📖 Training Arguments Guide / راهنمای آرگومان‌های آموزش

> **Complete reference for train.py command-line arguments**
>
> **train.py مرجع کامل آرگومان‌های خط فرمان**

📑 Table of Contents / فهرست مطالب

- Basic Training Parameters / پارامترهای پایه آموزش
- Loss Configuration / تنظیمات تابع هزینه
- Deformation & Augmentation / تغییر شکل و تقویت داده
- Patch & Seed Configuration / تنظیمات وصله و بذر
- System & I/O / سیستم و ورودی/خروجی
- Advanced Options / گزینه‌های پیشرفته

⚙️ Basic Training Parameters / پارامترهای پایه آموزش

`--device`

Default / پیش‌فرض: `cuda:0`

English: Specifies which GPU/CPU device to use for training.

- `cuda:0` - First GPU
- `cuda:1` - Second GPU
- `cpu` - CPU only (very slow)

فارسی: مشخص می‌کند از کدام GPU/CPU برای آموزش استفاده شود.

- `cuda:0` - اولین GPU
- `cuda:1` - دومین GPU
- `cpu` - فقط CPU (بسیار کند)

Example / مثال:

```
python train.py --device cuda:0
```

`--lr` (Learning Rate)

Default / پیش‌فرض: `0.03`

English: Controls how fast the adversarial patch is updated during training.

- Higher (0.05-0.1): Faster convergence but may be unstable

- Lower (0.01-0.02): Slower but more stable
- Recommended: 0.03 for most cases

فارسی: کنترل می‌کند که چقدر سریع وصله مخالف در طول آموزش به‌روز شود.

- بالاتر (0.1-0.05): همگرایی سریع‌تر اما ممکن است ناپایدار باشد
- پایین‌تر (0.02-0.01): کندتر اما پایدارتر
- توصیه شده: 0.03 برای اکثر موارد

Example / مثال:

```
python train.py --lr 0.03
```

`--lr_seed`

Default / پیش‌فرض: `0.01`

English: Learning rate for seed pattern optimization (when seed_type='variable' or 'langevin'). Controls how the initial patch pattern evolves during training.

فارسی: نرخ یادگیری برای بهینه‌سازی الگوی بذر (وقتی seed_type='variable' یا 'langevin'). کنترل می‌کند الگوی اولیه وصله چگونه در طول آموزش تکامل می‌یابد.

Example / مثال:

```
python train.py --seed_type variable --lr_seed 0.01
```

`--nepoch` (Number of Epochs)

Default / پیش‌فرض: `800`

English: Total number of training iterations.

- Quick test: 100-200
- Standard: 500-800
- Best results: 1000+

فارسی: تعداد کل تکرارهای آموزش.

- تست سریع: 100-200
- استاندارد: 500-800
- بهترین نتایج: 1000+

Example / مثال:

```
python train.py --nepoch 800
```

`--batch_size`

Default / پیش‌فرض: 2

English: Number of images processed simultaneously.

- Larger (4-8): Faster training, requires more GPU memory
- Smaller (1-2): Use when getting CUDA OOM errors
- Affects gradient stability

فارسی: تعداد تصاویر پردازش شده همزمان.

- بیشتر GPU بزرگ‌تر (4-8): آموزش سریع‌تر، نیاز به حافظه
- دریافت می‌کنید استفاده کنید CUDA OOM کوچک‌تر (1-2): وقتی خطای
- بر پایداری گرادیان تأثیر می‌گذارد

Example / مثال:

```
# For high-memory GPU
python train.py --batch_size 4

# For low-memory GPU (RTX 2060, etc.)
python train.py --batch_size 1
```

`--num_workers`

Default / پیش‌فرض: 4

English: Number of CPU threads for data loading.

- Set to number of CPU cores (max 8 recommended)
- Higher = faster data loading
- Reduce if getting CPU bottleneck

فارسی: تعداد رشته‌های CPU برای بارگذاری داده.

