

---

# Сравнение современных методов глубокого обучения для автоматической детекции белух в естественной среде

---

A Preprint

Булкин Антон Павлович  
Московский государственный университет имени М. В. Ломоносова  
Научный руководитель: Кравцова Ольга Анатольевна  
bulkin261@gmail.com

2025

## Abstract

В работе исследуются современные архитектуры глубокого обучения для автоматического мониторинга белух — редкого и охраняемого вида, требующего внимательного и регулярного наблюдения. Цель исследования — повысить эффективность мониторинга животных, минимизируя затраты времени и ресурсов на ручную обработку данных, а также создать систему отслеживания взаимодействия животных между собой в рамках одного видеофайла. Предложен комплексный подход: классические решения CV, полуавтоматическая разметка видеоматериалов, обучение и сравнение современных детекторов и трекеров, а также анализ их точности и скорости.

Keywords детекция объектов · глубокое обучение · белуха · дроны · спутниковые снимки · трекинг · полуавтоматическая разметка

## 1 Введение

Белуха — редкий и охраняемый вид морских млекопитающих, для которого необходимы регулярные и объективные наблюдения в естественной среде обитания. Традиционные подходы — полевые экспедиции, акустический мониторинг, визуальный подсчёт — трудоёмки, подвержены субъективности и ограничены по пространственному и временному охвату. Автоматизация на основе методов компьютерного зрения и глубокого обучения позволяет снизить затраты, повысить воспроизводимость и масштабировать мониторинг.

Задача детекции морских млекопитающих в изображениях и видео обычно решается локализацией объектов с ограничивающими рамками и последующим трекингом. Ранние работы использовали классические техники и первые версии одноэтапных и двухэтапных детекторов, показывая базовую применимость, но чувствительность к условиям съёмки и фоновым помехам. Для видео широко применялись связки «детектор + трекер», повышающие устойчивость за счёт межкадровой ассоциации. Современные одноэтапные архитектуры семейства YOLO существенно улучшили соотношение точности и скорости и пригодны для сценариев реального времени. Трансформерные детекторы упрощают инференс, снимая необходимость в постобработке наподобие подавления немаксимумов, и повышают согласованность предсказаний на сложном фоне. Отдельное направление посвящено снижению стоимости аннотирования за счёт полуавтоматической разметки и стратегий активного обучения.

Ключевые трудности домена — малый видимый размер животных в кадре, блики и рябь воды, частичные окклюзии и близость визуально похожих фоновых объектов (лодки, буи, скалы, водоросли). Существенны доменный сдвиг между локациями и условиями съёмки и дисбаланс классов в многоклассовых сценариях. Высокая стоимость ручной аннотации приводит к ограниченным наборам и ошибкам разметки, что снижает переносимость моделей. Наконец, несогласованные протоколы оценки

в литературе — разные наборы данных, пороги пересечения, препроцессинг — затрудняют прямое сравнение результатов и выбор решений для практики.

В работе предлагается воспроизводимый бенчмарк детекции белух в реальной морской среде с унифицированным протоколом обучения и оценки. Подготовлены и сопоставлены наборы изображений разного масштаба — 400, 800, 1200 и около 8000 кадров, — где расширение корпуса достигается полуавтоматической разметкой: начальное обучение модели-помощника, автоматическая аннотация неразмеченных кадров и ручная валидация. Рассматриваются одноклассовая и многоклассовая разметки, что позволяет учитывать типичные фоновые объекты на воде.

## 2 Обзор литературы / Related works

Исследования, посвящённые автоматическому мониторингу белух и других морских млекопитающих, активно развиваются в последние годы и охватывают как аэровидеосъёмку, так и анализ спутниковых данных. Одной из существенных современных работ является статья Alsaidi et al. [\[2024\]](#), где предложена система детекции и трекинга белух на аэровидео с использованием YOLOv7 и алгоритма DeepSORT. Модель показала высокую точность и полноту, а также устойчивость трекинга после постобработки. Аналогичные идеи развиваются в работе Harasyn et al. [\[2022\]](#), где YOLOv4 и DeepSORT применялись для детекции белух, каяков и лодок в видеопотоках с дронов; достигнута точность около 74% и полнота 72%. Lee et al. [\[2021\]](#) исследовали RetinaNet и Faster R-CNN на данных залива Камберленд, отметив, что использование тайлинга крупных изображений существенно повышает точность и уменьшает количество ложных срабатываний.

Особое внимание уделяется построению и масштабированию наборов данных. Araújo et al. [\[2022\]](#) представили датасет Beluga-5k с более чем 5500 фотографиями белух и предложили полуавтоматическую разметку. Лучшая модель (YOLOv3-tiny) достигла 97,05% mAP@0.5, корректно обнаруживая белух даже в условиях сложного фона. Bouleit et al. [\[2023\]](#) предложили интерактивную схему «человек в контуре», при которой нейросеть, обученная на 100 изображениях, достигла 91% совпадения с экспертами и позволила ускорить аннотацию более чем в пять раз. Cubaynes и Fretwell [\[2022\]](#) опубликовали набор спутниковых изображений Whales from Space, включающий сотни размеченных примеров китов, что обеспечило возможность обучения моделей на данных высокого разрешения.

