20. *Kingma D.P., Ba J.* Adam: A Method for Stochastic Optimization, *Int. Conf. Learn. Represent. (ICLR). May 2015*. Available at: http://arxiv.org/abs/1412.6980.
21. *Yue Z., et al.* Dual Adversarial Network: Toward Real-World Noise Removal and Noise Generation, In: Vedaldi A., et al. (Eds.) *Computer Vision – ECCV 2020*. Cham: Springer, 2020, pp. 41-58. ISBN: 978-3-030-58607-2.
22. *Wang Z., et al.* Image Quality Assessment: From Error Visibility to Structural Similarity, *IEEE Trans. Image Process*, 2004, Vol. 13, No. 4, pp. 600-612.
23. *Kousha S., et al.* Modeling sRGB Camera Noise with Normalizing Flows, *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR). Jun. 2022*, pp. 17442-17450. DOI: 10.1109/CVPR52688.2022.01694.
24. *Fu Z., Guo L., Wen B.* sRGB Real Noise Synthesizing with Neighboring Correlation-Aware Noise Model, *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2023, pp. 1683-1691. DOI: 10.1109/CVPR52729.2023.00168.
25. *Aff M., Brubaker M. A., Brown M.S.* HistoGAN: Controlling Colors of GAN-Generated and Real Images via Color Histograms, *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2020, pp. 7937-7946. Available at: https://api.semanticscholar.org/CorpusID:227151819.
26. *Li Y., et al.* LSDIR: A Large Scale Dataset for Image Restoration, *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, 2023, pp. 1775-1787. DOI: 10.1109/CVPRW59228.2023.00178.
27. *Zhang K., et al.* Beyond a Gaussian Denoiser: Residual Learning of Deep CNN for Image Denoising, *IEEE Trans. Image Process*, 2017, Vol. 26, No. 7, pp. 3142-3155. DOI: 10.1109/TIP.2017.2662206.
28. *Komatsu R., Gonsalves T.* Comparing U-Net Based Models for Denoising Color Images, *AI*, 2020, Vol. 1, No. 4, pp. 465-486. ISSN: 2673-2688. DOI: 10.3390/ai1040029. Available at: https://www.mdpi.com/2673-2688/1/4/29.
29. *Chu X., Chen L., Yu W.* NAFSSR: Stereo Image Super-Resolution Using NAFNet, *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW). Jun. 2022*, pp. 1239-1248.

Коваленко Алексей Сергеевич – Южный федеральный университет; e-mail: akov@sfedu.ru; г. Ростов-на-Дону, Россия; тел.: +79281217747; кафедра прикладной математики и программирования; ассистент.

Демяненко Яна Михайловна – Южный федеральный университет; e-mail: demyana@sfedu.ru; г. Ростов-на-Дону, Россия; тел.: +78632975111; кафедра прикладной математики и программирования; к.т.н.; доцент.

Kovalenko Aleksei Sergeevich – Southern Federal University; e-mail: akov@sfedu.ru; Rostov-on-Don, Russia; phone: +79281217747; the Department of Applied Mathematics and Programming; assistant lecturer.

Demyanenko Yana Mikhailovna – Southern Federal University; e-mail: demyana@sfedu.ru; Rostov-on-Don, Russia; phone: +78632975111; the Department of Applied Mathematics and Programming; cand. of eng. sc.; associate professor.

УДК 004.896 DOI 10.18522/2311-3103-2025-5-254-276

О.Б. Лебедев, Р.И. Черкасов

ПРИМЕНЕНИЕ ТЕХНОЛОГИЙ КОМПЬЮТЕРНОГО ЗРЕНИЯ В СИСТЕМАХ ОБРАБОТКИ ВИЗУАЛЬНОЙ ИНФОРМАЦИИ

Рассмотрено применение технологий искусственного интеллекта, в частности компьютерного зрения в системах обработки визуальной информации. Проведен комплексный анализ нейросетевых подходов к решению задач компьютерного зрения, включая систематизацию ключевых типов задач: классификацию изображений, детектирование объектов и семантическую сегментацию. Детально исследованы архитектурные принципы сверточных нейронных сетей с акцентом на механизмы извлечения пространственных признаков через сверточные слои, оптимизацию представления данных посредством операций пулинга и преобразование признаков в полносвязных слоях. Особое внимание уделено эволюции методов обнаружения объектов, где задача выбора модели рассмотрена как расширение классификации за счет интеграции регрессии пространственных координат, а также проведена оценка эффективности детекторов на основе метрик IoU, Precision, Recall и F1-score, демонстрирующих фундаментальный компромисс между точностью локализации и скоростью обработки. В качестве оптимального решения для систем реального времени представлен алгоритм YOLOv7, архитектура которого основана на разбиении входного

изображения на сетку S×S ячеек с прямым предсказанием параметров ограничивающих рамок (координаты центра, ширина, высота) и вероятностей классов для каждой ячейки, а также использовании специализированных слоёв (SPP, PANet) для мультимасштабной агрегации признаков. Структура нейронной сети подтверждает эффективность используемого подхода, обеспечивающего высокое быстродействие без критического снижения точности в стратегически важных приложениях видеонаблюдения, автономных систем и дополненной реальности. Проведено сравнительное исследование одноэтапных и двухэтапных детекторов с оценкой их производительности по ключевым метрикам. Особое внимание уделено практическим аспектам применения технологий компьютерного зрения в реальных системах обработки визуальной информации.

Глубокое обучение; сверточные нейронные сети; детектирование объектов; семантическая сегментация; метрики оценки; реальное время.

O.B. Lebedev, R.I. Cherkasov

APPLICATION OF COMPUTER VISION TECHNOLOGIES IN VISUAL INFORMATION PROCESSING SYSTEMS

This paper considers the application of artificial intelligence technologies, in particular computer vision, in visual information processing systems. A comprehensive analysis of neural network approaches to solving computer vision problems is carried out, including systematization of key types of problems: image classification, object detection and semantic segmentation. The architectural principles of convolutional neural networks are studied in detail with an emphasis on the mechanisms of spatial feature extraction through convolutional layers, optimization of data representation through pooling operations and feature transformation in fully connected layers. Particular attention is paid to the evolution of object detection methods, where the problem of model selection is considered as an extension of classification due to the integration of spatial coordinate regression, and an assessment of the effectiveness of detectors is carried out based on the IoU, Precision, Recall and F1-score metrics, demonstrating a fundamental trade-off between localization accuracy and processing speed. The YOLOv7 algorithm is presented as an optimal solution for real-time systems. Its architecture is based on splitting the input image into a grid of S×S cells with direct prediction of the bounding box parameters (center coordinates, width, height) and class probabilities for each cell, as well as the use of specialized layers (SPP, PANet) for multi-scale feature aggregation. The structure of the neural network confirms the effectiveness of the approach used, which ensures high performance without critically reducing accuracy in strategically important applications of video surveillance, autonomous systems, and augmented reality. A comparative study of one-stage and two-stage detectors was conducted with an assessment of their performance by key metrics. Particular attention is paid to the practical aspects of using computer vision technologies in real visual information processing systems.

Deep learning; convolutional neural networks; object detection; semantic segmentation; evaluation metrics; real time.

Введение. Современные достижения в области искусственного интеллекта совершили революцию в способе взаимодействия машин с окружающим миром, наделяя их способностью «видеть» и «понимать» визуальную информацию. Ключевым элементом этой революции стало компьютерное зрение – раздел искусственного интеллекта, позволяющий компьютерам и машинам анализировать, интерпретировать изображения и видеозаписи. Это не просто распознавание отдельных пикселей, а глубокое понимание содержимого визуальной информации: распознавание объектов, определение их местоположения, анализ взаимосвязей между ними, оценка контекста [1–3].

Компьютерное зрение опирается на мощные инструменты машинного обучения и сложные нейронные сети, которые, подобно человеческому мозгу, обучаются на огромных массивах данных. Эти сети «учат» компьютер отличать кошку от собаки, определять номерной знак автомобиля, распознавать лица людей, анализировать медицинские снимки и многое другое. Процесс обучения включает в себя предоставление сети множества примеров – фотографий, видеороликов, с подробным описанием их содержимого. Сеть анализирует эти данные, выявляя закономерности и устанавливая связи между пикселями и смыслом изображения. В результате компьютер приобретает способность самостоятельно обрабатывать новые, ранее не виденные, визуальные данные и выдавать точные и обоснованные результаты [3–5].

В отличие от человека, подверженного усталости, стрессам и субъективным ошибкам, системы компьютерного зрения работают непрерывно, обеспечивая высокую точность и скорость обработки огромных объемов визуальных данных. Например, анализируя потоки данных с дорожных камер, они могут в реальном времени отслеживать дорожную ситуацию, выявлять аварии, заторы и оптимизировать работу светофоров, что способствует повышению безопасности дорожного движения и снижению транспортных заторов. Это всего лишь малая часть примеров применения компьютерного зрения, его потенциал огромен, и с каждым днем эта технология находит все более широкое применение в самых разных сферах человеческой жизни [6].

Компьютерное зрение прошло путь от теоретической идеи до надежной технологии, активно стимулирующей инновации в различных сферах. Его эволюция отмечена ключевыми этапами: в 1950-1960-х годах началась разработка алгоритмов обработки визуальной информации, однако ограниченные вычислительные ресурсы сдерживали прогресс [7, 8].

Значительный рывок произошел в 1970-х с улучшением алгоритмов, таких как преобразование Хафа для детектирования линий и фигур, и появлением оптического распознавания символов (OCR), позволившего машинам читать текст [7, 8].

В 1980-1990-х годах машинное обучение стало важной составляющей, заложив основу для будущих прорывов. Настоящая революция случилась в 2000-2010-х годах с приходом глубокого обучения, которое коренным образом изменило способность машин интерпретировать визуальные данные, значительно расширив функционал в области идентификации объектов, анализа движения и решения сложных задач [7, 8].

Сегодняшний этап развития компьютерного зрения характеризуется бурным ростом, кардинально меняя методы решения задач в медицине, сфере беспилотников и концепции «умных» городов. Ключевую роль здесь играют модели, работающие мгновенно, как например YOLO (*You Only Look Once*) – они делают практическое применение «машинного зрения» одновременно и эффективным, и высокоточным [8]. «Сердцем» таких систем служат нейронные сети. Эти алгоритмы, по сути, копируют принципы работы человеческого мозга, чтобы «понимать» изображения. Среди них особой мощью обладают свёрточные нейронные сети (*CNN, Convolutional Neural Networks*), превосходно вычленяя мельчайшие детали вроде границ объектов или их характерных форм. Чтобы сделать визуальную информацию проще для анализа, применяют техники, например объединение (пулинг). Их задача – сконцентрироваться на самых информативных участках кадра. Затем последующие слои сети берут эту сжатую информацию и используют её для конкретных целей: опознать что-то или найти объект на изображении. Новейшие разработки, такие как продвинутые версии YOLO, созданные с упором на быстродействие и точность, позволяют обрабатывать картинки буквально на лету. Весь путь от «сырого» изображения до полезного вывода в компьютерном зрении обычно состоит из нескольких обязательных этапов [8–10].

