ML 대회 실험 운영 가이드

DAT RESEARCH

Astra와 Fable 협업에서 얻은 재사용 가능한 교훈 | 2026-09-13

목표는 시행착오를 없앤다고 약속하는 것이 아니라, 같은 실수를 반복하지 않고 적은 계산으로 검증 가능한 지식을 쌓는 것이다. 아래 항목은 다음 대회 시작 시 복사해 사용한다. 대회 규정과 데이터 형태가 바뀌면 다시 판단한다.

1. 시작 전에 고정할 1쪽 계약

- □ 목표 지표, 공식 평가 단위, 종료 시각, 제출 기회와 제출 주체를 기록한다.
- □ 팀 합류, 외부 데이터, 사전학습, 인터넷, 코드 및 가중치 라이선스, 최종 공개 조건을 공식 문서에서 확인한다.
- □ 원자료, 개인별 출력, 공개 가능한 집계의 경계를 구분한다. AI 도구 출력도 외부 전송이라는 점을 포함한다.
- □ 학습 프레임워크, Python 버전, uv.lock, GPU/CPU/디스크 예산과 실행 책임자를 기록한다.
- □ 제출 및 holdout 조회는 명시된 권한과 예산으로만 수행한다.
- □ 최근 규정 변경과 포럼 답변의 조회 시각, URL, 적용 결론을 보존한다.

코드가 MIT라고 weight나 pretraining dataset의 사용권까지 확인된 것은 아니다. 저장소 라이선스, 정확한 weight revision/hash, weight 약관, 데이터 권리, 재배포 의무를 각각 검토한다. 모델 이름만 보고 적격 판정을 내리지 않는다.

2. 데이터와 VALIDATION을 모델보다 먼저 설계

- □ 관측 단위와 독립 단위를 구분한다. 같은 환자, 장비, 기관, 세션, 시간 계열이 양쪽 fold에 섞이는지 확인한다.
- □ 배포 상황에 맞춰 GroupKFold, stratified group split, 시간 순서, leave-site-out 등 후보를 선택한다.
- □ 분할 후보는 최저 점수를 만드는 분할이 아니라 실제 배포 오차를 추정하는 분할인지 비교한다.
- □ training partition만으로 전처리 통계, 특징 선택, 정합 template, calibration을 fit한다.
- □ train/validation/test를 합쳐 정규화하거나 unlabeled test에 적응하지 않는다. 허용 여부와 선택 편향을 별도로 검토한다.
- □ 중복, 결측, 방향/좌표, 단위, intensity, label 정의, class imbalance, site/time 차이를 로컬 집계로 점검한다.
- □ 봉인 holdout은 탐색과 분리하고 조회 일지를 남긴다.

동일 split에서 많은 후보를 탐색하면 validation에 선택 편향이 누적된다. nested CV가 가능한지 비용을 검토하고, 어려우면 선별 결과를 개발 추정치로 명시한 뒤 고정 recipe의 독립 확인 절차를 계획한다. CV를 여러 번 했다는 사실이 선택 편향을 없애지는 않는다.

3. 모델 선택은 작은 데이터=전통 모델로 끝내지 않는다

유효 표본 수, 데이터 형태, 사전학습의 관련성, label 품질, 입력의 물리 구조, 모델 용량과 학습 예산을 함께 판단한다. 작은 의료영상에서는 3D CNN의 지역성 편향이 도움이 될 수 있지만, 작은 표 데이터와 작은 영상 데이터는 다른 문제다.

Grinsztajn et al., NeurIPS 2022은 중간 크기 표 데이터에서 tree 계열의 강점을 분석했다. 반면 TabPFN, Nature 2025는 작은 표 데이터에서도 foundation 모델이 강할 수 있음을 보였다. 이 논문은 의료영상에 TabPFN을 직접 쓰라는 근거가 아니다.

ViT, ICLR 2021은 큰 사전학습에서 작은 downstream task로 전이할 가능성을 보였다. DeiT, ICML 2021은 학습 recipe와 distillation의 데이터 효율을 보여 준다. 최신 구조와 강한 기준 모델을 모두 검토해야 한다.