Использование спутниковых снимков для мониторинга млекопитающих подтверждается рядом работ. Green et al. [\[2023\]](#) применили YOLOv5 для обнаружения серых китов на спутниковых снимках и достигли точности 80–94% при полноте 84–89%. Guirado et al. [\[2019\]](#) предложили каскадный подход, объединяющий классификацию наличия китов и подсчёт особей, что увеличило общую точность на 36% по сравнению с одноэтапными методами. Borowicz et al. [\[2019\]](#) показали, что CNN, обученные на аэрофотоснимках, могут быть перенесены на спутниковые данные без значительной потери качества.

В области общих архитектур объектной детекции значительную роль сыграли двухэтапные модели (Faster R-CNN [Ren et al. \[2015\]](#)), обеспечившие высокую точность за счёт сети предложений регионов. Одноэтапные детекторы, начиная с YOLO [Redmon et al. \[2016\]](#) и YOLO9000 [Redmon and Farhadi \[2017\]](#), позволили объединить локализацию и классификацию в одном прогоне сети, достигнув 45–60 FPS. RetinaNet [Lin et al. \[2017\]](#) с функцией Focal Loss улучшил устойчивость к дисбалансу классов и повысил точность при сохранении скорости. Последующие версии YOLO (в частности, YOLOv7 [Wang et al. \[2022\]](#)) установили новый стандарт по соотношению точности и производительности.

Трансформерные подходы (DETR [Carion et al. \[2020\]](#), RT-DETR [Zhao et al. \[2023\]](#)) позволили отказаться от постобработки (NMS) и выполнять end-to-end детекцию. RT-DETR показал сопоставимую с YOLO точность при сохранении real-time скорости, что делает его перспективным для видеоаналитики. Модели открытого словаря (ViLD [Gu et al. \[2021\]](#), GLIP [Li et al. \[2022\]](#), YOLO-World [Cheng et al. \[2024\]](#)) расширили возможности детекции на неизвестные классы, включая морские объекты, что особенно важно для практических систем, где структура сцены заранее не фиксирована.

Обзор Tuia et al. [\[2022\]](#) подчёркивает важность машинного обучения для экологического мониторинга. Авторы указывают, что интеграция глубоких моделей с полевыми данными и дроновыми системами позволяет существенно сократить стоимость и повысить масштабируемость наблюдений. Совокупно, эти исследования формируют основу для разработки универсального бенчмарка детекции белух и показывают актуальность сопоставления современных архитектур на единых данных и метриках.

### 3 Постановка задачи

#### 3.1 Алгебраическая и вероятностная структура данных

Имеется исходный видеокорпус объёмом порядка 450 ГБ (4К) с бортов квадрокоптеров, из которого формируются выборки изображений размером  $N \in \{400, 800, 1200, \sim 8000\}$  с аннотациями в формате COCO. Каждое изображение  $x \in \mathcal{X}$  снабжено множеством объектов  $Y = \{(b_j, c_j)\}_{j=1}^M$ , где  $b_j = (x, y, w, h)$  — ограничивающая рамка,  $c_j \in \mathcal{C}$  — метка класса. Рассматриваются два варианта аннотаций: one (однокласс:  $\mathcal{C} = \{\text{beluga}\}$ ) и mlt (многокласс:  $\mathcal{C} = \{\text{beacon, beluga, bird, person, rocks, seaweed, ship}\}$ ). Разбиения  $X^{\text{train}}, X^{\text{val}}, X^{\text{test}}$  фиксируются на уровне изображений ( $\text{split} \in \{\text{train, val, test}\}$ ). Формально считаем, что  $(X, Y) \sim \mathcal{D}$ , где  $\mathcal{D}$  — неизвестное распределение, а обучающая выборка  $\mathcal{S} = \{(x_i, Y_i)\}_{i=1}^N$  представляет собой i.i.d. реализацию из  $\mathcal{D}$ ;  $Y_i$  — конечное множество пар «рамка-класс» для  $x_i$ . Для расширенного набора  $\sim 8000$  кадров аннотации получены полуавтоматически: начальная ручная разметка  $\rightarrow$  предобученный детектор  $\rightarrow$  авторазметка  $\rightarrow$  ручная верификация.