Начало – это **получение изображений**: данные захватываются камерами или сенсорами, причем их качество напрямую «завязано» на возможностях датчика. Дальше идет **обработка изображений**: здесь собранные данные приводят в порядок – убирают шумы, подчеркивают контуры (например, через выделение границ), подготавливая их к глубокому анализу. Следующий шаг – **извлечение признаков**: на этом этапе выявляют и выдвигают на первый план критически важные элементы изображения – его формы, текстуры, уникальные черты. Завершающая стадия – **распознавание образов**: используя алгоритмы машинного обучения, система анализирует выделенные признаки, чтобы выполнить конечную задачу – найти объект, отследить его перемещение или классифицировать увиденное [8, 11–14].

Практическое применение моделей компьютерного зрения, включая YOLO-модели, охватывает множество отраслей. В здравоохранении технологии детектирования и классификации объектов ускоряют и повышают точность диагностики по медицинским снимкам, выявляя детали, невидимые человеческому глазу. Для автономных транспортных систем машинное зрение критически важно, обеспечивая распознавание дорожных знаков (в том числе текста на них), светофоров и пешеходов в режиме реального време-

ни. В сельском хозяйстве автоматизация процессов посева, полива и сбора урожая, а также мониторинг состояния растений для раннего выявления болезней или дефицита питательных веществ становятся возможными благодаря компьютерному зрению. На производственных линиях оно используется для контроля процессов, проверки качества продукции и автоматизированного наблюдения, повышая скорость, точность и снижая затраты за счет минимизации ошибок. По своей сути компьютерное зрение предоставляет машинам принципиально новый способ взаимодействия с окружающей средой, наделяя их способностью «видеть» и интерпретировать мир подобно человеку. Можно сказать, что на сегодняшний день использование компьютерного зрения позволяет увеличить эффективность и безопасность автономных мобильных объектов, более точно проводить диагностику в области медицины, анализировать активность покупателей различных товаров и выполнять сельскохозяйственные работы наиболее рациональным образом. Таким образом, несмотря на имеющиеся проблемы (стоимость внедрение таких технологий достаточно высока, надежность моделей машинного обучения низкая) применение технологий искусственного интеллекта, в частности компьютерного зрения, в различных сферах деятельности человека и промышленности в будущем является важной и перспективной задачей [15].

В работе излагается применение технологий искусственного интеллекта, в частности компьютерного зрения в системах обработки визуальной информации [8, 10–15].

Таксономия задач компьютерного зрения. Компьютерное зрение решает спектр задач возрастающей семантической сложности и пространственной детализации, формируя иерархию от глобального понимания сцены до точного анализа объектов. На первом уровне находится классификация изображений, где модель присваивает всей сцене единую метку (например, «кошка» или «городской пейзаж»), анализируя общее содержание без локализации элементов. Следующий уровень семантическая сегментация – требует пиксельной точности: каждый пиксель изображения классифицируется по категориям («дорога», «здание», «человек»), что позволяет разделять области по смыслу, но без различения отдельных экземпляров объектов одного класса (например, все пешеходы получают одинаковую метку). Более сложная задача обнаружения объектов добавляет к классификации пространственную локализацию: модель идентифицирует множественные объекты, рисуя вокруг них ограничительные рамки (*Bounding Boxes*) и присваивая классы (например, «автомобиль: координаты *X,Y*, ширина *W*, высота *H*»). Пик вершины иерархии занимает сегментация экземпляров, объединяющая детектирование и семантическую сегментацию: здесь не только определяются точные пиксельные границы каждого объекта, но и различаются отдельные экземпляры одного класса (например, выделение масками каждого из пяти яблок в корзине с присвоением уникальных идентификаторов). Эта прогрессия отражает эволюцию от абстрактного восприятия («что изображено?») через пространственный анализ («где находятся области?») к точечной работе с объектами («какие именно сущности присутствуют и каковы их границы?»), последовательно наращивая требования к детализации вывода и вычислительной сложности моделей [16, 17].

1. Классификация: присвоение изображению единой метки (например «кошка»).
2. Семантическая сегментация: пиксель-точная разметка изображения по классам (небо, дорога, здание).
3. Детектирование объектов: локализация и классификация множества объектов через ограничивающие рамки (*Bounding Boxes*).
4. Сегментация экземпляров: выделение пикселей каждого отдельного объекта (разные автомобили на парковке) [1,5].

Формально задача классификации определяется как поиск аппроксимирующей функции: $X \rightarrow Y$ на основе конечной обучающей выборки $X\text{^}N = \{x1, x2, ..., xN\}$, где X – пространство объектов (изображений), Y – множество меток классов. Признаковое описание объекта x формируется как вектор $(f_1(x), f_2(x), ..., f_k(x))$ в пространстве признаков [1, 2]:

$$D=Df_1 \times Df_2 \times \ldots \times D_{fk}.$$

Детектирование расширяет эту постановку, требуя одновременного решения задач классификации и регрессии координат [11, 14, 16].

Архитектурные основы сверточных нейронных сетей и эволюция детекторов. Сверточные нейронные сети (CNN) радикально изменили обработку изображений, сохраняя их пространственную структуру [1, 17–19]. Их ключевые компоненты:

♦ Сверточные слои: применяют обучаемые фильтры (ядра), извлекающие локальные признаки (края, текстуры). Операция свертки (рис. 1) генерирует карты признаков, глубина которых соответствует числу фильтров.

♦ Пулинговые слои (*Subsampling*): уменьшают пространственную размерность (*Max-Pooling*), повышая инвариантность к малым смещениям и снижая вычислительную сложность.

Полно связные слои: агрегируют высокоуровневые признаки для финальной классификации/регрессии.

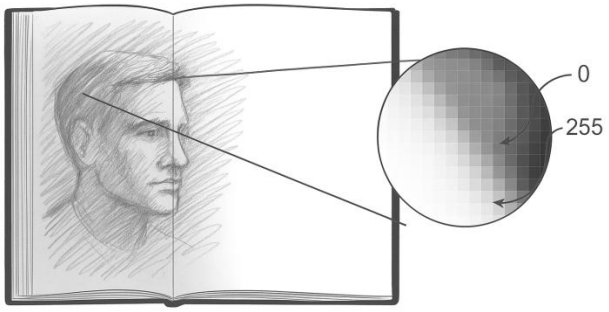

Рис. 1. Пример интерпретации изображения в числа

В области детектирования объектов доминируют две принципиально различные методологические парадигмы. Первая категория – двухэтапные детекторы, представленные семейством архитектур *R-CNN* (*Region-based Convolutional Neural Networks*), включая их эволюционные версии *Fast R-CNN* и *Faster R-CNN*. Эти системы функционируют по двухстадийной схеме: на начальном этапе генерируются «регион-предложения» (*Region Proposals*) – потенциальные зоны интереса, содержащие объекты-кандидаты; затем следует этап детальной обработки, где каждая предложенная область независимо классифицируется и подвергается регрессионному анализу для уточнения координат ограничивающих рамок [20].

Главное достоинство при использовании такого подхода заключается в получении довольно высокой точности при выполнении детектирования объектов. Это особенно важно если происходит распознавание сложных фрагментов с пересекающимися или же частично совмещенными объектами [9, 20, 21].

В то же время серьезный минус такого метода состоит в его повышенной затраты на вычислительные ресурсы. Такая зависимость приводит к недостаточной скорости обработки изображений, что сужает рамки использования данного метода в текущий момент времени (режим on-line) [8, 12].

Вторую категорию составляют одноэтапные детекторы, такие как *YOLO* (*You Only Look Once*) и *SSD* (*Single Shot MultiBox Detector*), которые реализуют принципиально иную философию обработки. Эти архитектуры выполняют прямое предсказание классов объектов и координат их ограничивающих рамок за единственный проход изображения через нейросеть (рис. 2), минуя стадию генерации «регион-предложений». Основное преимущество – исключительная скорость обработки, достигающая частот в десятки кадров в секунду, что позволяет использовать их в системах реального времени. Историческим недостатком являлось некоторое снижение точности, особенно при детектировании мелких объектов и сцен с высокой плотностью объектов [8, 13, 14]. Ярким представителем

этого направления является *YOLOv7* – современная эволюция семейства *YOLO*. Его архитектурная инновация заключается в разделении входного изображения на регулярную сетку *S*×*S*, где каждая ячейка независимо предсказывает несколько *bounding boxes* (*BBox*) и вероятности принадлежности к классам. Модель интегрирует специализированные модули, такие как *Spatial Pyramid Pooling* (*SPP*) для агрегации контекста разного масштаба и *Path Aggregation Network* (*PANet*) для эффективного комбинирования признаков из различных слоев (рис. 3). Ключевым прорывом *YOLOv* стала концепция «*trainable bag-of-freebies*» – набор методов оптимизации процесса обучения (например, продвинутая аугментация данных, специфические функции потерь), которые существенно повышают точность без увеличения вычислительных затрат на этапе инференса [7, 8, 14, 19, 21].

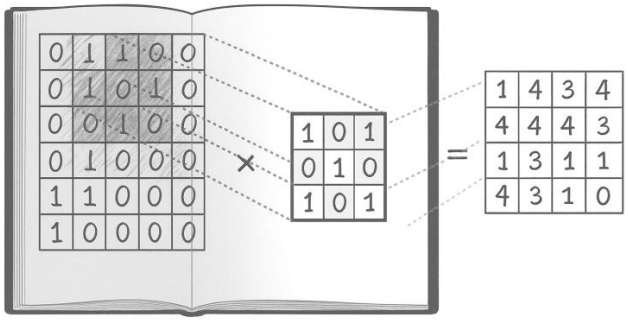

Рис. 2. Выполнение операции свертки

Сверточный слой служит фундаментальным структурным блоком сверточных нейронных сетей (*CNN*), выполняя операцию извлечения пространственных признаков посредством специализированных фильтров (ядер) – компактных матриц обучаемых весовых коэффициентов (типично 3×3 или 5×5 пикселей).

Принцип работы слоя заключается в систематическом скольжении каждого фильтра по всей поверхности входного изображения или карты признаков предыдущего слоя. На каждой позиции фильтр накладывается на локальную область (рецептивное поле), где происходит поэлементное умножение значений весов фильтра на соответствующие значения активаций под ним. Полученные произведения затем суммируются в единое скалярное значение, формирующее элемент новой выходной карты признаков (*feature map*).

Пространственная размерность этой карты динамически определяется тремя ключевыми параметрами: размером ядра (большие фильтры сильнее сокращают разрешение), шагом сдвига (*stride*) определяющим расстояние перемещения фильтра после каждой операции (стандартно 1-2 пикселя; увеличение шага резко уменьшает выходные размеры), и дополнением (*padding*) – добавлением нулевых или иных пикселей по границам входа для сохранения размерности карты или контроля степени её уменьшения [7, 8, 14–17].

Применение ансамбля разнородных фильтров в пределах одного слоя позволяет сети параллельно детектировать спектр низкоуровневых визуальных паттернов: отдельные фильтры специализируются на выделении ориентированных границ (вертикальных, горизонтальных, диагональных), угловых структур, текстурных особенностей или локальных контрастов.

