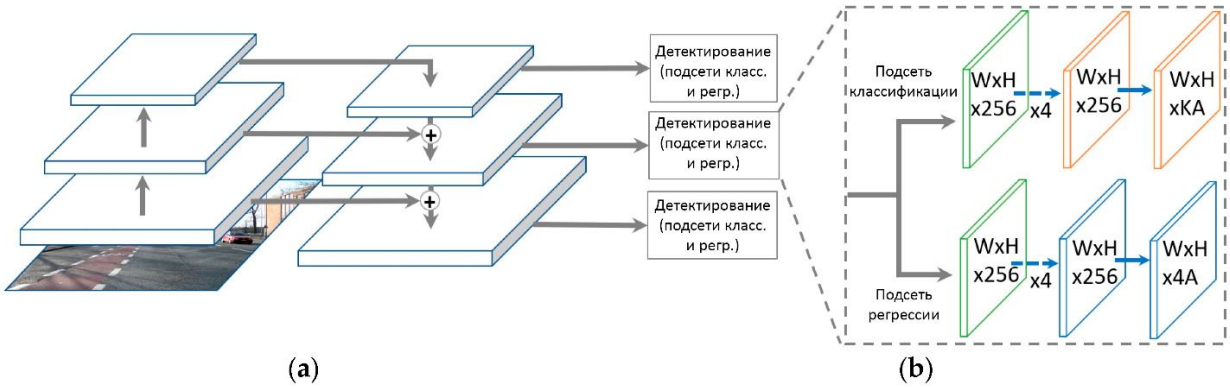


Рисунок 22 – Структурная модель детектора Retina-Net

В Retina-Net используется классическая сеть пирамид признаков, где карты признаков с разных слоев подгоняются под нужный размер с помощью метода ближайшего соседа и объединяются [36].

После модуля FPN располагается детектор, использующий для обнаружения объектов 3 набора карт признаков разного масштаба, поступающих из FPN. Подобно предыдущим датекторам, данные карты признаков покрываются фиксированными наборами ограничивающих прямоугольников и с помощью отдельных групп сверточных слоев производится классификация объектов в данных ограничивающих прямоугольниках и регрессия смещений по координатам для этих же рамок. Последним этапом работы метода Retina-Net является удаление дублирующих ограничивающих рамок с помощью алгоритма подавления не-максимумов [37].

Особенностью данного детектора является высокая точность обнаружения, превосходящая YOLOv3 и SSD, однако вектор модификаций, выбранный в процессе разработки данного детектора и нацеленный на повышении точности предшествующих детекторов, сильно увеличил

вычислительную сложность полученного в результате детектора и, соответственно, уменьшил скорость обнаружения объектов.

2.3.6 Обзор детектора RefineDet

Еще одним из рассматриваемых детекторов объектов на изображении является RefineDet (Single-Shot Refinement Neural Network for Object Detection), разрабатываемый с целью объединения преимущества одноступенчатых и двухступенчатых методов и преодолеть их недостатки [33]. Полная архитектура данного метода представлена на рисунке 23.

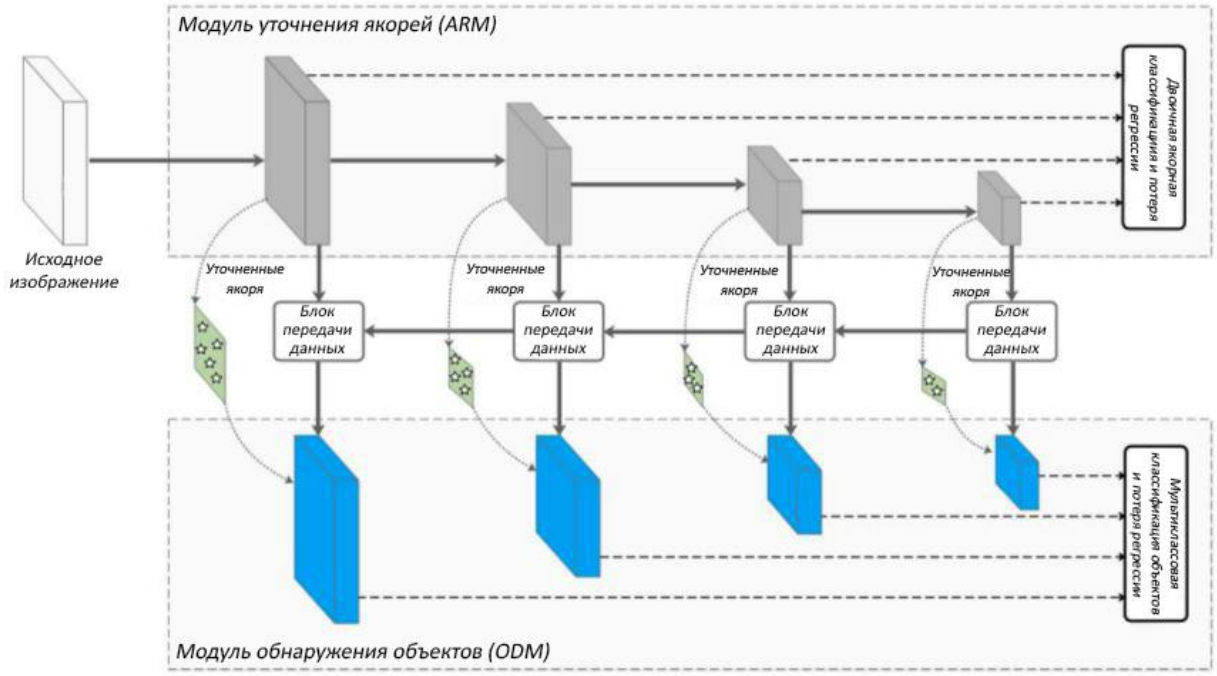


Рисунок 23 – Структурная модель детектора RefineDet

Представленный детектор состоит из двух частей: модуля уточнения якорных блоков (Anchor Refinement Module (ARM)) и модуля обнаружения объектов (Object Detection Module (ODM)) [33]. В первом модуле, также как в детекторах SSD и YOLO карты признаков с разных групп слоев сверточной нейронной сети покрываются группой якорных блоков фиксированного размера [12]. Модуль ARM отфильтровывает отрицательные якорные рамки, оцениваемые как фон, то есть происходит их бинарная классификации. Если

якорная рамка не соответствует фону, то для нее производится первичная регрессия координат ограничивающей рамки. После этого, отфильтрованные рамки передаются модулю ODM с помощью блока передачи данных [33]. Структура блока передачи данных представлена на рисунке 24.

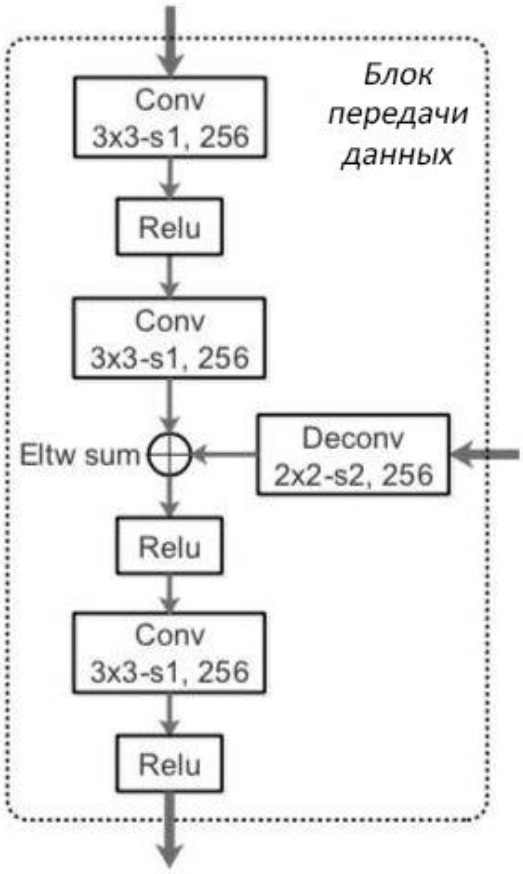


Рисунок 24 – Структура блока передачи данных

Данный блок предназначен для объединения карт признаков с разных слоев, подобно тому, как это делается в сетях пирамид признаков [35]. Полученные в результате объединения карты признаков передаются в модуль обнаружения объектов.

В модуле обнаружения объектов для полученных якорных рамок находится распределение вероятности по меткам классов, а также производится вторичная регрессии для нахождения смещений значений координат якорных блоков [33]. В конце работы данного детектора также

используется метод подавления не-максимумов для удаления дублирующих ограничивающих рамок.

В качестве экстрактора признаков в различных реализациях RefineDet используется как VGG-16, так и ResNet-101.

Преимуществом данной модели является то, что она сохраняет точность определения объектов на уровне двухступенчатых детекторов, таких как R-CNN и имеет скорость близкую к SSD.

2.3.7 Обзор детектора YOLOR

Детектор YOLOR (You Only Learn One Representation) не относится к семейству детекторов YOLO, но использует как основу YOLOv4 для своих улучшений. Самое главное принципиальное отличие YOLOR от всех других рассмотренных детекторов заключается в том, что в процессе обнаружения он использует не только явные знания, но и неявные [11]. Под явными знаниями подразумеваются те, которые могут быть получены в процессе обучения, а под неявными те, которые получены подсознательно. Проецируя данные понятия в область сверточных нейронных сетей, можно сказать, что явные знания — это признаки, извлекаемые из начальных слоев, а неявные знания – признаки, полученные из глубоких слоев [11]. Объединяя явные и неявные знания в процессе обнаружения, данный метод достигает высоких показателей скорости и точности обнаружения. Общий принцип работы данного подхода представлен на рисунке 25.

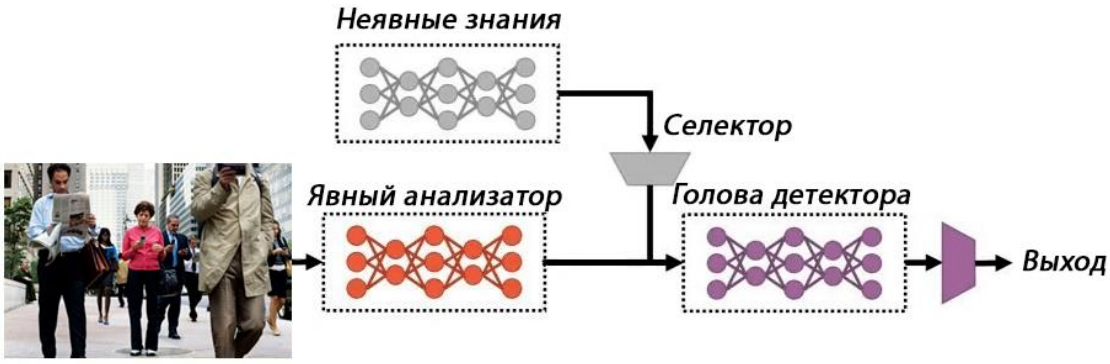


Рисунок 25 – Структурная модель детектора YOLOR

Явное обучение может производиться разными способами, в зависимости от решаемой задачи, но в случае с обнаружением объектов под явным обучением подразумевается обычное обучение фрагмента сети, отвечающего за получение карт признаков, с помощью обратного распространения ошибки для оптимизации его параметров [24].

Явный анализатор, показанный на рисунке 25, представляет из себя экстрактор признаков CSPDarknet-53 [5] в совокупности с сетью уточнения признаков PAN. Как было сказано в предыдущем разделе, PAN является дополнением FPN и усиливает пирамиду признаков точными сигналами локализации с низких уровней посредством добавления к FPN дополнительного восходящего пути [8]. Кроме того, для увеличения рецептивного поля и выделения важных признаков из изображения, YOLOR использует модуль SPP или иначе модуль объединения пространственных пирамид (Spatial Pyramid Pooling) [17]. Данный модуль представлен на рисунке 26.

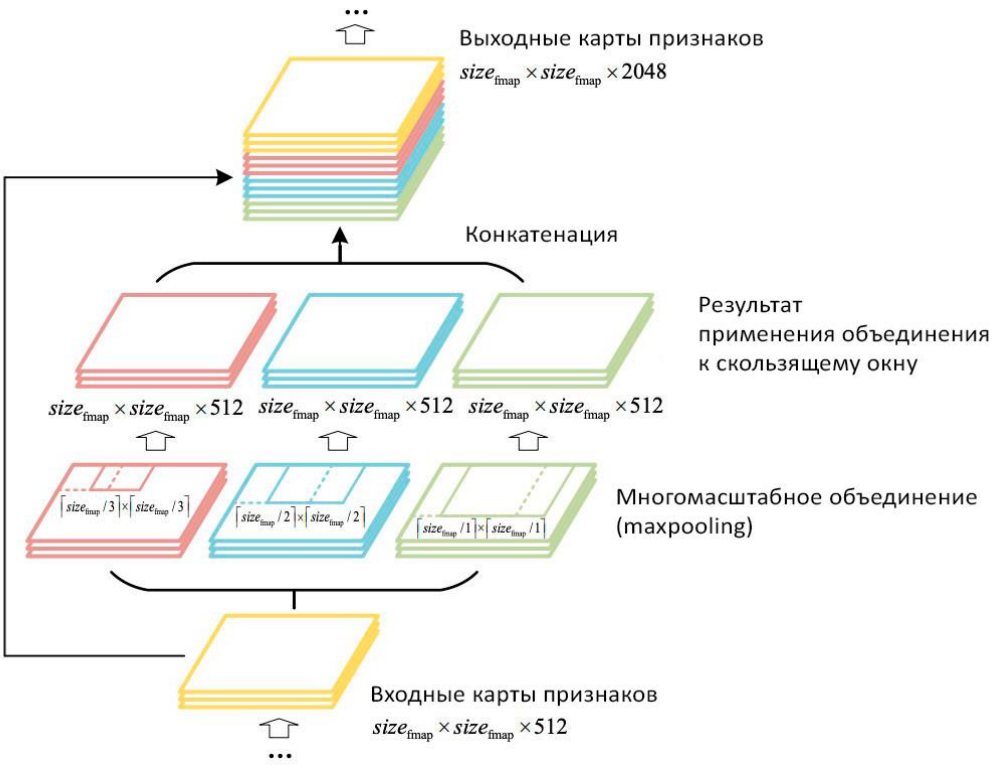


Рисунок 26 – Структура модуля SPP

Модуль SPP располагается после сети пирамид признаков и выполняет объединение по максимальному значению для окна, скользящего по картам признаков. Данная операция выполняется над одними и теми же входными данными несколько раз с различными размерами объединяющего ядра. Полученные в результате карты признаки склеиваются между собой с помощью конкатенации [17].

