HEPSCT deep learning 使用手册

> 适用版本：v0.1.0。部署版标题栏会显示 `v0.1.0+<commit>`，用于确认当前页面加载的是哪一次构建。
>
> 本文截图来自一次真实演示：数据根目录为 `4k_data`（4096×4096），训练只取每个 split 的前 4 张图、跑 20 个 epoch，推理使用一个已有的 `LILoss_22.pth` 权重。截图中的路径、运行名和数字都只是示例，实际使用时以你自己的数据为准。

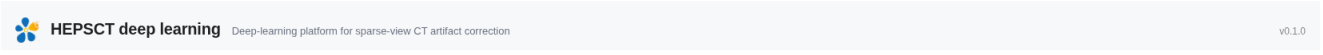
目录

1. 界面总览
2. Data 标签页
3. Train 标签页
4. Inference 标签页
5. Evaluation 标签页
6. Models 标签页
7. 典型工作流
8. 常见问题

---

1 界面总览

1.1 标题栏



元素	作用	说明
图标	HEPSCT 标识	固定在页面左上角
标题	`HEPSCT deep learning`	应用名称
副标题	`Deep-learning platform for sparse-view CT artifact correction`	说明用途：稀疏视角 CT 伪影校正的深度学习平台
版本号	右上角 `v0.1.0`	部署版形如 `v0.1.0+c8e336e`，即版本号 + 构建 commit。热更新后只有新打开的页面会显示新版本

1.2 标签页

整个应用分为五个标签页，从左到右依次是：

标签页	用途
Data	选择数据集、检查目录结构、浏览并预览训练/验证/测试图像
Train	配置超参数、启动/停止训练、观察损失曲线和每个 epoch 的记录
Inference	选择模型和权重，对一整个目录的图像做伪影校正
Evaluation	把预测结果与目标图像配对，计算 MAE/MSE/PSNR/SSIM/MSSSIM
Models	查看已注册的模型，或在当前内核中加载一个新的模型插件

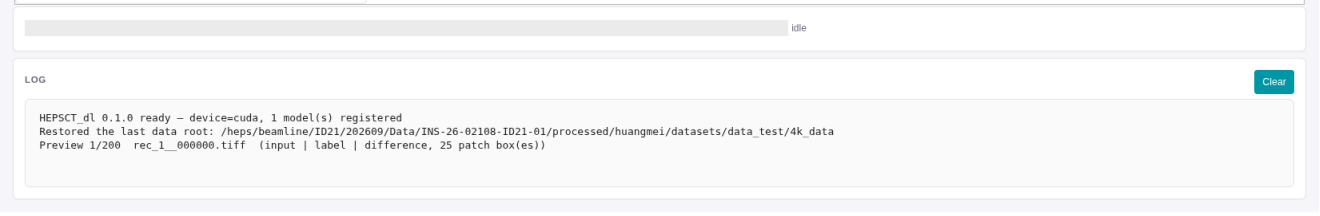
切换标签页不会丢失已填写的参数，正在运行的任务也不会中断；它们的状态在各自标签页里保留。

1.3 按钮颜色

按钮颜色表示任务的当前状态，不需要看文字也能判断：

颜色	状态	典型场景
蓝色	主操作、尚未运行	初始的「Start training」「Run inference」「Evaluate」
青色	辅助操作	「Refresh」「Load preview」「Clear」
红色	正在运行	任务进行中的开始按钮
绿色	成功完成	训练/推理/评估正常结束，或预览成功
橙色	完成但降级、被提前停止或出错	点击 Stop 后、输出目录全部被跳过、参数错误
灰色	不可用	没有任务可停时的「Stop」

1.4 进度条与状态文字



页面底部有一条**全局共享**的进度条，训练、推理、评估共用它，不会各自开一条。

元素	作用	说明
进度条	显示最近一次长任务的进度	空闲时显示 `idle`；任务结束后保留最后一条结果（如 `training finished`、`100/100 files`）

元素	作用	说明
右侧文字	进度详情	训练时包含 `epoch x/y`、当前阶段（train/val）、百分比、loss、每图耗时和 ETA；推理时包含完成数、百分比、每图耗时和 ETA；评估时显示 `完成数/总数 files`

需要特别注意的是：**训练时进度条显示的是「当前 epoch 内当前阶段」的进度**，每进入一个 epoch、或从训练阶段切到验证阶段，进度都会重新从 0 开始；整体进度要看文字里的 `epoch x/y`。推理和评估则是整个目录的进度。

单张图像预览（Inference 标签页的「Preview」）不占用这条进度条，结果直接写在 Run 卡片里。

1.5 日志面板

每个标签页底部都有同一个日志面板。

元素	作用	说明
面板	实时显示运行日志	**最新一行在最上面**；固定高度，约 7 行可见，超出可滚动
行数上限	自动丢弃最旧的内容	最多保留 400 行，超过后一次性丢掉最旧的四分之一，行为类似 `tail`
「Clear」	清空面板	只清显示，不会删除运行目录里的日志文件
日志级别	`INFO` / `WARN` / `ERROR`	`WARN` 和 `ERROR` 前缀醒目，通常直接说明下一步该改什么