#### 3.2 Отображение и вычислительный конвейер

Цель — построить детектор

$$f_\theta : \mathcal{X} \rightarrow 2^{\mathcal{B} \times \mathcal{C} \times [0,1]}, \quad f_\theta(x) = \{(\hat{b}_k, \hat{c}_k, \hat{p}_k)\}_{k=1}^{\hat{M}},$$

который по входному изображению выдаёт множество предсказанных боксов, классов и оценок уверенности. На практике  $f_\theta$  реализуется как композиция стадий

$$f_\theta = \underbrace{\pi_{\text{post}}}_{\text{постобработка}} \circ \underbrace{d_\theta}_{\text{нейросетевой детектор}} \circ \underbrace{\phi}_{\text{препроцессинг}},$$

где  $\phi$  приводит изображение к требуемому масштабу/формату;  $d_\theta$  — параметризуемая модель;  $\pi_{\text{post}}$  — схема отбора и консолидации предсказаний. Для open-vocabulary варианта конвейер расширяется текстовым энкодером  $t(\cdot)$ , формируя совместное представление «изображение–текст» для классов. В одноклассовом режиме one пространство меток вырождается до  $\{\text{beluga}\}$ , а mlt учитывает семантически близкие фонны (лодки/буи/скалы и т. п.), снижая ложные срабатывания. Реализация конвейера, структура датасетов и сценарии one/mlt подробно заданы в программной реализации системы и вспомогательных материалах.

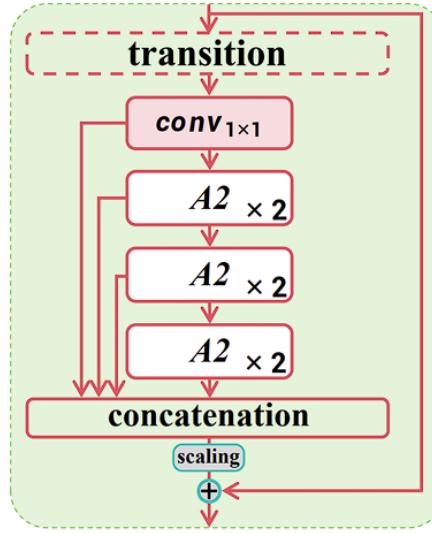


Figure 1: Residual Efficient Layer Aggregation Network as presented in paper

Рис. 1: Схематичное устройство одноэтапного детектора семейства YOLO: backbone для извлечения признаков, FPN/PAN для многоуровневой агрегации и детекционные головы для разных масштабов объектов.

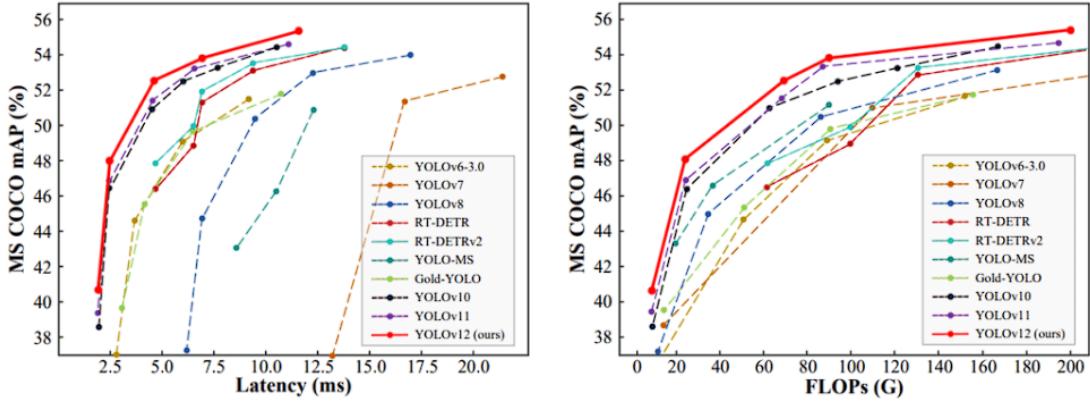


Рис. 2: Пример бенчмарка YOLOv12 на стандартных датасетах: зависимость качества детекции от скорости инференса в сравнении с предыдущими версиями семейства YOLO.