Неявные знания, представленные на рисунке 25, могут быть представлены одни из следующих способов:

- вектор, матрица или тензор;
- нейронная сеть, где неявное представление является линейной комбинацией вектора и весовой матрицы;
- матричная факторизация, где неявное представление является комбинацией базиса нескольких векторов и некоторого коэффициента.

Применение неявных знаний посредством операций конкатенации или сложения к результатам работы разных этапов сети, позволяет увеличить точность классификации, а также точность и скорость регрессии ограничивающей рамки объекта [11].

Голова детектора, используемая в YOLOR полностью аналогична используемой в YOLOv4: карты признаков полученные с модуля SPP покрываются фиксированным набором ограничивающих прямоугольников, размер которых находится автоматически с помощью алгоритма классификации k-NN и для каждого из них с помощью трех групп сверточных блоков производятся предсказания ограничивающей рамки, класса объекта и показателя объектности [5]. Аналогично другим детекторам с помощью алгоритма подавления не-максимумов удаляются лишние ограничивающие рамки.

Преимуществами данного детектора являются высокие показатели точности и скорости обнаружения объектов.

Таким образом, были рассмотрены особенности наиболее популярных детекторов объектов на изображении на основе СНС, таких как Faster R-CNN, R-FCN, YOLOv4, SSD, Retina-Net, RefineDet и YOLOR, а также выделены их преимущества и недостатки.

2.4 Сравнительный анализ современных детекторов объектов на изображении на основе сверточных нейронных сетей

Основными критериями оценки детекторов объектов являются их показатели точности и скорости обнаружения, именно поэтому при сравнении рассмотренных детекторов будет оцениваться количество обрабатываемых кадров в секунду и точность с помощью метрики mAP (mean average precision) [26]. Для проведения испытаний были взяты обученные модели детекторов, проверка которых производилась на наборе данных MS COCO, содержащим порядка 330 тысяч размеченных изображений, принадлежащим к 80 различным классам. Тестирование детекторов происходило на графическом процессоре Tesla K80 с помощью облачного сервиса Google Colab, предоставляющего бесплатный доступ к графическим процессорам из командной оболочки языка Python.

Результаты тестирования детекторов объектов на изображении представлены в таблице 3.

Таблица 3 - Результаты тестирования современных детекторов объектов на изображении

Название детектора	Скорость обнаружения (FPS)	Средняя точность распознавания (mAP)
Faster R-CNN	7	36,2
R-FCN	17,29	31,5
YOLOv4	62	43,2
SSD	46	33
Retina-Net	32,5	40,8
RefineDet	33,5	36,4
YOLOR	102	52,8

Диаграмма, демонстрирующая результаты работы детекторов представлена на рисунке 27.

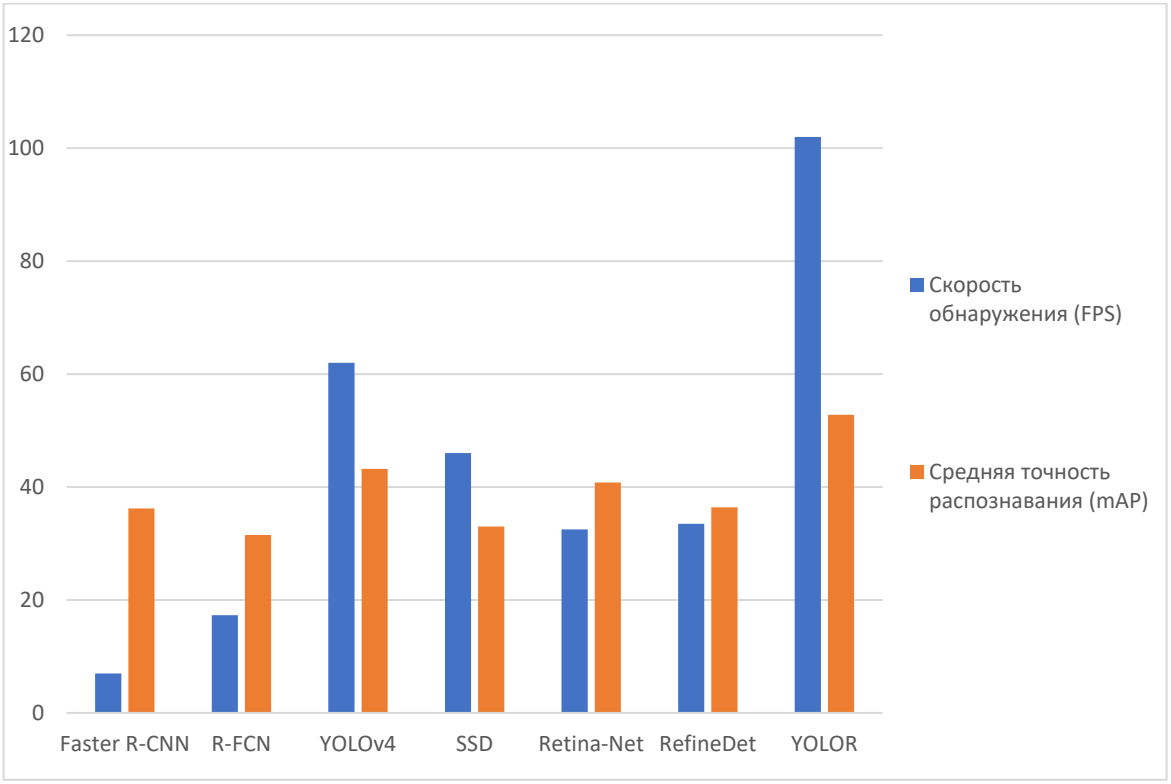


Рисунок 27 – Диаграмма основных показателей детекторов

Сравнивая результаты работы детекторов, можно сделать вывод, что двухступенчатые Faser R-CNN и R-FCN уступают одноступенчатым детекторам по скорости обнаружения, из чего следует их непригодность для использования в приложениях реального времени [27]. Наилучшим одноступенчатым детектором обнаружения объектов на изображении по результатам сравнительного анализа является YOLOR, поскольку превосходит другие экземпляры как в точности, так и в скорости обнаружения. Детектор объектов YOLOR является одним из самых современных решений в области компьютерного зрения и предлагает новаторский подход использования неявных знания в процессе обнаружения [11]. Кроме того, он занимает первое место в рейтинге детекторов объектов, предназначенных для работы в режиме реального времени.

Таким образом, был проведен сравнительный анализ основных детекторов объектов на изображении на основе сверточных нейронных сетей.

Вывод по разделу 2

Итак, в данном разделе был проведен анализ основных принципов работы детекторов объектов на изображении, основанных на СНС. Также был выполнен сравнительный анализ основных сверточных архитектур, выступающих в качестве структурных частей множества детекторов и напрямую влияющих на их производительность. Кроме того, были рассмотрены особенности реализации некоторых современных детекторов объектов на изображении, среди которых такие как: R-CNN, R-FCN, YOLO, SSD, RetinaNet, RefineDet и YOLOR. В рамках анализа данных детекторов были выявлены их преимущества и недостатки, а также проведено сравнение их показателей точности и скорости обнаружения объектов на изображении.

На основании результатов сравнительного был сделан вывод, что одноступенчатые детекторы являются наиболее подходящими для обнаружения объектов в режиме реального времени, поскольку обладают лучшим быстродействием. Кроме того, среди рассмотренных одноступенчатых детекторов объектов, наилучшим оказался YOLOR, обладающий самыми высокими показателями точности обнаружения и быстродействия.

3 Модификация архитектуры детектора объектов YOLOR

3.1 Выявление сильных сторон детектора YOLOR

Особенность детектора объектов YOLOR, как уже было отмечено ранее, состоит в совместном использовании явных и неявных знаний для обнаружения объектов. Именно благодаря неявным знаниям данный детектор достигает высоких показателей скорости и точности обнаружения объектов на изображении [11]. Неявные знания, могут быть представлены одни из следующих способов:

- вектор, матрица или тензор;
- нейронная сеть, где неявное представление является линейной комбинацией вектора и весовой матрицы;
- матричная факторизация, где неявное представление является комбинацией базиса нескольких векторов и некоторого коэффициента.

Для получения неявных знаний используют неявные нейронные представления, которые позволяют отображать дискретные входные значения на параметризованное непрерывное множество. Для преобразования неявного обучения в остаточную форму нейронных сетей используются равновесные модели [11].

Введение неявных знаний в модель позволяет получить следующие преимущества:

- уменьшение пространства многообразия обученного представления;
- выравнивание пространства ядра;
- дополнительные преимущества регрессии ограничивающей рамки.

Под пространством многообразия подразумевается, что любой объект может быть представлен как некоторая точка в пространстве и хорошо обученная модель для входных данных должна быть способна отыскать их проекцию в данное пространство [11]. Неявные знания позволяют уменьшить

это пространство, что увеличивает эффективность классификации. Схематично это представлено на рисунке 28.

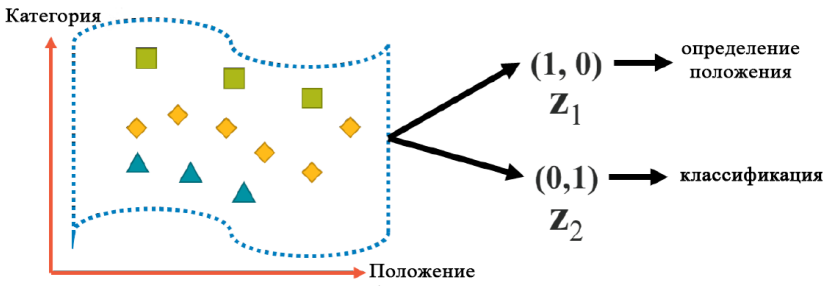


Рисунок 28 – Принцип уменьшения пространства многообразия

Второе получаемое преимущество - выравнивание пространства ядра, которое позволяет избежать проблемы рассогласования в многоголовочных сетях [11]. Наглядно необходимость данной операции представлена на рисунке 29.

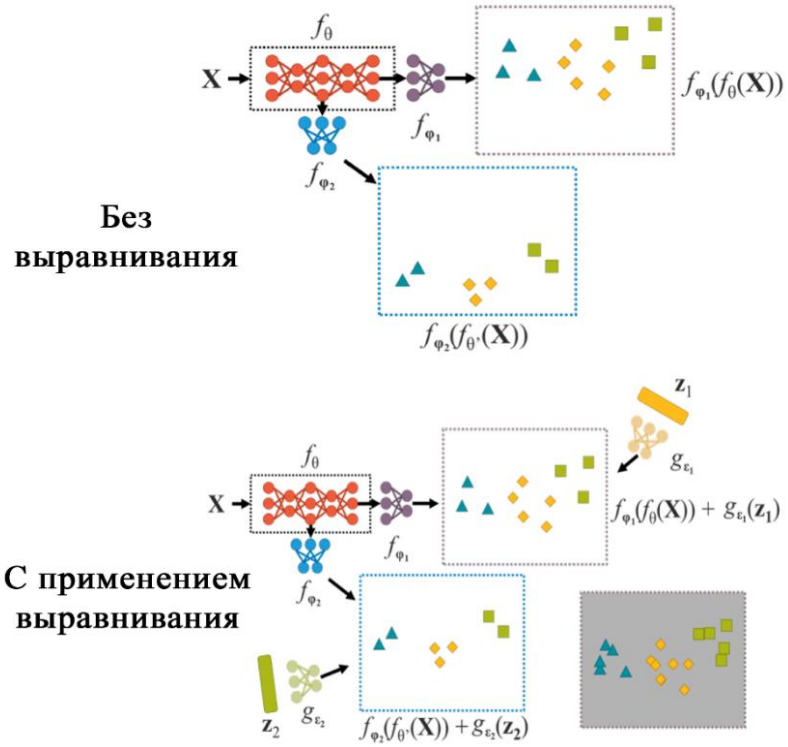


Рисунок 29 – Принцип выравнивания пространства ядра

Поскольку в детекторе YOLOR обнаружение происходит с помощью трех голов детектора на картах признаков разных масштабов и получаемых из разных частей нейронной сети, то объекты, относящиеся к одному и тому же классу, но имеющие разный размер могут обнаруживаться разными головками детектора, что приводит к рассогласованию пространства ядра. Из-за этого признаки больших и мелких объектов могут интерпретироваться по-разному, что приводит к ошибочным предсказаниям [11]. Выравнивание пространства ядра происходит посредством сложения или умножения неявного представления на карты признаков, получаемые после сети уточнения признаков.