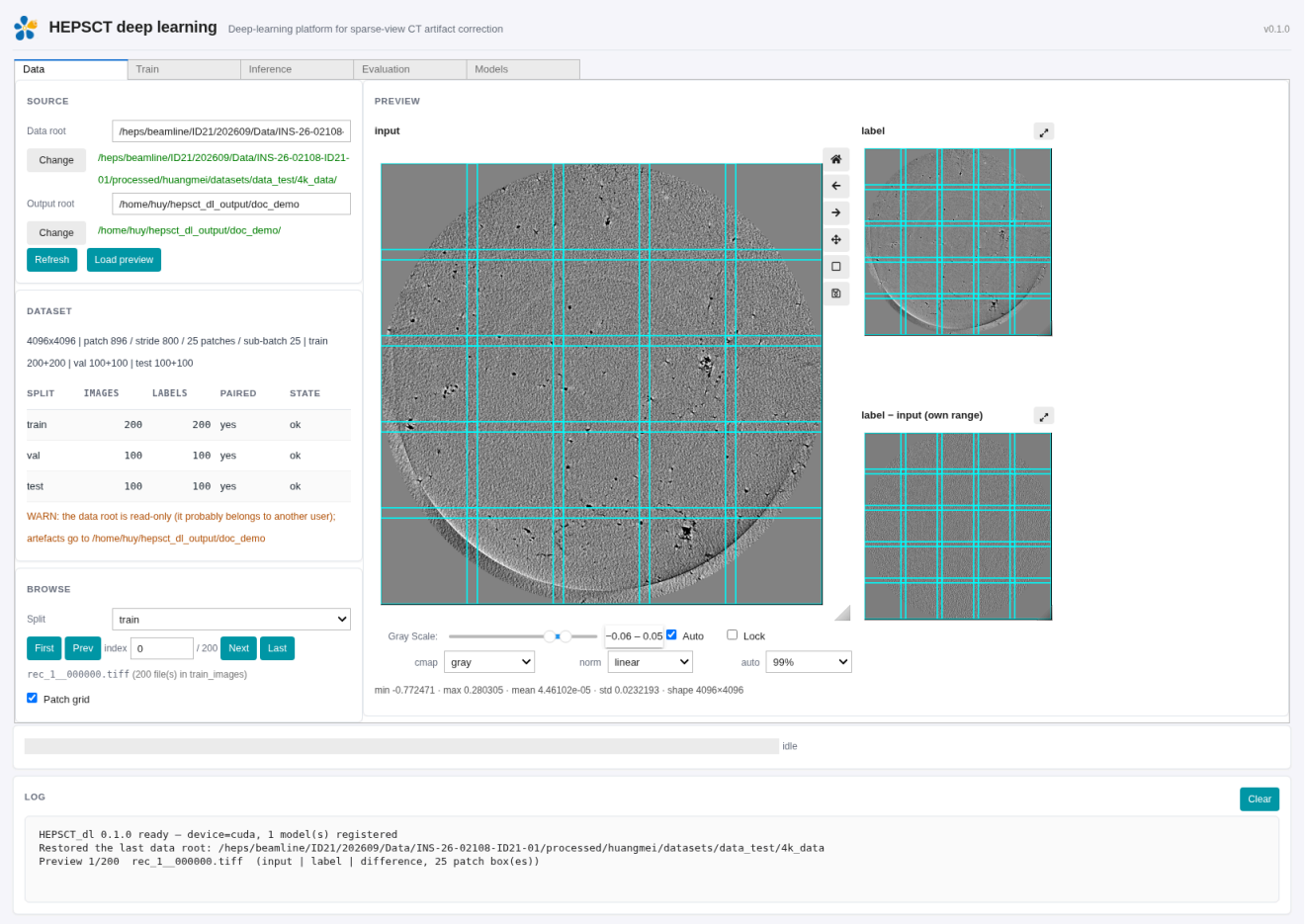
面板只是运行日志的一个窗口，**完整、按时间顺序**的日志在每个运行目录下：

- `train.log`：训练/推理/评估的主日志；
- `stdout.log`、`stderr.log`：子进程的标准输出与标准错误，用于排查崩溃。

标题栏右侧的悬停提示也写明了这一点。

---

2 Data 标签页



Data 标签页解决三个问题：目录是否被正确识别、每个 split 是否能一一配对、以及图像和 patch 切分是否就是你以为的那样。左侧是设置区，右侧是预览区。

2.1 Source：数据源

控件	作用	说明
Data root	数据集根目录	只需选一次，推理输入、评估目标、输出目录等其余路径都从这里推导。输入框里直接输入路径后按 **Enter** 或点击别处才会生效（不是每敲一个字符就重新扫描）
目录选择器	图形化选择目录	路径输入框下方是内嵌目录浏览器；按钮在未选路径时显示「Select」，已选路径时显示「Change」。只显示目录，进入后选中即可回填路径
选择器下方路径行	显示选择器当前所在目录	绿色表示该目录存在；黑色表示路径尚未创建或无法列出（例如还没创建的输出目录）。**输入框里的路径始终是应用实际使用的值**
Output root	运行产物写到哪里	留空表示写到数据根目录下——只有该目录属于你、可写时才可行
Refresh	重新探测	重新读取数据根目录：split、图像数量、尺寸、tiling 计划都会刷新；未被你手动改过的派生路径（输出根、推理输入/输出、评估预测/目标）也会重新推导

控件	作用	说明
Load preview	载入预览	显示当前 split 的第一张图；如果当前 split 没有图像，则切换到第一个有图像的 split

输出根目录的默认规则：

1. 数据根目录**可写**：输出根默认就是数据根；
2. 数据根目录**只读**（束线数据通常属于别的用户）：输出根默认是 `~/hepsct_dl_output/<数据根目录名>/`，日志里会给出 `WARN`；
3. 一旦你手动设置了输出根，Refresh 不会覆盖它。

数据根只读且输出根可写时，Dataset 卡片里的警告会直接写明产物去了哪里，例如：

```
WARN: the data root is read-only (it probably belongs to another user);
artefacts go to /home/huy/hepsct_dl_output/doc_demo
```

> 为什么不能总是写在数据旁边：束线数据往往是别人的、只读的，如果默认写进去，训练会在第一次写文件时失败。把产物集中到自己的 home 下，也是「不要改别人的数据」这条约束的落实。

2.2 Dataset：数据集概览

元素	作用
摘要行	一行给出图像尺寸、tiling 计划、各 split 的图像与标签数量。例如：`4096×4096 | patch 896 / stride 800 / 25 patches / sub-batch 25 | train 200+200 | val 100+100 | test 100+100`
表格	每个 split 一行，列为 `split`、`images`、`labels`、`paired`、`state`
警告	目录或配对有问题时的 `WARN` 行

`state` 的取值：

值	含义
`ok`	图像与标签数量一致，且文件名能一一配对
`images only`（橙色）	该 split 只有 `*_images`，没有 `*_masks`；可以推理，但不能作为训练/验证/评估的标签
`unpaired`（橙色）	有标签目录，但存在没有对应标签的图像，或没有对应图像的标签；配对**严格按文件名**

数据根的目录约定：

```
<data root>/
├── train_images/   train_masks/     # 训练对
├── val_images/     val_masks/       # 验证对
└── test_images/    test_masks/      # 测试对
```

- 至少要有 `train_images` 才能训练；`val` 缺失时，最佳模型改为跟随训练 loss，验证列显示 `–`。
- 只有 `test_images` 时仍然可以做推理。
- 常见警告包括：数据根只读、图像不是正方形、图像尺寸不在 tiling 表（当前支持 4096 与 6144）、某个 split 未配对、目录下没有 TIFF。

