Healthcare-focused SHAP Research Blueprint

Scope: Implement SHAP across tabular and time-series clinical models (UCI Diabetes + MIMIC-IV), evaluate grey areas (causality vs correlation, actionability, time-series interpretability), and produce reproducible experiments, metrics, visualizations, and a publishable write-up.

---

1. High-level objectives

1. Implement a consistent pipeline to train multiple model families (tree-based, classical, and deep sequence models) on clinical tabular and time-series datasets.
2. Compute and compare SHAP explanations (TreeSHAP, KernelSHAP, DeepSHAP, GradientSHAP) across models and tasks.
3. Identify and quantify grey areas: causal misattribution, instability across retrains and distributional shifts, actionability of explanations, and SHAP behavior on time-dependent data.
4. Propose and evaluate mitigation strategies (causal adjustments, windowed SHAP for sequences, aggregated visualizations, stability regularization).
5. Produce reproducible code, evaluation suite, and paper-quality figures.

---

2. Datasets & access

A. UCI Pima Indians Diabetes ("Diabetes")

- **Type:** Tabular, binary classification (diabetes onset). Small and easy to iterate on.
- **Why:** Good quick sanity checks for models vs. SHAP (coefficient vs. SHAP consistency), feature correlation experiments.
- **Preprocessing notes:** handle zeros-as-missing for some clinical measurements; standardize or rank-transform continuous features; encode categorical features (none in original Pima).

B. MIMIC-IV (ICU EHR; time-series + tabular)

- **Type:** Real-world clinical EHR with vitals, labs, medications, notes (if needed). Use for sequence prediction (e.g., in-hospital mortality, length-of-stay risk within first 48h), and static tasks (e.g., readmission risk).
- **Why:** Rich temporal structure, multi-modal features, heavy correlations — ideal to stress SHAP's weaknesses in causality and time.
- **Access:** Requires credentialed access through PhysioNet (CITI and data use agreement). Plan weeks for approvals.
- **Subset strategy:** Start with a cleaned ICU cohort (e.g., single-hospital adult ICU admissions) and focus on a limited prediction target to bound complexity.
- **Preprocessing notes:** define sampling window (first 24/48 hours), create time-binned features (e.g., hourly/4-hour aggregates), imputation strategy (forward-fill, last observed), mask indicators for missingness.

---

3. Models to implement

A. Tree-based (baseline; TreeSHAP available)

- XGBoost (preferred) — tuned with early stopping, max_depth/eta sweeps.
- Random Forest — classical baseline.

B. Linear / Logistic

- Logistic Regression with L1/L2 (sanity checks: SHAP vs coefficients).

C. Deep sequence models (for MIMIC-IV time series)

- LSTM/GRU with attention (predict using last hidden or attention-weighted pooling).
- Temporal Convolutional Network (TCN) or Transformer encoder for clinical sequences.

D. TabNet / Tabular DL

- TabNet or other recent tabular deep models — good middle-ground between tree and deep sequence models.

E. Probabilistic model (optional)

- Bayesian logistic regression or MC-Dropout MLP — to explore SHAP vs uncertainty mismatch.

---

4. SHAP variants & explainability techniques

- **TreeSHAP:** Fast exact (for trees). Use for XGBoost, RF.
- **KernelSHAP:** Model-agnostic, expensive. Use for small feature subsets and sanity checks.
- **DeepSHAP / GradientSHAP:** For neural networks (MLP, LSTM, CNN). Use with embedding-aware inputs for text/ordinal features.
- **Integrated Gradients:** Optional comparator for deep models.
- **Counterfactual explanations / anchors:** For actionability comparisons.

Notes: - For time-series: investigate **windowed SHAP** (compute local SHAP per time-bin or per-event and aggregate), and **feature-time interaction SHAP** (feature at time t as separate feature). - For embeddings (categorical or token inputs): attribute at embedding-dimension level then aggregate back to tokens/features.