Каждый независимый фильтр генерирует собственную карту признаков, а их совокупность образует многоканальный выходной тензор, где глубина соответствует количеству фильтров. Критическое свойство *CNN* проявляется в формировании иерархии абстракций через последовательность сверточных слоев:

♦ начальные слои реагируют на элементарные локальные паттерны (края, цветовые переходы, точки);

♦ промежуточные слои комбинируют эти примитивы в сложные конфигурации (углы, простые геометрические формы, текстурированные области);

♦ глубинные слои активируются на семантически значимых компонентах целых объектов (например, «колесо автомобиля» или «окно здания»), интегрируя информацию из обширных рецептивных полей.

Эта прогрессия обеспечивает переход от пиксельной обработки к семантическому пониманию сцены [6, 7, 9, 17–20]. Завершающие этапы архитектуры *CNN* часто включают полносвязные слои (иллюстрируемые на рис. 3), которые агрегируют высокоуровневые пространственные признаки для финальной классификации или регрессии.

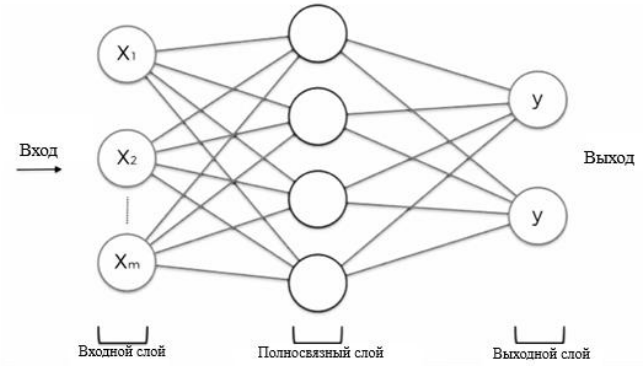

Рис. 3. Пример полностью связного слоя

Пулинговые слои (*Pooling Layers*) являются неотъемлемым компонентом архитектуры *CNN*, располагаясь непосредственно после сверточных операций для выполнения критически важной функции пространственного сжатия карт признаков. Наиболее распространенная операция *max-pooling* извлекает максимальное значение активации в пределах скользящего окна (типично 2×2 или 3×3 пикселя), что обеспечивает три фундаментальных преимущества: существенное снижение вычислительной сложности последующих слоев за счет уменьшения пространственного разрешения, повышение инвариантности модели к незначительным сдвигам, деформациям и шуму входных данных, а также контроль переобучения путем выделения наиболее устойчивых признаков. Альтернативные методы включают *average-pooling*, усредняющий значения в окне для сохранения фоновой информации, и *global average pooling*, выполняющий усреднение по всей карте признаков и часто заменяющий полносвязные слои в современных архитектурах для минимизации параметров. Выбор стратегии зависит от задачи: *max-pooling* доминирует при выделении ключевых признаков, тогда как *average*-пулинг эффективен для текстурных областей [8, 14].

Типичная архитектура *CNN* формируется чередованием сверточных и пулинговых слоев, создающих иерархическое представление данных, которое завершается полносвязным слоем (*Fully Connected Layer*), преобразующим многомерные карты признаков в плоский вектор для финальной классификации или регрессии. В задачах классификации изображений размерность выходного слоя строго соответствует количеству целевых классов. Однако для решения задачи детектирования объектов – ключевого направления компьютерного зрения, требующего не только идентификации, но и точной пространственной локализации объектов, – архитектура требует существенной модификации. Выходные слои в таких моделях должны генерировать координаты ограничивающих рамок (*bounding boxes*) и ассоциированные с ними метки классов, что реализуется через специализированные архитектурные модули [8, 14–19].

Эволюция методов детектирования объектов сформировала две принципиально различные архитектурные парадигмы. Двухэтапные детекторы (*R-CNN, Fast R-CNN, Faster R-CNN*) реализуют каскадную обработку: на первом этапе алгоритмически или

нейросетевыми методами генерируются регионы интереса (*Region Proposals*) – зоны-кандидаты, потенциально содержащие объекты; на втором этапе каждый регион независимо проходит через *CNN* для классификации объекта и регрессионного уточнения координат его ограничивающей рамки. Хотя этот подход обеспечивает эталонную точность детектирования, его ключевой недостаток – экстремальные вычислительные затраты – делает его непрактичным для систем, требующих работы в реальном времени. В контраст этой методологии одноэтапные детекторы (*YOLO, SSD*) применяют радикально иную стратегию, предсказывая классы объектов и параметры их ограничивающих рамок (координаты центра, ширину, высоту) за единый прямой проход изображения через нейросеть (рис. 4). Их неоспоримое преимущество – высокая скорость обработки (десятки кадров в секунду) – обеспечивает работу в реальном времени, однако традиционно достигается за счет некоторого снижения точности, особенно заметного при детектировании мелких объектов или в сценах с высокой плотностью объектов [8–15, 21].

Принцип работы одноэтапных детекторов (рис. 4) основан на концепции глобального анализа изображения как единого семантического поля. Вместо выделения дискретных регионов-кандидатов, нейросеть разделяет изображение на пространственную сетку, где каждая ячейка напрямую предсказывает вероятности классов и параметры *bounding boxes* для фиксированного числа объектов. Этот подход фундаментально устраняет ключевое узкое место двухэтапных методов – необходимость ресурсоемкой обработки тысяч перекрывающихся регионов-кандидатов, что и обеспечивает прорывное быстродействие при сохранении конкурентоспособного качества детектирования. Двухэтапные детекторы, такие как *R-CNN* (*Regions with Convolutional Neural Networks*) и его модификации (*Fast R-CNN, Faster R-CNN*), работают в два этапа [8, 12–18].

Рис. 4. Пример работы алгоритма обнаружения объектов

Сначала они генерируют предложения областей (*region proposals*), которые потенциально содержат объекты. Затем, сверточная сеть обрабатывает эти области, чтобы классифицировать и уточнить их границы. Двухэтапные детекторы обычно обеспечивают более высокую точность, но работают медленнее, чем одноэтапные [7, 9, 11]. Выбор между одноэтапными и двухэтапными методами зависит от конкретных требований задачи – баланса между скоростью и точностью, рис. 5.

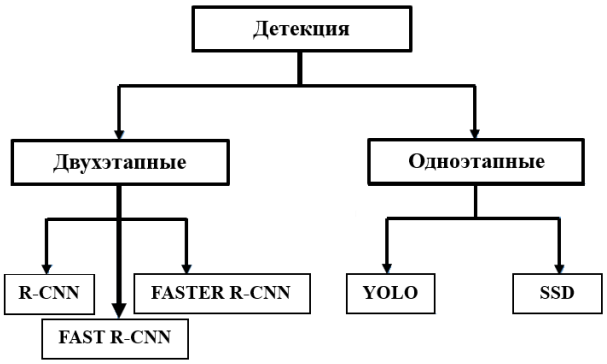

Рис. 5. Схема детекции

Прорывные архитектурные инновации в области сверточных нейронных сетей радикально преобразили ландшафт компьютерного зрения [1, 8, 11, 14]. Архитектура *ResNet* (Residual Networks) преодолела «проблему исчезающих градиентов» через введение остаточных связей (*skip connections*), позволяющих эффективно обучать сети с глубиной в сотни слоёв и достигать рекордной точности на сложных наборах данных. Параллельно развивались *Inception*-модули, оптимизирующие вычислительную эффективность за счет параллельных сверток разного масштаба, и *EfficientNet*, балансирующий глубину/ширину/разрешение через составное масштабирование. Эти достижения не только подняли планку качества распознавания, но и расширили сферы применения *CNN* за пределы компьютерного зрения: в обработке естественного языка (анализ текстовых последовательностей), временных рядов (прогнозирование), медицинской диагностике (анализ снимков), автономном транспорте (сенсорная интерпретация) и других областях ИИ [13, 19]. Непрерывные исследования фокусируются на создании устойчивых к шуму моделей, ресурсоэффективных архитектур для edge-устройств и системах реального времени с минимальной задержкой [18, 19].

Одноэтапные детекторы, особенно семейство *YOLO* (*You Only Look Once*), стали эталоном эффективности в задачах, требующих мгновенного анализа видеопотоков, рисунок 6. Разработанный Джозефом Редмоном в 2015 году [14–16, 18–21], алгоритм эволюционировал до *YOLOv7* – современного лидера по соотношению скорость/точность. Его операционный принцип базируется на:

1. Дискретизации изображения на сетку $S \times S$ ячеек.
2. Прогнозировании параметров для каждого анкора (*bounding box*):
   - Координаты центра относительно ячейки.
   - Ширина/высота относительно размеров анкора.
   - Вероятность присутствия объекта.
   - Распределение вероятностей по классам.
3. Синтезе признаков через специализированные слои (*SPP*, *PANet*), оптимизированные для мультимасштабного детектирования [7, 8, 14]. Архитектурные особенности *YOLOv7* (рис. 7) включают:

- *Backbone* (*CSPDarknet53*) для извлечения признаков.
- *Neck* (*PANet*) для агрегации признаков разного уровня.
- *Head* с анкор-базированным предсказанием рамок/классов.
- «*Trainable bag-of-freebies*» – техники обучения, повышающие точность без роста вычислительных затрат [14]. Эта комбинация обеспечивает непревзойденную скорость (до 160 FPS на GPU) при сохранении конкурентоспособной точности (55.9% AP на COCO), делая *YOLOv7* оптимальным выбором для внедрения в реальных системах видеонаблюдения, робототехники и дополненной реальности [21].

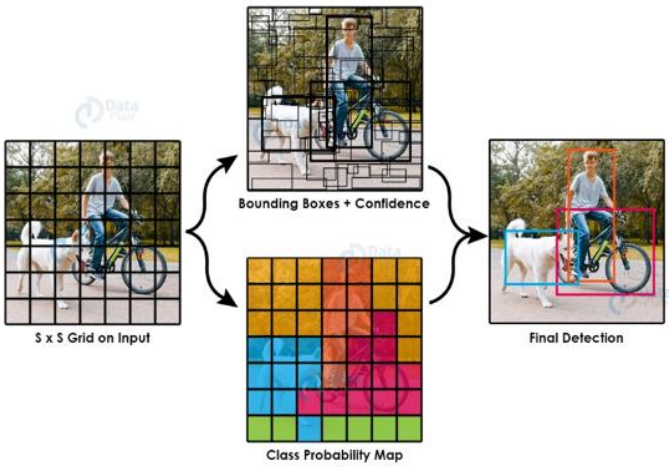

Рис. 6. Принцип работы алгоритма YOLO

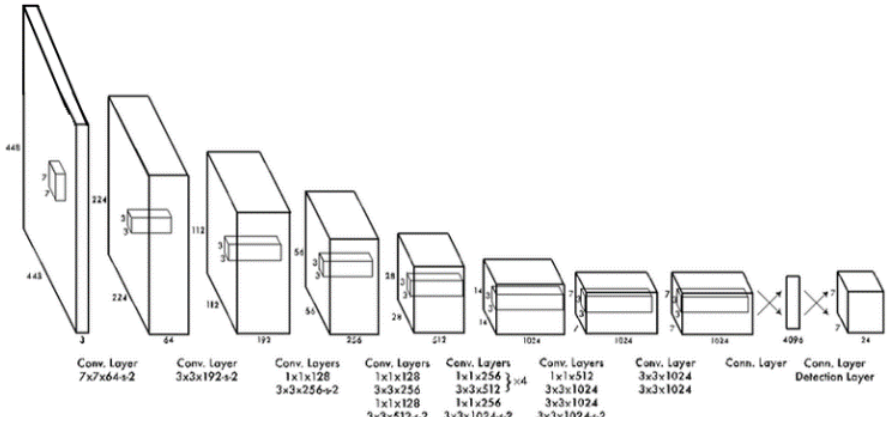

Рис. 7. Архитектура нейронной сети для алгоритма YOLO

Система метрик для оценки детекторов. Корректная постановка задачи машинного обучения требует четкого определения критериев оценки эффективности алгоритма. Для задач классификации изображений базовым и вычислительно экономичным метриком служит доля верно классифицированных примеров (*Accuracy*). Формально, при наличии размеченного датасета с эталонными метками и предсказаниями модели, *Accuracy* рассчитывается как процент объектов, для которых прогноз совпал с истинным значением [8, 10, 12, 16–18].