2.3 Browse：按 split 浏览

控件	作用	说明
Split	选择要浏览的 split	下拉框只列出有图像的 split；切换后自动显示该 split 的第 0 张
First	第一张	等价于把 index 设为 0
Prev	上一张	到第一张后不再前进
index	图像序号	**从 0 开始**；可以直接输入数字回车跳转，越界会被夹到有效范围内
/ N	总数	当前 split 的 TIFF 数量
Next	下一张	到最后一张后不再前进
Last	最后一张	跳到该 split 的最后一张
文件名行	当前图像	显示文件名和该目录内的文件数量，例如 `rec_1__000000.tiff (200 file(s) in train_images)`
Patch grid	显示 patch 网格	默认开启。把网络实际看到的切块画在所有预览窗格上：4096 图像为 896 像素 patch、步长 800，共 25 块，相邻块重叠 96 像素。检查切分是否合理时开着；只看图像细节时可以关掉

2.4 Preview：图像预览

预览区是三窗格对比：

窗格	内容	用途
input	模型输入（稀疏视角/含伪影图像）	大图
label	目标图像（完整视角/校正后图像）	缩略图，与 input 共用灰度窗口
label − input (own range)	标签减输入	缩略图，就是网络需要学习的**校正量**；使用自己的自动灰度范围

缩略图标题右侧的 ↗ 按钮可以把该窗格换到大图位置，原来的大图移到右侧栏。这样可以全尺寸查看 label 或差值，同时继续用同一组对比度控件调节。

灰度显示控件只出现在大图下方，作用范围如下：

控件	作用	说明
Gray Scale	手动设置显示窗口 `[vmin, vmax]`	两端滑块可拖动；松开鼠标后才重绘（`continuous_update=False`），避免拖动过程中反复重算
Auto	自动计算显示窗口	默认开启。按 auto 百分位从当前图像重新计算窗口
Lock	锁定当前窗口	勾选后换图不再自动重算，所有图像都用同一个灰度范围，便于逐张比较亮度变化
cmap	颜色映射	`gray`、`viridis`、`inferno`、`magma`、`plasma`、`turbo`；伪彩便于看微弱结构
norm	数值归一化	`linear`、`log`、`power`。log/power 要求数据非负；遇到负值会自动回退到 linear 并在日志中说明
auto	自动窗口的百分位	`100%`、`99%`、`95%`、`90%`，默认 **99%**：取中间 99% 的像素（0.5%–99.5% 分位）作为窗口，抑制极亮点和坏像素对显示的干扰
统计行	`min · max · mean · std · shape`	针对当前大图的整幅数据，不是显示窗口内的数据；`nan/inf` 像素会被排除并单独计数
工具栏（右侧竖排）	matplotlib 图像工具	Home 复位视图、Back/Forward 视图历史、Pan 平移、矩形缩放、Download plot（把当前视图导出为图片）
光标读数	鼠标位置的像素值	在大图上移动时显示 `x … y … I …`；移出图像后保留最后一个读数

如果整幅图都是 `nan/inf`，窗格会是空白，日志会明确警告；如果是 log/power 遇到负值，会自动回退 linear，而不是画出一张错误的图。

2.5 路径的自动推导与会话记忆

探测数据根之后，应用会自动填充：

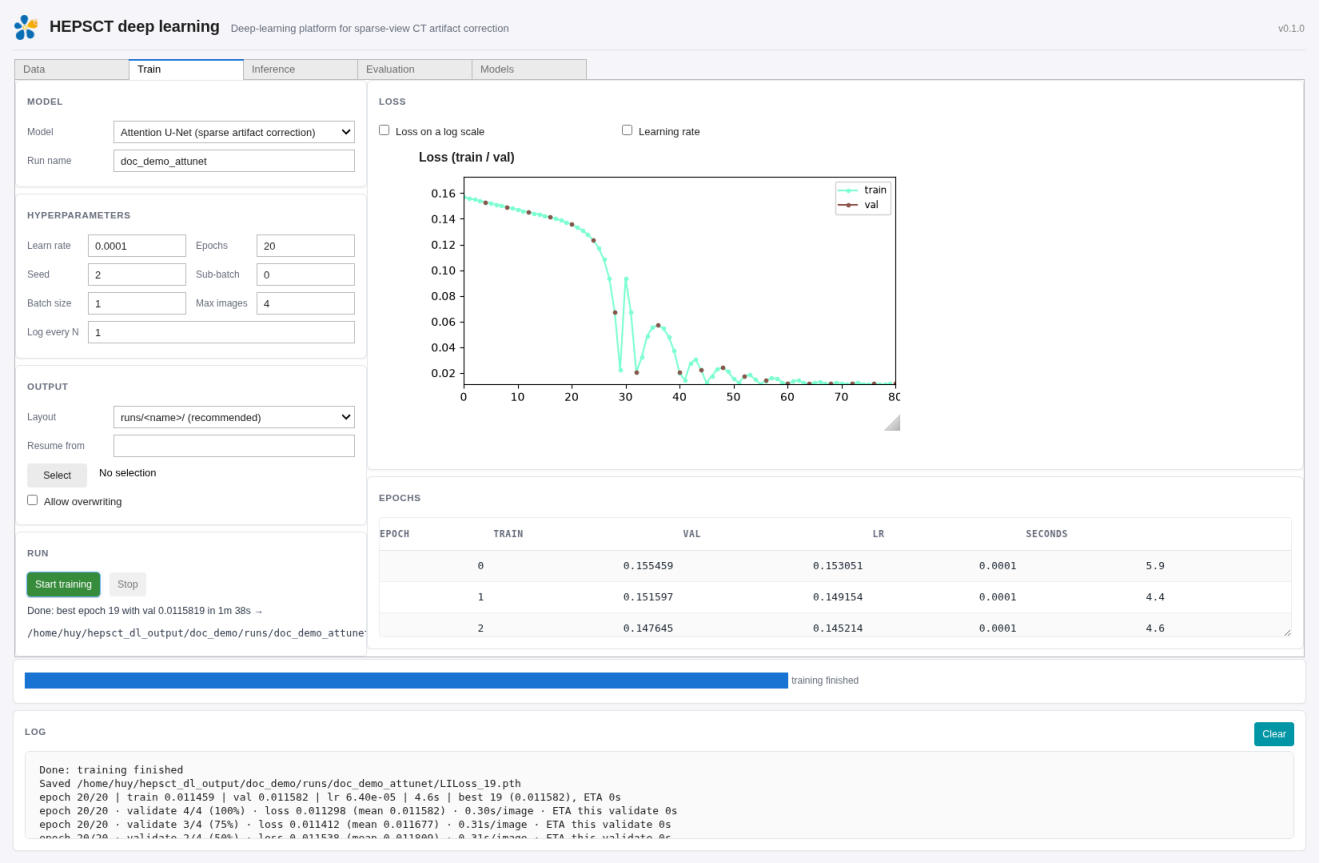
字段	默认值
Output root	数据根（可写时），否则 `~/hepsct_dl_output/<数据根目录名>`
Inference → Input dir	`<data root>/test_images`（如果存在）
Inference → Output dir	`<output root>/runs/predict/<YYYYmmdd_HHMMSS>`（运行时才创建）
Evaluation → Prediction dir	`<data root>/test_predict`（如果里面有 TIFF），否则用推理的默认输出目录
Evaluation → Target dir	`<data root>/test_masks`（如果存在）