- تنظیم کنید (حداکثر 8 توصیه می‌شود CPU برابر با تعداد هسته‌های)
- بالاتر = بارگذاری داده سریع‌تر
- دریافت کردید کاهش دهید CPU اگر گلوگاه

Example / مثال:

```
python train.py --num_workers 8
```

💰 Loss Configuration / تنظیمات تابع هزینه

`--loss_type`

Default / پیش‌فرض: `max_iou`

English: Type of detection loss computation.

Type	Description	When to Use
`max_iou`	Score of box with maximum IoU	**Recommended default**
`max_conf`	Maximum confidence score	Simple, may be unstable
`softplus_max`	Smooth max using softplus	More stable gradients
`softplus_sum`	Weighted sum of all boxes	Aggregate all detections

فارسی: نوع محاسبه تابع هزینه تشخیص.

نوع	توضیحات	چه موقع استفاده کنیم
`max_iou`	امتیاز جعبه با بیشترین IoU	**پیش‌فرض توصیه شده**
`max_conf`	حداکثر امتیاز اطمینان	ساده، ممکن است ناپایدار باشد
`softplus_max`	حداکثر هموار با softplus	گرادیان‌های پایدارتر
`softplus_sum`	مجموع وزنی همه جعبه‌ها	تجمیع همه تشخیص‌ها

Example / مثال:

```
python train.py --loss_type max_iou
python train.py --loss_type softplus_max  # For more stable training
```

`--tv_loss`

Default / پیش‌فرض: `1.0`

English: Total Variation (TV) loss weight for patch smoothness.

- Higher (2.0-5.0): Smoother patches, less noisy
- Lower (0.1-0.5): More aggressive patterns, may be noisy
- 0: Disable TV regularization (not recommended)

فارسی: وزن تابع هزینه Total Variation (TV) برای هموارسازی وصله.

- بالاتر (2.0-5.0): وصله‌های هموارتر، نویز کمتر
- پایین‌تر (0.1-0.5): الگوهای تهاجمی‌تر، ممکن است نویزی باشد
- 0: غیرفعال کردن TV (توصیه نمی‌شود) منظم‌سازی

Example / مثال:

```
python train.py --tv_loss 1.0
```

`--real_loss`

Default / پیش‌فرض: `0.5`

English: Weight for 3D rendering loss (realistic scenarios with mesh deformation). Controls importance of 3D-rendered examples vs 2D patch applications.

فارسی: وزن تابع هزینه رندر سه‌بعدی (سناریوهای واقع‌گرایانه با تغییر شکل مش). اهمیت نمونه‌های رندر شده سه‌بعدی در مقابل کاربردهای وصله دوبعدی را کنترل می‌کند.

Example / مثال:

```
python train.py --real_loss 0.5
```

`--patch_loss`

Default / پیش‌فرض: `0.5`

English: Weight for 2D patch loss (simple planar transformations). Balances between 2D and 3D training paths.

فارسی: وزن تابع هزینه وصله دوبعدی (تبدیلات مسطح ساده). بین مسیرهای آموزش دوبعدی و سه‌بعدی تعادل ایجاد می‌کند.

Example / مثال:

```
# Focus more on 3D realism
python train.py --real_loss 0.7 --patch_loss 0.3

# Focus more on 2D attacks
python train.py --real_loss 0.3 --patch_loss 0.7
```

`--train_iou`

Default / پیش‌فرض: `0.45`

English: IoU threshold for considering a detection as valid target.

- Higher (0.5-0.7): Only attack well-aligned boxes
- Lower (0.3-0.4): Attack more boxes, may include false positives

فارسی: آستانه IoU برای در نظر گرفتن یک تشخیص به عنوان هدف معتبر.

