GALACTIC: GLOBAL AND LOCAL COUNTERFACTUALS FOR TIME-SERIES CLASSIFICATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Time-series classifiers inform decisions from clinical diagnosis to industrial monitoring, yet the usual explanations, saliency over timesteps and discriminative subsequences, indicate what supported a prediction rather than what would change it. Counterfactual explanations address this, but existing methods for time-series classifiers describe individual predictions and leave open how recurring temporal changes connect classes across a population. Summarizing many local counterfactuals is the natural route to a population-level answer, and we show why it falls short: an objective asking only for the smallest change to one series drives that change toward the decision boundary, leaving no margin for similar series, and selecting among such changes cannot add a margin the objective never asked for. We introduce GALACTIC, the first unified local-to-global counterfactual framework for differentiable multiclass time-series classifiers. At the local level, GALACTIC-L asks each counterfactual to be as small as possible while crossing the decision boundary by a margin shared across the structural subgroup of its series. At the global level, GALACTIC-G selects rather than generates: it accepts the perturbations of any local generator and keeps a small reusable set for each directed class transition. It expresses coverage, sparsity, and perturbation magnitude in one unit, bits, under a Minimum Description Length criterion, so no weights between them are set, and greedy selection carries the $(1 - 1/e)$ guarantee under stated conditions. Across ten UCR datasets, eight local generators, and six adapted global methods, the perturbations of GALACTIC-L cover on average more of each class transition, at a shorter description length, than those of any competing generator.

Keywords: Counterfactual Explanations, Time-Series Classification, Explainability, Global Explanations.

1 INTRODUCTION

In high-stakes domains, spanning from cardiac diagnosis to fault detection on industrial equipment, time-series classification plays an integral role (Foumani et al., 2024). Yet returning only a label, such as "abnormal recording" or "degrading machine", leaves clinicians and operators with a finding and no route to act on. Understanding what led to the model's decision is important, and most explanation methods highlight which parts of a series supported its prediction as a relevance score per timestep (Zhou et al., 2021; Guillemé et al., 2019; Leung et al., 2023), a discriminative subsequence (Ye & Keogh, 2009), or evidence built into the classifier itself (Early et al., 2024). However, a simple label, even with some explanations, still does not reveal what specific changes in the signal would flip the model's verdict, making counterfactual explanations essential.

In generic classification settings, a counterfactual explanation answers exactly this question: what is the *minimal* change leading to a different prediction (Wachter et al., 2017)? For time series, however, additional requirements apply; a useful explanation must also have small change amplitudes and be confined to the temporal regions that support the transition from one class to another, rather than being scattered throughout the series. There are multiple local counterfactual explanation methods for time-series (Delaney et al., 2021; Li et al., 2022; Bahri et al., 2022b; Höllig et al., 2022; Wang et al., 2024; Refoyo & Luengo, 2024; Yamaguchi et al., 2024; Li et al., 2023; 2024), and none dominates every quality dimension (Li et al., 2026). However, because these methods operate strictly on a per-instance basis, they fail to reveal whether structurally similar time-series share consistent

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

pathways across class boundaries. Global counterfactual explanations could address this gap: a curated, small set of actions covering many instances at once (Rawal & Lakkaraju, 2020; Kavouras et al., 2026; Fragkathoulas et al., 2025). However, current global explanation methods are designed for tabular data and fail to account for time-series-specific constraints, such as temporal continuity.

Moreover, local explanations do not necessarily serve as good building blocks for global explanations: **per-instance optimality is antagonistic to cross-instance reuse**. A local objective that asks for the minimal change to a single series is satisfied the moment the prediction flips; consequently, its "nearest counterfactual" approaches the decision boundary with a margin close to zero, pointing in a direction specific to that series. Even for practical local generators, which are often heuristics and need not return a point near the boundary, a perturbation is not guaranteed to remain valid outside the instance it was built for. A related fragility is well documented in tabular recourse, where small action perturbations can invalidate a minimal recourse (Pawelczyk et al., 2023). Simply pooling locally optimal counterfactuals therefore inherits this missing margin rather than repairing it.

