

HF-Bench • DL vs. classical ML

Anthony Nguyen

1 • PROBLEM

what and why

Heart failure (HF) is one of the leading drivers of preventable hospital readmission in the United States, with roughly 20–25% of patients returning within 30 days of discharge and associated costs exceeding \$26 billion annually. Accurate risk stratification at the point of discharge could enable targeted interventions — closer follow-up, medication adjustment, home health referrals — that reduce readmissions and improve patient outcomes. A substantial body of literature applies machine learning to this problem, with most published models relying on classical approaches: logistic regression, random forest, or gradient-boosted trees (XGBoost/LightGBM). Deep learning methods — particularly attention-based architectures (TabTransformer, FT-Transformer) and purpose-built tabular DL models (TabNet) — have shown promise on general tabular benchmarks but remain underexplored in the clinical HF readmission setting. Critically, most existing comparisons are conducted on single-institution datasets, rarely report subgroup performance across demographic groups, and do not systematically evaluate the interpretability-performance tradeoff that clinicians actually care about. This project fills that gap: we conduct the first systematic head-to-head benchmark of classical ML versus deep learning for 30-day HF readmission prediction on MIMIC-IV, with explicit evaluation of fairness across demographic subgroups and clinical interpretability via SHAP attribution.

2 • APPROACH

model / method, and why this one

We benchmark five model families on an identical MIMIC-IV HF cohort, held to a strict train/validation/test split (temporal, to simulate real deployment): Model Family Hardware Why included Logistic Regression Classical CPU Clinical standard; maximally interpretable baseline XGBoost Classical (GBDT) CPU Current SOTA on tabular EHR; sets the bar DL must beat LightGBM Classical (GBDT) CPU Faster GBDT alternative; cross-checks XGBoost results TabNet Deep Learning MI300X Attention-based tabular DL with built-in feature selection FT-Transformer Deep Learning MI300X Feature Tokenizer + Transformer; SOTA on tabular DL benchmarks All models are trained on identical feature sets. Hyperparameters are tuned via Optuna (100 trials per model) on the validation set. Feature importance is assessed for all models using SHAP — TreeExplainer for classical models, GradientExplainer for DL models — enabling a direct interpretability comparison across paradigms. Subgroup analyses by race/ethnicity, sex, and age group (≥ 65 vs < 65) are conducted for all five models. This surfaces not only which model performs best overall, but which model performs most equitably across patient populations — a contribution largely absent from prior benchmarks. The core research questions are: (1) Does deep learning outperform classical ML on this task? (2) If so, at what cost to interpretability and fairness? (3) Which model a clinician should actually deploy?

3 • DATASET

source, size, preprocessing

Target compute: 1x AMD Instinct MI300X (deep learning arms); CPU-only for classical ML arms. Est. total wall-clock: 2–3 days. Target journal: npj Digital Medicine / PLOS ONE / JAMIA Open All data comes from MIMIC-IV v2.2, a publicly available de-identified EHR dataset from Beth Israel Deaconess Medical Center, accessed via PhysioNet credentialed access (no DUA beyond PhysioNet credentialing, typically approved in 1–7 days). Cohort definition: • Adults (≥ 18 years) with primary ICD-10 diagnosis of heart failure (I50.x) • Discharged alive following an inpatient hospitalization • At least one prior admission in MIMIC-IV (required for historical feature computation) • Estimated

cohort: ~15,000–25,000 admissions • Outcome (label): readmitted within 30 days of discharge — binary (1 = readmitted, 0 = not) MIMIC-IV module Features extracted admissions / patients Age, sex, race/ethnicity, insurance type, admission source, length of stay diagnoses_icd Comorbidities: diabetes, CKD, COPD, AFib, hypertension (one- hot); Charlson Comorbidity Index labevents BNP/NT-proBNP, creatinine, sodium, hemoglobin, eGFR at discharge + missingness flags prescriptions Loop diuretic dose, ACE inhibitor/ARB/ARNI, beta-blocker, total medication count chartevents Discharge weight, systolic BP, heart rate, O2 saturation Engineered features Admissions in prior 6 months, BNP delta (admission→discharge), days since last admit Preprocessing: median imputation for missing labs + binary missingness indicators, one-hot encoding for categoricals, StandardScaler normalization (fit on train only, applied to val/test), SMOTE oversampling on training fold only. Temporal train/val/test split: 70% / 15% / 15% by admission date.

4 • METRICS

how you'll know it worked

Experiment Metric Primary benchmark (all 5 models) AUROC, AUPRC — primary performance metrics; target AUROC ≥ 0.75 Clinical operating point Sensitivity + specificity at 80% specificity threshold; PPV, NPV Calibration Brier score + calibration plot (reliability diagram) for all models Fairness / subgroup AUROC stratified by race/ethnicity, sex, age ≥ 65 vs < 65 — for all 5 models Interpretability Mean |SHAP| feature ranking; top-10 feature table; SHAP summary plot per model DL vs classical gap Delta AUROC (FT-Transformer – XGBoost); paired bootstrap significance test Ablation All models trained without engineered features vs full feature set; delta AUROC Seeds: 42, 123, 7, 0, 256. All metrics reported as mean $\pm$ 95% CI across 5 seeds. Hyperparameter tuning performed on validation set only; final numbers reported on held-out temporal test set.