- بالاتر (0.5-0.7): فقط جعبه‌های کاملاً هم‌تراز را حمله کند
- پایین‌تر (0.3-0.4): جعبه‌های بیشتری را حمله کند، ممکن است شامل مثبت‌های کاذب شود

Example / مثال:

```
python train.py --train_iou 0.45
```

🔄 Deformation & Augmentation / تغییر شکل و تقویت داده

`--tps2d_range_t`

Default / پیش‌فرض: `50.0`

English: Translation range for 2D Thin-Plate Spline (TPS) deformation. Controls how much the 2D patch can shift/move.

- Higher: More position variation
- Lower: Patch stays closer to original position

فارسی: محدوده انتقال برای تغییر شکل TPS دوبعدی (Thin-Plate Spline). کنترل می‌کند وصله دوبعدی چقدر می‌تواند جابجا شود.

- بالاتر: تنوع موقعیت بیشتر
- پایین‌تر: وصله نزدیک‌تر به موقعیت اصلی می‌ماند

Example / مثال:

```
python train.py --tps2d_range_t 50.0
```

`--tps2d_range_r`

Default / پیش‌فرض: `0.1`

English: Rotation/scale range for 2D TPS deformation. Controls warping intensity for 2D patches.

- Higher (0.2-0.5): More aggressive warping
- Lower (0.05-0.1): Subtle deformations

فارسی: محدوده چرخش/مقیاس برای تغییر شکل TPS دوبعدی. شدت انحراف برای وصله‌های دوبعدی را کنترل می‌کند.

- بالاتر (0.2-0.5): انحراف تهاجمی‌تر
- پایین‌تر (0.05-0.1): تغییر شکل‌های ظریف

Example / مثال:

```
python train.py --tps2d_range_r 0.1
```

`--tps3d_range`

Default / پیش‌فرض: `0.15`

English: Deformation range for 3D mesh TPS transformation. Controls body pose variation and cloth wrinkles.

- Higher (0.2-0.3): Extreme poses and wrinkles
- Lower (0.05-0.1): Subtle body movements

فارسی: محدوده تغییر شکل برای تبدیل TPS مش سه‌بعدی. تنوع حالت بدن و چروک‌های پارچه را کنترل می‌کند.

- بالاتر (0.2-0.3): حالت‌ها و چروک‌های شدید
- پایین‌تر (0.05-0.1): حرکات ظریف بدن

Example / مثال:

```
python train.py --tps3d_range 0.15
```

`--disable_tps2d`

Default / پیش‌فرض: `False` (enabled)

English: Disable 2D TPS deformation during training. Use when you only want 3D realistic rendering without 2D augmentation.

فارسی: غیرفعال کردن تغییر شکل TPS دوبعدی در طول آموزش. وقتی فقط رندر سه‌بعدی واقع‌گرایانه می‌خواهید بدون تقویت دوبعدی استفاده کنید.

Example / مثال:

```
python train.py --disable_tps2d  # Only 3D deformation
```

`--disable_tps3d`

Default / پیش‌فرض: `False` (enabled)

English: Disable 3D mesh TPS deformation during training. Use for faster training with only 2D planar transformations.

فارسی: غیرفعال کردن تغییر شکل TPS مش سه‌بعدی در طول آموزش. برای آموزش سریع‌تر با تنها تبدیلات مسطح دوبعدی استفاده کنید.

Example / مثال:

```
python train.py --disable_tps3d  # Only 2D transformations
```

`--resample_type`

Default / پیش‌فرض: `None`

English: Type of resampling strategy for data augmentation. Determines how training images are selected each epoch.

فارسی: نوع استراتژی نمونه‌برداری مجدد برای تقویت داده. تعیین می‌کند چگونه تصاویر آموزشی در هر epoch انتخاب شوند.

Example / مثال:

```
python train.py --resample_type random
```

🎨 Patch & Seed Configuration / تنظیمات وصله و بذر

`--seed_type`

Default / پیش‌فرض: `fixed`

English: Strategy for patch initialization and evolution.