We address these gaps with GALACTIC, a unified local-to-global counterfactual framework for time-series classification, organized as a hierarchy spanning individual instances, structurally similar subgroups, classes, and the classifier as a whole. At the *local* level, GALACTIC-L makes reusability part of the objective rather than an assumed property of the resulting perturbations: a counterfactual must be valid and minimal for its own series *and* reach a margin shared by its structural subgroup. At the *global* level, GALACTIC-G selects rather than generates: it takes candidate perturbations from any source and keeps a compact reusable subset per directed transition under a Minimum Description Length objective (Rissanen, 1978), which measures coverage and complexity in bits, with no trade-off weights to tune. Our *contributions* are:

1. **Problem formulation.** We formalize counterfactual explainability for time-series classification at the series (local) level, requiring each counterfactual to reach a margin shared by its structural subgroup, and at the class-transition (global) level as selection among local candidates.

2. **Local counterfactuals with structural constraints.** We introduce a constrained gradient optimization that uses subgroup-specific temporal importance and requires a common margin within each structural subgroup, so that counterfactuals are built to remain valid across similar series.

3. **Global counterfactual summaries via MDL.** We formalize an MDL selection objective for choosing the perturbations of each directed class transition that is free of manually tuned trade-off weights: coverage, sparsity, and perturbation magnitude are expressed in one unit, bits, unlike multi-objective formulations. We characterize the submodularity of its reduction, deriving a greedy selection strategy with an approximation bound under stated conditions.

4. **Adaptive hierarchical refinement.** We define a hierarchical selection that splits the budget across the structural subgroups of each transition by size, so the number of perturbations adapts to the structure of each transition, and the smaller candidate space lets selection scale.

5. **Thorough experimental evaluation.** We show that GALACTIC-L gives better candidate pools for global selection than the local baselines, and GALACTIC-G more concise, higher-coverage summaries than the adapted global baselines, on ten UCR datasets and three classifier architectures.

2 RELATED WORK

Local counterfactual explanations for time series. Local counterfactual methods for time series differ mainly in how they search for a change that flips the prediction while keeping the series temporally coherent. NATIVE GUIDE takes the nearest series of another class and copies a subsequence of it (Delaney et al., 2021); TSEVO evolves changes over segments with multi-objective evolutionary search (Höllig et al., 2022); GLACIER runs gradient descent restricted by local or global importance vectors (Wang et al., 2024); SUB-SPACE evolves sparse subsequence masks, extended to multivariate series by MULTI-SPACE and to amortized generation by FASTPACE (Refoyo & Luengo, 2024; 2026; Refoyo et al., 2026); MASCOTS edits a symbolic representation of the series and decodes it back (Płudowski et al., 2026); other methods restrict edits to contiguous intervals (Yamaguchi et al., 2024) or guide them with shapelets and learned saliency (Li et al., 2022; 2023; 2024). Further methods steer the change toward the nearest cluster of another class (Ghosh et al., 2024), use cross-aligned perturbations (Hosseinzadeh et al., 2025), or search a latent space to keep the result plausible (Huang et al., 2024; Li et al., 2025). A recent benchmark finds no method best on validity,

proximity, sparsity, plausibility, and runtime at once (Li et al., 2026). All of them build a change for one series and never test the change on other series. GALACTIC-L adds this to the objective: each change must raise the target-class score of its series to a margin shared by structurally similar series.