Дополнительными преимуществами использования неявных знаний, по словам авторов детектора, является их сложение с результатами обнаружения для предварительного определения смещения центральной координаты ограничивающей рамки или их умножение для выявления гиперпараметров якорных блоков [11]. Наглядно данные функции представлены на рисунке 30.

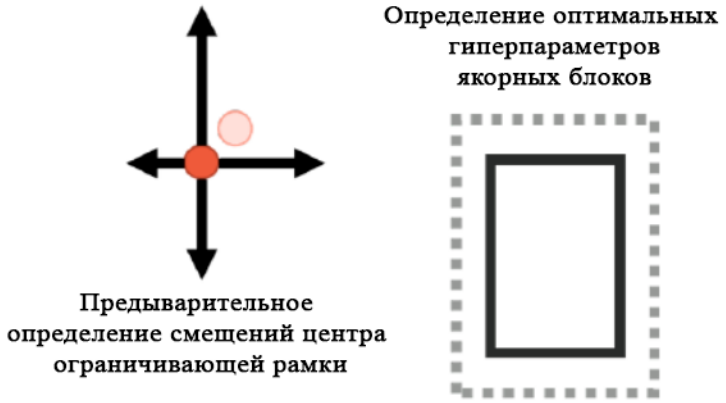


Рисунок 30 – Дополнительные преимущества от использования неявных знаний

То, для каких целей и в каком месте неявные знания применяются в модели YOLOR представлено на рисунке 31. В модели YOLOR неявные знания применяются к выходам слоя PAN для выравнивания признаков, к

результирующим предсказаниям детектора для их уточнения и для каждой головы детектора для улучшения оптимизации функции потерь [11].

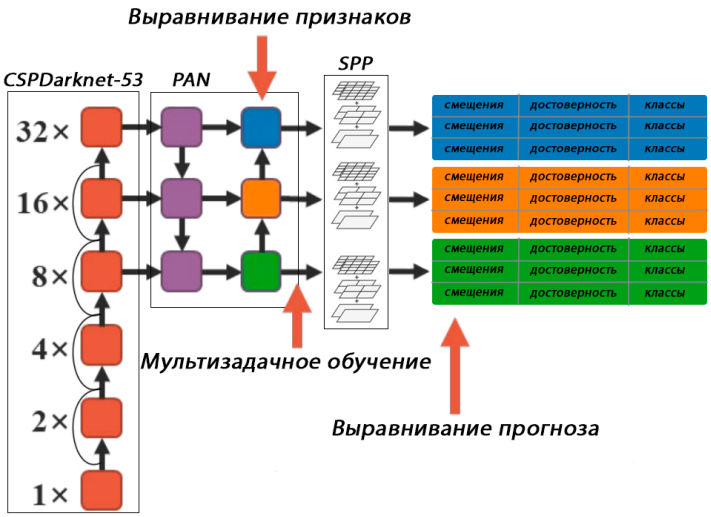


Рисунок 31 – Применение неявных знаний в модели YOLOR

Таким образом, была рассмотрена основная особенность детектора YOLOR – использование неявных знаний в процессе обнаружения объектов, значительно повышающих производительность детектора, а также определены цели их применения к различным структурным частям детектора. Подход использования неявных знаний в процессе обнаружения будет также использоваться в модифицированной версии детектора. Несмотря на преимущества данного детектора над другими как в точности, так и в скорости обнаружения, он обладает некоторыми недостатками.

3.2 Предлагаемые улучшения

3.2.1 Улучшение экстрактора признаков

Детектор YOLOR использует в качестве экстрактора признаков СНС под названием CSPDarknet-53 [5]. Данная архитектура объединяет в себе модульность и остаточные соединения между сверточными блоками,

унаследованные от ResNet с межэтапными, частичными соединениями подобно DenseNet. Благодаря коротким соединениям происходит усечение градиентного потока, что улучшает распространение ошибки на более ранние слои в процессе обучения, тем самым повышая качество и скорость обучения [18]. Добавление межэтапных частичных соединений в данную архитектуру, позволяет устранить узкие места в вычислениях [10]. Структура CSPDarknet-53 представлена на рисунке 32.

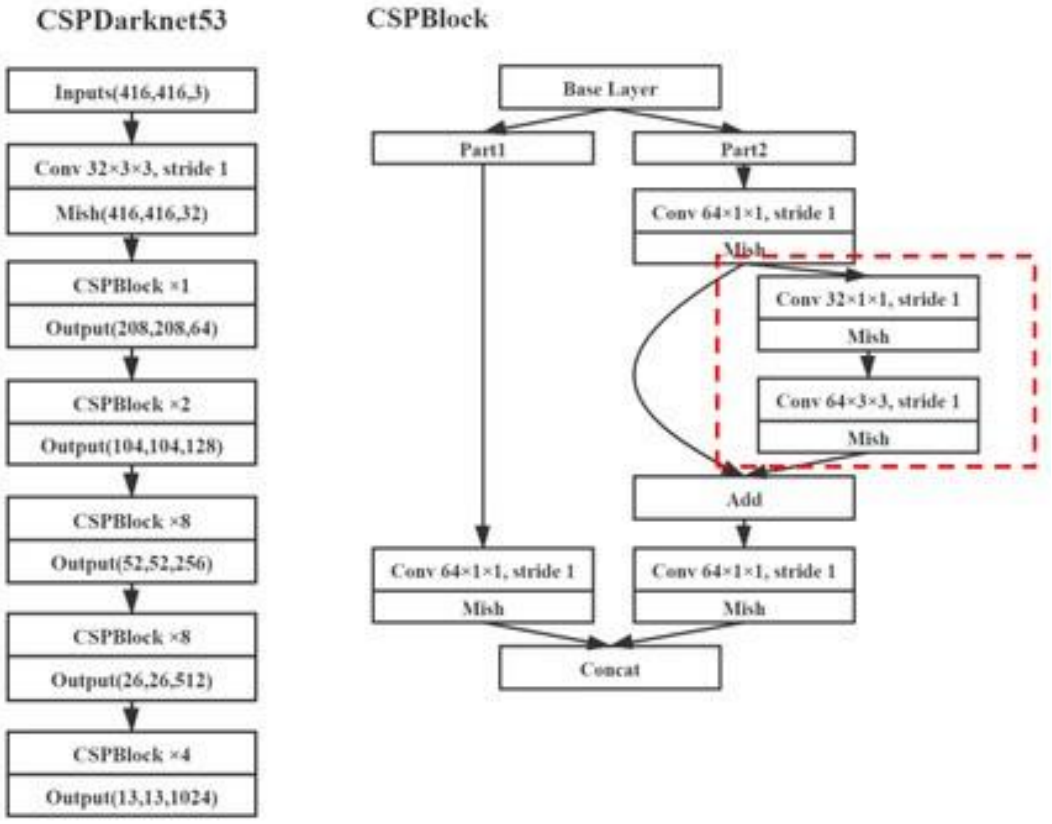


Рисунок 32 – Структурная модель экстрактора признаков CSPDarknet-53

При сравнительном анализе экстракторов признаков, проведенного в разделе 2.2, было выявлено, что архитектура Darknet-53, модификация которой используется в YOLOR, уступает по точности ResNeXt, Res2Net и DPN [41]. Следовательно, использование одного из этих экстракторов признаков может повысить точность детектора. Для увеличения скорость обнаружения, необходимо использовать экстракторы признаков, обладающие

минимальным количеством слоев и вычислительной сложностью. Исходя из этого, в качестве претендентов на роль экстрактора признаков были выбраны экстрактор признаков CSPResNeXt, внедряющий межэтапные частичных соединения в ResNeXt подобно CSPDarknet, и экстрактор признаков Res2Net [31]. Сравнение основных характеристик CSPResNeXt и Res2Net, а также их сопоставление с характеристиками оригинального экстрактора признаков CSPDarknet представлено в таблице 4.

Таблица 4 - Результаты сравнительного анализа экстракторов признаков

Наименование модели	Количество параметров, млн.	FLOPs, млрд.	Точность (Топ-1)
CSPDarkNet-53	27,6	13	77,2
CSPResNeXt-50	20,5	3,4	77,9
CSPResNeXt-101	38,2	7,3	79,7
Res2Net-50	23	3,8	77,9
Res2Net-101	42	7,6	79,2

Как видно из таблицы, CSPResNeXt и Res2Net превосходят экстрактор признаков CSPDarknet-53 по точности, следовательно, использование одного из этих экстракторов в YOLOR способно повысить точность детектора. Причем более глубокие архитектуры со 101 слоем должны значительно повысить точность модели. В то время как более короткие архитектуры с 50 слоями должны поспособствовать увеличению скорости модели, незначительно увеличивая точность [39].

Стоит отметить, что архитектуры CSPResNeXt и Res2Net со 151 слоем не рассматриваются в качества экстрактора признаков, поскольку дают небольшой прирост к точности по сравнению с менее глубокими архитектурами, но значительно повышают вычислительную сложность, что в результате может негативно сказаться на скорости работы детектора [39].

Хотя говорится, что архитектуры CSPResNeXt и Res2Net являются 50 и 101-слойными, на самом деле они содержат гораздо большее количество

слоев. Постфиксы 50 и 101 говорят о том, что количество сверточных блоков в архитектуре соответствует ResNet-50 и ResNet-101 соответственно.

Предложенная в ResNeXt концепция «сеть в нейроне» позволяет разбивать карты признаков на несколько наборов для проведения независимых сверток, в дальнейшем объединяя их. Вводится новое измерение, называемое «мощность» регулирующее количество таких независимых ветвей сверток [29]. Подобная архитектура экстрактора на практике показывает значительное увеличение разнообразности признаков и, следовательно, точности классификации, без увеличения количества параметров и сложности сети.

Особенности иерархических частичных соединений, используемых в Res2Net, позволяют изменять рецептивные поля на более детальном уровне для захвата мелких и глобальных признаков [31]. Данная архитектура также вводит новое измерение, называемое «масштаб», регулируемое ширину иерархической структуры в блоке.

Данные архитектуры имеют общую структуру с оригинальной моделью ResNet, представленную на рисунке 33.

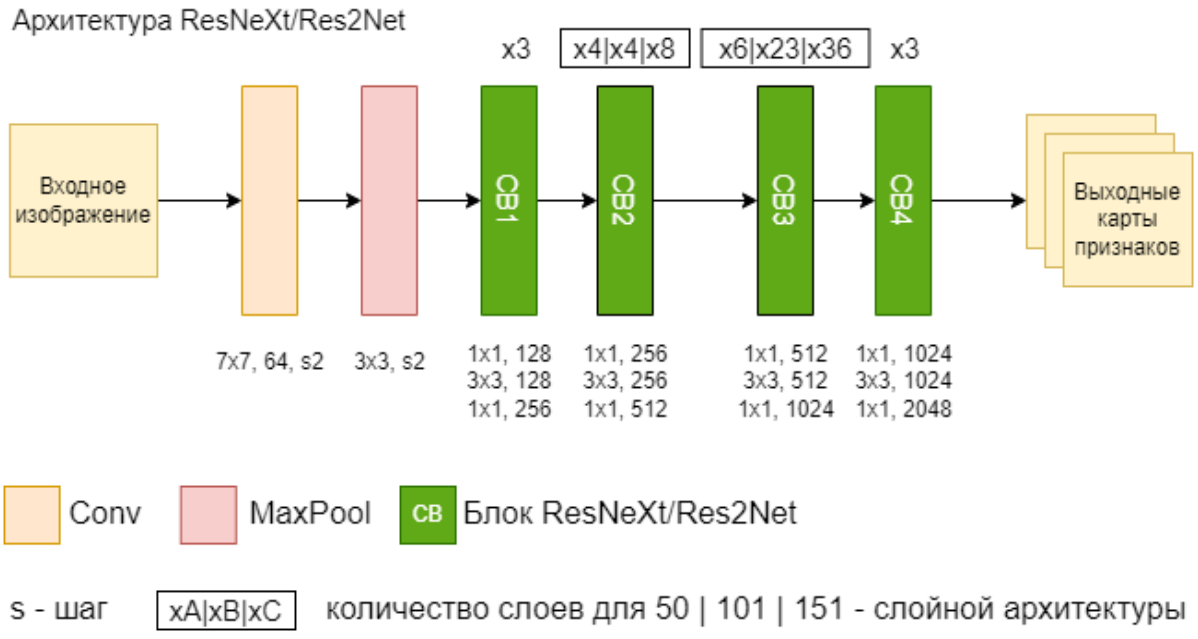


Рисунок 33 – Структурная модель ResNeXt и Res2Net

Различие данных архитектур состоит во внутреннем строении их блоков, обозначенных как «СВ» на рисунке 33. Особенности построения блоков ResNeXt и Res2Net представлены на рисунке 34.