---

5. Experiments (detailed)

Experiment group 1 — Sanity & baseline

- Train Logistic, XGBoost, RF, MLP on Diabetes dataset.
- Compute SHAP (TreeSHAP for trees, Kernel/DeepSHAP for others).
- Verify consistency: coefficients ↔ SHAP ranking for linear model; permute features to check SHAP sensitivity.
- Deliverable: table of feature importance ranks by method + correlation matrix.

Experiment group 2 — Stability & retrain variance

- For each model, train N runs with different seeds (N=10–30). Compute SHAP for same test set and measure stability metrics (see Metrics section).
- Vary training data (bootstrap samples) to measure dataset-sensitivity.
- Deliverable: stability plots, variance decomposition.

Experiment group 3 — Correlated features & causal probe

- Create simulated correlated features using Diabetes or MIMIC subset (e.g., create X_corro=0.9*X_primary + noise).
- Train models and compare SHAP attribution between causal feature and correlation proxy.
- Build a simple DAG for a subset of variables; apply back-door adjustments (e.g., conditioning) when computing conditional SHAP approximations.
- Deliverable: demonstration of misattribution & proposed causal-adjusted SHAP heuristic.

Experiment group 4 — Time-series interpretability

- On MIMIC-IV, define task: mortality within 48–72h using first 24h data.
- Train sequence models (LSTM/Transformer) and tree-based models on time-aggregated features.
- Compute **time-resolved SHAP**: treat (feature, time_bin) as separate features; compute SHAP and then aggregate over time to produce heatmaps.
- Evaluate differences between sequence-model SHAP (DeepSHAP/GradientSHAP) and aggregated-tree SHAP.
- Deliverable: time-by-feature heatmaps and disagreement analysis.

Experiment group 5 — Distribution shift & OOD

- Simulate shift: remove a subpopulation or change measurement noise; measure how SHAP explanations change compared to prediction performance.
- Deliverable: sensitivity curves showing explanation drift vs model performance.

Experiment group 6 — Actionability & counterfactual SHAP

- For a few high-impact cases, compute counterfactuals (minimal actionable change leading to predicted outcome flip). Compare which features show high SHAP vs which are actionable per counterfactual.
- Deliverable: case studies showing mismatches and proposing an "actionable importance" score.

Experiment group 7 — Computational profiling

- Measure SHAP runtime/memory across methods and models for increasing feature space (F up to a few hundreds). Report practical limits and heuristics (e.g., feature grouping, sampling budget).
- Deliverable: performance table and recommended heuristics.

---

6. Metrics (quantitative evaluation)

Design or adopt metrics to quantify explanation quality and grey areas:

- **Fidelity**: how well the explanation predicts model output (e.g., leave-one-out or additive approximation error).
- **Infidelity / Sensitivity**: perturbation-based tests measuring change in model output vs explained contribution.
- **Stability / Robustness**: mean pairwise similarity (Spearman rank or Kendall tau) of SHAP vectors across retrains or bootstrap samples.
- **Causal misattribution score**: on simulated DAGs with ground-truth causal effects, measure fraction of importance assigned to non-causal proxies.
- **Actionability alignment**: overlap between top-k SHAP features and top-k actionable features from counterfactual analysis.
- **Computational cost**: runtime and memory as a function of F and model complexity.

---

7. Visualization & reporting

- **Global viewers**: barplots of mean absolute SHAP, summary beeswarm plots, dependence plots.
- **Local viewers**: force plots for individual predictions, cohort explained cases.
- **Time-series**: time × feature heatmaps, animated or small-multiples to show temporal evolution.
- **Stability dashboards**: violin/box plots of importance ranks across seeds.
- **Causality reports**: DAG visualizations with SHAP-weight overlays.

Produce publication-quality figures (vector formats) and interactive notebooks for exploration.

---

8. Reproducibility & code structure

Repo layout (suggested)

```
/README.md
/data/
/notebooks/
/scripts/
  - train_model.py
  - compute_shap.py
  - evaluate_metrics.py
/src/
  - models/
  - data/
  - explanations/
  - metrics/
/experiments/
  - config_ymls/
/docs/
/paper/
```

Key libs: Python 3.10+, scikit-learn, xgboost, lightgbm, pytorch (or tensorflow), pytorch-lightning (optional), shap, captum (for deep attributions), pandas, numpy, matplotlib, seaborn (for quick viz), plotly (interactive), networkx (DAGs).

