

Image Models

`mlumina.image_models`

Обёртка `YoloImageModel` над `ultralytics.YOLO` (`src/mlumina/image_models/YoloImageModel.py`). Даёт единый интерфейс для инференса, обучения, распределённого (Ray) обучения/инференса и подбора гиперпараметров через Optuna.

??????????

```
from ultralytics import YOLO
from mlumina.image_models import YoloImageModel

model = YoloImageModel(YOLO("yolov8n.pt"))
```

`YoloImageModel(model: Any)` принимает **уже созданный экземпляр** `ultralytics.YOLO`, а не путь к файлу — конструктора вида `YoloImageModel(model_path="yolov8n.pt")` не существует (это расхождение со старым текстом корневого README, см. [known-issues.md](#)).

??????????

```
results = model.predict(images, conf=0.5) # kwargs пробрасываются в ultralytics.YOLO.predict
```

Параллельный инференс через Ray:

```
import ray
ray.init()

results = model.predict_ray(images=[...], num_workers=2, conf=0.5)
```

`predict_ray` требует предварительного `ray.init()`, иначе бросает `RuntimeError`. Список изображений делится на `num_workers` батчей; **каждый Ray-воркер создаёт `YOLO("yolov8n.pt")` и подгружает в неё `state_dict` текущей модели** — архитектура жёстко зашита на yolov8n. Если обёрнутая модель — не yolov8n-совместимая архитектура,

`load_state_dict` не сработает или результат будет некорректным. Нет метода `predict_parallel`.

????????

```
result = model.train(data="dataset.yaml", epochs=10, imgsz=640)
```

Распределённое обучение с усреднением весов (federated-style):

```
import ray
ray.init()

worker_results = model.train_ray(data="dataset.yaml", num_workers=2, epochs=5)
```

`train_ray` обучает `num_workers` независимых копий модели (снова через хардкод `YOLO("yolov8n.pt")`), затем усредняет их веса (`_average_states`, простое поэлементное усреднение тензоров) и загружает результат в `self.model`. Это **не** синхронное distributed training — усреднение приближённое и не эквивалентно, например, DDP или Ray Train.

?????? ??????????????????

```
study = model.tune_optuna(data="dataset.yaml", n_trials=10, direction="minimize")
```

Перебирает `lr`, `batch`, `imgsz`, `epochs` через Optuna; для каждого trial создаёт независимую копию модели (опять `YOLO("yolov8n.pt")`). Возвращает `optuna.study.Study`.

????????

```
model.save("models/best.pt")
```

???????? ??????????????

- Хардкод базовой архитектуры `"yolov8n.pt"` в `predict_ray`/`train_ray`/`tune_optuna` — методы корректны только для моделей, совместимых по форме весов с yolov8n.
- `ray.is_initialized()` проверяется только в `predict_ray`/`train_ray`, не в `tune_optuna` (там Ray не используется вовсе — тюнинг локальный, несмотря на то что модуль в целом "про Ray").

Revision #1

Created 2026-09-23 06:40:05 UTC by Маслов Алексей

Updated 2026-09-23 06:40:06 UTC by Маслов Алексей