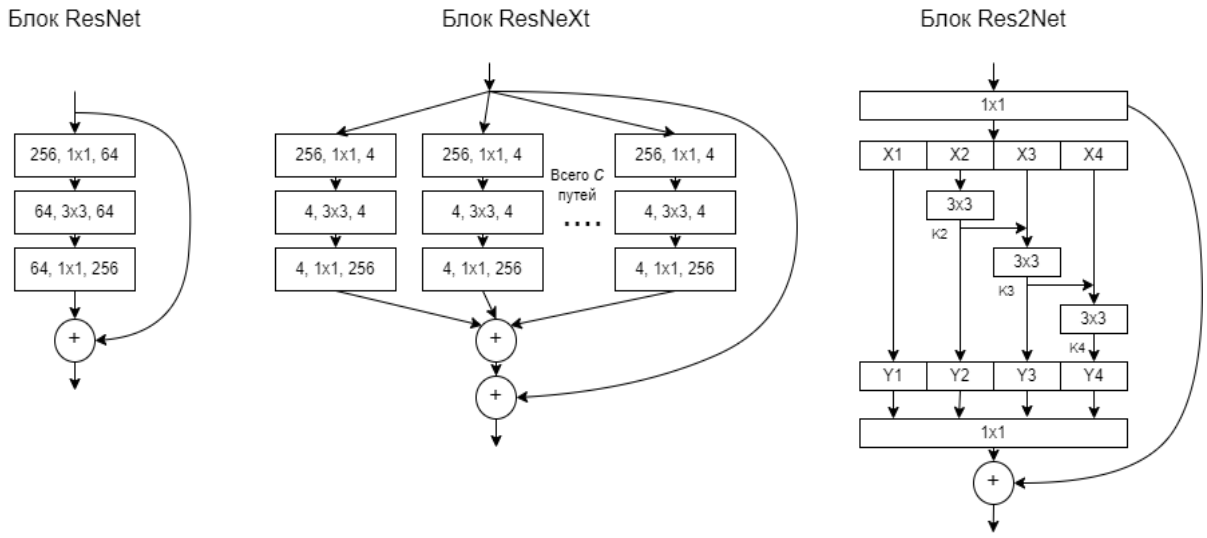


Рисунок 34 – Структура блоков ResNeXt и Res2Net

Каждый из представленных блоков обладает своими особыми преимуществами, поэтому было решено сконструировать сверточную архитектуру, объединяющую блоки Res2Net и ResNeXt и содержащую оба дополнительных измерения: масштаб и мощность. Экстрактор признаков, интегрирующий оба блока посредством встраивания блоков ResNeXt внутрь блока Res2Net, позволит получить следующие преимущества:

– сохранение части низкоуровневых признаков и сокращение вычислительной сложности, подобно DenseNet, благодаря первой ветви Res2Net, сохраняющей часть входных данных в первоначальном виде [10];
– улучшение обнаружения мелких и крупных объектов, благодаря увеличенному размеру рецептивного поля вследствие масштабирования признаков [31];

– повышение точности классификации с сохранением вычислительной сложности внутри Res2Net за счет использования групповых сверток ResNeXt, увеличивающих разнообразие выделяемых признаков [29];
– увеличение скорости обучения модели за счет использования остаточных соединений между блоками, способствующих ускорению оптимизации ранних слоев в процессе обратного распространения ошибки [18].

Сверточная модель, полученная в результате объединения Res2Net и CSPResNeXt была названа Res3Net. Структура блока данной модели представлена на рисунке 35.

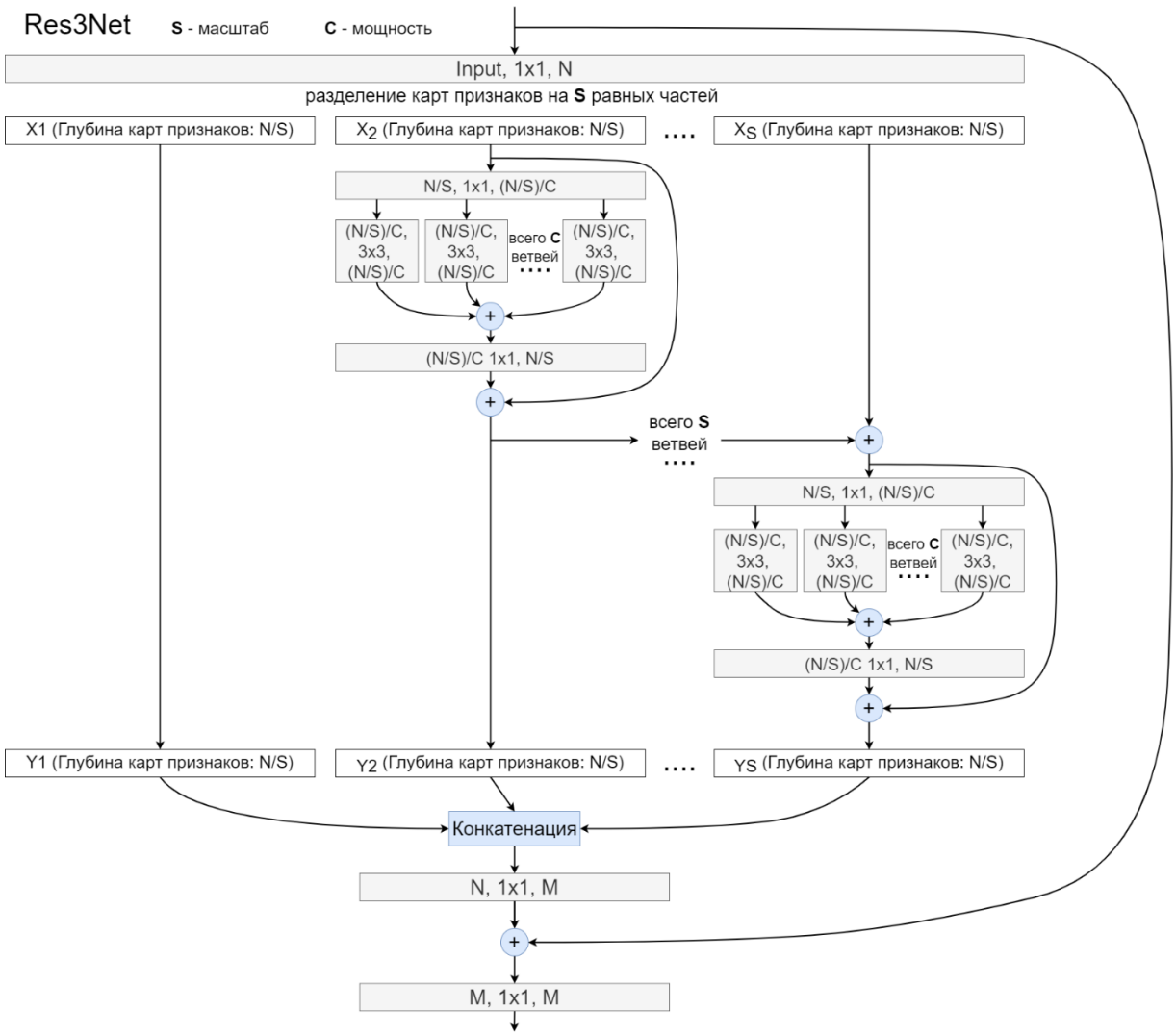


Рисунок 35 – Структура блока Res3Net

Часто, показатель мощности в архитектуре CSPResNeXt принимает значение 8, 16 или 32. От этого показателя зависит количество сверточных фильтров, используемых в сверточных блока [29]. Поскольку ResNeXt внедряется внутрь Res2Net, где происходит разбиение входной карты признаков на несколько частей, необходимо выбирать такую конфигурацию, чтобы увеличить количество сверточных фильтров в параллельных ветвях CSPResNeXt. Использование большого количества параллельных операций сверток с наименьшим набором фильтров снижает эффективность выделения признаков [29]. Также, при выборе показателя масштаба, используемого в блоке Res2Net, необходимо учитывать, что большой размер иерархической структуры приведет к значительному повышению вычислительной сложности [31]. Исходя из этого, показатели масштаба и мощности были выбраны 4 и 8 соответственно, что приводит лишь к незначительному увеличению вычислительной сложности по сравнению с исходными экстракторами признаков.

Таким образом, были даны предложения по улучшению детектора путем замены сети выделения признаков CSPDarknet53 на смоделированную сеть Res3Net, объединяющую в себе преимущества более точных экстракторов признаков CSPResNeXt и Res2Net.

3.2.2 Улучшение сети уточнения карт признаков

Уточнение признаков, выполняющееся в шее детектора, является не менее важным этапом, чем их извлечение. Правильное конструирование шеи детектора может значительно повысить производительность модели [35]. Почти все методы уточнения карт признаков основаны на сетях пирамид признаков и используют восходящие и нисходящие пути с объединением одномасштабных карт признаков из соседних слоев [8, 20, 35].

В качестве сети уточнения признаков в YOLOR используется архитектура PAN, представленная на рисунке 36.

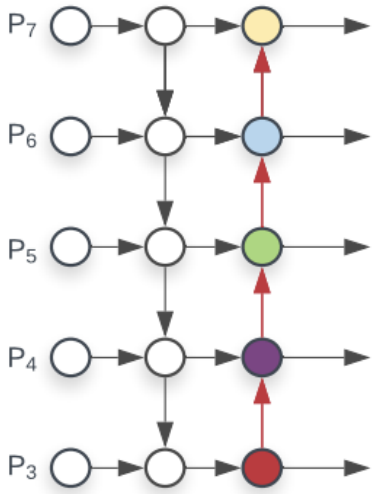


Рисунок 36 – Структурная модель сети уточнения признаков PAN

Данная архитектура сети является довольно мощной сетью уточнения признаков, однако в совокупности с CSPDarknet-53 производит недостаточное увеличение рецептивного поля для обнаружения крупных объектов [11]. Именно поэтому в YOLOR используется блок SPP для расширения рецептивного поля. Однако, как показывают современные исследования в области компьютерного зрения, использование слоев объединения по максимальному значению в глубоких сверточных архитектурах, во-первых, способствует потере важной пространственной информации, а во-вторых, понижает устойчивость детектора к вариативности позы объектов на изображении (проблема ухудшения дисперсии вариативности) [22]. Из-за этого детектор хуже обнаруживает сложные объекты при изменении их пространственного положения. Одним из способов увеличения размера рецептивного поля является использование глубоких экстракторов признаков, однако это влечет за собой значительное снижение скорости обнаружения объектов на изображении [26]. Исходя из этого, самым оптимальным решением данной проблемы является использование модульных сетей уточнения признаков наподобие BiFPN и TPN, способных гибко

масштабировать размер рецептивного поля в зависимости от количества используемых модульных блоков [20].

Из результатов сравнительного анализа сетей уточнения признаков, представленного в таблице 2 из пункта 2.2, видно, что модульная сеть уточнения признаков BiFPN обладает максимальным показателем повышения точности исходной модели ResNet и составляет 12,6%, в то время как PAN повышает точность на 11%. Кроме того, BiFPN имеет на 22% меньше обучаемых параметров. Именно поэтому данная архитектура была выбрана для внедрения в YOLOR вместо связки PAN+SPP. Архитектура сети уточнения признаков BiFPN состоит из чередующихся блоков, внутреннее которых представлено на рисунке 37.

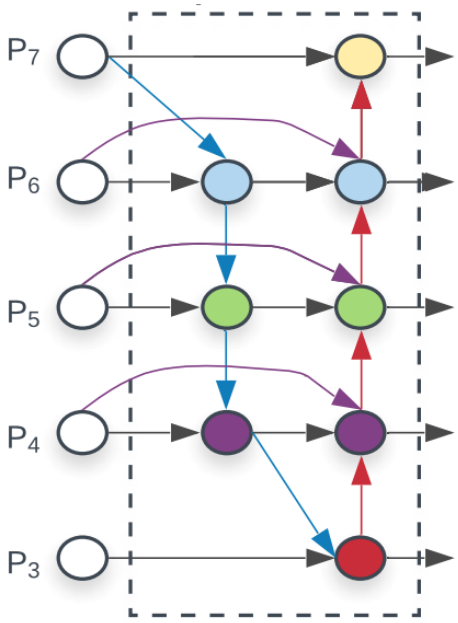


Рисунок 37 – Структурная модель блока BiFPN

Данная СНС подобно PAN является двунаправленной, однако из нее удалены узлы, имеющие один вход и, следовательно, не участвующие в объединении признаков. Также, добавление межэтапного соединения между входным и выходным слоями, находящимися на одном уровне, позволяет без лишних затрат на вычисление объединять больше признаков [20].

Поскольку СНС Res3Net, используемая в качестве экстрактора признаков должна достаточно хорошо расширять рецептивное поле за счет иерархической сверточной структуры, нет необходимости использовать большое количество блоков BiFPN [15]. Именно поэтому будет использовано только 2 чередующихся блока BiFPN, что должно быть достаточно для получения большого количества объединенных признаков высокого уровня.

Поскольку объединяемые в BiFPN карты признаков с разных слоев имеют различное разрешение, это может привести к тому, что одни карты признаков будут оказывать большее влияние на результат, нежели другие. Для решения данной проблемы ко всем объединяемым картам признаков добавляются дополнительные весовые коэффициенты, значение которых ограниченно в диапазоне от 0 до 1, и которые также, как и другие оптимизируются в процессе обучения [20]. Данное объединение называется быстрым нормализованным объединением и описывается формулой (2).

$$O = \sum_i \frac{w_i}{e + \sum_j w_j} \times I_i \,, \tag{2}$$

где O – выходные карты признаков;

w_i – обучаемый весовой коэффициент для i-го набора карт признаков;

e – малое значение, препятствующее делению на 0 в случае зануления весового коэффициента;

w_j – множество всех весовых коэффициентов;

I_i – входной набор карт признаков с индексом i.