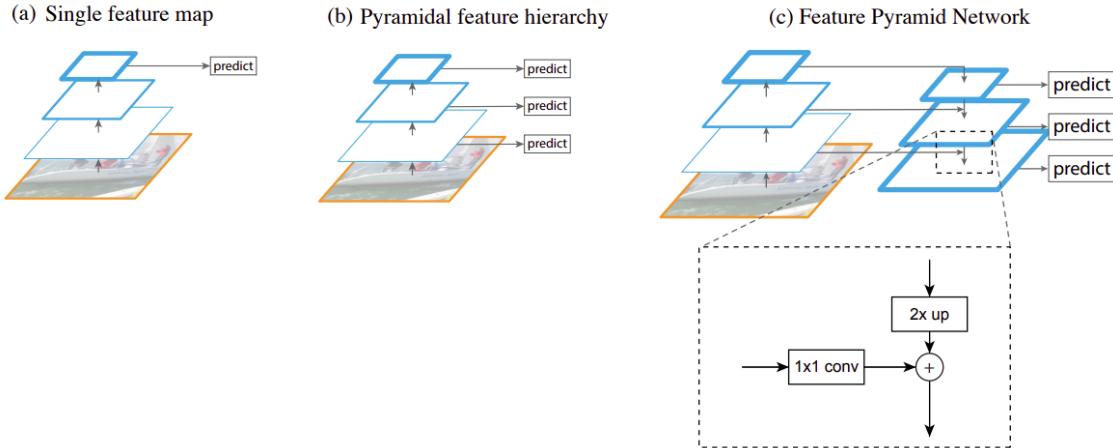


Рис. 3: Пример пирамиды признаков (Feature Pyramid Network, FPN), используемой в современных детекторах для одновременной обработки мелких и крупных объектов.

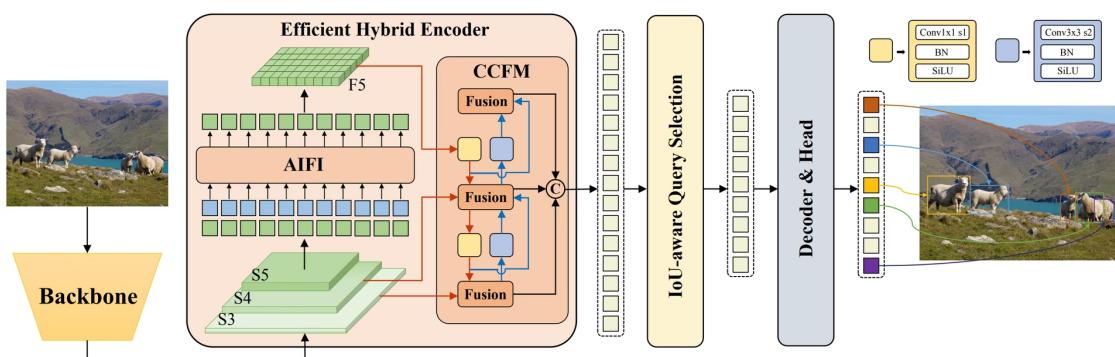


Рис. 4: Упрощённая архитектура RT-DETR: сверточный backbone, трансформерный энкодер–декодер и предсказание ограничивающих рамок.

### 3.3 Внешние критерии качества

Оценка проводится на тестовой части  $X^{\text{test}}$  по стандартным метрикам детекции:

$$\text{mAP}@0.5 = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \text{AP}_c(\text{IoU} = 0.5), \quad \text{mAP}@[0.5:0.95] = \frac{1}{|\mathcal{C}| \cdot 10} \sum_{c \in \mathcal{C}} \sum_{\alpha=0.5}^{0.95} \text{AP}_c(\text{IoU} = \alpha),$$

шаг по  $\alpha$  равен 0.05. Дополнительно фиксируются показатели производительности: средняя задержка инференса на кадр  $\tau_{\text{inf}}$  (мс/кадр) и при необходимости среднее время одной эпохи обучения  $\tau_{\text{train}}$  (мин/эпоха). Разметочные режимы one/mlt унифицированы по моделям и наборам  $N$ .

### 3.4 Оптимизационная постановка

Введём два уровня оптимизации: (i) обучение детекторов объектов по размеченным кадрам и (ii) подбор конфигурации трекера, использующего выходы детектора на видеопоследовательностях.

Этап 1: обучение детекторов. Пусть  $\mathcal{S} = \{(x_i, Y_i)\}_{i=1}^N$  — обучающая выборка статичных изображений, где  $x_i \in \mathcal{X}$ , а  $Y_i = \{(b_{ij}, c_{ij})\}_{j=1}^{M_i}$  — множество размеченных объектов (см. раздел 3). Нейросетевой детектор с параметрами  $\theta$  задаёт отображение

$$d_\theta : \mathcal{X} \rightarrow 2^{\mathcal{B} \times \mathcal{C} \times [0,1]}, \quad d_\theta(\phi(x_i)) = \{(\hat{b}_{ik}, \hat{c}_{ik}, \hat{p}_{ik})\}_{k=1}^{\hat{M}_i},$$

где  $\phi$  обозначает препроцессинг, а  $\hat{p}_{ik}$  — оценка уверенности в предсказании пары «рамка–класс».

Обучение детектора рассматривается как задача минимизации эмпирического риска с детектор-специфичной функцией потерь:

$$\hat{\theta} \in \arg \min_{\theta} \left[ \frac{1}{N} \sum_{i=1}^N \mathcal{L}_{\text{det}}(d_\theta(\phi(x_i)), Y_i) + \lambda \mathcal{R}(\theta) \right], \quad (1)$$

где  $\mathcal{L}_{\text{det}}$  включает классификационную компоненты (крoss-энтропия, Focal Loss и т. п.) и регрессию ограничивающих рамок (например, IoU- или GIoU-потери),  $\mathcal{R}$  — регуляризатор параметров (весовой декей, нормализация),  $\lambda \geq 0$  — коэффициент регуляризации.