Оценка алгоритмов детектирования объектов существенно сложнее, так как требует одновременного учета корректности классификации *и* точности локализации. Ключевые метрики включают:

1. ***Intersection over Union* (IoU):** Количественная мера совпадения предсказанной ограничивающей рамки с эталонной (*Ground Truth, GT*). Рассчитывается как отношение площади пересечения рамок к площади их объединения $IoU = Area(Pred \cap GT) \,/\, Area(Pred \cup GT)$. Значения варьируются от 0 (полное несовпадение) до 1 (идеальное наложение) (рис. 8). Порог IoU (часто 0.5) определяет, считается ли обнаружение успешным (*True Positive*).

2. **Классификация ошибок детектирования (для одного класса):**

- ***True Positive* (*TP*):** Объект присутствует на изображении и корректно обнаружен моделью (IoU ≥ порог).
- ***False Negative* (*FN*):** Объект присутствует, но модель его не обнаружила.
- ***False Positive* (*FP*):** Модель ошибочно «обнаружила» объект в отсутствие такового или при некорректной локализации IoU $<$ порог [10].
- ***Precision* (Точность):** Доля истинных объектов среди всех обнаруженных: $P = TP \,/\, (TP + FP)$. *Высокое значение:* минимум ложных срабатываний.
- ***Recall* (Полнота):** Доля обнаруженных объектов от общего числа истинных: $R = TP \,/\, (TP + FN)$.

Эти метрики образуют основу для комплексной оценки качества детекторов, связывая качество локализации (через IoU) с классификационной надежностью (через P и R). На рисунке 8 визуализируется принцип расчета IoU для различных сценариев перекрытия *BBox* [14–16].

***Precision* (Точность)** определяется как отношение корректно обнаруженных объектов (*True Positives, TP*) ко всем предсказаниям модели:

$$(TP + False\ Positives, FP): Precision = TP \,/\, (TP + FP).$$

Диапазон значений: [0, 1]. Высокий Precision (близкий к 1) указывает на минимальное количество ложных срабатываний – большинство предсказанных ограничивающих рамок точно соответствуют реальным объектам. *Recall* (Полнота) отражает долю обнаруженных

объектов относительно общего числа истинных объектов в данных: $Recall = TP / (TP + False\ Negatives, FN)$. Диапазон значений: [0, 1]. Высокий *Recall* (близкий к 1) сигнализирует о редких пропусках объектов – модель охватывает почти все целевые сущности на изображении [7, 9, 11].

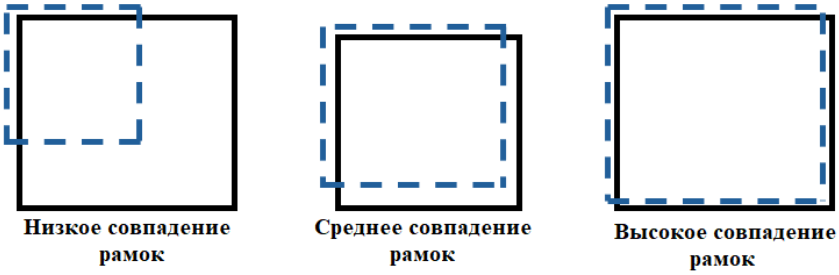

Рис. 8. Примеры влияния положения рамок

Взаимодополняемость и интерпретация метрик *Precision* и *Recall* выявляют фундаментальный компромисс в работе детекторов объектов. В сценарии высокого *Recall* при низком *Precision* модель демонстрирует высокую полноту охвата, корректно идентифицируя большинство целевых объектов (минимизируя пропуски, *FN*), однако генерирует значительное количество ложных срабатываний (высокий *FP*). Такое поведение приемлемо в приложениях, где критически важна минимизация пропусков угроз даже ценой ложных тревог, например, в системах видеобезопасности или скрининге опасных предметов. Напротив, сценарий низкого *Recall* при высоком *Precision* характеризуется исключительной точностью предсказаний (минимум ложных срабатываний, низкий *FP*), но сопровождается высоким уровнем пропущенных объектов (значительные *FN*). Эта ситуация типична для областей с катастрофическими последствиями ложных положительных результатов, таких как медицинская диагностика по снимкам, где неверное заключение недопустимо, однако допустим риск невыявления части аномалий [8, 11, 14–16]. Баланс между этими метриками определяется спецификой задачи и допустимым уровнем ошибок каждого типа. Ключевой принцип валидации**:** поскольку алгоритмы машинного обучения оптимизируются под обучающие данные, объективная оценка требует тестирования на отложенной выборке (*holdout set*) – данных, полностью исключённых из процесса обучения. Это предотвращает оптимистичные искажения метрик и гарантирует репрезентативность результатов [18].

Системы обнаружения объектов в реальном времени. На сегодняшний день, обнаружение объектов в реальном времени является одной из самых важных и актуальных задач, решаемых с помощью методов машинного обучения. Именно решение данной задачи, является основой для построения современных систем компьютерного зрения. Здесь можно привести множество примеров, это и отслеживание движения нескольких объектов одновременно, автономные подвижные составы и транспортные средства, робототехнические изделия, медицинская аналитика, в части касающейся исследования рентгеновских снимков и т.д. Технически, реализовать такую систему возможно, как с использованием вычислительной мощности непосредственно устройства локально, так и с применением облачных вычислений, при условии наличия широкополосного доступа к глобальной сети [18].

Одними из самых актуальных научно-технических вопросов на данный момент, по праву считаются перепараметризация моделей и динамическое назначение меток. При этом, научные изыскания в данных направлениях породили множество новых вызовов и проблем, касающихся совершенствования систем обнаружения объектов. В статье постараемся выделить некоторые из них, а также привести возможные пути решения, которые могут оказаться эффективными при решении, как научных, так и прикладных задач. В процессе перепараметризации моделей, происходит анализ стратегий процесса перепараметризации, которые применяются к различным сетевым слоям, с использованием ме-

тода обратного распространения ошибки или, иными словами, метода вычисления градиента. В частности, рассматривается возможность точного планирования перепараметризации модели [18].

Достаточно популярная в последнее время технология динамического присвоения меток, при обучении многослойной модели, зарекомендовала себя с лучшей стороны при решении множества прикладных задач. Но и она не лишена недостатков. Здесь можно сказать о том, что не до конца рассмотрен вопрос, каким образом назначаются динамические цели для выводов различных слоев, либо ветвей при использовании ансамблевых моделей. Ниже, рассмотрим один из предлагаемых методов решения данной проблемы, принципиально назначающий метки «от общего к частному», или, как дословно звучит перевод данного подхода «от грубого к тонкому» [19].

Самыми распространенными на данный момент моделями обнаружения объектов в реальном времени можно считать модели *YOLO* и *FCOS*. Основные параметры, которыми должна обладать успешная модель для решения данной задачи являются: развитая сетевая архитектура, обладающая высоким быстродействием; усовершенствованные методы интеграции признаков; наиболее совершенные методы обнаружения; надежная функция потерь; высокоэффективные методы присвоения меток; совершенные методы обучения. Однако, внедрение современных высокотехнологичных элементов, в структуру модели, ведет и к появлению новых проблем и особенностей в работе, одним из решений которых может являться метод «*bag-of-freebies*» или «мешок слов» [19, 20].

Рассмотрим, как работают методы перепараметризации модели. В рамках подходов к использованию данных методов, обычно происходит объединение нескольких вычислительных модулей в одно целое, на этапе инференса, т.е. тогда, когда модель уже не обучается, а использует выводы предыдущих итераций обучения, для принятия решений на основе новых данных. Часто, метод перепараметризации модели относят к ансамблевым методам, которые в свою очередь делят на два типа: ансамбль уровня модулей и ансамбль уровня модели. Приведем описание самых распространенных подходов к перепараметризации на уровне модели. В рамках одного из них, происходит обучение на нескольких одинаковых моделей с разными обучающими данными, затем полученные веса усредняются и получают вывод. В другом случае происходит взвешенное усреднение весов моделей, на разном количестве итераций. При перепараметризации на уровне модели происходит разделение модуля на несколько одинаковых частей при обучении, как бы разветвляя их, а при инференсе, т.е. получении вывода, наоборот объединяет разветвленные модули в один эквивалентный модуль. Но стоит сказать, что далеко не все перепараметризованные модули могут быть успешно использованы в современных архитектурах [19, 20].

Рассмотрим метод масштабирования модели, как один из способов изменить размеры уже созданной модели, и адаптировать ее к решению специфичных задач. В рамках данного метода используются различные коэффициенты масштабирования по параметрам: разрешение; глубина, или, иными словами, количество слоев; количество каналов; количество пирамид признаков (*FPN*) [15]. Данные коэффициенты используются в целях достижения оптимального соотношения между количеством параметров, сложностью вычислений, скоростью инференса и точностью. Одним из популярных методов масштабирования модели является поиск архитектуры нейронной сети (*NAS*) [15, 17]. В рамках данного метода поиск подходящих коэффициентов масштабирования происходит автоматически, в пространстве поиска, при этом отсутствует необходимость задавать жесткие ограничения [18].

Несмотря на то, что метод является универсальным и достаточно популярным, он не лишен недостатков. Например, для того чтобы завершить поиск коэффициентов масштабирования, модели требуется выполнить множество сложных вычислений. В некоторых источниках рассматривается связь между коэффициентами масштабирования и количеством параметров, вкупе с количеством операций, при этом производится оценка правил, с целью получения более точных коэффициенты масштабирования. На данный момент, практически все методы масштабирования анализируют коэффициенты по от-

дельности [15]. Это происходит из-за того, что практически все актуальные архитектуры *NAS* работают с коэффициентами масштабирования, которые имеют достаточно низкий уровень корреляции. Например, модели, такие как *DenseNet* и *VoVNe*t, принцип работы которых основан, в том числе, на конкатенации, во время масштабирования, меняют ширину входа некоторых слоев сети [21, 22].