Заключительная структурная часть архитектуры YOLOR - голова детектора не нуждается в изменении и остается аналогична используемой в YOLOv4.

Измененная модель YOLOR с учетом всех предложений по улучшению архитектуры представлена на рисунке 38.

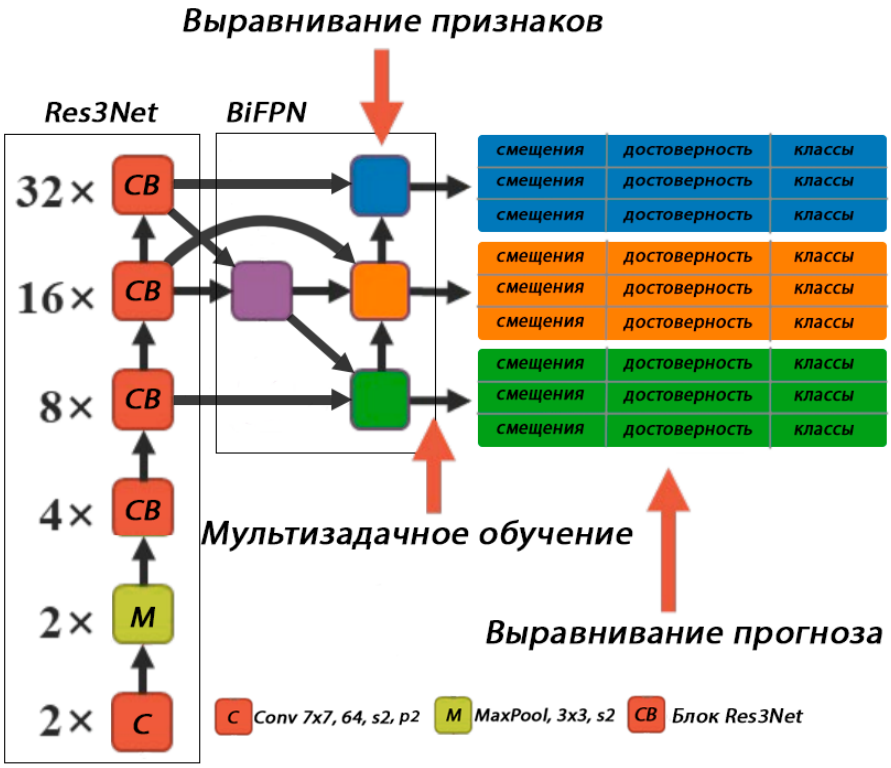


Рисунок 38 – Архитектура измененной модели YOLOR

Таким образом, были даны предложения по улучшению архитектуры модели детектора YOLOR, которые способны повысить точность и скорость обнаружения объектов на изображении: заменить сети выделения признаков CSPDarknet53 на смоделированную сеть Res3Net, кроме того, сеть уточнения карт признаков PAN совместно с модуль объединения пространственных пирамид заменить на сеть уточнения признаков BiFPN.

3.3 Описание математической модели модифицированного детектора YOLOR

Огромную долю любого детектора объектов на изображении составляют операции свертки, выполняющиеся во время прямого прогона изображения через архитектуру сверточной нейронной сети [24]. Визуально данная

операция была представлена на рисунках 5-6. Математически операция свертки может быть представлена формулой (3).

$$C_{m,n}^{i} = f(b + \sum_{q \in Q_i} \sum_{k=0}^{RF-1} \sum_{l=0}^{RF-1} X_{m+k,n+l}^{q} \times KL_{k,l}^{q}), \quad (3)$$

где $C_{m,n}^{i}$ – выход нейрона, на слое i с индексами m, n;

f – функция активации;

b – смещение;

Q_i – множество карт признаков предыдущего слоя;

RF – размер рецептивного поля нейрона;

X – входное значение нейрона;

KL – множество фильтров.

Для уменьшения размерности карты признаков, полученной в результате свертки, часто используют операцию подвыборки по максимальному значению [24]. С математической точки данная операция описывается формулой (4).

$$h_{i,j} = \max\{x_{i+k-1,j+l-1} \; \forall \; 1 \leq k \leq m \; and \; 1 \leq l \leq m\}, \quad (4)$$

где h – результирующая матрица;

m – высота и ширина ядра подвыборки.

Результирующим слоем в сверточном блоке является слой активации, который производит нормализацию входного сигнала и представляет результат работы сети в нужном диапазоне [22]. В предлагаемом детекторе в качестве функции активации была выбрана функция Mish, представленная в формуле (5).

$$f(x) = x \times \tanh(\ln(1 + e^{x})), \quad (5)$$

где x – карта признаков, полученная в результате свертки.

Выбор данной функции активации вместо часто используемой функции ReLU обусловлен улучшенной производительностью как с точки зрения стабильности во время обучения, так и точности.

Графическое представление данной функции активации приведено на рисунке 39.

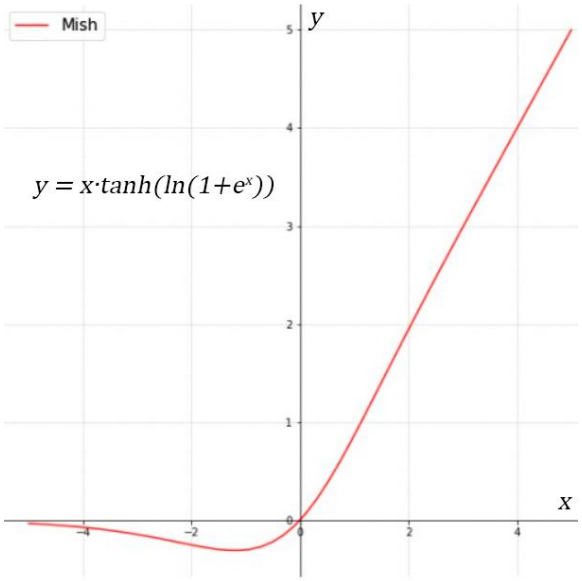


Рисунок 39 – Графическое представление функции активации Mish

При проектировании сверточной нейронной сети, необходимо учитывать изменение размера карт признаков при прямом проходе по сети, особенно, если производится объединение карт с разных слоев [26]. Размер карт признаков, получаемых в результате свертки между набором фильтров и входными картами признаков, может быть рассчитан по формуле (6).

$$C_{tensor} = \left(\left[\frac{n_H + 2p - f}{s} + 1\right], \left[\frac{n_W + 2p - f}{s} + 1\right], FC\right), \quad (6)$$

где C_{tensor} – размерность карт признаков, представленных в виде 3-х мерного тензора;

n_H, n_W – высота и ширина входной матрицы соответственно;

p – размер рамки нулевого заполнения;

f – размер фильтра свертки (квадратная матрица);

s – размер шага свертки;

FC – количество фильтров в ядре свертки.

Размер матрицы, полученной в результате операции подвыборки, рассчитывается также, как для операции свертки по формуле (6).

В процессе работы модель делает (5 + c) предсказаний для каждой ячейки сетки, где c – количество классов, распознаваемых моделью. В 5 первых предсказываемых значений входят смещения координат центра ограничивающей рамки, ее высоты и ширины, а также оценка достоверности [5]. Ограничивающая рамка, генерируемая на основе предсказанных смещений, представлена на рисунке 40.

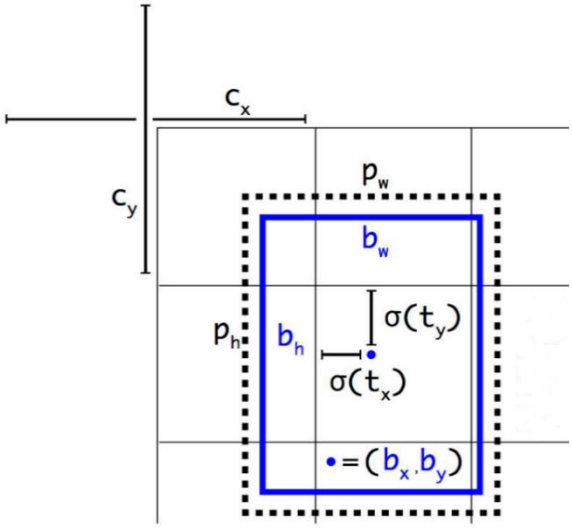


Рисунок 40 – Ограничивающая рамка, генерируемая на основе предсказанных смещений

Параметры координат ограничивающей рамки, представленной на рисунке 40, рассчитываются аналогично формулам (7) – (10).

$$b_x = \sigma(t_x) + c_x, \quad (7)$$

$$b_y = \sigma(t_y) + c_y, \quad (8)$$

$$b_w = p_w \times e^{t_w}, \quad (9)$$

$$b_h = p_h \times e^{t_h}, \quad (10)$$

где b_x, b_y, b_w, b_h - значения координат предсказанной ограничивающей рамки;

t_x, t_y, t_w, t_h - смещения, предсказанные моделью;

c_x, c_y – координаты верхнего левого угла ячейки сетки, к которой относится якорная рамка;

p_w, p_h - ширина и высота якорной рамки;

σ – сигмовидная функция, унифицирующая предсказания в пределах интервала [0, 1] (ускоряет сходимость сети).

Оценка достоверности, предсказываемая для каждой ограничивающей рамки, рассчитывается по формуле (11).

$$\sigma(t_o) = \Pr(object) \times IOU(b, object), \quad (11)$$

где $\sigma(t_o)$ – показатель достоверности ограничивающей рамки;

$\Pr(object)$ – условная вероятность, что ограничивающая рамка содержит объект;

$IOU(b, object)$ – индекс пересечения, между объектом и предсказанной ограничивающей рамкой, рассчитываемый по формуле (1).

В качестве классификатора в YOLOR используется логистическая регрессия для прогнозирования вероятности отношения объекта к каждому классу, поэтому оставшиеся *c*-значений представляют распределение вероятностей по классам [11].

Последним шагом прямого прохода сверточный нейронной сети является расчет общей ошибки обнаружения, состоящей из ошибок

локализации, классификации и уверенности [5]. В детекторе YOLOR для расчета ошибок используются функции, предложенные в YOLOv3. Ошибка классификации рассчитывается как перекрестная энтропия между двумя распределениями вероятностей и представлена в формуле (12).

$$\sum_{i=0}^{K\times K}\sum_{j=0}^{M} I_{ij}^{obj} \sum_{c\in classes} [\hat{p}_i(c)\log\big(p_i(c)\big) + (1-\hat{p}_i(c))\log\big(1-p_i(c)\big)], \qquad (12)$$

где K – размер сетки;

M – количество якорных блоков на каждую ячейку сетки;

I_{ij}^{obj} – функция, возвращающая 1, если за обнаружение объекта отвечает j-я ограничивающая рамка в ячейке i, или 0 в ином случае

c – класс из множества обнаруживаемых классов;

$p_i(c)$ – истинное значение присутствия класса c в ячейке i;

$\hat{p}_i(c)$ – предсказанная условная вероятность присутствия класса c в ячейке i.

Ошибка локализации рассчитывается как средняя квадратическая ошибка по формуле (13).

$$\lambda_{coord}\sum_{i=0}^{K\times K}\sum_{j=0}^{M} I_{ij}^{obj}\,(2-w_i\times h_i)[(x_i-\hat{x}_i)^2+(y_i-\hat{y}_i)^2] + \\ + \lambda_{coord}\sum_{i=0}^{K\times K}\sum_{j=0}^{M} I_{ij}^{obj}\,(2-w_i\times h_i)\left[(w_i-\hat{w}_i)^2+\left(h_i-\hat{h}_i\right)^2\right], \qquad (13)$$

где λ_{coord} – положительный весовой коэффициент для ошибок координат ограничивающей рамки (по умолчанию обычно равен 5);

x_i, y_i, w_i, h_i – истинные значения координат центра, высоты и ширины ограничивающей рамки;

$\hat{x}_i, \hat{y}_i, \hat{w}_i, \hat{h}_i$ – предсказанные моделью значения координат центра, высоты и ширины ограничивающей рамки.

При расчете ошибки уверенности, также, как и при расчете ошибки классификации, рассчитывается перекрестная энтропия [15]. Расчет ошибки уверенности представлен в формуле (14).

$$\sum_{i=0}^{K\times K}\sum_{j=0}^{M} I_{ij}^{obj}\left[\hat{C}_i \log(C_i) + \left(1-\hat{C}_i\right)\log\left(1-C_i\right)\right] - \\ -\lambda_{noobj}\sum_{i=0}^{K\times K}\sum_{j=0}^{M} I_{ij}^{obj}\left[\hat{C}_i \log(C_i) + \left(1-\hat{C}_i\right)\log\left(1-C_i\right)\right], \tag{14}$$

где λ_{noobj} – коэффициент для увеличения ошибки при обнаружении объекта, являющегося фоном (по умолчанию обычно равен 0.5);

C_i – истинный показатель достоверности для рамки j в ячейке i;

$\hat{C}_i$ – предсказанный моделью показатель достоверности для рамки j в ячейке i.