这些自动值只会在字段为空、或仍等于上一次自动推导值时写入；**一旦你手动改过，刷新不会把它改回去。**

上次成功使用的 `data_root`、`output_root` 和 `sub_batch_size` 会记住，下次打开页面自动恢复。恢复时先恢复输出根、再恢复数据根，避免只读数据集先触发一次错误的 home 回退警告。

---

3 Train 标签页



左侧是训练配置，右侧是损失曲线和逐 epoch 记录。

3.1 Model

控件	作用	说明
Model	选择要训练的模型插件	列表来自插件注册表；每个插件声明自己支持的图像尺寸、loss、metric 和 checkpoint 命名
Run name	运行名称	决定 `<output root>/runs/<name>/`。留空则自动生成 `<model>_<YYYYmmdd_HHMMSS>`；如果填了 Resume from 且留空，则沿用被恢复运行的名字

3.2 Hyperparameters

控件	默认值	作用与说明
Learn rate	`1e-4`	Adam 的学习率。调度器沿用原程序：每 10 个 epoch 乘以 0.8
Epochs	`60`	训练轮数。启动后修改这个框不会影响正在运行的训练
Seed	`2`	随机种子，用于数据顺序等，便于复现
Sub-batch	`0`	一次前向传播处理的 patch 数。`0` 表示按 tiling 计划取：4096 为 25，6144 为 10（沿用原表）。启动前有显存预检，放不下会拒绝启动并给出建议值
Batch size	`1`	一个优化步包含的**整图**数量。原程序用 1；网络实际一次前向的批量是 Sub-batch，显存取决于二者乘积 `sub_batch × batch_size`
Max images	`0`	`0` 表示使用全部图像；填一个小数字则每个 split 只取**按文件名排序的前 N 张**，用于低成本试跑（smoke test）
Log every N	`10`	每 N 个 batch 往日志面板写一行。一个 batch 是一个优化步；batch size 为 1 时即每 N 张图一行。日志面板仍按这个频率刷新，但**进度条每个 batch 都会前进**；填 `1` 表示每个 batch 都记一行

显存与速度的经验：

- `sub_batch × batch_size` 决定了单步的显存占用；
- 图像是 4096 或 6144 时，整图训练本身的显存需求就很高，先把 `Max images` 设为 2–4 试通流程，再放开；
- 预检失败时不要盲目重试，按日志给出的建议值调整 Sub-batch。

3.3 Output

控件	作用	说明
Layout	产物目录布局	`runs/<name>/ (recommended)`：所有产物都在 `<output root>/runs/<name>/` 下；`epoch/ + loss files in the data root (legacy)`：checkpoint 写到 `<output root>/epoch/`，损失曲线写到 `<output root>/Local_loss_line.png` 和 `Local_loss.txt`，兼容原程序（输出根默认等于数据根，所以这个选项的文案写作 data root；实际始终写到当前输出根）
Resume from	断点续训	可填一个**运行目录**或**某个 checkpoint 文件**。目录形式优先用 `last.pth`，没有则用 `best.pth`；留空表示从头训练
Allow overwriting	允许覆盖已有权重	默认关闭。关闭时，若本次运行可能覆盖已有的权重文件，会在启动前直接拒绝，而不是跑到一半把旧模型覆盖掉

> 无论选哪种 Layout，运行记录 `config.json`、`history.csv`、`last.pth`、`best.pth` 和日志都固定在 `<output root>/runs/<name>/`，因此 legacy 布局下的运行同样可以续训和追溯。

3.4 Run

控件	作用	说明
Start training	启动训练	训练在**独立进程**中运行，界面仍可切换标签页、查看日志；按钮变红表示正在运行
Stop	请求停止	在**当前正在处理的图像**结束后生效，不会强杀进程。正在进行的 epoch 会保留已算出的部分，记录为 `partial`，且该 epoch 不做验证；`best.pth` 仍是此前最优的权重；再次 Start 并 Resume 会从下一个 epoch 继续。点击一次后按钮禁用，重复点击没有额外作用
状态行	本次训练的结果	正常结束：`Done: best epoch 19 with val 0.0115819 in 1m 38s → <run dir>`；主动停止显示 `Stopped` 并注明 `(stopped early)`；参数错误显示红色 `ERROR: …`

3.5 Loss 卡片

控件	作用	说明
Loss (train / val)	损失曲线	训练曲线每个 **batch** 一个点（batch size 为 1 时即每张图）；验证 loss 在每个 epoch 结束时以点标在同一条横轴上（横轴是 batch 序号）。没有验证集时不会画一条空的 val 曲线
Loss on a log scale	对数纵轴	收敛后 loss 的绝对值很小，线性坐标下后段几乎是一条平线；对数坐标能看清后段变化。遇到非正值时该点无法画在对数轴上
Learning rate	显示学习率曲线	默认隐藏。学习率只在调度器触发时阶跃下降，看一次即可；勾选后在 loss 图下方显示

3.6 Epochs 卡片

逐 epoch 的表格，可滚动：

列	含义
`epoch`	epoch 序号（从 0 开始）
`train`	该 epoch 训练集平均 loss
`val`	该 epoch 验证集 loss；没有验证集时显示 `–`（表示没有测量，不是数值错误）
`lr`	该 epoch 使用的学习率
`seconds`	该 epoch 耗时