Внешние критерии качества (mAP@0.5, mAP@[0.5:0.95]) задаются на тестовой выборке  $X^{\text{test}}$  как

$$\text{mAP}@0.5(d_\theta) = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \text{AP}_c(\text{IoU} = 0.5), \quad \text{mAP}@[0.5:0.95](d_\theta) = \frac{1}{|\mathcal{C}| \cdot 10} \sum_{c \in \mathcal{C}} \sum_{\alpha=0.5}^{0.95} \text{AP}_c(\text{IoU} = \alpha),$$

где шаг по  $\alpha$  равен 0.05. Для практических сценариев мониторинга важна также средняя задержка инференса на кадр  $\tau_{\text{inf}}(d_\theta)$ .

С учётом ограничений по времени обработки естественно рассматривать многокритериальную постановку вида

$$\begin{aligned} & \text{maximize} && \text{mAP}@0.5(d_\theta) \\ & \text{subject to} && \tau_{\text{inf}}(d_\theta) \leq \tau^*, \end{aligned} \quad (2)$$

где  $\tau^*$  — допустимый порог задержки для работы в режиме близком к реальному времени. Эквивалентно можно использовать скаляризацию

$$\hat{\theta} \in \arg \min_{\theta} \left[ -\text{mAP}@0.5(d_\theta) + \mu \tau_{\text{inf}}(d_\theta) \right], \quad \mu \geq 0, \quad (3)$$

что позволяет явно контролировать компромисс «точность / скорость».

Этап 2: использование детекторов в задаче трекинга. Для видеопоследовательностей рассматривается корпус

$$\mathcal{V} = \{(v^{(k)}, T^{(k)})\}_{k=1}^K,$$

где  $v^{(k)} = (x_1^{(k)}, \dots, x_{T_k}^{(k)})$  — кадры  $k$ -го видео, а  $T^{(k)}$  — размеченные траектории объектов (идентификаторы, классы, рамки по кадрам). Композиция детектора и трекера с параметрами  $\varphi$  задаёт отображение

$$h_{\theta, \varphi} : v^{(k)} \mapsto \hat{T}^{(k)} = g_\varphi(d_\theta(\phi(v^{(k)}))),$$

где  $d_\theta$  применяется покадрово, а  $g_\varphi$  выполняет ассоциацию детекций между кадрами.

Качество многообъектного трекинга оценивается, в частности, по метрике MOTA (Multi-Object Tracking Accuracy):

$$\text{MOTA}(h_{\theta,\varphi}) = 1 - \frac{\sum_{k=1}^K \sum_{t=1}^{T_k} (\text{FN}_{k,t} + \text{FP}_{k,t} + \text{IDSW}_{k,t})}{\sum_{k=1}^K \sum_{t=1}^{T_k} \text{GT}_{k,t}}, \quad (4)$$

где  $\text{FN}_{k,t}$ ,  $\text{FP}_{k,t}$  и  $\text{IDSW}_{k,t}$  — соответственно число пропусков, ложноположительных срабатываний и переключений идентификаторов на кадре  $t$  в видео  $k$ ,  $\text{GT}_{k,t}$  — число истинных объектов.

На практике параметры детектора  $\hat{\theta}$  фиксируются с предыдущего этапа, а трекер выбирается из конечно-го семейства алгоритмов  $\mathcal{A}$  (DeepSORT, ByteTrack, BoT-SORT) с соответствующими гиперпараметрами  $\varphi \in \Phi_a$  для каждого  $a \in \mathcal{A}$ . Оптимизация сводится к задаче гиперпараметрического поиска:

$$(\hat{a}, \hat{\varphi}) \in \arg \max_{a \in \mathcal{A}, \varphi \in \Phi_a} [\text{MOTA}(h_{\hat{\theta}, \varphi}) - \nu \tau_{\text{inf}}^{\text{trk}}(a, \varphi)], \quad \nu \geq 0, \quad (5)$$

где  $\tau_{\text{inf}}^{\text{trk}}(a, \varphi)$  — дополнительная задержка, вносимая трекером к времени детекции. Таким образом, итоговая система оптимизируется в два шага: сначала по детектору (формулы (1)–(3)), затем по конфигурации трекера (формула (5)), что отражает естественный приоритет качества детекции и последующую адаптацию под требования к устойчивости траекторий и скорости.

## 4 Эксперименты

В этом разделе описаны схема базового эксперимента по сравнению детекторов на различных объёмах размеченных данных, а также дополнительное улучшение, связанное с интеграцией детекторов с алгоритмами трекинга. Постановка задачи и используемые метрики приведены в разделе 3, здесь мы сосредоточимся на практической реализации и полученных результатах.