Во многих источниках, посвященных построению эффективных архитектур сетевых моделей, среди основных факторов влияющих на качество их работы, выделяют оптимизацию количества параметров, объем вычислений и вычислительную сложность [18–22]. Так же, зачастую, анализируется влияние соотношения каналов ввода/вывода, разветвленности архитектуры, а также результаты воздействия элементарных операций на скорость вывода. Некоторыми авторами учитывается роль функции активации при масштабировании модели, вместе с тем, уделяется внимание и количеству элементов в выходных тензорах сверточных слоев [14, 17, 22]. На рис. 9 представлена схема модели *CSPVoNet*, являющейся одним из вариантов исполнения модели *VoVNet*. Здесь следует отметить, что помимо вышеупомянутых подходов, архитектура *CSPVoNet* регулирует професс градиентного спуска таким образом, чтобы веса различных слоев обучались с учетом специфики извлекаемых ими признаков. Именно поэтому, градиентный подход позволяет получать более точные выводы гораздо быстрее. Модель *ELAN* на рис. 9 анализирует еще один подход к проектированию архитектуры, при котором за счет контроля длины путей распространения градиента, достигается более эффективное обучение глубоких слоев и улучшается сходимость модели. Рассмотрим еще одну модель *Extended-ELAN* (*E-ELAN*) (рис. 9), построенную на основе *ELAN*.

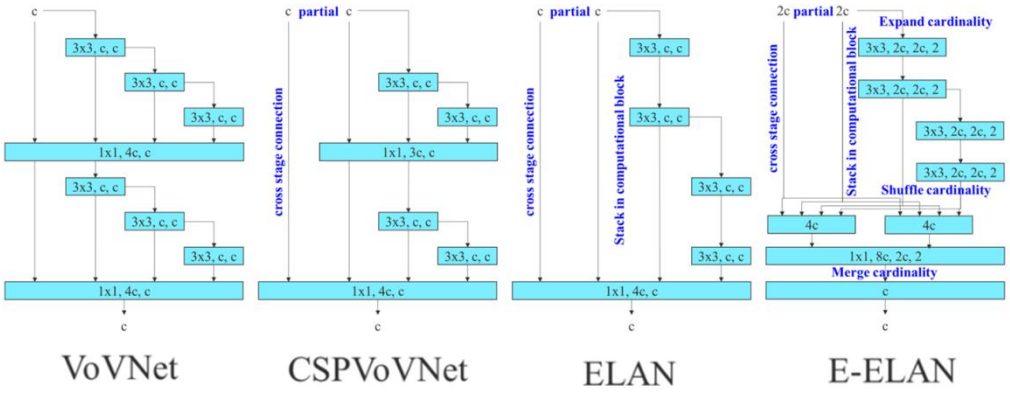

Рис. 9. Сети агрегации слоев

В независимости от того, какое значение будет иметь параметр длины путей распространения градиента, а также от того какое количество вычислительных блоков используется в модели *ELAN,* показатели ее работы остаются стабильными. Однако, в том случае, если процесс объединения вычислительных блоков будет происходить без использования критерия остановки, данная стабильность работы будет нарушена, а параметрические коэффициенты могут снизится. Модель *E-ELAN* задействует расширение, перестановки и масштабирование признаков для того, что способствует непрерывному и устойчивому росту обучаемости сети. При этом исходный градиентный спуск не будет нарушен. С точки зрения подхода к построению, модель *E-ELAN* корректирует только архитектуру вычислительного блока, в то время, как архитектура переходного слоя остается совершенно неизменной [20–22].

Зачастую, при построении современных моделей, придерживаются стратегии, при которой используется принцип групповой свертки для расширения канала и увеличения мощности вычислительного блока. Предлагается применение одного и того же параметра группы и расширения канала ко всем вычислительным блокам вычислительного слоя. После чего, карта признаков, которая рассчитывалась каждым вычислительным блоком

перемешивается в g групп в соответствии с заданным параметром группы g, а затем объединяется. В то же время, количество каналов в каждой группе карты признаков будет ровно таким же, как и количество каналов в исходной архитектуре. Так же, происходит добавление g групп карт признаков для повышения уникальности данных. Кроме того, архитектура модели *E-ELAN* может перенаправлять потоки данных между группами вычислительных блоков, что способствует извлечению более разнообразных признаков [18–22].

Основными целями процесса масштабирования модели, в свою очередь, является настройка некоторых атрибутов модели, а также, что логично, создания моделей различных масштабов, с целью получения требуемых выводов с приемлемой скоростью. Например, в модели масштабирования *EfficientNet* в расчет берутся ширина, глубина и разрешение. Если же рассматривать масштабированную модель *YOLOv4*, то здесь масштабирование заключается в настройке количества этапов. Во многих источниках анализируется влияние базовой («*vanilla*») и групповой сверток, на количество параметров и вычислительных операций, при масштабировании модели по ширине и глубине [13, 16, 22].

Указанные выше методы, в основном используются в архитектурах *PlainNet или ResNet.* При изменении масштаба в данных архитектурах, степень входящей и исходящей связности каждого слоя остается неизменной, из-за чего становится возможным провести независимый анализ влияния каждого коэффициента масштабирования на количество параметров и сложность вычислений [18, 20, 22]. В случае применения данных методов к архитектуре, основанной на объединении признаков, изменение масштаба по глубине приводит к соответствующему изменению числа входящих связей слоя преобразования, расположенного непосредственно после вычислительного блока, как это показано на рис. 10,a,b.

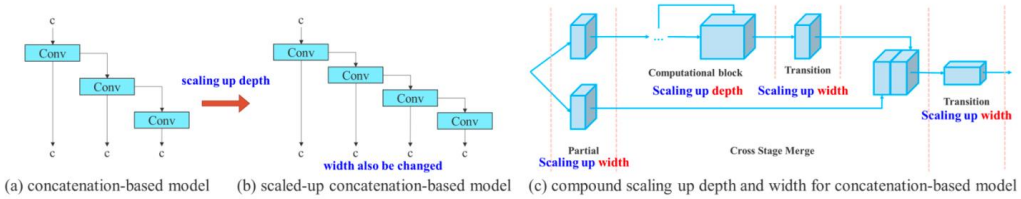

Рис. 10. Масштабирование для моделей на основе объединения

Исходя из вышесказанного, можно сделать вывод, что фактически полностью отсутствует возможность анализа влияния различных коэффициентов масштабирования по отдельности, для моделей на основе объединения. При этом, необходимо рассматривать коэффициенты в совокупности. Для примера рассмотрим задачу увеличения глубины масштабирования. Данный подход может привести к изменению соотношения между входным и выходным каналами переходного слоя, что может привести к снижению уровня аппаратных ресурсов модели. Поэтому целесообразно введение иных методов масштабирования составной модели на основе объединения. В процессе масштабирования коэффициента глубины вычислительного блока, также необходимо рассчитывать изменение числа выходных каналов данного блока [13, 20, 22]. При этом выполняется масштабирование коэффициента ширины с такими же скорректированными параметрами на переходных слоях, что позволяет сохранить свойства, которыми модель обладала в первоначальном виде, с соблюдением оптимальных структурных параметров, рис. 10,c.

Несмотря на то, что метод *RepConv* демонстрирует отличную производительность на многослойной модели *VGG*, при применении к моделям с архитектурами *ResNet*, *DenseNet* его точность существенно снижается. Зачастую, для того чтобы понять, как перепараметризированная свертка должна сочетаться с различными сетями, используется подход на основе градиентного бустинга, совместно с применением запланированной перепараметризированной свертки [18, 22].

Рассмотрим метод *RepConv* подробнее. Фактически, он объединяет свертку 3×3, свертку 1×1 и идентичное соединение в одном сверточном слое [12, 19, 20–22]. В результате анализа использования *RepConv* и других архитектур, установлено, что такой же

подход в *RepConv* разрушает остаток в *ResNet* и объединение в *DenseNet,* что, в свою очередь обеспечивает увеличение разнообразия градиентов для различных карт признаков. Исходя из этого, используется *RepConv* без подобного подхода (*RepConvN*), для проектирования архитектуры запланированной перепараметризированной свертки. В случае, когда сверточный слой, с остаточным или конкатенационным соединением заменяется перепараметризированной сверткой, не должно наблюдаться идентичного соединения. На рис. 11 показан пример реализации перспективной перепараметризованной свертки, на базе *PlainNet* и *ResNet.*

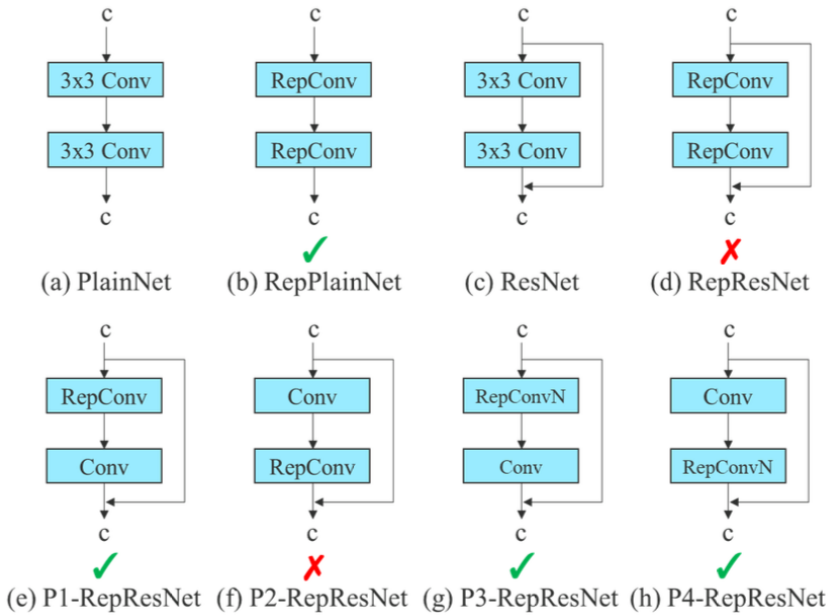

Рис. 11. Альтернативные схемы перепараметризированной модели

Остановимся на методе глубокого супервизирования, широко применяемом при обучении глубоких нейронных сетей [19]. Его основная концепция заключается в добавлении дополнительных выходных слоев в промежуточных уровнях сети, где вспомогательные функции потерь способствуют более устойчивому обновлению весов и повышению скорости сходимости обучения. Даже для таких архитектур как *ResNet* и *DenseNet,* сходимость которых обычно находится на достаточно высоком уровне, применение глубокого супервизирования может значительно улучшить производительность модели при решении целого ряда задач. На рис. 12,a,b показаны соответственно, архитектура детектора объектов без применения глубокого супервизирования и с ним.