列	含义
（空列）	当前最优 epoch 标 `<b>best</b>`；被 Stop 截断的 epoch 标 `partial`（橙色）

表格按 epoch 顺序排列，不提供排序；要看整体趋势请看左侧曲线。底部状态行里的 `best epoch` 与表格中的 `best` 一致。

3.7 训练产物

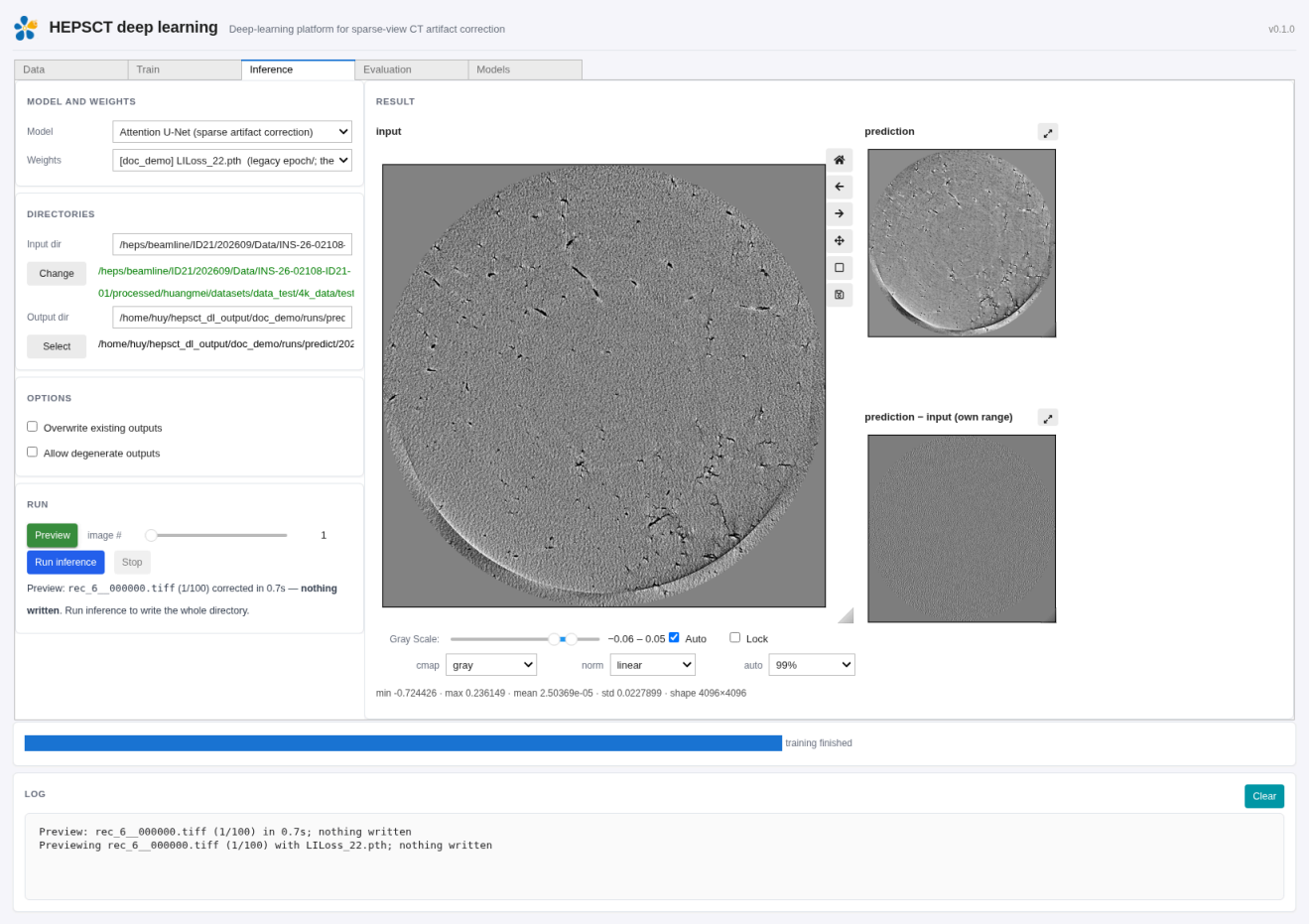
一次 `runs` 布局的训练结束后，运行目录里会有：

文件	内容
`config.json`	本次运行的完整配置（数据根、超参数、模型、设备等）
`history.csv`	逐 epoch 的 loss、学习率、耗时、是否最优、是否 partial
`last.pth`	每个 epoch 都写，续训的起点（含权重、优化器、调度器状态）
`best.pth`	当前最优权重（有验证集时跟随 val loss，否则跟随 train loss）
`LILoss_<epoch>.pth`	插件命名的权重，**只在被监控的 loss 创新低时保存**（有验证集时是 val loss，没有时是 train loss）；文件名里的数字是 epoch，不是排名
`train.log` / `stdout.log` / `stderr.log`	主日志、标准输出、标准错误
`.lock`	运行锁，避免两个进程同时写同一个运行目录

legacy 布局下，`LILoss_<epoch>.pth` 写到 `<output root>/epoch/`，损失曲线写到输出根目录的 `Local_loss_line.png` / `Local_loss.txt`，其余记录仍在 `runs/<name>/`。

---

4 Inference 标签页



推理的流程是：切块（tile）→ 逐块预测 → 合并（merge）→ 按原文件名写出。左侧选择模型、权重和目录，右侧显示一张输入、预测和二者差值的对比。

4.1 Model and weights

控件	作用	说明
Model	推理使用的模型	必须是所选权重训练时使用的模型。若 checkpoint 记录了自己的模型名，选错会在启动前报错，而不是在加载权重时抛出一堆 unexpected keys
Weights	选择权重文件	扫描**数据根和输出根**，列出可用的 checkpoint：`runs/<run>/best.pth`、`last.pth`，以及 legacy 的 `epoch/*.pth`（runs 布局里插件命名的 `LILoss_<epoch>.pth` 不单独列出；最优和最新状态已分别由 `best.pth`、`last.pth` 提供）。标签会带上信息：`epoch 19, best val 0.0115819, attunet_artifact`；legacy 文件会注明「数字是 epoch，不是排名」。来自非数据根的条目带 `[目录名]` 前缀

权重列表在数据根/输出根重新探测后、以及训练结束后自动刷新。

4.2 Directories

控件	作用	说明
Input dir	待校正图像目录	目录里的**每个 TIFF** 都会被处理，输出保持同名。默认指向 `test_images`
Output dir	预测结果目录	默认是 `<output root>/runs/predict/<时间戳>`，运行前不存在也没关系，第一次写入时会创建。**不允许与 Input dir 相同**：那会把预测写回原图，覆盖掉输入