5 • COMPUTE

training time, memory, dependencies

Classical ML arms (logistic regression, XGBoost, LightGBM) run entirely on CPU and require no special hardware. Deep learning arms (TabNet, FT-Transformer) use the MI300X for training and hyperparameter search. Phase Hardware Estimated wall-clock MIMIC-IV data pull + preprocessing CPU ~2–4 hours (one-time) (mimic-code SQL + pandas) Classical ML training + Optuna tuning CPU ~1–2 hours total (100 trials each) TabNet training + Optuna tuning (100 1x MI300X ~4–6 hours trials, 5 seeds) FT-Transformer training + Optuna tuning 1x MI300X ~6–8 hours (100 trials, 5 seeds) SHAP attribution (all 5 models, test set) CPU / MI300X ~1–2 hours Subgroup analysis + figure generation CPU ~1 hour Total 1x MI300X node ~1.5–2 days wall-clock with buffer GPU memory: TabNet and FT-Transformer on a ~20,000-patient tabular dataset fit comfortably within 16 GB VRAM. The MI300X (192 GB HBM3) is far above what is strictly required, but allows large Optuna search spaces and full-batch training without minibatch approximation artifacts. Storage: ~5–10 GB for relevant MIMIC-IV tables + preprocessed .parquet cache. Well under any reasonable budget. Software / dependencies (all pip-installable): • Python 3.10+, pandas, numpy, scipy, scikit-learn, xgboost, lightgbm • pytorch-tabnet, pytorch-frame (FT-Transformer implementation) • shap, optuna, imbalanced-learn, matplotlib, seaborn • ROCm 6.x + official PyTorch ROCm wheel (for MI300X) • mimic-code SQL scripts for cohort extraction (MIT-licensed) • No Docker required; standard Python venv. No CUDA-only kernels.

6 • DELIVERABLES

model, paper, demo

By end of the two-week window: • Reproducible cohort extraction pipeline (SQL + Python) from MIMIC-IV • Trained and tuned models for all 5 model families with full cross-validation results • Head-to-head benchmark table: AUROC, AUPRC, Brier score, sensitivity, specificity for all models • Subgroup fairness table: AUROC by race/ethnicity, sex, and age group for all 5 models • SHAP summary plots and top-10 feature tables for all models (classical TreeExplainer + DL GradientExplainer) • ROC and precision-recall curves overlaid for all 5 models • Draft methods + results sections formatted for npj Digital Medicine Longer-term (post-sprint): • Full manuscript submitted to npj Digital Medicine, PLOS

ONE, or JAMIA Open • Abstract submitted to AMIA Annual Symposium or AHA Scientific Sessions • Public GitHub repo with fully reproducible pipeline, MIT-licensed I am intentionally not promising that deep learning beats XGBoost — a negative result (classical ML matches or outperforms DL on this task) is equally publishable and arguably more clinically useful. The goal is a clean, honest, reproducible benchmark with equity findings.

7 • RISK

what breaks first, and plan B

what breaks first, and plan B Risk Likelihood Mitigation MIMIC-IV PhysioNet Medium Apply immediately (typically 1–7 days). Use credentialing delayed MIMIC-IV demo (100-patient public subset, no credentialing) to build and validate full pipeline in parallel. No time lost. DL models underperform Medium This is a valid, publishable finding — ‘classical ML XGBoost on this task matches DL on tabular EHR data with better interpretability’ is a useful clinical result. Frame as the honest benchmark it is. XGBoost arm is the safety net. FT-Transformer or TabNet Medium Isolate DL training behind unit-tested modules. Fall fails to converge / back to MLP (simpler DL baseline) if attention- hyperparameter sensitivity based models misbehave. ROCm compatibility tested in smoke run before full training. SHAP GradientExplainer is Medium Use SHAP KernelExplainer as fallback (model- slow or unstable on DL agnostic, slower but reliable). Limit explainability models analysis to a random 500-patient subset of the test set if runtime is prohibitive. Subgroup sample sizes too Low Report CIs explicitly. Collapse granular race small for reliable AUROC categories to broader groups if $n < 100$. Flag small- estimates n subgroups rather than omitting them — reviewers accept this with honest reporting. Class imbalance (~20% Low SMOTE on training fold + `class_weight='balanced'` positive) degrades minority- for classical models + `pos_weight` in DL loss. class recall Report AUPRC alongside AUROC. Tune threshold to clinical operating point rather than defaulting to 0.5. I will check in at the 1-week mark with classical ML results and Stage-1 DL training numbers so we can decide together whether to push for the full 5-model benchmark or trim to the strongest 3.