Таким образом, общая ошибка представлена в формуле (15).

$$Loss = \lambda_{coord}\sum_{i=0}^{K\times K}\sum_{j=0}^{M} I_{ij}^{obj}\left(2-w_i\times h_i\right)\left[\left(x_i-\hat{x}_i\right)^2+\left(y_i-\hat{y}_i\right)^2\right] + \\ \lambda_{coord}\sum_{i=0}^{K\times K}\sum_{j=0}^{M} I_{ij}^{obj}\left(2-w_i\times h_i\right)\left[\left(w_i-\hat{w}_i\right)^2+\left(h_i-\hat{h}_i\right)^2\right] - \\ \sum_{i=0}^{K\times K}\sum_{j=0}^{M} I_{ij}^{obj}\left[\hat{C}_i \log(C_i) + \left(1-\hat{C}_i\right)\log\left(1-C_i\right)\right] - \\ \lambda_{noobj}\sum_{i=0}^{K\times K}\sum_{j=0}^{M} I_{ij}^{obj}\left[\hat{C}_i \log(C_i) + \left(1-\hat{C}_i\right)\log\left(1-C_i\right)\right] - \tag{15}$$

$$\sum_{i=0}^{K\times K}\sum_{j=0}^{M} I_{ij}^{obj} \sum_{c\in classes} [\hat{p}_i(c)\log\big(p_i(c)\big) + (1-\hat{p}_i(c))\log\big(1-p_i(c)\big)].$$

Поскольку обнаружение в модели YOLOR производится в 3-х разных масштабах, то и ошибки рассчитываются для каждого предсказания. После этого сумма всех ошибок используется как общая ошибка для оптимизации модели с помощью алгоритма обратного распространения ошибки [5].

Обычно функция обучения нейронных сетей, в том числе и сверточных, может быть представлена формулой (16).

$$\begin{aligned} y &= f_\theta(x) + E, \\ E &\to min \end{aligned} \tag{16}$$

где y – решаемая задача (реальный ответ);

f – работа нейронной сети;

θ – набор параметров весовых коэффициентов;

E – ошибка предсказания.

Минимизируя ошибку, результаты работы сети лучше аппроксимируют реальную задачу [4].

В отличие от остальных детекторов, в YOLOR используются неявные знания, принимающие участие в работе модели, которые представляют из себя некий вектор, матрицу, тензор, или отдельную сеть и также нуждаются в оптимизации [11]. Функция обучения сети, использующей неявные знания, может быть представлена формулой (17).

$$\begin{aligned} y &= f_\theta(x) + E + g_\varphi\big(E_{ex}(x), E_{im}(z)\big), \\ E &+ g_\varphi(E_{ex}(x), E_{im}(z)) \to min \end{aligned} \tag{17}$$

где g_φ – некая функция, описывающая операцию, производимую над явным и неявным знанием (объединение, умножение, конкатенация);

E_{ex} и E_{im} – операции, моделирующие явную и неявную ошибку соответственно;

z – неявные знания, представляемые в виде константной структуры данных (вектор, матрица, тензор).

Стоит отметить, что сети, для формирования как явного, так и неявного представления оптимизируются одинаково с помощью алгоритма обратного распространения ошибки [11].

Основная идея алгоритма обратного распространения ошибки состоит в поиске глобального минимума в многомерном пространстве весовых коэффициентов с помощью движения в противоположную сторону от направления градиента [24]. Для нахождения этого минимума находятся частные производные по всем весовым коэффициентам, что позволяет определить их вклад в общую ошибку и правильно скорректировать. Представив сверточную нейронную сеть в виде графа, можно описать этап обратного распространения ошибки для операции свертки рисунком 41.

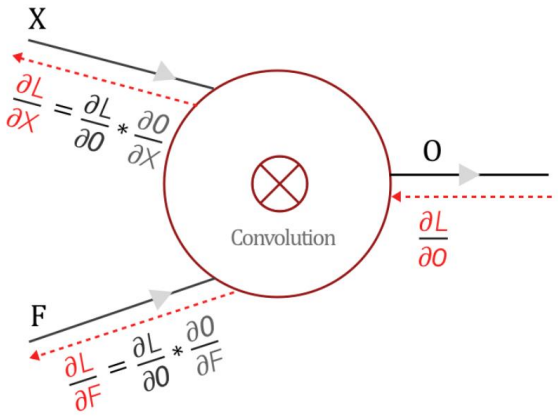


Рисунок 41 – Обратное распространение ошибки в сверточной нейронной сети

Как видно из рисунка 41, входные значения X и F являются картой признаков и фильтром соответственно. Результат свертки в виде выходной карты признаков обозначен как O. Градиент ошибки по отношению к выходу O представлен как $\frac{\partial L}{\partial O}$. Вклад входной карты признаков и фильтра в ошибку L обозначены как $\frac{\partial L}{\partial X}$ и $\frac{\partial L}{\partial F}$, и находятся как произведение $\frac{\partial L}{\partial O}$ на $\frac{\partial O}{\partial X}$ и $\frac{\partial O}{\partial F}$ соответственно. Частная производная ошибки L по фильтру F используется для оптимизации весовых коэффициентов фильтра F по формуле (18).

$$F_{new} = F - \eta \frac{\partial L}{\partial F}, \tag{18}$$

где F_{new} – обновленная матрица весовых коэффициентов;

F – матрица весовых коэффициентов, используемая при предыдущем прямом проходе;

η – скорость обучения.

Для расчета $\frac{\partial L}{\partial F}$ используется формула (19).

$$\frac{\partial L}{\partial F_k} = \sum_{i=1}^{M} \sum_{j=1}^{M} \frac{\partial L}{\partial O_{ij}} \times \frac{\partial O_{ij}}{\partial F_k}, \tag{19}$$

где k – индексы элемента фильтра F;

M – размер фильтра.

Из приведенной формулы видно, что градиент $\frac{\partial L}{\partial F}$ получается с помощью той же операции свертки между входной картой признаков X и градиентом ошибки $\frac{\partial L}{\partial O}$.

Частная производная ошибки L по входной карте признаков X используется для передачи ошибки в более ранние слои и обновления используемых там фильтров [24]. Для расчета $\frac{\partial L}{\partial F}$ используется формула (20).

$$\frac{\partial L}{\partial X_k} = \sum_{i=1}^{M} \sum_{j=1}^{M} \frac{\partial L}{\partial O_{ij}} \times \frac{\partial O_{ij}}{\partial X_k}. \quad (20)$$

Исходя из данной формулы следует, что градиент $\frac{\partial L}{\partial X}$ также находится с помощью свертки между повернутого на 180 градусов фильтра F и градиентом ошибки $\frac{\partial L}{\partial O}$.

Таким образом, применение алгоритма обратного распространения в сверточной нейронной сети есть ни что иное, как произведение операция свертки в обратном порядке.

Использование остаточных соединений позволяет предотвращать затухание градиента и более быстро распространять ошибку на ранние слои [18]. Математически остаточные соединения можно представить в виде формулы (21).

$$H(x) = O(x) + x, \quad (21)$$

где H(x) – карты признаков, получаемые в результате объединения карт признаков F(x) и x;

F(x) карты признаков, получаемые непосредственно с предыдущего слоя;

x – карты признаков, получаемые с более ранних слоев с помощью остаточного соединения.

Стоит отметить, что в качестве операции объединения может выступать как сложение, так и конкатенация [11].

Исходя из формулы (21) следует, что градиенты ошибок по объединяемым картам признаков рассчитываются по формулам (22), (23).

$$\frac{\partial L}{\partial x} = \frac{\partial L}{\partial H} \times \frac{\partial H}{\partial x} = \frac{\partial L}{\partial H} \times \left(\frac{\partial O}{\partial x} + 1\right) = \frac{\partial L}{\partial H} \times \frac{\partial O}{\partial x} + \frac{\partial L}{\partial H}, \quad (22)$$

$$\frac{\partial L}{\partial O} = \frac{\partial L}{\partial H} \times \frac{\partial H}{\partial O} = \frac{\partial L}{\partial H} \times \left(1 + \frac{\partial x}{\partial O}\right) = \frac{\partial L}{\partial H} \times \frac{\partial x}{\partial O} + \frac{\partial L}{\partial H}. \quad (23)$$

Именно так происходит оптимизация всех весовых коэффициентов представленной модели.

Таким образом, была описана математическая модель модифицированного детектора YOLOR.

3.4 Оценка производительности модифицированной архитектуры детектора объектов YOLOR

Оценка производительности модифицированного детектора YOLOR включает в себя оценку точности и скорости производимых им обнаружений. Также для определения степени влияния произведенных изменений на качество обнаружения объектов необходимо провести сравнение результатов работы модифицированного и оригинального детекторов. Для сравнения с оригинальным детектором YOLOR были выбраны 2 модификации, отличающиеся глубиной экстрактора признаков и содержащие Res3Net-50 и Res3Net-101. Данные модификации были обозначены как YOLOR-LM (LM – light modification) и YOLOR-DM (DM – deep modification) соответственно.

Реализация детектора производилась на языке Python с использованием библиотеки машинного обучения PyTorch, содержащей множество функций и алгоритмов для создания архитектур любой сложности. Еще одной важной особенностью данной библиотеки является поддержка технологии CUDA, позволяющая производить вычисления с использованием графических процессоров.

Для обучения моделей был использован графический процессор Tesla T4, обладающий 16 гигабайтами видеопамяти, 2560 ядрами CUDA и 320 тензорными ядрами. В качестве обучающего набора данных использовался датасет COCO, содержащий порядка 330 тысяч размеченных изображений,

разделенных на 80 классов. Изображения из данного набора были поделены на тренировочный, тестовый и оценочный наборы данный в соотношении 70%, 25% и 5% соответственно.

Для оцениваемых моделей было произведено сравнение основных архитектурных особенностей, таких как количество слоев, количество обучаемых параметров и вычислительная сложность, выражаемая в количестве выполняемых операций с плавающей запятой. Данные характеристики приведены в таблице 5.

Таблица 5 - Характеристики архитектур YOLOR, YOLOR-LM и YOLOR-DM

Название модели	Количество слоев	Количество параметров, млн.	BFLOPs, млрд.
YOLOR	529	52.9	120,6
YOLOR-LM	2783	35.7	84
YOLOR-DM	5554	54.7	145

Как видно из таблицы 5, самой легковесной моделью является YOLOR-LM, несмотря на большее количество слоев, чем в стандартном YOLOR. Это связано с тем, что YOLOR-LM использует меньшее количество весовых фильтров в сверточных слоях благодаря иерархично-ветвистой структуре. В связи с этим также уменьшено количество обучаемых параметров и вычислительная сложность по сравнению с оригинальным детектором. Модель YOLOR-DM, напротив, является самой глубокой архитектурой с количеством слоев почти вдвое большим, чем в модификации LM. Кроме того, данная модель обладает самой большой вычислительной сложностью и количеством обучаемых параметров.

Обучение детекторов производилось на цветных изображениях с разрешением 512 на 512 пикселей. Размер партии обучения, определяющий количество изображений на основе которых вычисляется общая ошибка и происходит обновление весовых коэффициентов, был выбран равный 32. Несмотря на то, что показатель скорость обучения является динамическим в течении всего процесса обучения, для большинства эпох он составляет 0,01.

Данная скорость обучения является наиболее оптимальной и позволяет модели более корректно производить оптимизацию весовых коэффициентов, предотвращая расхождение процесса обучения [18].

На рисунке 42 приведен график изменения общей ошибки обнаружения на тестовом наборе данных в процессе обучения, выполняющимся в течении 10 эпох.

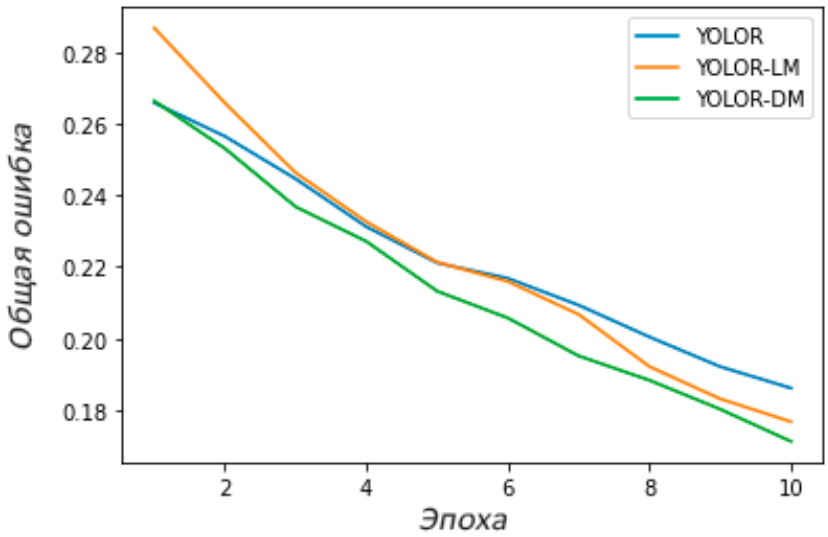


Рисунок 42 – Динамика изменения общей ошибки обнаружения в процессе обучения длительностью 10 эпох

Как видно из рисунка 42, модели YOLOR-LM и YOLOR-DM сходятся быстрее, чем модель YOLOR. Это говорит о более быстром снижении ошибок классификации, локализации и объектности, составляющих общую ошибку [28]. Более быстрая сходимость этих ошибок в модифицированных алгоритмах может заключаться в повышении точности предсказания классов или ограничивающих рамок крупных и мелких объектов благодаря увеличению размера рецептивного поля с помощью связки Res3Net+BiFPN.

Для оценки точности моделей обнаружения объектов используется метрика AP (average precision) [26]. Данная метрика основана на двух показателях – точность (precision) и отзыв (recall), расчет которых производится по формулам (24) и (25).

$$precision = \frac{TP}{TP + FP}, \quad (24)$$

$$recall = \frac{TP}{TP + FN}, \quad (25)$$

где TP – все правильные обнаружения, произведенное моделью;

FP – обнаружения, произведенные моделью, которые не являются правильными;

FN – объекты, которые не были обнаружены моделью.