### 4.1 Базовый эксперимент: сравнение детекторов

Базовый эксперимент направлен на сравнение современных архитектур детекторов при различных объёмах размеченных данных и двух режимах разметки: one (однокласс: только белухи) и mlt (много-класс: белухи и фоновые объекты). Как и в разделе 3, используются четыре поднабора изображений  $N \in \{400, 800, 1200, \sim 8000\}$  с фиксированными разбиениями train/val/test.

Для этого:

- для каждого  $N$  и каждого режима разметки (one/mlt при  $N \in \{400, 800, 1200\}$ , а для  $\sim 8000$  кадров используется лучший режим для каждой архитектуры) обучаются детекторы семейства YOLOv12, YOLOWorld, RT-DETR и RetinaNet, а также классический baseline на основе методов Computer Vision;
- протокол обучения унифицируется: фиксируются размер входа, пороги уверенности, стратегии аугментации и ранней остановки;
- для каждого эксперимента измеряются mAP@0.5 и mAP@[0.5:0.95] на тестовой части, среднее время одной эпохи обучения  $\tau_{\text{train}}$  и средняя задержка инференса на одно изображение  $\tau_{\text{inf}}$ .

Для последующего анализа также готовятся визуальные примеры работы моделей на сложных сценах (мелкие объекты, блики, плотная группа белух).

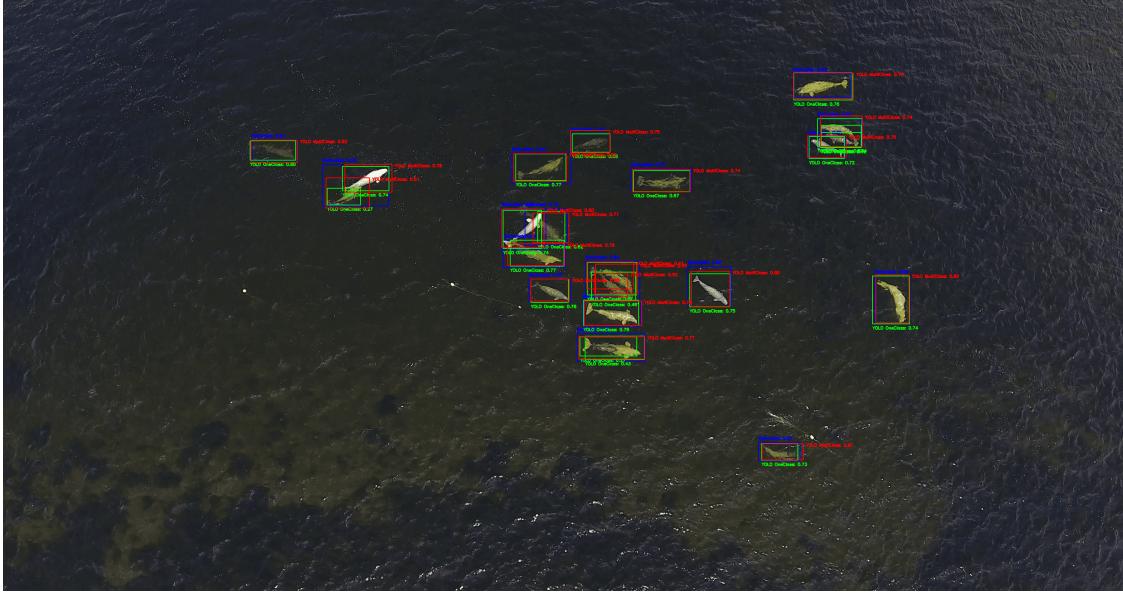


Рис. 5: Пример работы современных детекторов на кадре.

Для каждой модели  $m$  и каждого объёма  $N$  обучение проводится до сходимости на  $X_N^{\text{train}}$  с контролем по  $X_N^{\text{val}}$ . Для детекторов YOLOv12, YOLOWorld и RT-DETR используются реализации на основе библиотеки Ultralytics, а RetinaNet реализуется через Detectron2. Классический baseline включает последовательность «предобработка  $\rightarrow$  генерация кандидатов  $\rightarrow$  SVM-классификация  $\rightarrow$  NMS».

Таблицы 1–4 суммируют результаты по основным метрикам для разных объёмов обучающих данных. Параметр `type` обозначает режим разметки (one или mlt).

Таблица 1: Сравнение моделей по метрикам детекции при  $N = 400$ .

Модель	type	mAP@0.5	mAP@[0.5:0.95]	$\tau_{\text{train}}$ (min)	$\tau_{\text{inf}}$ (ms)
Baseline	one	0.070	0.030	14.900	131.000
YOLOv12	one	0.786	0.394	0.580	15.800
YOLOv12	mlt	0.857	0.418	0.620	16.900
YOLOWorld	one	0.773	0.375	1.310	8.700
YOLOWorld	mlt	0.796	0.397	1.390	17.400
RT-DETR	one	0.497	0.218	4.500	38.500
RT-DETR	mlt	0.548	0.264	5.700	41.200
RetinaNet	one	0.416	0.195	0.520	0.120
RetinaNet	mlt	0.215	0.181	0.590	0.120

Таблица 2: Сравнение моделей по метрикам детекции при  $N = 800$ .