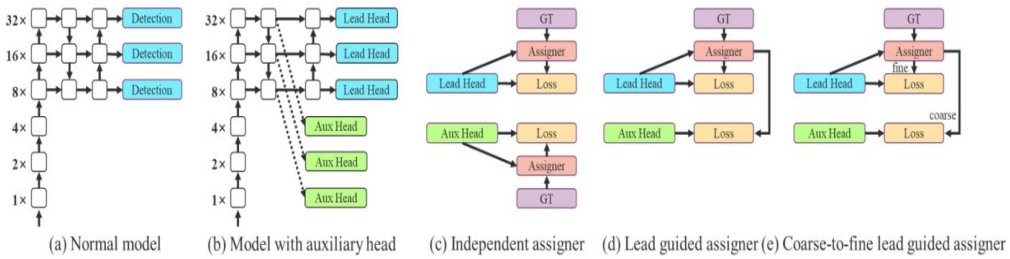

Рис. 12. Архитектура детектора объектов без применения глубокого супервизирования (a) и с ним (b)

Рассмотрим процесс присвоения меток. Ранее при обучении глубоких нейронных сетей метки напрямую соответствовали эталонным данным, формируя жёсткие (one-hot) метки в соответствии с исходной разметкой обучающей выборки. В последнее время, особенно в задачах, связанных с обнаружением объектов, всё чаще используется информация о распределении выходных вероятностей модели. Эти данные сопоставляются с эталонными метками, что позволяет с помощью вычислительных и оптимизационных методов формировать более устойчивые «мягкие» метки. Например, *YOLO* использует алгоритм *IoU* с прогнозной регрессией, которая вводит дополнительные ограничения и эталонные данные, в качестве «мягкой» метки объектности. Существуют альтернативные подходы, при которых результаты прогнозирования сети сопоставляются с эталонными данными, после чего алгоритм формирует «мягкие» метки для последующего обучения модели [19–22].

Модель с применением глубоко супервизирования, должна быть обучена на целевых задачах, вне зависимости от положения в слое главного и вспомогательного выходного слоя. Здесь важно понять, каким образом присваивать мягкую метку главному и вспомогательному выходному слою. На рис. 12 отражены результаты наиболее распространённого на данный момент метода, который заключается в разделении главного и вспомогательного выходного слоя, а затем использовании их собственных результатов прогнозирования и фактических данных для присвоения меток [20]. Далее рассмотрим новый метод присвоения меток, который направляет как вспомогательный, так и главный выходной слой с помощью прогнозирования поведения главного выходного слоя [22]. Здесь, прогнозирование поведения главного выходного слоя, используется в качестве некоего ориентира, для генерации иерархических меток от грубых к точным, которые используются соответственно для обучения главного и вспомогательного выходного слоя. Данные стратегии присвоения меток с глубоким супервизированием показаны на рис. 12,d,e соответственно.

Назначение меток с помощью главного выходного слоя в основном рассчитывается на базе результатов его прогнозирования и реальных данных, а также позволяет сгенерировать «мягкие» метки посредством процесса оптимизации. Этот набор «мягких» меток будет использоваться в качестве целевой модели обучения как для вспомогательного, так и для главного выходных слоев. Это происходит потому, что главный выходной слой обладает относительно сильной способностью к обучению, поэтому сгенерированная им «мягкая» метка должна быть более репрезентативной в отношении распределения и корреляции между исходными данными и целевыми параметрами. Кроме того, такое обучение можно рассматривать как разновидность обобщенного остаточного обучения. Позволяя более простому вспомогательному выходному слою напрямую обучаться на информации, которую уже усвоил главный выходной слой. Теперь он сможет сосредоточиться на обучении на остаточной информации, которая еще не была усвоена [19].

При создании меток с помощью грубой и точной привязки по главному выходному слою также используется прогнозируемый результат главного выходного слоя и реальную информацию для генерации «мягкой» метки. В процессе работы модели формируются два типа «мягких» меток: «точные» и «грубые». «Точные» метки совпадают с мягкими метками, генерируемыми алгоритмом присвоения под контролем главного выходного слоя, тогда как «грубые» метки создаются путем расширения числа положительных целей за счет ослабления ограничений в процессе присвоения положительных примеров [14, 20, 22]. Причина этого заключается в том, что способность к обучению вспомогательного выходного слоя не так сильна, как у главного выходного слоя, и чтобы избежать потери информации, на которой необходимо обучиться, стоит сосредоточится на оптимизации работы вспомогательного выходного слоя, для решения задачи обнаружения объектов. Что касается вывода главного выходного слоя, можно отфильтровать результаты с высокой точностью из результатов с высоким качеством воспроизведения в качестве окончательного вывода [6, 18].

Здесь следует отметить, что, если дополнительный вес «грубой» метки будет близок к весу «точной» метки, это может привести к неверному предварительному прогнозу. Поэтому, чтобы эти дополнительные «грубые» положительные метки оказывали мень-

шее влияние, следует ввести ограничения в декодер, для того чтобы дополнительные «грубые» положительные метки не могли создавать «мягкие» метки идеально. Данный механизм позволяет динамически регулировать важность «точных» и «грубых» меток в процессе обучения и делает оптимизируемую верхнюю границу «точных» меток всегда выше, нежели «грубых» меток [6, 10, 19].

В качестве базовых моделей были взяты предыдущая версия YOLO и современный детектор объектов YOLOR. Ранее было проведено сравнение рассматриваемых моделей YOLOv7 и базовых моделей, обученных с использованием одинаковых настроек, из результатов было выведено, что по сравнению с YOLOv4, YOLOv7 задействует на 75% меньше параметров, производит на 36% меньше вычислений и обеспечивает на 1,5% более высокий показатель средней точности (AP). По сравнению с современным YOLOR-CSP, YOLOv7 использует на 43% меньше параметров, производит на 15% меньше вычислений и на 0,4% более высокий показатель средней точности. В производительности мини-модели по сравнению с YOLOv4-tiny-31, YOLOv7-tiny уменьшает количество параметров на 39% и объем вычислений на 49%, но сохраняет тот же показатель средней точности. На облачной GPU-модели модель по-прежнему может иметь более высокий показатель средней точности при уменьшении количества параметров на 19% и объема вычислений на 33% [6, 11, 13, 22].

Также проводилось сравнение рассматриваемого в статье метода с современными детекторами объектов для обычных графических процессоров и мобильных графических процессоров, из результатов которого можно увидеть, что рассматриваемый метод показывает лучшее соотношение скорости и точности. Если сравнить YOLOv7-tiny-SiLU с YOLOv5-N (r6.1), рассматриваемый метод на 127 fps быстрее и на 10,7% точнее по AP. Кроме того, YOLOv7 имеет 51,4% AP при частоте кадров 161 fps, тогда как PPYOLOE-L с таким же AP имеет частоту кадров только 78 fps. С точки зрения использования параметров, YOLOv7 на 41% меньше, чем PPYOLOE-L. Если сравнить YOLOv7-X со скоростью вывода 114 fps с YOLOv5-L (r6.1) со скоростью вывода 99 fps, YOLOv7-X может улучшить AP на 3,9%. Если сравнить YOLOv7- X с YOLOv5-X (r6.1) аналогичного масштаба, то скорость вывода YOLOv7-X будет на 31 fps выше. Кроме того, с точки зрения количества параметров и вычислений, YOLOv7-X сокращает 22% параметров и 8% вычислений по сравнению с YOLOv5-X (r6.1), но улучшает AP на 2,2%.

При сравнении YOLOv7 с YOLOR с использованием входного разрешения 1280, скорость вывода YOLOv7-W6 на 8 кадров в секунду выше, чем у YOLOR-P6, а коэффициент обнаружения также увеличен на 1%. Что касается сравнения между YOLOv7-E6 и YOLOv5-X6 (r6.1), первый показывает прирост показателя средней точности на 0,9%, по сравнению со вторым, использует на 45% меньше параметров и производит на 63% меньше вычислений, а скорость вывода увеличивается на 47%. YOLOv7-D6 имеет скорость вывода, близкую к YOLOR-E6, но улучшает показатель средней точности на 0,8%. YOLOv7-E6E показывает скорость вывода, близкую к YOLOR-D6, но улучшает показатель средней точности на 0,3%.

При использовании различных стратегий масштабирования модели для изменения значения масштаба в большую сторону [6, 12, 19–22], были получены ряд экспериментальных данных. Среди них можно выделить рассматриваемый в статье метод составного масштабирования, основная идея которого заключается в увеличении глубины вычислительного блока в 1,5 раза и ширины блока перехода в 1,25 раза. Если сравнить его с методом, при котором масштабировалась только ширина, то можно увидеть, что показатель средней точности увеличился на 0,5% с меньшим количеством параметров и меньшим объемом вычислений. Если его методом, который масштабирует только глубину, будет видно, что рассматриваемый метод требует увеличения количества параметров только на 2,9% и объема вычислений на 1,2%, что позволяет улучшить показатель средней точности на 0,2%. Также можно отметить, что комбинированная стратегия масштабирования позволяет более эффективно использовать параметры и вычисления.

Чтобы проверить обобщенность предлагаемой нами модели с перепараметризацией, будем использовать ее для проверки на модели на основе конкатенации и модели на основе остаточных значений соответственно [21, 22]. Модель на основе конкатенации и

модель на основе остаточных значений, которые мы выбрали для проверки, это 3-слойная *ELAN* и *CSPDarknet* соответственно. В эксперименте с моделью на основе конкатенации были заменены слои свертки 3×3 в разных позициях в 3-слойной *ELAN* на *RepConv*, ее подробная конфигурация показана на рис. 13.

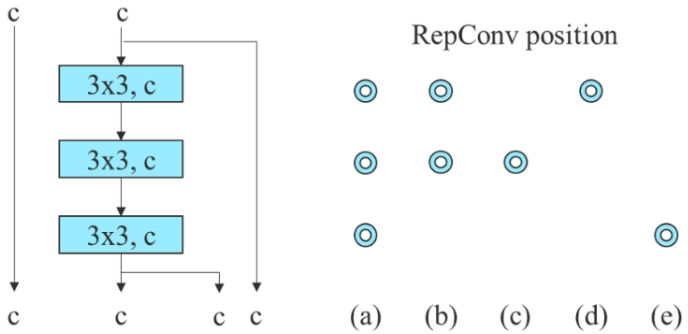

Рис. 13. Экспериментальный RepConv 3-слойный ELAN

Из результатов, проведенных исследований получено, что все более высокие значения AP присутствуют в нашей перепараметризованной модели. В эксперименте, посвященном модели на основе остаточных значений, в виду того, что исходный темный блок не имеет блока свертки 3×3, соответствующего выбранной стратегии проектирования, были дополнительно разработаны обратный темный блок для эксперимента, архитектура которого показана на рис. 14. Поскольку *CSP-Darknet* с темным блоком и обратным темным блоком имеет точно такое же количество параметров и операций, их сравнение является корректной задачей. Результаты полученные в ходе работы модели полностью подтверждают, что предложенная запланированная перепараметризованная модель одинаково эффективна на модели на основе остаточных значений.

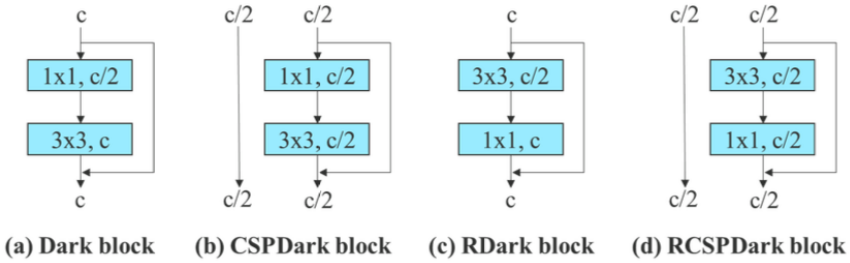

Рис. 14. Обратная модель CSP-Dsrknet

Рассмотрим случай отсутствия вспомогательного выходного слоя. В этом сценарии проводится сравнение общей стратегии независимого присвоения меток с методами, управляемыми главным выходным слоем. В результате исследований было установлено, что модели, увеличивающие потери вспомогательного выхода, способны существенно улучшить общую производительность. Более того, стратегия присвоения меток под контролем главного выходного слоя демонстрирует лучшую производительность по сравнению с общей стратегией независимых меток. Стратегия, при которой вспомогательный выход обучается на «грубых» метках, а главный выход – на «точных» метках, показывает наилучшие результаты во всех экспериментах. На рис. 15 приведена карта вероятностей объектов, предсказанная различными методами на вспомогательной и главной головках. Анализ карты показывает, что обучение вспомогательного выходного слоя на мягких метках, управляемых главным выходным слоем, способствует более эффективному извлечению остаточной информации главным выходным слоем из согласованных целей.