Для определения того, к какому классу отнести обнаружение – к TP (true positive) или к FP (false positive) используется метрика пересечение по объединению (intersections over union), которая была представлена в формуле 1 пункта 2.1. Вначале задается некоторое пороговое значение от 0 до 1, если результат пересечения по объединению между предсказанной ограниченной рамки и реальной рамкой объекта равен этому пороговому значению или превосходит его, то данное обнаружение классифицируется как TP, в ином случае – как FP [26]. Значение IoU равное 1 означает, что предсказание полностью соответствует действительной ограничивающей рамке объекта. IoU равное 0 говорит о том, что предсказание не пересекается с действительной рамкой объекта [27]. Чаще всего пороговое значение для определения TP устанавливается в 0,5, более редко используют 0,75 и 0,95.

Если обученный детектор имеет высокие показатели точности и отзыва, это говорит о том, что он правильно обнаруживает большую часть объектов. В иных случаях высокий показатель отзыва совместно с низким показателем точности говорит о преобладающем количестве неверных обнаружений, а низкий показатель отзыва совместно с высоким показателем точности сигнализирует о том, что большая часть объектов не была обнаружена [26].

После расчета показателей точности и отзыва для каждого изображения из тестового набора данных, строится кривая из точек с координатами, соответствующими этим показателям для каждого класса объектов. Для определения средней точности (average precision) обнаружения конкретного

класса, необходимо найти площадь под кривой точность-отзыв для данного класса [27]. Расчет площади под кривой производится по формуле (26).

$$AP = \int_0^1 p(r)dr, \tag{26}$$

где p(r) – показатель точности, соответствующий отзыву на кривой.

Для нахождения средней точности детектора по всем классам показатели AP складываются и делятся на количество классов [26]. Данная точность называется усредненной средней точностью или математическим ожиданием средней точности и обозначается как mAP (mean average precision).

Кривые точности-отзыва и значения mAP для детекторов YOLOR, YOLOR-LM и YOLOR-DM представлены на рисунке 43.

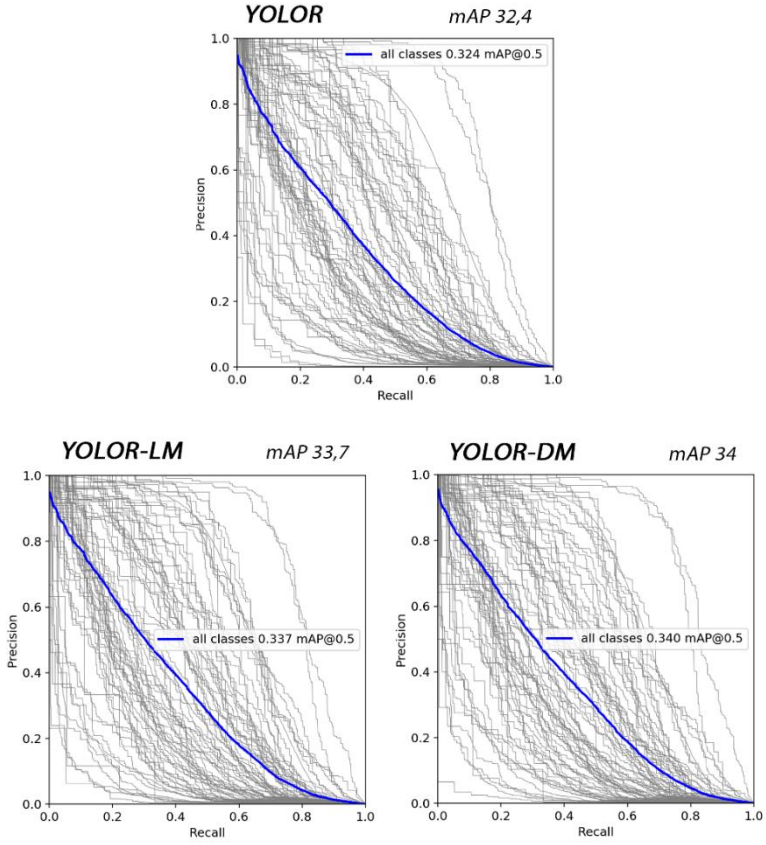


Рисунок 43 – Показатели точности моделей, обученных в течении 10 эпох

Как видно из рисунка 43, модели модифицированных детекторов YOLOR обладают большей точностью обнаружения объектов на проверочной выборке. Точность модели YOLOR-LM составляет 33,7 mAP, что выше на 4%, чем у детектора YOLOR, показатель mAP которого равен 32,4. Модель YOLOR-DM обладает 34 mAP, что выше на 5% по отношению к оригинальному детектору YOLOR.

Для оценки скорости детекторов была рассчитана скорость обнаружения объектов и скорость работы алгоритма NMS, также был произведен расчет количества обрабатываемых кадров в секунду (FPS) для оценки пригодности детектора в приложениях реального времени. Результаты представлены в таблице 6.

Таблица 6 - Показатели скорости работы детекторов YOLOR, YOLOR-LM и YOLOR-DM

Название модели	Скорость обнаружения, мс.	Скорость работы алгоритма NMS, мс.	Общая скорость, мс.	FPS
YOLOR	3,8	6,1	9,8	102
YOLOR-LM	3,1	5,7	8,8	113
YOLOR-DM	6,2	6,7	12,9	77

Как видно из результатов, представленных в таблице, модель YOLOR-LM обладает самой высокой скоростью обнаружения, что обуславливается меньшей вычислительной сложностью из-за наименьшего количества производимых операций с плавающей запятой. Модель YOLOR-DM, напротив, обладает самой низкой скоростью обнаружения за счет увеличенной сложности своей архитектуры.

Линейчатая диаграмма с показателями производительности детекторов YOLOR, YOLOR-LM и YOLOR-DM представлена на рисунке 44. Диаграмма показывает, что детектор YOLOR-LM превосходит оригинальный детектор YOLOR как по скорости, так и по точности обнаружения, в то время как

детектор YOLOR-DM обладает лучшей точность, уступая двум другим детекторам в скорости обнаружения.

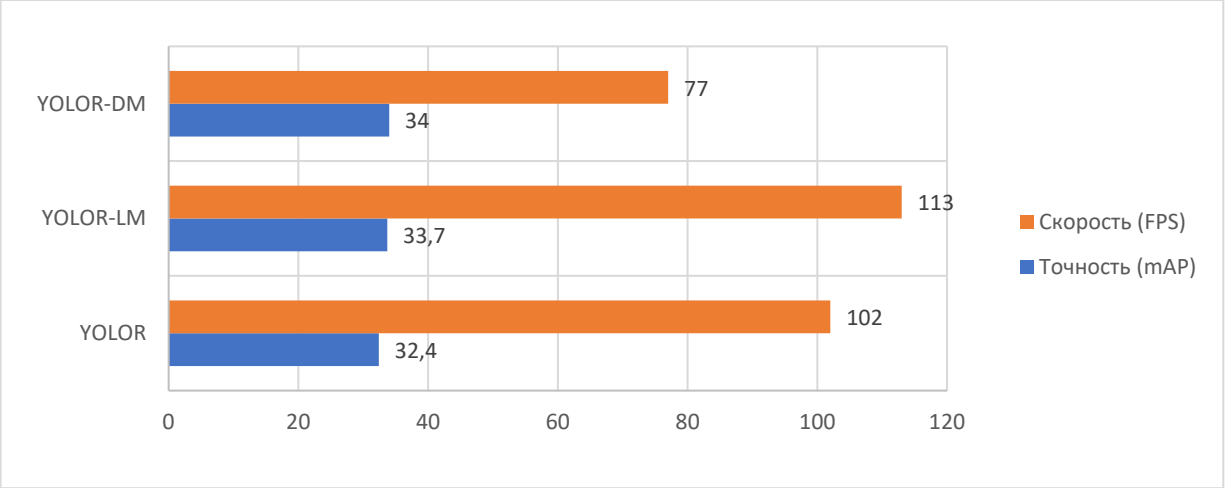


Рисунок 44 – Диаграмма производительности детекторов YOLOR, YOLOR-LM и YOLOR-DM

Для визуального определения качества производимых обнаружений объектов было проведено тестирование работы детекторов на одних и тех же проверочных наборах данных. Результаты обнаружения объектов представлены на рисунке 45, который еще раз подтверждает, что детекторы YOLOR-LM и YOLOR-DM, полученные в результате модификации, обладают большей точностью обнаружения объектов, чем YOLOR. Из полученных результатов можно судить, что продолжение обучения рассмотренных моделей позволит еще больше снизить общую ошибку обнаружения объектов детекторами, повышая при этом их показатели mAP. Однако, общая картина сравнения производительности детекторов между собой не измениться. Для детектора YOLOR, обученного в течении определенного количества эпох, найдется такой детектор YOLOR-LM, превосходящий его в точности и скорости обнаружения и такой детектор YOLOR-DM, превосходящий оба этих детектора в точности обнаружения.

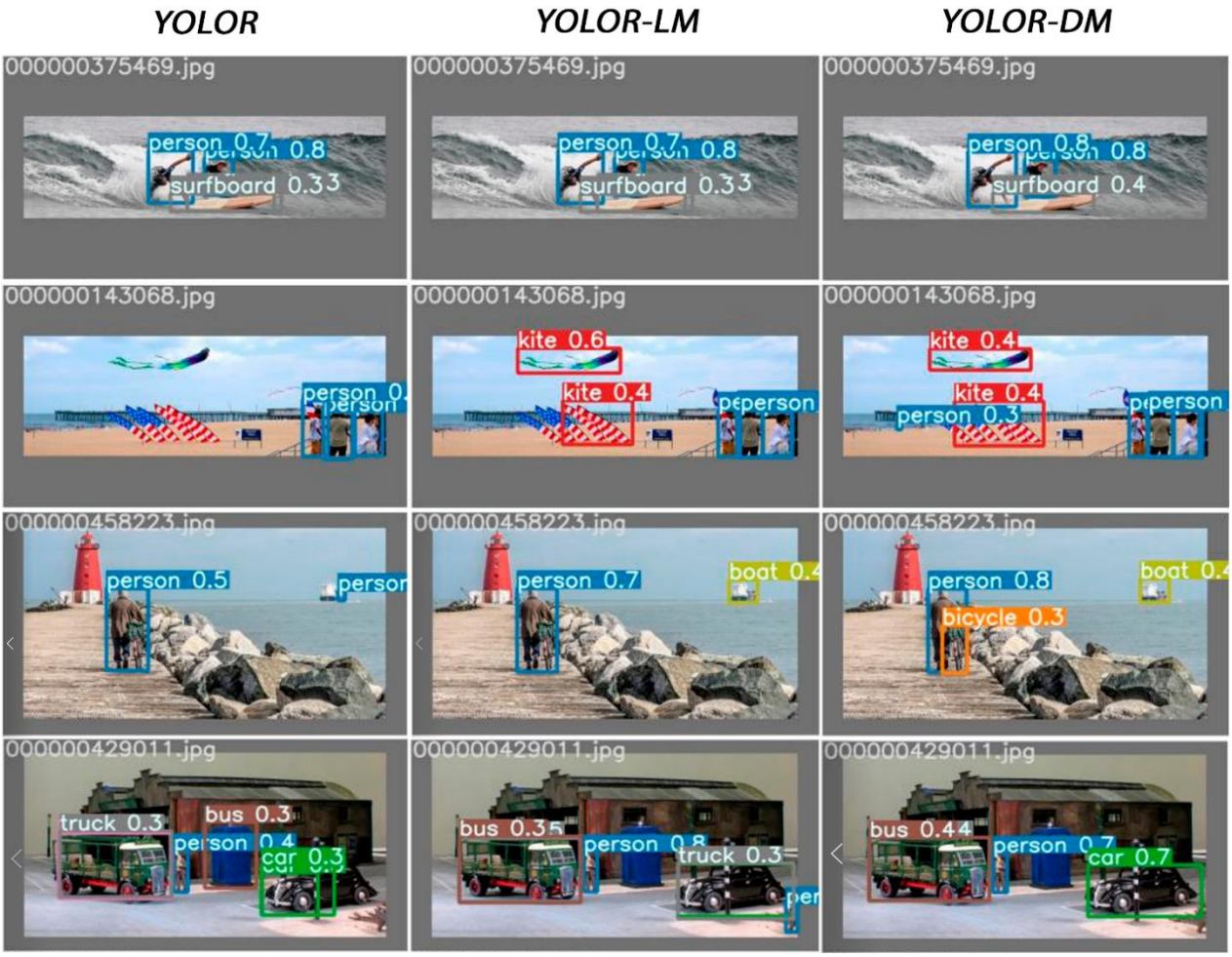


Рисунок 45 - Результаты обнаружения объектов на изображении детекторами YOLOR, YOLOR-LM и YOLOR-DM

Вывод по разделу 3

Итак, в данном разделе была проанализирована сильная сторона детектора YOLOR - применение неявных знаний в процессе обнаружения объектов, которые также будут использоваться в предлагаемом детекторе объектов, поскольку способствуют повышению точности и скорости обнаружения. На основе структурного анализа YOLOR было предложено несколько улучшений.