Notebooks: one exploratory notebook per experiment group; scripts for production runs.

---

9. Compute & resources

- **Local dev:** laptop or small workstation for Diabetes experiments.
- **GPU:** required for deep models on MIMIC-IV (NVIDIA GPU w/ 12+ GB recommended). Use cloud (AWS/GCP/Azure) or institutional cluster for large runs.
- **Storage:** MIMIC subset may require tens to hundreds of GB depending on included data modalities.

---

10. Ethical, privacy & regulatory considerations

- MIMIC data is sensitive. Follow data use agreement, deidentification rules, and institutional review processes.
- Avoid publishing patient-level traces or small cohort counts that risk reidentification.
- Be explicit in the paper about SHAP's limits (correlation not causation) and avoid prescriptive medical claims without clinical validation.

---

11. Timeline & milestones (suggested 6–9 months plan)

Month 0–1: Access MIMIC, set up repo, run Diabetes baseline experiments.

Month 2–3: Train tree & linear baselines on MIMIC aggregated features; implement TreeSHAP & KernelSHAP prototypes; stability experiments.

Month 4: Implement sequence models (LSTM/Transformer), DeepSHAP/GradientSHAP pipelines; time-resolved SHAP analyses.

Month 5: Causal-simulation experiments, counterfactual SHAP case studies, OOD shifts.

Month 6: Computational profiling, finalize metrics, make figures; draft paper.

Month 7–9: Revisions, reproduce key experiments, prepare supplementary materials, submission.

---

12. Potential deliverables

- Reproducible code repository with configs and scripts.
- Notebooks for interactive exploration.
- Benchmark tables for SHAP methods across models and datasets.
- Figures: stability plots, time-resolved SHAP heatmaps, causality misattribution demos.

- A draft manuscript (arXiv / ML conference submission).

---

13. Common pitfalls & mitigations

- **Pitfall:** Long waits for MIMIC access. *Mitigation:* start Diabetes and simulated experiments immediately.
- **Pitfall:** KernelSHAP computational explosion. *Mitigation:* use feature grouping, background dataset sampling, or limiting to local regions.
- **Pitfall:** Misinterpreting SHAP as causal. *Mitigation:* always include DAG-based caveats and counterfactual checks.

---

14. Quick starter checklist (first 2 weeks)

- [ ] Obtain/confirm MIMIC access and download subset (or prepare simulated ICU-like data).
- [ ] Clone repo skeleton and set up environment.
- [ ] Run Diabetes training + TreeSHAP + KernelSHAP sanity checks.
- [ ] Implement SHAP stability measurement script.

---

15. Appendix: example commands / snippets

Train XGBoost (sketch)

```
# train_model.py (sketch)
import xgboost as xgb
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
dtrain = xgb.DMatrix(X_train, label=y_train)
dtest = xgb.DMatrix(X_test, label=y_test)
params = {"objective": "binary:logistic", "max_depth": 6, "eta": 0.05}
model = xgb.train(params, dtrain, num_boost_round=1000, evals=[(dtest,
'eval')], early_stopping_rounds=50)
model.save_model('xgb.model')
```

Compute TreeSHAP (sketch)

```
import shap
import xgboost
model = xgboost.Booster()
model.load_model('xgb.model')
explainer = shap.TreeExplainer(model)
shap_values = explainer.shap_values(X_test)
shap.summary_plot(shap_values, X_test)
```

Compute DeepSHAP (sketch)

```
import shap
import torch
# assume pytorch model
explainer = shap.DeepExplainer(model, background)  # background: sample
tensor
shap_values = explainer.shap_values(X_test_tensor)
```

---

16. Suggested next interactive steps (tell me which you want now)

- I can **generate the repo skeleton and a few starter notebooks** (train + SHAP) for you.
- I can **produce the Diabetes baseline runs** here and show summary SHAP plots (quick experiment).
- I can **draft the methods section** of a paper from this plan.

Tell me which of the above you want me to do immediately and I will start (I will run Diabetes experiments locally in this session if you choose that).