Type	Description	Use Case
`fixed`	Start from fixed pattern, don't update	**Most common**
`random`	Random initialization each epoch	Exploration
`variable`	Learnable seed pattern	Advanced optimization
`langevin`	Stochastic gradient descent for seed	Research

فارسی: استراتژی برای مقداردهی اولیه و تکامل وصله.

نوع	توضیحات	مورد استفاده
`fixed`	شروع از الگوی ثابت، به‌روز نشود	**رایج‌ترین**
`random`	مقداردهی تصادفی هر epoch	اکتشاف
`variable`	الگوی بذر قابل یادگیری	بهینه‌سازی پیشرفته
`langevin`	گرادیان کاهشی تصادفی برای بذر	تحقیقاتی

Example / مثال:

```
python train.py --seed_type fixed
python train.py --seed_type variable --lr_seed 0.01
```

`--clamp_shift`

Default / پیش‌فرض: `0`

English: Shift value for color clamping during patch generation. Adjusts the valid color range for patch pixels.

فارسی: مقدار shift برای محدودسازی رنگ در طول تولید وصله. محدوده رنگ معتبر برای پیکسل‌های وصله را تنظیم می‌کند.

Example / مثال:

```
python train.py --clamp_shift 0.1
```

`-p, --patch`

Default / پیش‌فرض: `texture/heart.png`

English: Path to initial patch image. Use to:

- Resume training from checkpoint
- Fine-tune existing patch
- Start with specific pattern instead of random

فارسی: مسیر تصویر وصله اولیه. استفاده کنید برای:

- ادامه آموزش از checkpoint
- تنظیم دقیق وصله موجود
- شروع با الگوی خاص به جای تصادفی

Example / مثال:

```
# Start with custom pattern
python train.py --patch texture/custom_pattern.png

# Resume from previous training
python train.py --patch results/yolov5/patch_epoch_500.png
```

🏗️ Architecture & Configuration / معماری و پیکربندی

`--arch`

Default / پیش‌فرض: `yolov2`

English: Target detector architecture to attack.

Architecture	Config File	Description
`yolov2`	`v2.yaml`	YOLOv2 detector
`yolov3`	`v3.yaml`	YOLOv3 detector
`yolov5`	`v5.yaml`	**YOLOv5 (recommended)**
`yolov11`	`v11.yaml`	Latest YOLO version
`rcnn`	`faster_rcnn.yaml`	Faster R-CNN
`deformable-detr`	`ddetr.yaml`	Deformable DETR

فارسی: معماری تشخیص‌دهنده هدف برای حمله.

معماری	فایل تنظیمات	توضیحات
`yolov2`	`v2.yaml`	YOLOv2 تشخیص‌دهنده
`yolov3`	`v3.yaml`	YOLOv3 تشخیص‌دهنده
`yolov5`	`v5.yaml`	**YOLOv5 (توصیه شده)**
`yolov11`	`v11.yaml`	آخرین نسخه YOLO
`rcnn`	`faster_rcnn.yaml`	Faster R-CNN
`deformable-detr`	`ddetr.yaml`	Deformable DETR

Example / مثال:

```
python train.py --arch yolov5 --cfg configs/baseline/v5.yaml
```

`-cfg, --cfg`

Default / پیش‌فرض: `configs/baseline/v2.yaml`

English: Path to YAML configuration file containing:

- Dataset paths
- Model weights location
- Attack target class
- Input image size
- Other detector-specific settings

فارسی: مسیر فایل پیکربندی YAML حاوی:

- مسیرهای مجموعه داده

- موقعیت وزن‌های مدل
- کلاس هدف حمله
- اندازه تصویر ورودی
- سایر تنظیمات خاص تشخیص‌دهنده

Example / مثال:

```
python train.py --cfg configs/baseline/v5.yaml
```

💾 System & I/O / سیستم و ورودی/خروجی

`--save_path`

Default / پیش‌فرض: `results/demo`

English: Directory to save training outputs:

- `patch_epoch_*.png` - Generated patches
- `composite/` - Sample renders
- `runs/` - TensorBoard logs

فارسی: دایرکتوری برای ذخیره خروجی‌های آموزش:

- `patch_epoch_*.png` - وصله‌های تولید شده
- `composite/` - نمونه رندرها
- `runs/` - لاگ‌های TensorBoard

Example / مثال:

```
python train.py --save_path results/yolov5_person_attack
```

`--checkpoints`

Default / پیش‌فرض: `0`

English: Resume training from specific epoch checkpoint.