В качестве экстрактора признаков была выбрана сконструированная в ходе текущего исследования сверточная архитектура Res3Net, объединяющая преимущества CSPResNeXt и Res2Net, которые обладают лучшей производительностью по сравнению с экстрактором признаков CSPDarknet.

Шея детектора YOLOR, состоящая из сети уточнения признаков PAN и блока пространственного объединения признаков SPP, была заменена на блочную сеть уточнения признаков BiFPN, которая позволяет гибко масштабировать размер рецептивного поля, повышая точность обнаружения и сохраняет пространственную информацию об объектах, которая могла бы быть утеряна при использовании SPP.

Также были описаны математические методы, применяемые в детекторе YOLOR.

В данном разделе были описаны основные параметры, используемые при обучении оцениваемых детекторов объектов, а также описано окружение, в котором оно проводилось. Для обученных в течении 10 эпох моделей детекторов объектов YOLOR-LM, YOLOR-DM и YOLOR был проведен сравнительный анализ производительности. В качестве оцениваемых параметров выступали точность обнаружения, рассчитываемая по метрике AP, и скорость обнаружения (FPS). В результате сравнительного анализа было выявлено, что модификация YOLOR-LM на 4% точнее и на 30% быстрее по сравнению с оригинальным детектором YOLOR. Другая модификация YOLOR-DM, обладающая самой глубокой сетевой структурой повышает точность на 5% по сравнению с детектором YOLOR, однако медленнее него на 25%.

Заключение

В ходе выполнения магистерской диссертации были рассмотрены основные классы методов обнаружения объектов на изображении: на основе машинного и глубокого обучения. Методы, основанные на машинном обучении, применяются в основном для узкого круга задач. Основными недостатками данных методов является низкая скорость распознавания и плохая обобщающая способность. Лучшим подходом для обнаружения объектов на изображении является метод, основанный на глубоком обучении, а именно сверточные нейронные сети.

Для оценки влияния отдельных компонентов детектора на точность и скорость обнаружения был проведен анализ основных принципы работы детекторов объектов на изображении на основе СНС и выполнен сравнительный анализ основных сверточных архитектур, выступающих в качестве структурных частей множества детекторов. Также, были рассмотрены особенности реализации некоторых современных детекторов объектов на изображении, таких как: R-CNN, R-FCN, YOLO, SSD, RetinaNet, RefineDet и YOLOR. В рамках сравнительного анализа данных детекторов были описаны их преимущества и недостатки, а также произведена оценка их показателей точности и скорости обнаружения объектов на изображении. На основании результатов сравнительного был сделан вывод, что одноступенчатые детекторы являются более подходящими для обнаружения объектов в режиме реального времени, за счет лучшего быстродействия. Кроме того, среди рассмотренных детекторов объектов, наилучшим оказался YOLOR, обладающий самыми высокими показателями точности и скорости обнаружения.

Для улучшения детектора YOLOR был проведен анализ его сильных и слабых частей. Главной особенностью детектора YOLOR является применение неявных знаний в процессе обнаружения объектов, которая была

сохранена в модифицированном детекторе объектов, поскольку способствуют повышению точности и скорости обнаружения.

В процессе модификации детектора YOLOR было сделано несколько улучшений: в качестве экстрактора признаков была выбрана сконструированная в ходе текущего исследования сверточная архитектура Res3Net, объединяющая преимущества CSPResNeXt и Res2Net, которые обладают лучшей производительностью по сравнению с экстрактором признаков CSPDarknet, также шея детектора YOLOR, состоящая из сети уточнения признаков PAN и блока пространственного объединения пирамид SPP, была заменена на блочную сеть уточнения признаков BiFPN, которая позволяет гибко масштабировать размер рецептивного поля, повышая точность обнаружения и сохраняет пространственную информацию об объектах, которая могла бы быть утеряна при использовании SPP.

Для оценки были обучены две модификации детектора: YOLOR-LM и YOLOR-DM, отличающиеся глубиной сетевой архитектуры. В качестве оцениваемых параметров выступали точность обнаружения, рассчитываемая по метрике AP, и скорость обнаружения (FPS). В результате сравнительного анализа было выявлено, что модификация YOLOR-LM на 4% точнее и на 30% быстрее по сравнению с оригинальным детектором YOLOR. Другая модификация YOLOR-DM, обладающая самой глубокой сетевой архитектурой, повышает точность на 5% по сравнению с детектором YOLOR, однако медленнее него на 25%.

Из полученных результатов исследования следует, что для детектора YOLOR, найдется такой детектор YOLOR-LM, превосходящий его в точности и скорости обнаружения и такой детектор YOLOR-DM, превосходящий оба этих детектора в точности обнаружения, но обладающий меньшим быстродействием.

Таким образом, была подтверждена гипотеза о том, что производительность детектора во многом зависит от правильного конструирования сверточных архитектур, входящих в его состав.

Список используемой литературы

1. Бурков А. Машинное обучение без лишних слов / Андрей Бурков - Питер СПб, 2020. – 192 с.

2. Вьюгин В. Математические основы машинного обучения и прогнозирования / Владимир Вьюгин. - МЦНМО, 2014. - 304 с.

3. Гелиг А., Матвеев А. Введение в математическую теорию обучаемых распознающих систем и нейронных сетей. Учебное пособие / Аркадий Гелиг, Алексей Матвеев - Издательство СПбГУ, 2014. – 224 с.

4. Abu-Mostafa Y. Learning From Data / Yaser S. Abu-Mostafa, Malik Magdon-Ismail, Hsuan-Tien Lin – AMLBook. – 2012.-Jan. -C. 213.

5. Alexey Bochkovskiy. YOLOv4: Optimal Speed and Accuracy of Object Detection, 2020 // arXiv [Электронный ресурс]: открытый архив научных статей. URL: https://arxiv.org/abs/2004.10934 (дата обращения 10.06.2021).

6. Amit Kumar Sinha. Application of Deep Learning in Object Detection: Application of Deep Learning in Object Detection using Tensorflow // Amit Kumar Sinha, Adarsha Ruwali, Abhilash Jha. LAP LAMBERT. - 2017. -Dec. -C. 56.

7. Bishop C. Pattern Recognition and Machine Learning (Information Science and Statistics) / Christopher M. Bishop - Springer-Verlag New York Inc. -2007.-Feb. -C. 738.

8. Can Zhang. PAN: Towards Fast Action Recognition via Learning Persistence of Appearance, 2020 // arXiv [Электронный ресурс]: открытый архив научных статей. URL: https://arxiv.org/abs/2008.03462 (дата обращения 10.12.2021).

9. Cedric Picron. Trident Pyramid Networks: The importance of processing at the feature pyramid level for better object detection, 2021 // arXiv [Электронный ресурс]: открытый архив научных статей. URL: https://arxiv.org/abs/2110.04004 (дата обращения 10.12.2021).

10. Chien-Yao Wang. CSPNet: A New Backbone that can Enhance Learning Capability of CNN, 2019 // arXiv [Электронный ресурс]: открытый архив

научных статей. URL: https://arxiv.org/abs/1911.11929 (дата обращения 01.12.2021).

11. Chien-Yao Wang. You Only Learn One Representation: Unified Network for Multiple Tasks, 2021 // arXiv [Электронный ресурс]: открытый архив научных статей. URL: https://arxiv.org/abs/2105.04206 (дата обращения 01.12.2021).

12. Christian Szegedy. Scalable, High-Quality Object Detection, 2015 // arXiv [Электронный ресурс]: открытый архив научных статей. URL: https://arxiv.org/abs/1412.1441 (дата обращения 10.12.2020).

13. Edward E. David, Eyes and Ears for Computers, 1962 // IEEE Xplore [Электронный ресурс]: открытый архив научных статей. URL: https://ieeexplore.ieee.org/document/4066820 (дата обращения 10.12.2020).

14. Goodfellow I. Deep Learning (Adaptive Computation and Machine Learning series) / Ian Goodfellow, Yoshua Bengio, Aaron Courville - The MIT Press. -2016.-Nov. -C. 800.

15. Harrington P. Machine Learning in Action / Peter Harrington - Manning Publications. – 2012.-April. -C. 384.

16. Jifeng Dai. R-FCN: Object Detection via Region-based Fully Convolutional Networks, 2016 // arXiv [Электронный ресурс]: открытый архив научных статей. URL: https://arxiv.org/abs/1605.06409 (дата обращения 10.06.2021).

17. Kaiming He. Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition, 2014 // arXiv [Электронный ресурс]: открытый архив научных статей. URL: https://arxiv.org/abs/1512.03385 (дата обращения 10.12.2021).

18. Kaiming He. Deep Residual Learning for Image Recognition, 2015 // arXiv [Электронный ресурс]: открытый архив научных статей. URL: https://arxiv.org/abs/1512.03385 (дата обращения 10.12.2021).

19. Kelleher J. Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Worked Examples, and Case Studies / John D. Kelleher, Brian Mac Namee, Aoife D`Arcy - The MIT Press. – 2015.-July. – C. 624.

20. Mingxing Tan. EfficientDet: Scalable and Efficient Object Detection, 2019 // arXiv [Электронный ресурс]: открытый архив научных статей. URL: https://arxiv.org/abs/1911.09070 (дата обращения 10.12.2021).

21. Mitchell T. Machine Learning / Tom Mitchell – Mc Graw Hill India. - 2017.– Mar. – C. 432.

22. Pardhu Thottempudi. Novel Approach for detection of objects in surveillance videos // Pardhu Thottempudi. LAP LAMBERT. - 2017. -Dec. -C. 64.

23. Pramod J. Deore. Real Time Video Processing and Object Detection on Mobile // Pramod J. Deore, Shailaja Arjun Patil, Sunil B. Chaudhari. LAP LAMBERT. - 2017. -May. -C. 64.

24. Rashid T. Make Your Own Neural Network / Tariq Rashid - CreateSpace Independent Publishing Platform. – 2016. – C. 222.

25. Rojas R. Neural Networks: A Systematic Introduction / Raul Rojas, Peter Varga - Springer Berlin Heidelberg. – 1996.-Jul. -C. 522.

26. Ross Girshick. Rich feature hierarchies for accurate object detection and semantic segmentation, 2013 // arXiv [Электронный ресурс]: открытый архив научных статей. URL: https://arxiv.org/abs/1311.2524 (дата обращения 10.06.2021).

27. Roth, P.M. Survey of Appearance-Based Methods for Object Recognition // P.M. Roth, M. Winter – Technical Report ICG-TR-01/16, Institute

28. Russell S. Artificial Intelligence: Pearson New International Edition: A Modern Approach / Stuart Russel, Norvig Peter – Pearson. – 2013.-Aug. -C. 1104.

29. Saining Xie. Aggregated Residual Transformations for Deep Neural Networks, 2016 // arXiv [Электронный ресурс]: открытый архив научных статей. URL: https://arxiv.org/abs/1611.05431 (дата обращения 10.12.2021).

30. Shalev-Shwarthz S. Understanding Machine Learning: From Theory to Algorithms / Shai Shalev-Shwartz, Shai Ben-David - Cambridge University Press. - 2014.-May. -C. 415.

31. Shang-Hua Gao. Res2Net: A New Multi-scale Backbone Architecture, 2019 // arXiv [Электронный ресурс]: открытый архив научных статей. URL: https://arxiv.org/abs/1904.01169 (дата обращения 10.12.2021).

32. Shaoqing Ren. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, 2015 // arXiv [Электронный ресурс]: открытый архив научных статей. URL: https://arxiv.org/abs/1506.01497 (дата обращения 10.06.2021).

33. Shifeng Zhang. Single-Shot Refinement Neural Network for Object Detection, 2017 // arXiv [Электронный ресурс]: открытый архив научных статей. URL: https://arxiv.org/abs/1711.06897 (дата обращения 10.06.2021).

34. Sibt ul Hussain. Machine Learning Methods for Visual Object Detection // Sibt ul Hussain. Editions universitaires europeennes. -2012. -March. -C. 160.

35. Tsung-Yi Lin. Feature Pyramid Networks for Object Detection, 2016 // arXiv [Электронный ресурс]: открытый архив научных статей. URL: https://arxiv.org/abs/1612.03144 (дата обращения 10.12.2021).

36. Tsung-Yi Lin. Focal Loss for Dense Object Detection 2017 // arXiv [Электронный ресурс]: открытый архив научных статей. URL: https://arxiv.org/abs/1708.02002 (дата обращения 10.06.2021).

37. Wei Liu. SSD: Single Shot MultiBox Detector, 2016 // arXiv [Электронный ресурс]: открытый архив научных статей. URL: https://arxiv.org/abs/1512.02325 (дата обращения 10.06.2021).

38. Witten I. Data Mining: Practical Machine Learning Tools and Techniques / Ian H. Witten, Eibe Frank, Mark A. Hall - Morgan Kaufmann. – 2011.-Jan. -C. 664.

39. Xiaoyue Jiang. Deep Learning in Object Detection and Recognition. // Xiaoyue Jiang, Abdenour Hadid, Yanwei Pang, Eric Granger, Xiaoyi Feng. Springer. -2020. -Nov. -C. 240.

40. Yigithan Dedeoglu. lgorithms for Smart Video Surveillance: Moving Object Detection, Tracking and Classification // Yigithan Dedeoglu. LAP LAMBERT. -2010. -Sept. -C. 108.

41. Yunpeng Chen. Dual Path Networks, 2017 // arXiv [Электронный ресурс]: открытый архив научных статей. URL: https://arxiv.org/abs/1707.01629 (дата обращения 10.12.2021).