갈래	빠른 기준	최신 후보를 고려할 방식
표 데이터	정규화 선형 모델, GBDT	tabular Transformer/foundation, 라이선스와 데이터 제약 확인
2D 이미지	작은 CNN 또는 적격 pretrained CNN	ViT/ConvNeXt/자기지도 backbone의 frozen 및 fine-tune 대조
3D/의료영상	작은 3D CNN, 물리적으로 타당한 특징	3D Transformer, 적격 2D/3D 사전학습, local-global 표현
시계열	선형/통계 모델, 작은 convolution	시간 누수 없는 Transformer와 관련 pretrained 모델
multimodal	각 modality 단독	단독 대비 결합 이득과 결측 modality 처리 대조

- □ 시작 단계에 최신 논문, 저자 코드, 상위 입상자의 공개 solution을 함께 조사한다.
- □ 최신 후보 2~3개와 강한 기준 모델을 우선순위표로 고른다. 이 숫자는 시작용 예시이며 대회 크기에 맞춘다.
- □ from-scratch, frozen feature, partial fine-tuning, full fine-tuning을 구분한다.
- □ 라이선스 적격성과 modality 적합성을 통과한 후보만 다운로드한다.
- □ 실패한 설정을 해당 모델 계열 전체의 실패로 일반화하지 않는다.

4. 실험 트리: 넓게 선별하고 유망한 원인을 깊게 확인

1. 기준 모델의 end-to-end 학습과 공식 입력/출력 경로를 먼저 동작시킨다.
2. 입력, encoder, pooling, head, 목적함수, optimizer를 별도 가설 축으로 나눈다.
3. 각 노드에 부모 recipe, 바꿀 변수, 기대 방향, 비용, 중단 기준을 기록한다.
4. 선별 단계는 비교 가능한 pilot을 사용하되 유망 후보의 전체 validation을 계획한다.
5. 채택 실패 이유를 남긴다. 새로운 근거가 없으면 같은 실험을 반복하지 않는다.
6. 블록 구조가 좁혀지면 제한된 HPO를 수행하고, 최종 단계에서만 작은 앙상블의 비용과 이득을 검토한다.

구조 탐색 중 같은 모델을 수십 seed로 평균하면 구조 자체가 좋아졌는지 알기 어렵고 GPU 시간을 소모한다. fold validation은 하나의 recipe를 평가하는 절차이며 seed bag과 다르다. 앙상블이 필요하면 각 단독 성능, OOF 잔차 보완성, inference 비용을 함께 확인한다.

5. 통제실험 카드 템플릿

필드	작성 내용
질문	무엇을 배우려는가
부모	기준 recipe/source/split 지문
변경	한 변수 또는 명시적 요인 조합
고정	seed, input, step, optimizer, batch, 평가 및 checkpoint 규칙
1차 지표	호환되는 validation log loss 등
보조 지표	fold별 차이, 시간, 메모리, train 곡선, calibration
채택 기준	사전에 정한 실용 이득과 비용, 실패 시 다음 행동
결과	모든 완료 fold, 누락/실패/조기 종료, 선택 노출
해석	관측 사실 / 가능한 설명 / 미검증 원인
다음	기각, 확인, 수정, 보류 중 하나

같은 seed라도 연산 그래프나 dropout 호출 수가 바뀌면 RNG 흐름이 달라질 수 있다. 엄격한 대조가 필요하면 모델 초기화, 데이터 shuffle, augmentation의 RNG stream을 분리하고 합성 fixture에서 일치를 확인한다.

6. 최신 탐색 알고리즘도 비용과 가정을 검토

□ 작은 search space에 강한 prior를 주고 random/TPE 등 단순 탐색을 비교 기준으로 둔다.

□ multi-fidelity pruning 전에 짧은 epoch 순위가 충분한 학습 후 순위와 관련 있는지 확인한다.