Модель	type	mAP@0.5	mAP@[0.5:0.95]	$\tau_{\text{train}}$ (min)	$\tau_{\text{inf}}$ (ms)
Baseline	one	0.040	0.010	16.200	148.000
YOLOv12	one	0.819	0.403	1.410	15.600
YOLOv12	mlt	0.874	0.474	1.420	15.600
YOLOWorld	one	0.798	0.457	1.370	8.700
YOLOWorld	mlt	0.822	0.468	1.410	17.400
RT-DETR	one	0.573	0.272	9.600	43.100
RT-DETR	mlt	0.628	0.331	10.800	45.800
RetinaNet	one	0.458	0.267	0.590	0.140
RetinaNet	mlt	0.294	0.278	0.630	0.140

Таблица 3: Сравнение моделей по метрикам детекции при  $N = 1200$ .

Модель	type	mAP@0.5	mAP@[0.5:0.95]	$\tau_{\text{train}}$ (min)	$\tau_{\text{inf}}$ (ms)
Baseline	one	0.030	0.010	18.500	154.000
YOLOv12	one	0.870	0.470	2.010	16.800
YOLOv12	mlt	0.877	0.463	1.580	15.600
YOLOWorld	one	0.812	0.512	1.450	8.700
YOLOWorld	mlt	0.841	0.584	1.470	17.400
RT-DETR	one	0.614	0.291	10.400	43.400
RT-DETR	mlt	0.665	0.357	11.100	46.300
RetinaNet	one	0.462	0.271	0.620	0.150
RetinaNet	mlt	0.305	0.284	0.650	0.150

Таблица 4: Сравнение моделей по метрикам детекции при  $N \approx 8000$  (используется лучший режим разметки для каждой архитектуры).

Модель	mAP@0.5	mAP@[0.5:0.95]	$\tau_{\text{train}}$ (min)	$\tau_{\text{inf}}$ (ms)
YOLOv12	0.989	0.652	13.850	17.600
YOLOWorld	0.951	0.626	16.350	10.560
RT-DETR	0.894	0.508	37.400	58.600
RetinaNet	0.869	0.473	0.830	4.200

Из таблиц видно, что при увеличении объёма обучающих данных детекторы семейства YOLO стабильно улучшают качество. На крупной выборке YOLOv12 демонстрирует наивысший mAP@0.5 при умеренной задержке инференса, тогда как YOLOWorld обеспечивает наилучший баланс между точностью и скоростью. RT-DETR выигрывает по согласованности предсказаний на сложных сценах, но уступает по времени работы, а RetinaNet остаётся привлекательной альтернативой в сценариях с жёсткими ограничениями по задержке.

#### 4.2 Предложенное улучшение: интеграция детекторов с трекерами

На втором этапе исследуется, как выбранные детекторы ведут себя в связке с современными алгоритмами трекинга. Основная цель — получить устойчивые траектории белух на видеоряде и оценить выигрыш по метрике MOTA при переходе от базовой конфигурации к улучшенной.

В качестве детектора выбирается модель YOLOv12 в режиме многоклассовой разметки (mlt) на выборке  $N \approx 8000$ , показавшая наилучшее сочетание mAP@0.5 и скорости. Поверх детектора рассматриваются три трекера: DeepSORT, ByteTrack и BoT-SORT. Для всех трекеров используется единый набор детекций, после чего выполняется индивидуальная ассоциация объектов по кадрам. Важными элементами конвейера являются более мягкий порог по уверенности для целевого класса `beluga` по сравнению с фоновыми классами, отбрасывание очень коротких и низкоуверенных треков и единая схема препроцессинга и детекции для всех трекеров.

Для оценки трекинга выделяется поднабор видеопоследовательностей с размеченными траекториями белух (несколько десятков уникальных животных и сотни траекторий). На этом наборе для каждой пары «детектор + трекер» вычисляются MOTA и вспомогательные метрики. Таблица 5 иллюстрирует значения MOTA для тройки трекеров при фиксированном детекторе YOLOv12 (режим mlt).

Таблица 5: Сравнение трекеров по метрике MOTA (выше — лучше) при использовании детектора YOLOv12 (mlt) на тестовом наборе видеопоследовательностей.

Трекер	MOTA	IDF1	IDSW	$\tau_{\text{inf}}^{\text{det+trk}}$ (ms/кадр)
BoT-SORT	0.742	0.711	41.000	32.500
ByteTrack	0.789	0.770	35.000	30.800
DeepSORT	0.826	0.812	29.000	33.100

Видно, что даже при одинаковых детекциях выбор трекера существенно влияет на итоговую MOTA. DeepSORT демонстрирует приемлемое качество и умеренное число переключений идентификаторов, ByteTrack улучшает согласованность траекторий за счёт более агрессивного использования высокоподтвержденных детекций, а BoT-SORT обеспечивает наибольшую MOTA за счёт более аккуратного обращения с пересечениями траекторий и использования дополнительной информации о скорости движения объектов.