Global and group counterfactual explanations. Global counterfactual explanations summarize, with a small set of actions, how the predictions of a whole population can be changed. ARES and its accelerated variant express these actions as interpretable rules (Rawal & Lakkaraju, 2020; Ley et al., 2022); GLOBE-CE learns a few translation directions, each scaled per instance (Ley et al., 2023); CET learns a decision tree that splits the population and assigns one action per leaf (Kanamori et al., 2022); T-CREX extracts rule-based regional counterfactuals from a surrogate model (Bewley et al., 2024); and GLANCE groups instances and chooses their actions jointly under coverage and cost (Kavouras et al., 2026). Group-oriented work adds feasibility and fairness across subpopulations (Kavouras et al., 2023; Fragkathoulas et al., 2025). All of them are designed for tabular features and a binary outcome, so none models temporal structure or more than one target class. GALACTIC-G instead selects, per directed class transition, a compact subset of the candidate perturbations of any local method under a Minimum Description Length (MDL) objective (Rissanen, 1978), which expresses coverage, sparsity and perturbation magnitude in bits, with no trade-off weights to tune. A related demonstration paper (Fragkathoulas et al., 2026) presents an interactive system for exploring local and global counterfactual explanations for time-series clustering. The present work instead develops the methodology for differentiable multiclass classifiers: a local-to-global formulation, subgroup-consistent generation through shared target margins, its theoretical properties, and a systematic empirical evaluation.

3 PROBLEM DEFINITION

Let $\mathcal{T} = \mathbb{R}^T$ be the space of univariate series of length T. A series $\boldsymbol{x} = [x_1, \ldots, x_T] \in \mathcal{T}$ is partitioned into M contiguous segments $\mathcal{S}(\boldsymbol{x}) = \{I_1, \ldots, I_M\}$ by breakpoints $0 = \tau_0 < \cdots < \tau_M = T$, with $I_m = \{\tau_{m-1} + 1, \ldots, \tau_m\}$. Given $X = \{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^N \subset \mathcal{T} \times \{1, \ldots, K\}$, let a differentiable multiclass classifier have logits $z_\theta : \mathcal{T} \to \mathbb{R}^K$, probabilities $f_\theta = \mathrm{softmax}(z_\theta)$, and prediction $\hat{y}(\boldsymbol{x}) = \arg\max_c z_{\theta,c}(\boldsymbol{x})$. $X_y = \{\boldsymbol{x} \in X : \hat{y}(\boldsymbol{x}) = y\}$ is the set of series predicted as y. The **target margin** $m_t(\boldsymbol{x}) = z_{\theta,t}(\boldsymbol{x}) - \max_{c \neq t} z_{\theta,c}(\boldsymbol{x})$ measures how far $\boldsymbol{x}$ lies inside class t: $\hat{y}(\boldsymbol{x}) = t$ exactly when $m_t(\boldsymbol{x}) > 0$.

3.1 LOCAL COUNTERFACTUAL EXPLANATIONS

For $\boldsymbol{x} \in X_y$, the target class $t(\boldsymbol{x}) = \arg\max_{c \neq y} z_{\theta,c}(\boldsymbol{x})$ is the runner-up class. The **directed transition** $y \to t$ is the change of prediction from y to t, and its **transition population** $X_{y \to t} = \{\boldsymbol{x} \in X_y : t(\boldsymbol{x}) = t\}$ holds the series predicted as y with runner-up t. A counterfactual perturbation $\boldsymbol{\delta} \in \mathbb{R}^T$ changes $\boldsymbol{x}$ into $\boldsymbol{x}^{cf} = \boldsymbol{x} + \boldsymbol{\delta}$ with $\hat{y}(\boldsymbol{x} + \boldsymbol{\delta}) = t$, and its size is measured by a cost $d(\boldsymbol{x}, \boldsymbol{x} + \boldsymbol{\delta})$ (Euclidean distance). Standard formulations treat each instance in isolation, following Wachter et al. (2017): given $\boldsymbol{x} \in X_{y \to t}$, find $\boldsymbol{\delta}^*$ attaining, or approaching to any tolerance, the infimum of $d(\boldsymbol{x}, \boldsymbol{x} + \boldsymbol{\delta})$ subject to $\hat{y}(\boldsymbol{x} + \boldsymbol{\delta}) = t$. The cheapest perturbations therefore approach the decision boundary: for any cost that grows as a perturbation is scaled up, no valid perturbation attains the infimum, and perturbations approaching the infimum have $m_t(\boldsymbol{x} + \boldsymbol{\delta}) \to 0$ (Proposition 1). Such a perturbation flips $\boldsymbol{x}$ with no margin left to flip similar series. Reusable perturbations should instead reach inside the target class. We therefore require, on top of **validity** ($\hat{y}(\boldsymbol{x} + \boldsymbol{\delta}) = t$), **margin consistency**: reaching a reference margin $\gamma_t > 0$ shared by a structural subgroup $G \subseteq X_{y \to t}$ of series with a common temporal layout and boundary.