- □ 느리게 학습하는 pretrained fine-tune나 깊은 모델을 초반 loss만으로 모두 제거하지 않는다.
- □ 탐색 대상과 고정 요인을 구분하고 architecture/lr/augmentation/seed를 동시에 바꾼 결과를 한 요인의 효과라고 하지 않는다.
- □ trial DB와 recipe, source, 데이터 분할 지문을 저장하고 중단 후 중복 학습을 방지한다.
- □ 동일 validation에 tuning된 ensemble/calibration은 다시 그 점수만으로 채택하지 않는다.

PriorBand, ICLR 2024는 전문가 prior와 multi-fidelity 탐색의 결합을 다룬다. 후속 방법을 조사할 때는 실제 성능뿐 아니라 wall time, 순위 보존, 예산, 병렬 환경을 비교한다. 특정 HPO 라이브러리를 설치한 것만으로 효율적인 탐색을 구현한 것은 아니다.

7. 학습 안정성과 성능 저하를 구분

- □ optimizer, learning rate, warm-up, schedule, batch/accumulation, weight decay를 기록한다.
- □ dropout, label smoothing, mixup, consistency를 한꺼번에 추가하지 않는다.
- □ AMP, gradient clipping, compiled graph가 수치/재현성에 영향을 주는지 점검한다.
- □ early stopping의 monitor와 patience, best/last/EMA 선택을 구분한다.
- □ train과 validation 곡선에서 적합 부족, 과적합, 수치 실패를 분리한다.
- □ normalization은 수식과 위치를 구분한다. head LN 제거와 Transformer Pre-LN은 다른 실험이다.
- □ SSL/준지도 학습은 합법적 unlabeled pool, validation 제외 정책, teacher 생성 partition과 pseudo-label 누수를 먼저 명시한다.

Warm-up이 항상 필요하거나 항상 불필요한 것은 아니다. dropout=0.2가 크다는 인상도 실험으로 확인한다. EMA는 장식이 아니며 decay와 총 update 수의 관계를 검토해야 한다.

8. 자원 운영: 학습 중단을 실험 비용으로 계산

- □ 실제 GPU 모델의 공식 thermal/power 기준과 telemetry를 확인한다.
- □ CPU 데이터 처리, GPU 학습, 디스크 I/O 시간을 따로 측정한다.
- □ worker 수와 CPU thread를 늘릴 때 throughput 및 메모리, CPU oversubscription을 함께 본다.
- □ GPU 병렬 작업은 VRAM뿐 아니라 전력, 온도, 처리량과 실험 간 간섭을 확인한 후 허용한다.
- □ 진행 중 recipe의 소스를 덮어쓰지 않는다. 새 버전 route를 만들고 안전한 job 할당 경계에서 적용한다.
- □ 재개 가능한 checkpoint, optimizer/RNG 상태, 고정 source hash와 retry 상한을 둔다.
- □ STOP/host fault를 원인 확인 없이 지우지 않는다. healthy training을 중단해 복구 시험하지 않는다.
- □ 호스트 업데이트, 재부팅, 종료는 연구 자동화가 임의로 수행하지 않게 한다.

- □ 디스크 저수위 경고와 추정 증가량을 표시한다. 불필요한 공개 pretrained cache는 provenance/의존성을 확인한 뒤 정리한다.
- □ best, 재개 중, 검증된 제출 및 재현에 필요한 weight는 보존한다. 만료 데이터의 삭제는 대회별 체크리스트로 관리한다.

재부팅 원인을 로그 없이 특정 모델이나 명령 탓으로 확정하지 않는다. 원인 조사와 재발 방지 정책은 별개로 기록한다.