- 0: Start from scratch
- N: Load patch_epoch_N.png and continue

فارسی: ادامه آموزش از checkpoint epoch خاص.

- 0: شروع از ابتدا
- N: بارگذاری patch_epoch_N.png و ادامه

Example / مثال:

```
# Resume from epoch 500
python train.py --checkpoints 500 --save_path results/yolov5
```

`--lr_decay`

Default / پیش‌فرض: `1.1`

English: Learning rate decay factor for main patch optimization. LR is multiplied by (1/lr_decay) every few epochs.

- Higher (2.0): Aggressive decay
- Lower (1.1): Gentle decay
- 1.0: No decay

فارسی: ضریب کاهش نرخ یادگیری برای بهینه‌سازی وصله اصلی. LR در هر چند epoch (1/lr_decay) ضرب می‌شود.

- بالاتر (2.0): کاهش تهاجمی
- پایین‌تر (1.1): کاهش ملایم
- 1.0: بدون کاهش

Example / مثال:

```
python train.py --lr 0.03 --lr_decay 1.1
```

`--lr_decay_seed`

Default / پیش‌فرض: `2.0`

English: Learning rate decay factor for seed pattern (when seed_type='variable'). Typically decays faster than main patch LR.

فارسی: ضریب کاهش نرخ یادگیری برای الگوی بذر (وقتی seed_type='variable'). معمولاً سریع‌تر از LR وصله اصلی کاهش می‌یابد.

Example / مثال:

```
python train.py --seed_type variable --lr_seed 0.01 --lr_decay_seed 2.0
```

`-sp, --save_process`

Default / پیش‌فرض: `True`

English: Save intermediate patch images during training. Creates patch_epoch_0.png, patch_epoch_100.png, etc. Disable to save disk space.

و ... ایجاد می‌کند. برای patch_epoch_0.png، patch_epoch_100.png. **فارسی:** ذخیره تصاویر وصله میانی در طول آموزش صرفه‌جویی در فضای دیسک غیرفعال کنید.

Example / مثال:

```
python train.py --save_process  # Save all
python train.py  # Also saves (default=True)
```

`-n, --board_name`

Default / پیش‌فرض: `None` (auto-generated)

English: Custom name for TensorBoard logs and saved patches. If not specified, uses timestamp.

فارسی: نام سفارشی برای لاگ‌های TensorBoard و وصله‌های ذخیره شده. اگر مشخص نشود، از timestamp استفاده می‌کند.

Example / مثال:

```
python train.py --board_name yolov5_person_exp1
```

`-d, --debugging`

Default / پیش‌فرض: `False`

English: Enable debugging mode.

- Skips TensorBoard server startup
- Useful for development/testing
- Still logs metrics to files

فارسی: فعال کردن حالت اشکال‌زدایی.

- راه‌اندازی سرور TensorBoard را رد می‌کند
- برای توسعه/تست مفید است
- هنوز معیارها را در فایل‌ها ثبت می‌کند

Example / مثال:

```
python train.py --debugging
```

`-np, --new_process`

Default / پیش‌فرض: `False`

English: Start new TensorBoard server process instead of reusing existing. Use when previous TensorBoard didn't shut down properly.

فارسی: شروع فرآیند سرور TensorBoard جدید به جای استفاده مجدد از موجود. وقتی TensorBoard قبلی به درستی خاموش نشده استفاده کنید.