4.3 Options

控件	默认	作用与说明
Overwrite existing outputs	关	关闭时，输出目录里已存在的同名文件会被**跳过**，并在结束状态里报告 `skipped N`。若全部被跳过，状态会变成橙色「Nothing written」，避免把「什么都没做」误认为成功
Allow degenerate outputs	关	关闭时，如果某张预测几乎是常数（预测的像素标准差不到输入标准差的 1%），会在第一张就停止并报错，不写这一张。这通常意味着权重与数据不匹配（模型不对、归一化不同）。确认这是预期行为时才勾选，允许写出

4.4 Run

控件	作用	说明
Preview	单张预览	用当前权重校正输入目录中的一张图，结果显示在右侧 Result 卡片；**不写任何文件**，也不需要 Output dir。按钮成功变绿
image # 滑块	选择预览哪一张	**从 1 开始**计数；范围跟随 Input dir 里的文件数。默认第 1 张
Run inference	对整个目录推理	启动后进度条显示整体完成数、百分比、每图耗时和 ETA；按钮变红
Stop	请求停止	在**当前正在写出的图像**结束后生效；已经写出的文件保留。再次运行并勾选 Overwrite 可补齐
状态行	结果	正常：`Done: wrote 100, skipped 0 → <output dir>`；跳过全部或提前停止：橙色说明；出错：红色 `ERROR: …`

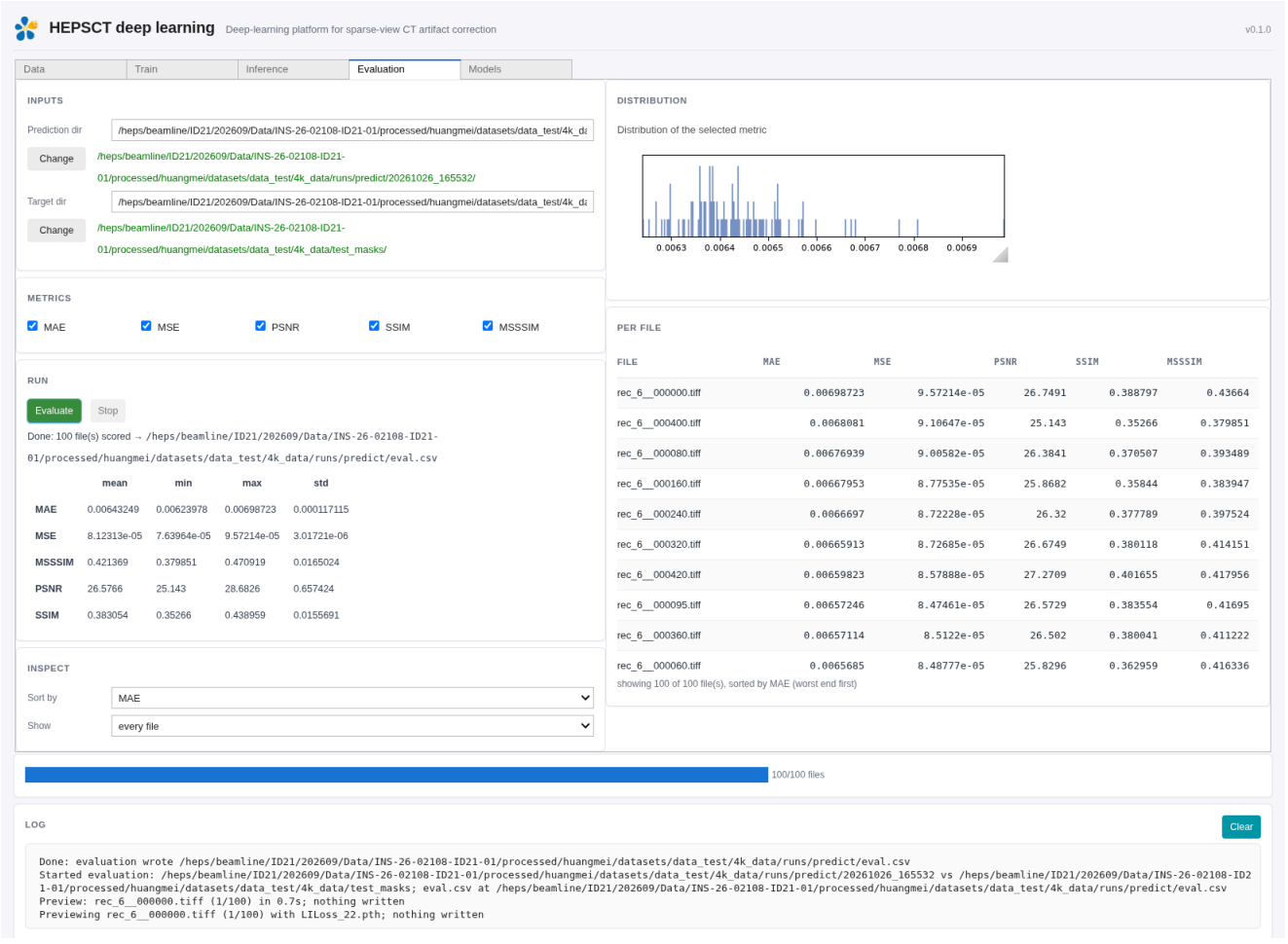
4.5 Result 卡片

窗格	内容	说明
input	原始图像	大图；灰度窗口、cmap、norm、Auto/Lock 的控件只出现在这里
prediction	网络输出	缩略图，**跟随 input 的对比度设置**，便于直接比较
prediction − input (own range)	预测减输入	缩略图，使用**自己的自动范围**。校正量通常很小，如果强行套用 input 的窗口会变成一片灰，所以单独定标

和 Data 预览一样，任意窗格的 ↗ 可以把它换到大图位置，方便全尺寸检查；悬停可读像素值。一次推理结束后，Result 会显示**最后写出**的那张图；单张 Preview 则显示所预览的那张。

---

5 Evaluation 标签页



评估把预测目录与目标目录**按文件名一一配对**，逐文件计算所选指标，并给出汇总、分布和最差/最好的若干文件。

5.1 Inputs

控件	作用	说明
Prediction dir	预测结果目录	这个目录里的每个 TIFF 都会被评分
Target dir	目标（真值）目录	必须为每个预测文件提供**同名**文件。只要有一个预测找不到同名目标，整次评估在开始前就报错，并给出第一个缺失的文件名——不会静默地少评几个文件