Problem 1 (Subgroup-Consistent Local Counterfactuals)**.** *Given a subgroup $G \subseteq X_{y \to t}$ with target t and reference margin $\gamma_t > 0$, find, for every $\boldsymbol{x}_i \in G$, $\boldsymbol{\delta}_i^* = \arg\min_{\boldsymbol{\delta}_i} d(\boldsymbol{x}_i, \boldsymbol{x}_i + \boldsymbol{\delta}_i)$ subject to $m_t(\boldsymbol{x}_i + \boldsymbol{\delta}_i) \geq \gamma_t$, which implies $\hat{y}(\boldsymbol{x}_i + \boldsymbol{\delta}_i) = t$.*

Each perturbation is computed for one series and must reach γ_t on that series only; subgroup members share γ_t, not perturbations. The shared margin makes a flip of other members possible but not required, and Section 5.2 measures how often such flips occur. Requiring only $m_t > 0$ instead of $m_t \geq \gamma_t$ gives back the per-instance counterfactual above.

3.2 Transition-Specific Global Explanations

The valid local counterfactuals of a transition form its candidate pool $\Delta_{y\to t} = \{\boldsymbol{\delta}_i : \boldsymbol{x}_i \in X_{y\to t},\ \hat{y}(\boldsymbol{x}_i + \boldsymbol{\delta}_i) = t\}$. A global explanation is a compact subset $S \subseteq \Delta_{y\to t}$ that describes how the series of $X_{y\to t}$ can be moved to class t. A series is covered by S if at least one perturbation in S flips it: $\mathrm{cov}(X_{y\to t}, S) = \{\boldsymbol{x} \in X_{y\to t} : \exists \boldsymbol{\delta} \in S,\ \hat{y}(\boldsymbol{x} + \boldsymbol{\delta}) = t\}$. The **effectiveness** of S is the fraction of covered series: $\mathrm{eff}(X_{y\to t}, S) = |\mathrm{cov}(X_{y\to t}, S)|/|X_{y\to t}|$. The **flipping cost** of a covered series is the cost of the cheapest perturbation in S that flips it, $\mathrm{cost}_{\mathrm{flip}}(\boldsymbol{x}, S) = \min_{\boldsymbol{\delta}\in S:\hat{y}(\boldsymbol{x}+\boldsymbol{\delta})=t} d(\boldsymbol{x}, \boldsymbol{x}+\boldsymbol{\delta})$, and the **average flipping cost** averages it over the covered series: $\mathrm{afc}(X_{y\to t}, S) = \frac{1}{|\mathrm{cov}|}\sum_{\boldsymbol{x}\in\mathrm{cov}} \mathrm{cost}_{\mathrm{flip}}(\boldsymbol{x}, S)$.