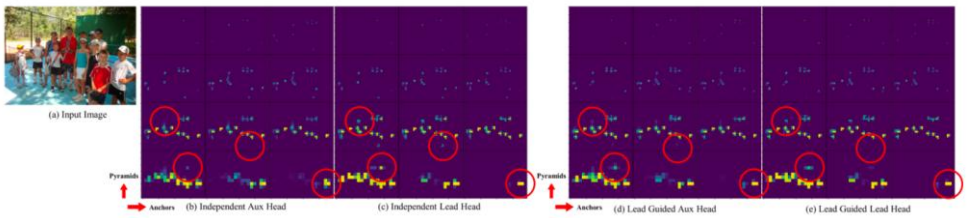

Рис. 15. Карта объектности, предсказанная различными методами на вспомогательном и главном выходном слое

YOLOv7 использует несколько уровней пирамид для многомасштабного прогнозирования объектов [22]. Вспомогательный выходной слой может быть подключен к средней пирамиде для обучения, что обеспечивает восстановление информации, потенциально утрачиваемой при прогнозировании на следующем уровне пирамиды. По этим причинам имеет смысл использовать частично подключенный вспомогательный выходной слой. Метод, при котором вспомогательный выходной слой обучается на грубых метках, а главный – на точных, демонстрирует наибольшую эффективность вспомогательного обучения в архитектуре *E-ELAN*. Вспомогательный выходной слой подключается после одного из наборов карт признаков перед их объединением, что позволяет весам вновь сгенерированных карт признаков не обновляться напрямую через вспомогательную функцию потерь [8, 14, 19, 22]. Такая конструкция обеспечивает каждой пирамиде главной головы возможность получать информацию о объектах различных размеров.

Заключение. Обнаружение объектов в реальном времени является одной из наиболее актуальных и востребованных задач компьютерного зрения и машинного обучения. Данные технологии находят применение в самых разных областях: от систем видеонаблюдения и автономных транспортных средств до робототехнических комплексов и медицинской аналитики, включая анализ рентгеновских и МРТ-изображений. Современные алгоритмы позволяют одновременно отслеживать движение нескольких объектов, обеспечивать безопасную навигацию автономных транспортных средств, анализировать состояние растений и животных, а также решать сложные задачи в промышленной автоматизации. Реализация таких систем может происходить как на локальных вычислительных устройствах, так и с использованием облачных вычислений при наличии широкополосного доступа к сети. Однако эффективная работа моделей в реальном времени требует баланса между скоростью вывода, точностью обнаружения и количеством используемых вычислительных ресурсов, что делает разработку легковесных и оптимизированных моделей особенно актуальной.

Одним из основных направлений современного исследования является перепараметризация моделей, включающая адаптацию весов и структуры нейронной сети для повышения эффективности обучения и инференса. Этот процесс подразумевает использование методов ансамблирования и усреднения весов моделей, а также динамическое назначение меток, что позволяет моделям работать с более точными и адаптивными целевыми функциями. Перепараметризация может происходить как на уровне отдельных вычислительных модулей, так и на уровне всей модели, когда несколько одинаковых моделей обучаются на различных данных, после чего их веса объединяются для формирования единого вывода. Данный подход улучшает качество прогнозов и способствует устойчивости модели к шуму в данных, однако не все архитектуры позволяют эффективно применять перепараметризацию, особенно если они содержат сложные блоки конкатенации или остаточных связей.

Особое внимание в последние годы уделяется динамическому присвоению меток, когда модель формирует «мягкие» метки на основе вероятностного распределения прогнозов и эталонных данных. Такой подход позволяет улучшить процесс обучения глубоких сетей, повышая точность и стабильность работы вспомогательных и основных выходных слоев. В частности, методы, использующие иерархическое назначение меток «от

грубого к тонкому», позволяют главным слоям передавать информацию о прогнозируемых объектах вспомогательным слоям, оптимизируя обучение на остаточной информации. Это особенно важно для многомасштабных моделей, где различные уровни пирамид признаков обрабатывают объекты разных размеров.

В качестве современных моделей обнаружения объектов в реальном времени наиболее широко применяются *YOLO* и *FCOS*. Модели *YOLOv7* и *YOLOR* демонстрируют высокую производительность и точность, обеспечивая эффективную работу даже на мобильных и встроенных устройствах. *YOLOv7*, в частности, сочетает в себе продуманную архитектуру с высокоскоростными слоями агрегации признаков, используя *SPP* и *PANet* для мультимасштабной обработки объектов. Одной из особенностей YOLOv7 является применение запланированной перепараметризации сверток (*RepConv*), что позволяет объединять в одном слое свертку 3×3, свертку 1×1 и идентичное соединение, увеличивая разнообразие градиентов и улучшая устойчивость модели. При этом важно учитывать совместимость *RepConv* с различными архитектурами, так как применение данного метода к сетям с остаточными соединениями или конкатенацией может потребовать специальных корректировок.

Масштабирование моделей также играет ключевую роль в оптимизации работы нейросетей. Оно позволяет адаптировать архитектуру под конкретные задачи, регулируя глубину, ширину, разрешение входного изображения и количество пирамид признаков. Например, подход *NAS* позволяет автоматически подбирать коэффициенты масштабирования для оптимального соотношения между точностью, количеством параметров и вычислительной сложностью. Однако в моделях, основанных на объединении признаков, требуется учитывать влияние различных коэффициентов масштабирования в совокупности, так как изменение глубины или ширины отдельных блоков может нарушить баланс входных и выходных каналов и снизить эффективность модели.

Легковесные модификации *YOLOv7* позволяют решать задачи обнаружения объектов на устройствах с ограниченными ресурсами. Интеграция с *ShuffleNetV2* и *Vision Transformer* снижает количество параметров и повышает скорость инференса, сохраняя при этом точность модели. Специализированные модели, такие как *LWMD-YOLOv7* для анализа терагерцовых изображений, *DGS-YOLOv7-Tiny* для обнаружения вредителей и заболеваний растений, *ECF-YOLOv7-Tiny* для улучшения слияния признаков, и *LWS-YOLOv7* для обнаружения объектов на водной поверхности, демонстрируют высокую производительность в прикладных задачах, сочетая скорость работы и точность. Эти модели используют разнообразные методы оптимизации: от новых функций потерь до модифицированных архитектурных блоков, обеспечивающих устойчивое извлечение признаков и эффективное обучение на ограниченных данных.

Применение глубокого супервизирования также позволяет улучшить обучение нейронных сетей, добавляя промежуточные выходные слои, где вспомогательные функции потерь ускоряют сходимость и повышают точность прогнозов. Главный и вспомогательный выходные слои могут обучаться на «точных» и «грубых» мягких метках соответственно, что обеспечивает более полное использование информации и предотвращает потерю важных признаков. Такие методы особенно эффективны в архитектурах *E-ELAN* и *CSPVoNet,* где контроль длины путей распространения градиента и организация вычислительных блоков позволяют ускорить обучение глубоких слоев и повысить устойчивость к переобучению.

Сравнительные исследования показывают, что *YOLOv7* превосходит предшествующие модели по ряду ключевых показателей. Например, по сравнению с *YOLOv4, YOLOv7* использует на 75% меньше параметров, на 36% меньше вычислений и обеспечивает прирост средней точности на 1,5%. В сравнении с *YOLOR-CSP, YOLOv7* достигает более высокой точности при меньшем объеме вычислений и меньшем количестве параметров. Аналогичные результаты наблюдаются и при использовании мини-моделей для мобильных устройств: *YOLOv7-tiny* сохраняет точность, значительно снижая требования к вычислительным ресурсам.

Кроме того, современный подход к присвоению меток позволяет использовать прогнозы главного выходного слоя для формирования иерархических мягких меток для вспомогательных слоев. Это обеспечивает более эффективное извлечение информации и улучшает работу всей модели. Механизмы динамического регулирования важности «точных» и «грубых» меток позволяют оптимизировать процесс обучения и избежать неверных прогнозов на вспомогательных уровнях сети. Практические эксперименты показывают, что такой подход улучшает показатели точности и полноты, особенно при обработке сложных мультимасштабных изображений, что имеет критическое значение для приложений в области безопасности, автономной навигации и сельского хозяйства.

Таким образом, современные исследования демонстрируют, что применение методов легковесного масштабирования, перепараметризации, глубокого супервизирования и динамического присвоения меток позволяет создавать высокоэффективные детекторы объектов, способные работать в реальном времени на устройствах с ограниченными вычислительными ресурсами. Эти подходы не только повышают точность и скорость работы моделей, но и открывают новые возможности для практического применения в самых разнообразных областях, от анализа медицинских изображений до автономного управления транспортом и мониторинга окружающей среды. Перспективы дальнейших исследований включают разработку универсальных методов масштабирования, интеграцию с мобильными и облачными вычислительными платформами, а также создание гибридных моделей, способных сочетать точность детекции с минимальными вычислительными затратами.

В данной работе проведен комплексный анализ нейросетевых подходов к решению задач компьютерного зрения, включая систематизацию ключевых типов задач: классификацию изображений, детектирование объектов и семантическую сегментацию. Детально исследованы архитектурные принципы сверточных нейронных сетей с акцентом на механизмы извлечения пространственных признаков через сверточные слои, оптимизацию представления данных посредством операций пулинга и преобразование признаков в полносвязных слоях. Особое внимание уделено эволюции методов обнаружения объектов, где задача выбора модели рассмотрена как расширение классификации за счет интеграции регрессии пространственных координат, а также проведена оценка эффективности детекторов на основе метрик *IoU, Precision, Recall* и *F1-score*, демонстрирующих фундаментальный компромисс между точностью локализации и скоростью обработки. В качестве оптимального решения для систем реального времени представлен алгоритм *YOLOv7*, архитектура которого основана на разбиении входного изображения на сетку $S \times S$ ячеек с прямым предсказанием параметров ограничивающих рамок (координаты центра, ширина, высота) и вероятностей классов для каждой ячейки, а также использовании специализированных слоёв (*SPP, PANet*) для мультимасштабной агрегации признаков. Структура нейронной сети подтверждает эффективность используемого подхода, обеспечивающего высокое быстродействие без критического снижения точности в стратегически важных приложениях видеонаблюдения, автономных систем и дополненной реальности.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Николенко С., Кадурин А., Архипельская Е.* Глубокое обучение. Погружение в мир нейронных сетей. – 2-е изд. – СПб.: Питер, 2023. – 576 с.
2. *Davies E.R., Turk M.A.* Advanced Methods and Deep Learning in Computer Vision. – Academic Press, 2022. – 690 p.
3. *Géron A.* Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. – O'Reilly Media, 2022. – 870 p.
4. *Горячкин Б.С., Китов М.А.* Компьютерное зрение: современные тенденции // Цифровая обработка сигналов. – 2023. – № 1. – С. 45-62.
5. *Bovik A.C.* Handbook of Image and Video Processing. – Academic Press, 2023. – 1200 p.
6. *Кочанов Д.Н.* Тенденции развития компьютерного зрения на основе глубокого обучения // Искусственный интеллект в технических системах: Сб. трудов XII Международной научно-технической конференции. – М.: МГТУ им. Н.Э. Баумана, 2023. – С. 112-119.