目标目录里多出来的文件不会参与评分（评分以预测目录的文件列表为准）。

5.2 Metrics

控件	作用	说明
指标复选框	选择要计算的指标	默认 MAE、MSE、PSNR、SSIM、MSSSIM。列表由模型插件声明，应用只提供自己能真正计算的指标；插件声明了但应用不支持的指标不会被列出，而是在下方以说明文字提示
已勾选项	刷新注册表后保留	重新加载模型或刷新时，已经勾选的指标不会丢

指标方向：

指标	越大越好?
MAE、MSE	否，越小越好
PSNR、SSIM、MSSSIM	是，越大越好

5.3 Run

控件	作用	说明
Evaluate	开始评分	按钮变红；进度条显示 `done/total files`。所有文件评分完成后变成绿色
Stop	请求停止	在**当前正在评分的文件**结束后生效；已经算出的文件保留，汇总基于这些文件。若一张都没算完就停止，不会报错，状态如实说明
状态行	结果	`Done: 100 file(s) scored → …/eval.csv`；没有可写位置时会给出额外警告

结果文件 `eval.csv` 的写入位置：

1. 优先写到**预测目录的上一级目录**（截图中的 `runs/predict/eval.csv` 就是这种情况）；
2. 上一级不可写时，写到 `~/hepsct_dl_output/<上一级目录名>/eval.csv`；
3. 两处都不可写时，仍然完成计算并在界面显示数字，但日志会警告没有写出 CSV。

CSV 的列是 `name` 加上各个已选指标，一行一个文件。

5.4 Inspect

控件	作用	说明
Sort by	选择排序和绘图使用的指标	下拉框列出本次计算的所有指标
Show	显示多少行	`every file`、`worst 10`、`worst 50`、`best 10`。`worst` 对 MAE/MSE 指误差最大，对 PSNR/SSIM/MSSSIM 指分数最低；`best` 反之

表格脚注会写清楚当前显示的数量、总数、排序指标和方向，例如 `showing 100 of 100 file(s), sorted by MAE (worst end first)`。

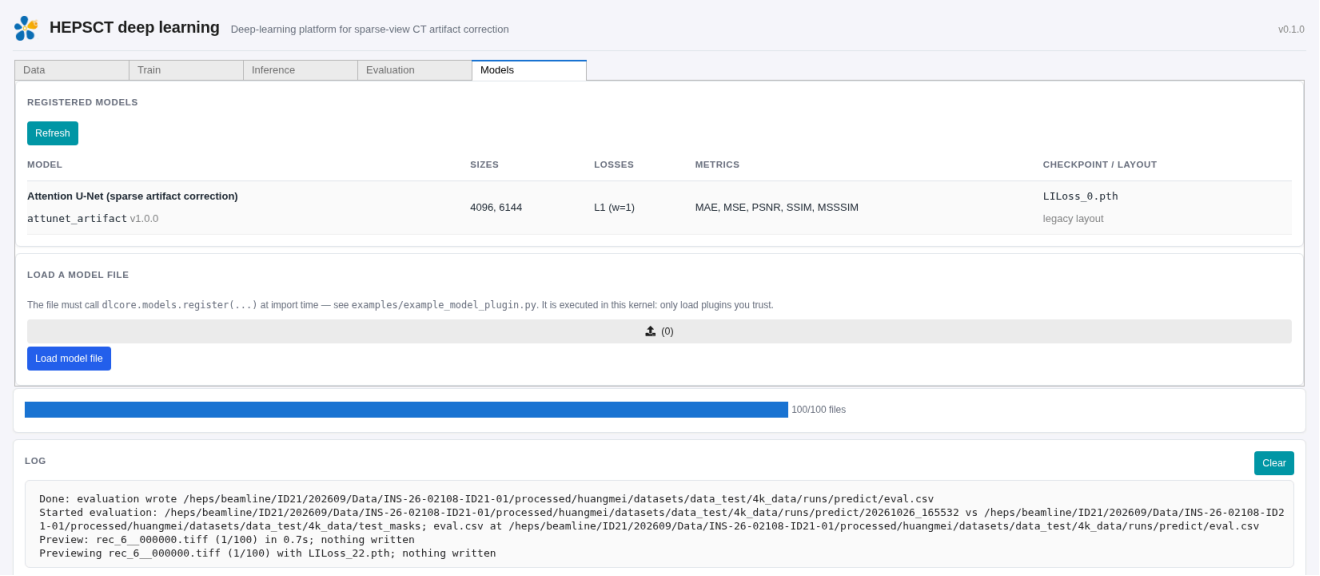
5.5 Distribution 与 Per file

区域	内容
Distribution	所选指标在**全部**已评分文件上的直方图（不受 Show 的限制，切换指标时重绘）；用于判断是否存在少数异常差的文件
Per file	逐文件指标表，列依次为文件名、MAE、MSE、PSNR、SSIM、MSSSIM。PSNR 为无穷大时显示 `∞`（预测与目标完全一致）。表格可滚动；切换 Sort by / Show 会立即重排

上方的汇总表给出每个指标的 `mean`、`min`、`max`、`std`。标准差只在有限值上计算（PSNR 出现 `∞` 时不会把统计量变成 `nan`）。

---

6 Models 标签页



6.1 Registered models

控件	作用	说明
Refresh	重新读取插件注册表	模型插件在当前内核中注册；加载新插件后如果表格没更新，点这里
表格 `model`	模型名称与版本	第一行是显示名，第二行是注册键与版本，例如 `attunet_artifact v1.0.0`
`sizes`	支持的图像尺寸	插件能为哪些尺寸生成 tiling 计划（当前内置插件支持 4096、6144）
`losses`	训练损失	例如 `L1 (w=1)`
`metrics`	插件声明的指标	会出现在 Evaluation 标签页的指标复选框里
`checkpoint / layout`	checkpoint 命名与布局	例如 `LILoss_0.pth` 与 `legacy layout`，说明该插件的权重如何命名、从哪种目录结构读取

6.2 Load a model file

控件	作用	说明
文件选择框	选择一个 `.py` 文件	只接受单个 `.py`
Load model file	在当前内核中执行该文件	文件必须在**导入时**调用 `dlcore.models.register(...)`；注册成功后，模型立即出现在 Train 和 Inference 的 Model 下拉框中。参考 `examples/example_model_plugin.py`
状态行	加载结果	成功显示注册的模型名；失败显示红色错误