Treating coverage, cost and sparsity as separate objectives yields a Pareto front, from which a solution is picked by manually tuned weights (Fragkathoulas et al., 2025; Kavouras et al., 2026). We instead use the *Minimum Description Length* (MDL) principle (Rissanen, 1978): the best summary is the one that describes the transition population in the fewest bits, measured by the two-part code length $L(S, X_{y\to t}) = L(S) + L(X_{y\to t} \mid S)$. The **model code** $L(S)$ is the cost of describing the selected perturbations: bits for how many timesteps each changes and by how much, plus a p_{sz}-bit pointer for each covered series and each perturbation:

$$L(S) = \sum_{\boldsymbol{\delta}\in S} \left[\log_2(\|\boldsymbol{\delta}\|_0 + 1) + \log_2(\|\boldsymbol{\delta}\|_1 + 1)\right] + \left(|\mathrm{cov}(X_{y\to t}, S)| + |S|\right) p_{sz}.$$

The **data code** $L(X_{y\to t} \mid S)$ charges each uncovered series as an outlier, at the code length of the most expensive candidate $\boldsymbol{\delta}^* \in \Delta_{y\to t}$:

$$L(X_{y\to t} \mid S) = |X_{y\to t} \setminus \mathrm{cov}(X_{y\to t}, S)| \cdot \left[\log_2(\|\boldsymbol{\delta}^*\|_0 + 1) + \log_2(\|\boldsymbol{\delta}^*\|_1 + 1) + 2p_{sz}\right].$$

Problem 2 (Global Transition Summary)**.** *Given transition population $X_{y\to t}$, candidate pool $\Delta_{y\to t}$, and cardinality budget μ, find $S^*_{y\to t} = \arg\min_{S\subseteq\Delta_{y\to t}} L(S) + L(X_{y\to t} \mid S)$ subject to $|S| \le \mu$.*

4 Galactic Framework

Galactic has two stages, summarized in Figure 1: Galactic-L generates local explanations under structural masks, and Galactic-G selects reusable perturbations per directed class transition.

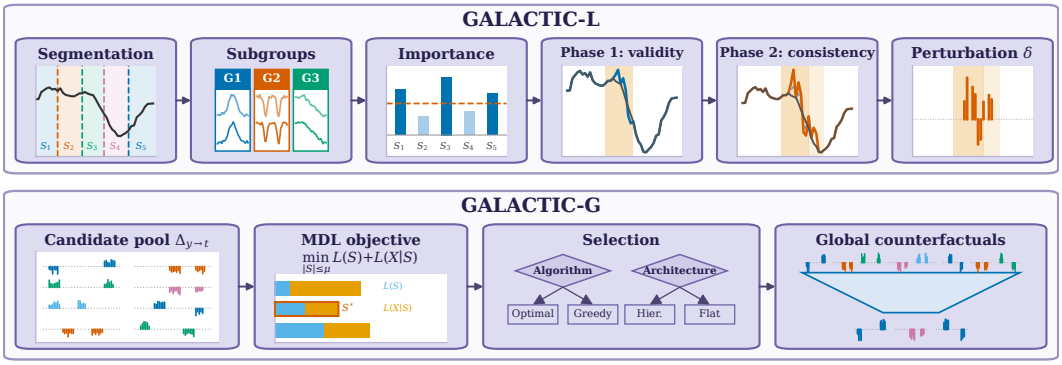


Figure 1: Overview of Galactic: Galactic-L (segmentation, interval scoring, two-phase masked search) and Galactic-G (MDL selection per class transition).

4.1 Galactic-L: Structural Local Explanations

Segmentation and structural subgrouping. Series of the same class can differ in shape and in how the classifier separates them from other classes. We partition each series into contiguous segments $\mathcal{S}(\boldsymbol{x})$ using NNSegment (Sivill & Flach, 2022), which detects change points from nearest-neighbor index discontinuities in a matrix profile (Yeh et al., 2016), ranked by local mean and

variance shifts. We then cluster X_y on two complementary signals: (1) *morphological*, using the segment layout $\mathcal{S}(\boldsymbol{x})$ to group series with shared temporal shape; and (2) *directional*, using pooled target-margin gradients $\nabla_{\boldsymbol{x}} m_{t(\boldsymbol{x})}(\boldsymbol{x})$ to group series whose margin reacts to changes in the same regions.