7. *Zhang J., Li C., Wan X.* Real-Time Safety Helmet Detection in Complex Construction Environments // IEEE Transactions on Industrial Informatics. – 2023. – Vol. 19 (10). – P. 10034-10043.
8. *Redmon J., Farhadi A.* YOLOv7: An Incremental Improvement // arXiv:1804.02767 [cs.CV]. – 2023. – URL: https://github.com/ultralytics/ultralytics.
9. *Лебедев В.Б., Лебедев О.Б.* Композитные многоагентные системы для распознавания изображений в реальном времени // Информатика и системы управления. – 2022. – № 3 (73). – С. 77-89.
10. *Dudarev D.S., Dudarev K.S., Motaylenko L.V.* Computer Vision: A Retrospective Analysis of Evolution and Impact // IEEE Access. – 2024. – Vol. 12. – P. 11245-11260.
11. *Padilla R., Passos W.L., Dias T.L.* A Comparative Analysis of Object Detection Metrics with a Companion Open-Source Toolkit // Electronics. – 2021. – Vol. 10 (3). – P. 279-284.
12. *Dyachenko R.A., Dovgal V.V., Gura D.A.* Comparative Analysis of YOLOv7 and U-Net for Remote Sensing Image Segmentation // IEEE Geoscience and Remote Sensing Letters. – 2024. – Vol. 21. – P. 125-142.
13. *Wang Z., Wang P., Li Y.* Deep Learning for Face Recognition in Unconstrained Environments: A Survey // ACM Computing Surveys. – 2023. – Vol. 55 (9). – Article 188.
14. *Wang C.-Y., Bochkovskiy A., Liao H.-Y.* YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors // Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). – 2023. – P. 7464-7475.
15. *Vaswani A., Shazeer N., Parmar N.* Attention Is All You Need // Advances in Neural Information Processing Systems 30 (NIPS 2017). – 2017. – P. 67-84.
16. *Liu Y., Sun P., Wergeles N.* A Survey and Performance Evaluation of Deep Learning Methods for Small Object Detection // Expert Systems with Applications. – 2021. – Vol. 172. – P. 357-369.
17. *Казначеева А.А., Власенко О.М., Эпов А.А.* Алгоритм управления мехатронной станцией сортировки изделий с применением системы компьютерного зрения // Электронный научный журнал «Инженерный вестник Дона». – 2025. – № 7 (127).– С. 133-143.
18. *Чжен А., Казари А.* Машинное обучение. Конструирование признаков. – М.: Бомбора, 2024. – 240 с.
19. *Небаба С.Г., Марков Н.Г.* Сверточные нейронные сети семейства YOLO для мобильных систем компьютерного зрения // Компьютерные исследования и моделирование. – 2024. – № 3. – С. 615-631.
20. *Трубин А.Е. и др.* Методика предобработки данных машинного обучения для решения задач компьютерного зрения // Прикладная Информатика. – 2022. – № 4. – С. 36-39.
21. *Васильев М.Е., Шалимов А.С., Савина О.А.* Обзор версий YOLO: одноэтапная модель сверточной нейронной сети // Universum: технические науки: электронный научный журнал. – 2025. – № 6 (135). – URL: https://7universum.com/ru/tech/archive/item/20293.
22. *Красноперова А.С., Твердохлебов А.С., Карташов А.А., Вебер В.И., Куприц В.Ю.* Исследование эффективности применения моделей нейронных сетей YOLO для распознавания объектов на радиолокационных изображениях // Russian Technological Journal. – 2025. – 13 (4). – С. 25-36. – https://doi.org/10.32362/2500-316X-2025-13-4-25-36. – EDN: WVWVCJ.

REFERENCES

1. *Nikolenko S., Kadurin A., Arkhipel'skaya E.* Glubokoe obuchenie. Pogruzhenie v mir neyronnykh setey [Deep learning. Immersion in the world of neural networks]. 2nd ed. Saint Petersburg: Piter, 2023, 576 p.
2. *Davies E.R., Turk M.A.* Advanced Methods and Deep Learning in Computer Vision. Academic Press, 2022, 690 p.
3. *Géron A.* Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. O'Reilly Media, 2022, 870 p.
4. *Goryachkin B.S., Kitov M.A.* Komp'yuternoe zrenie: sovremennye tendentsii [Computer vision: modern trends], *Tsifrovaya obrabotka signalov* [Digital signal processing], 2023, No. 1, pp. 45-62.
5. *Bovik A.C.* Handbook of Image and Video Processing. Academic Press, 2023, 1200 p.
6. *Kochanov D.N.* Tendentsii razvitiya komp'yuternogo zreniya na osnove glubokogo obucheniya [Trends in the development of computer vision based on deep learning], *Iskusstvennyy intellekt v tekhnicheskikh sistemakh: Sb. trudov XII Mezhdunarodnoy nauchno-tekhnicheskoy konferentsii* [Artificial intelligence in technical systems: Proceedings of the XII International scientific and technical conference]. Moscow: MGTU im. N.E. Baumana, 2023, pp. 112-119.
7. *Zhang J., Li C., Wan X.* Real-Time Safety Helmet Detection in Complex Construction Environments, *IEEE Transactions on Industrial Informatics*, 2023, Vol. 19 (10), pp. 10034-10043.
8. *Redmon J., Farhadi A.* YOLOv7: An Incremental Improvement, *arXiv:1804.02767 [cs.CV]*, 2023. Available at: https://github.com/ultralytics/ultralytics.

9. *Lebedev V.B., Lebedev O.B.* Kompozitnye mnogoagentnye sistemy dlya raspoznavaniya izobrazheniy v real'nom vremeni [Composite multi-agent systems for image recognition in real time], *Informatika i sistemy upravleniya* [Computer Science and Control Systems], 2022, No. 3 (73), pp. 77-89.
10. *Dudarev D.S., Dudarev K.S., Motaylenko L.V.* Computer Vision: A Retrospective Analysis of Evolution and Impact, *IEEE Access*, 2024, Vol. 12, pp. 11245-11260.
11. *Padilla R., Passos W.L., Dias T.L.* A Comparative Analysis of Object Detection Metrics with a Companion Open-Source Toolkit, *Electronics*, 2021, Vol. 10 (3), pp. 279-284.
12. *Dyachenko R.A., Dovgal V.V., Gura D.A.* Comparative Analysis of YOLOv7 and U-Net for Remote Sensing Image Segmentation, *IEEE Geoscience and Remote Sensing Letters*, 2024, Vol. 21, pp. 125-142.
13. *Wang Z., Wang P., Li Y.* Deep Learning for Face Recognition in Unconstrained Environments: A Survey, *ACM Computing Surveys*, 2023, Vol. 55 (9), Article 188.
14. *Wang C.-Y., Bochkovskiy A., Liao H.-Y.* YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023, pp. 7464-7475.
15. *Vaswani A., Shazeer N., Parmar N.* Attention Is All You Need, *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, 2017, pp. 67-84.
16. *Liu Y., Sun P., Wergeles N.* A Survey and Performance Evaluation of Deep Learning Methods for Small Object Detection, *Expert Systems with Applications*, 2021, Vol. 172, pp. 357-369.
17. *Kaznacheeva A.A., Vlasenko O.M., Epov A.A.* Algoritm upravleniya mekhatronnoy stantsiey sortirovki izdeliy s primeneniem sistemy komp'yuternogo zreniya [Algorithm for controlling a mechatronic station for sorting products using a computer vision system], *Elektronnyy nauchnyy zhurnal «Inzhenernyy vestnik Dona»* [Electronic scientific journal «Engineering Bulletin of the Don»], 2025, No. 7 (127), pp. 133-143.
18. *Chzhen A., Kazari A.* Mashinnoe obuchenie. Konstruirovanie priznakov [Machine learning. Feature engineering]. Moscow: Bombora, 2024, 240 p.
19. *Nebaba S.G., Markov N.G.* Svertochnye neyronnye seti semeystva YOLO dlya mobil'nykh sistem komp'yuternogo zreniya [Convolutional neural networks of the YOLO family for mobile computer vision systems], *Komp'yuternye issledovaniya i modelirovanie* [Computer Research and Modeling], 2024, No. 3, pp. 615-631.
20. *Trubin A.E. i dr.* Metodika predobrabotki dannykh mashinnogo obucheniya dlya resheniya zadach komp'yuternogo zreniya [Methodology for preprocessing machine learning data for solving computer vision problems], *Prikladnaya Informatika* [Applied Informatics], 2022, No. 4, pp. 36-39.
21. *Vasil'ev M.E., Shalimov A.S., Savina O.A.* Obzor versiy YOLO: odnoetapnaya model' svertochnoy neyronnoy seti [Review of YOLO versions: one-stage model of convolutional neural network], *Universum: tekhnicheskie nauki: elektronnyy nauchnyy zhurnal* [Universum: technical sciences: electronic scientific journal], 2025, No. 6 (135). Available at: https://7universum.com/ru/tech/archive/item/20293.
22. *Krasnoperova A.S., Tverdokhlebov A.S., Kartashov A.A., Veber V.I., Kuprits V.Yu.* Issledovanie effektivnosti primeneniya modeley neyronnykh setey YOLO dlya raspoznavaniya ob"ektov na radiolokatsionnykh izobrazheniyakh [Efficiency of YOLO neural network models applied for object recognition in radar images], *Russian Technological Journal*, 2025, 13 (4), pp. 25-36. Available at: https://doi.org/10.32362/2500-316X-2025-13-4-25-36. EDN: WVWVCJ.

Лебедев Олег Борисович – МИРЭА – Российский технологический университет; e-mail: lebedev.ob@mail.ru; г. Москва, Россия; тел.: 89085135512; кафедра информатики; д.т.н.; профессор.

Черкасов Роман Иванович – МИРЭА – Российский технологический университет; e-mail: cherkasov@mirea.ru; г. Москва, Россия; тел.: 89518286127; кафедра информатики; к.т.н.; доцент.

Lebedev Oleg Borisovich – MIREA – Russian University of Technology; e-mail: lebedev.ob@mail.ru; Moscow, Russia; phone: +79085135512; the Department of Computer Science; dr. of eng. sc.; professor.

Cherkasov Roman Ivanovich – MIREA – Russian University of Technology; e-mail: cherkasov@mirea.ru; Moscow, Russia; phone: +79518286127; the Department of Computer Science; cand. of eng. sc.; associate professor.