9. 감독 모델과 구현 모델의 역할

- □ 감독자 1명이 연구 질문, 근거, 채택을 책임진다. 현재 소유자를 역할 파일에 명시한다.
- □ 구현자는 입력/출력, 수정 파일, 금지 범위, 합성 검증 기준이 정해진 작은 작업을 맡는다.
- □ 실제 dispatch, 반환, 검수, runtime 적용을 기록한다. 메모만으로 위임 완료라고 하지 않는다.
- □ 문맥에는 소스와 허용된 집계만 전달한다. 환자/개인 데이터는 로컬 계산 경계를 유지한다.
- □ 도구 접근을 차단할 수 없다면 lower model은 도구 없이 source proposal만 반환한다.
- □ 사용량 소진, 인증 문제, 정확한 모델 미지원, 네트워크 오류를 구분한다.
- □ supervisor 인계는 미완료 파일, 활성 PID/경계 상태, pending 가설과 다음 판정까지 포함한다.
- □ 모델의 답변은 새 근거와 검수 대상이다. 실행 완료나 자동 코드 적용 권한으로 취급하지 않는다.

AGENTS.md는 짧은 경계와 관련 문서 경로 중심으로 유지한다. 모든 수정마다 전체 역사나 긴 지침을 읽게 하지 않는다. 작은 UI 수정에 전체 ML 테스트를 반복하지 않고 영향 있는 계약 검사를 선택한다.

10. 모니터는 사람이 오해하지 않게 설계

- □ 현재 가설과 실제 job, fold, 학습 단계, update 시각을 함께 표시한다.
- □ epoch ETA와 전체 실험 ETA를 구분한다. 시간 범위는 조기 종료/냉각에 따라 달라짐을 표시한다.
- □ heartbeat와 iteration progress를 분리해 긴 epoch를 멈춤으로 오인하지 않게 한다.
- □ 완료 결과를 평가 범위별로 묶고 숫자형 log loss 오름차순으로 정렬한다.
- □ 진행 중 결과는 색과 텍스트로 구분하고 완료된 전체 CV와 섞지 않는다.
- □ 모델별 입력, block, pooling, head, loss를 그림으로 표시한다.
- □ 결과를 원장, 연구 메모, 다음 행동과 연결한다. 공개 결과는 allowlist로만 내보낸다.
- □ 외부 공개 site에 로컬 데이터 서비스나 원격접속 credential을 연결하지 않는다.

11. 제출 실패를 줄이는 체크리스트

- □ 정확한 archive hash를 고정하고 루트 entrypoint와 asset 경로를 확인한다.
- □ 공식 container와 CPU/GPU별 허용 환경에서 입력부터 결과 파일까지 확인한다.
- □ 경로 계약, 출력 filename/schema/order, 확률 범위, 누락 행과 중복을 검증한다.
- □ 검증에 사용한 package와 실제 올릴 package가 동일한지 재확인한다.
- □ 플랫폼 시작 log의 빈 directory listing만으로 실제 데이터 전달 방식을 추정하지 않는다.
- □ 오류를 상수 확률로 덮으면 성공처럼 보이는 나쁜 제출이 될 수 있다. 허용된 fallback과 실패 수 집계, 전체 입력 불가 시 명확한 실패 정책을 구분한다.
- □ 재빌드는 deterministic archive 또는 content manifest를 사용한다.
- □ 제출 결과, CV/holdout과의 차이, 남은 횟수와 실제 error 원인을 남긴다.
- □ CV→LB gap 몇 건으로 보편적인 점수 보정치나 달성 한계를 만들지 않는다.

12. 종료와 다음 대회로 가져갈 것

- □ 최종 코드/uv.lock/source/hash/모델 provenance와 추론 계약을 보존한다.
- □ 실패 가설, 실패 원인에 대한 불확실성, 유효했던 대조와 낭비한 계산을 기록한다.
- □ 개인/환자 자료 없는 재사용 템플릿과 synthetic tests만 별도 번들로 내보낸다.
- □ 모델의 이름보다 정보 보존, 평가 설계, 계산 비용, 데이터 권리를 다음 프로젝트에 전달한다.
- □ 다음 대회의 시작 계약은 새 규정으로 다시 작성한다.

이 대회에서 일반화할 교훈은“CNN을 쓰라”가 아니다. 작은 데이터에서도 최신 모델을 충분히 검토하되, 비교할 기준과 변경한 요인, 실제 비용, 실패를 남기는 방식이 먼저라는 것이다.