Example / مثال:

```
python train.py --new_process
```

🎯 Complete Examples / مثال‌های کامل

Example 1: Basic YOLOv5 Attack / حمله پایه YOLOv5

```
python train.py \
    --arch yolov5 \
    --cfg configs/baseline/v5.yaml \
    --save_path results/yolov5_basic \
    --nepoch 800 \
    --loss_type max_iou \
    --lr 0.03 \
    --batch_size 2
```

Example 2: High-Quality 3D Attack / حمله سه‌بعدی با کیفیت بالا

```
python train.py \
    --arch yolov5 \
    --cfg configs/baseline/v5.yaml \
    --save_path results/yolov5_3d_realistic \
    --nepoch 1000 \
    --real_loss 0.7 \
    --patch_loss 0.3 \
    --tps3d_range 0.2 \
    --loss_type softplus_max \
    --lr 0.025 \
    --tv_loss 2.0 \
    --batch_size 4
```

English: Focus on realistic 3D rendering with more mesh deformation and smoother patches.

فارسی: تمرکز بر رندر سه‌بعدی واقع‌گرایانه با تغییر شکل بیشتر مش و وصله‌های هموارتر.

Example 3: Fast 2D Attack (GPU Limited) / حمله سریع دوبعدی (محدود GPU)

```
python train.py \
    --arch yolov5 \
    --cfg configs/baseline/v5.yaml \
    --save_path results/yolov5_fast \
    --nepoch 500 \
    --disable_tps3d \
    --patch_loss 1.0 \
    --real_loss 0.0 \
    --lr 0.05 \
    --batch_size 1 \
    --num_workers 2
```

English: Only 2D transformations, smaller batch, faster convergence for limited hardware.

فارسی: فقط تبدیلات دوبعدی، batch محدود سخت‌افزار برای سریع‌تر همگرایی، کوچک‌تر.

Example 4: Multi-Detector Universal Attack / حمله جهانی چند تشخیص‌دهنده

```
# Train against YOLOv5
python train.py --arch yolov5 --cfg configs/baseline/v5.yaml --save_path
results/universal --nepoch 800

# Fine-tune with YOLOv3
python train.py --arch yolov3 --cfg configs/baseline/v3.yaml --patch
results/universal/patch_epoch_799.png --save_path results/universal_v3 --nepoch
200 --lr 0.01

# Fine-tune with Faster R-CNN
python train.py --arch rcnn --cfg configs/baseline/faster_rcnn.yaml --patch
results/universal_v3/patch_epoch_199.png --save_path results/universal_final --
nepoch 200 --lr 0.01
```

English: Progressive fine-tuning across multiple detectors for transferable attacks.

فارسی: تنظیم دقیق پیشرونده در چندین تشخیص‌دهنده برای حملات قابل انتقال.

Example 5: Resume from Checkpoint / ادامه از Checkpoint

```
python train.py \
    --arch yolov5 \
    --cfg configs/baseline/v5.yaml \
    --save_path results/yolov5_resumed \
    --checkpoints 500 \
    --nepoch 1000 \
    --lr 0.02 \
    --board_name yolov5_resumed_from_500
```

English: Continue training from epoch 500 with adjusted learning rate.

فارسی: ادامه آموزش از epoch 500 با نرخ یادگیری تنظیم شده.

🔍 Parameter Tuning Guide / راهنمای تنظیم پارامترها

🩺 Common Issues & Solutions / مشکلات رایج و راه‌حل‌ها

Loss not decreasing / هزینه کاهش نمی‌یابد

English Solutions:

- Increase `--lr` to 0.05
- Change `--loss_type` to `softplus_max`
- Reduce `--tv_loss` to 0.5
- Check dataset has target class

راه‌حل‌های فارسی:

- `--lr` را به 0.05 افزایش دهید
- `--loss_type` را به `softplus_max` تغییر دهید
- `--tv_loss` را به 0.5 کاهش دهید
- کلاس هدف دارد dataset بررسی کنید

CUDA Out of Memory / تمام شد CUDA حافظه

English Solutions:

```
--batch_size 1          # Reduce batch size
--num_workers 2         # Reduce workers
--disable_tps3d         # Disable 3D rendering
```