> 安全提示：插件文件会在当前内核里直接执行，等同于运行其中的任意 Python 代码。只加载你信任的文件。

插件是当前页面会话级别的：页面刷新/内核重启后，需要重新加载（随应用自带的插件会自动注册）。

---

7 典型工作流

下面是一条从零到评估的完整路径，括号里是每一步要看的地方。

1. **确认数据（Data）**
    - 在 Data root 填入数据集根目录，按 Enter；
    - 确认 Dataset 摘要的尺寸和 tiling 计划正确，表格各 split 为 `ok`；
    - 若数据根只读，确认 Output root 指向你自己的目录；
    - 用 Split / 滑块浏览几张，确认 input、label 和差值符合预期，Patch grid 覆盖完整。
2. **试跑训练（Train）**
    - 选择 Model，填写 Run name；

- 先把 Max images 设为 4、Epochs 设为 2–5，Log every N 设为 1，确认流程能跑通；
- 正式训练时把 Max images 改回 0、Epochs 设为需要的轮数；
- 观察 Loss 曲线：训练和验证是否一起下降、有没有发散；需要看后段细节时勾选对数坐标；
- 训练结束后记下状态行里的 `best epoch` 和 `val`，以及运行目录。

3. **单张检查（Inference）**
   - Model 选训练时用的模型；Weights 选 `best.pth`（或状态行给出的最佳 `LILoss_<epoch>.pth`）；
   - Input dir 默认是 `test_images`；
   - 点 Preview，拖动 image # 看几张，确认 prediction 和差值合理；
   - 这一步不写文件，可以放心反复看。
4. **整目录推理（Inference）**
   - 确认 Output dir（默认是带时间戳的新目录）；
   - 第一次运行保持 Overwrite 关闭；如果目录里已有结果，会跳过而不是覆盖；
   - Run inference，观察进度条和 ETA；需要中止时点 Stop。
5. **评估（Evaluation）**
   - Prediction dir 指向刚写出的目录；
   - Target dir 指向 `test_masks`；
   - 勾选需要的指标，Evaluate；
   - 用 Sort by + Show 看 `worst 10`，再结合 Distribution 判断误差是普遍偏高还是个别文件异常；
   - 需要留档时使用 `eval.csv`。
6. **换模型或加模型（Models）**
   - 内置模型不需要操作；
   - 自定义模型：在 Models 标签页选择一个 `.py`（导入时调用 `dlcore.models.register(...)`），点 Load model file，再回到 Train 选择它。

---

8 常见问题

Q：数据根目录只读，训练/推理会不会失败?
不会，只要你设置了可写的 Output root。默认会落到 `~/hepsct_dl_output/<数据根目录名>/`。Dataset 卡片的警告在输出根可写时会显示实际位置。

Q：我在 Data root 里输入了路径，但下面的信息没变。
输入框是提交式更新：按 **Enter** 或点击别处后才探测。只敲字符不会触发扫描。

Q：路径行显示 `No selection`，但输入框里明明有路径。
输入框是权威值。选择器只是辅助填写的工具；它无法列出尚未创建的输出目录时，路径行可能无法高亮，但应用仍会按输入框的路径写入（第一次写时创建目录）。

Q：训练时进度条为什么总是从头开始?
进度条显示的是**当前 epoch 的当前阶段**（train 或 val），每进入一个新的 epoch/阶段就重置。整体进度看右侧文字里的 `epoch x/y`。

Q：点 Stop 之后为什么没有立刻停?
Stop 是协作式的：它在当前正在处理的图像（训练）、当前正在写出的图像（推理）或当前正在评分的文件（评估）结束后生效，这样不会留下半个文件或一段无法续训的状态。

Q：推理显示 `Nothing written`。
输出目录里已经存在全部同名文件，而 Overwrite 没勾选。勾选「Overwrite existing outputs」或换一个输出目录。应用特意把这种情况标成橙色，避免把「全部跳过」当成成功。

Q：推理第一张就报 `degenerate prediction`。
预测几乎是常数，通常是权重与数据不匹配（模型选错、checkpoint 不对、归一化不同）。先确认 Model/Weights 的对应关系；如果这确实是预期输出（例如输入本身接近空白），再勾选「Allow degenerate outputs」。

Q：评估报 `prediction(s) have no target`。
预测目录里存在目标目录没有的同名文件。评估严格按文件名配对；请确认预测文件和目标文件来自同一个测试集、命名一致。目标目录里多出的文件不影响。

Q：评估没有生成 `eval.csv`。
预测目录的上一级不可写，且 home 下的回退目录也不可写时会这样。数字仍会显示，日志会说明。把预测放到自己的目录下再评估即可。

Q：权重列表里的 `[xxx]` 前缀是什么?
表示这个 checkpoint 来自**非数据根**的目录（通常是输出根）。例如数据根只读、产物写在 `~/hepsct_dl_output/doc_demo` 时，列表会显示 `[doc_demo] runs/...`。不同根下的同名文件不会被去重。

Q：为什么 `val` 列是 `–`?
该运行没有验证集（没有 `val_images/val_masks`），所以没有测量验证 loss。最佳模型改为跟随训练 loss，这不是数值错误。

Q：图像尺寸既不是 4096 也不是 6144。
当前 tiling 表只覆盖这两种尺寸，不能训练/推理。Dataset 卡片会给出警告；要支持新尺寸需要给插件增加 tiling 计划。

Q：日志面板只显示最近的几百行，之前的去哪了?
面板是实时 tail（最多 400 行，超出丢弃最旧的四分之一）。完整历史在运行目录的 `train.log`（以及 `stdout.log` / `stderr.log`）。

Q：调整了预览的灰度，为什么 label 也跟着变，而差值窗格没变?
label 与 input 共用同一个灰度窗口，这样两张图可以直接比较；差值（label − input、prediction − input）使用自己的自动范围，因为校正量的数值范围和原图完全不同，套用原图窗口会把它压成一片灰。

Q：训练/推理/评估期间可以关闭或刷新页面吗?
不建议。voila 为每个页面启动一个内核，刷新会重新加载应用，界面上的进度、状态和当前内核里的自定义插件都会丢失。长任务请保持页面打开；需要确认进度时也可以直接查看运行目录的日志文件。

Q：标题栏版本号后面的 `+xxxxxxx` 是什么?
部署构建的 commit 短哈希。热更新部署后，新打开的页面会显示新构建；已经打开的页面仍运行旧构建，需要刷新。