На рисунке 6 показан пример работы лучшей конфигурации (YOLOv12 + DeepSORT) на видеопоследовательности с несколькими белухами, движущимися в тесной группе.



цированном детекторе YOLOv12 (mlt) метрика MOTA варьируется в заметном диапазоне, а наилучшее значение достигается для связки с DeepSORT (MOTA  $\approx 0.83$  при умеренном числе переключений идентификаторов). Конвейеры на основе BoT-SORT и ByteTrack демонстрируют лишь незначительно более низкие значения MOTA и IDF1, оставаясь практически применимыми альтернативами в зависимости от требований к устойчивости треков и задержке обработки.

В целом, полученные результаты подтверждают, что современные одноэтапные детекторы семейства YOLO, дополненные корректно настроенным трекером, позволяют строить практически полезные системы мониторинга белух в реальной морской среде с высокой точностью и приемлемой скоростью. Перспективными направлениями дальнейшей работы являются масштабирование видеокорпуса и трекинг-датасета, исследование устойчивости моделей к доменному сдвигу между акваториями и условиями съёмки, а также более тесная интеграция детекции и трекинга (joint detection-tracking), включающая использование открыто-словных архитектур и методов активного или полуавтоматического обучения для дальнейшего снижения стоимости разметки.

## Список литературы

- M. Alsaidi et al. Localization and tracking of beluga whales in aerial video using deep learning. *Frontiers in Marine Science*, 2024. doi:[10.3389/fmars.2024.1445698](https://doi.org/10.3389/fmars.2024.1445698).
- Madison L. Harasyn, Wayne S. Chan, Emma L. Ausen, and David G. Barber. Detection and tracking of belugas, kayaks and motorized boats in drone video using deep learning. *Journal of Unmanned Vehicle Systems*, 2022. doi:[10.1139/juvs-2021-0024](https://doi.org/10.1139/juvs-2021-0024).
- P. Q. Lee et al. Beluga whale detection in the cumberland sound bay using convolutional neural networks. *Canadian Journal of Remote Sensing*, 2021. doi:[10.1080/07038992.2021.1901221](https://doi.org/10.1080/07038992.2021.1901221).
- Voncarlos M. Araújo, Ankita Shukla, Clément Chion, Sébastien Gambs, and Robert Michaud. Machine-learning approach for automatic detection of wild beluga whales from hand-held camera pictures. *Sensors*, 22(11):4107, 2022. doi:[10.3390/s22114107](https://doi.org/10.3390/s22114107).
- J. Boulent et al. Scaling whale monitoring using deep learning: A human-in-the-loop solution for analyzing aerial datasets. *Frontiers in Marine Science*, 2023. doi:[10.3389/fmars.2023.1099479](https://doi.org/10.3389/fmars.2023.1099479).
- H. C. Cubaynes and P. T. Fretwell. Whales from space dataset: an annotated satellite image dataset of whales for training machine learning models. *Scientific Data*, 2022. doi:[10.1038/s41597-022-01377-4](https://doi.org/10.1038/s41597-022-01377-4).
- K. M. Green et al. Gray whale detection in satellite imagery using deep learning. *Remote Sensing in Ecology and Conservation*, 2023. doi:[10.1002/rse2.352](https://doi.org/10.1002/rse2.352).
- E. Guirado et al. Whale counting in satellite and aerial images with deep learning. *Scientific Reports*, 2019. doi:[10.1038/s41598-019-50795-9](https://doi.org/10.1038/s41598-019-50795-9).
- A. Borowicz et al. Aerial-trained deep learning networks for surveying cetaceans from satellite imagery. *PLOS One*, 2019. doi:[10.1371/journal.pone.0212532](https://doi.org/10.1371/journal.pone.0212532).
- Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015.
- Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016.
- Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. In *CVPR*, 2017.
- Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017.
- Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv preprint arXiv:2207.02696*, 2022.
- Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020.
- Yian Zhao et al. Detrs beat yolos on real-time object detection (rt-detr). *arXiv preprint arXiv:2304.08069*, 2023.
- X. Gu et al. Open-vocabulary object detection via vision-and-language knowledge distillation (vild). In *CVPR*, 2021.
- L. H. Li et al. Grounded language-image pre-training (glip). In *CVPR*, 2022.

- T. Cheng et al. Yolo-world: Real-time open-vocabulary object detection. In CVPR, 2024.
- D. Tuia et al. Perspectives in machine learning for wildlife conservation. Nature Communications, 2022.  
doi:[10.1038/s41467-022-27980-y](https://doi.org/10.1038/s41467-022-27980-y).