راه‌حل‌های فارسی:

```
--batch_size 1          # کاهش اندازه batch
--num_workers 2         # کاهش workers
--disable_tps3d         # غیرفعال کردن رندر سه‌بعدی
```

Training too slow / آموزش بسیار کند

English Solutions:

```
--nepoch 500            # Reduce epochs
--disable_tps3d         # Skip 3D rendering
--batch_size 4          # Increase batch (if GPU allows)
--num_workers 8         # More data loading threads
```

راه‌حل‌های فارسی:

```
--nepoch 500            # کاهش تعداد epochs
--disable_tps3d         # رد کردن رندر سه‌بعدی
--batch_size 4          # افزایش batch (اگر GPU اجازه دهد)
--num_workers 8         # رشته‌های بیشتر برای بارگذاری داده
```

Patch too noisy / وصله بسیار نویزی

English Solutions:

```
--tv_loss 2.0           # Increase smoothness
--lr 0.02               # Reduce learning rate
--loss_type softplus_max  # Smoother optimization
```

راه‌حل‌های فارسی:

```
--tv_loss 2.0           # افزایش هموارسازی
--lr 0.02               # کاهش نرخ یادگیری
--loss_type softplus_max  # بهینه‌سازی هموارتر
```

📊 Monitoring Training / نظارت بر آموزش

TensorBoard

English: Launch TensorBoard to monitor training progress:

```
tensorboard --logdir results/yolov5/runs/
```

Open browser: http://localhost:6006

فارسی: TensorBoard را برای نظارت بر پیشرفت آموزش راه‌اندازی کنید:

```
tensorboard --logdir results/yolov5/runs/
```

مرورگر را باز کنید: http://localhost:6006

Output Files / فایل‌های خروجی

English:

```
results/yolov5/
├── patch_epoch_0.png       # Initial patch
├── patch_epoch_100.png     # Intermediate patches
├── patch_epoch_799.png     # Final patch (USE THIS!)
├── composite/              # Sample rendered images
│   ├── epoch_0.png
│   └── epoch_799.png
└── runs/                   # TensorBoard logs
    └── events.out.tfevents.*
```

فارسی:

```
results/yolov5/
├── patch_epoch_0.png       # وصله اولیه
├── patch_epoch_100.png     # وصله‌های میانی
├── patch_epoch_799.png     # وصله نهایی (از این استفاده کنید!)
├── composite/              # تصاویر رندر شده نمونه
│   ├── epoch_0.png
│   └── epoch_799.png
└── runs/                   # لاگ‌های TensorBoard
    └── events.out.tfevents.*
```

🎓 Advanced Topics / موضوعات پیشرفته

Variable Seed Training / آموزش بذر متغیر

English: Allows the base pattern to evolve during training:

```
python train.py \
    --seed_type variable \
    --lr 0.03 \
    --lr_seed 0.01 \
    --lr_decay 1.1 \
    --lr_decay_seed 2.0 \
    --nepoch 1000
```

فارسی: به الگوی پایه اجازه می‌دهد در طول آموزش تکامل یابد:

Langevin Dynamics / دینامیک Langevin

English: Stochastic optimization for exploring pattern space:

```
python train.py \
    --seed_type langevin \
    --lr_seed 0.005 \
    --nepoch 1500
```

فارسی: بهینه‌سازی تصادفی برای کاوش فضای الگو:

📚 Further Reading / مطالعه بیشتر

English:

- Training and Inference Guide for complete workflow
- Attack Methods for loss function details
- Add New Detector for architecture customization

فارسی:

- راهنمای آموزش و استنتاج برای گردش کار کامل
- روش‌های حمله برای جزئیات تابع هزینه
- افزودن تشخیص‌دهنده جدید برای سفارشی‌سازی معماری

🎯 **Happy Adversarial Patch Training! / آموزش موفق وصله مخالف!**