On the standardized concatenation of both signals, K-medoids clustering (Kaufman & Rousseeuw, 1990) with silhouette selection identifies the structural subgroups $\mathcal{G}_y = \{G_{y,1}, \ldots, G_{y,R_y}\}$. For any $\boldsymbol{x} \in G_{y,r}$, its target class $t(\boldsymbol{x}) = \arg\max_{c \neq y} z_{\theta,c}(\boldsymbol{x})$ induces the transition subgroup $G_{y,r\to t}$, the G of Problem 1.

Phase 1: Validity. *(i) Importance-guided perturbation.* We build a timestep importance mask $\boldsymbol{w} \in \{0,1\}^T$ from two interval scores computed over the subgroup: block permutation, which replaces an interval with the same interval from a random population member and measures the drop in source- and target-class evidence; and the magnitude of the target-margin gradient on the interval, averaged over the transition subgroup. Rather than a penalty in the loss, $\boldsymbol{w}$ is applied as a hard constraint on the gradient, $\nabla \leftarrow \nabla_{\boldsymbol{x}'}\mathcal{L} \odot \boldsymbol{w}$, so the optimizer cannot alter low-importance regions and the characteristic shape of the series is preserved. We consider five strategies for $\boldsymbol{w}$, defined formally in Section B.5: (1) *accuracy-budgeted*, thresholding the permutation accuracy drop at a fixed global quantile; (2) *source-confidence*, keeping the regions that support the current class y; (3) *target-confidence*, keeping those required to reach t; (4) *transition-combined*, aligning source and target breakpoints and scoring each interval by its source, target and transition evidence; and (5) *doubling-support*, our default, which ranks the combined intervals and admits them whole, doubling the number admitted at each stage until the flip succeeds or the edited fraction reaches a ceiling ρ. Doubling-support starts from the top-ranked half of the admitted intervals and widens only on failure. We also use *Unconstrained*, with no mask, as a baseline. The schedule of candidate supports is computed once per subgroup, so all members of the subgroup search the same nested supports and differ only in the stage at which they flip.

Algorithm 1 GALACTIC-L

Input: $\boldsymbol{x}, f_\theta, G_{y,r}, G_{y,r\to t}, \bar{\boldsymbol{g}}, \rho, \eta, N, \kappa, \gamma_t$
Output: valid $\boldsymbol{x}^{cf}$, or failure

Phase 1: Validity
$y \leftarrow \hat{y}(\boldsymbol{x})$
$t \leftarrow \arg\max_{c \neq y} z_{\theta,c}(\boldsymbol{x})$
$(\mathcal{I}, s) \leftarrow$ ALIGNSCORES$(G_{y,r}, X_t, G_{y,r\to t})$
$\boldsymbol{w} \leftarrow$ RANKEDMASK$(\mathcal{I}, s, \rho)$
$\boldsymbol{x}' \leftarrow \boldsymbol{x} + \boldsymbol{\xi} \odot \boldsymbol{w}$, $\boldsymbol{\xi} \sim \mathrm{Normal}(0, \sigma^2)$
$\boldsymbol{x}^{cf} \leftarrow$ None; $\mathcal{L}_{\min} \leftarrow \infty$
for $i = 1$ to N **do**
  $\mathcal{L}_i \leftarrow \lambda_1 \|\boldsymbol{x}' - \boldsymbol{x}\|_2 - \lambda_2 \log(f_{\theta,t}(\boldsymbol{x}') + \zeta) + \lambda_3(1 - \cos(\boldsymbol{x}' - \boldsymbol{x}, \bar{\boldsymbol{g}}))$
  if $\hat{y}(\boldsymbol{x}') = t$ **and** $f_{\theta,t}(\boldsymbol{x}') \geq \kappa$ **and** $\mathcal{L}_i < \mathcal{L}_{\min}$ **then**
    $\boldsymbol{x}^{cf} \leftarrow \boldsymbol{x}'$; $\mathcal{L}_{\min} \leftarrow \mathcal{L}_i$
  end if
  $\boldsymbol{x}' \leftarrow$ ADAMSTEP$(\boldsymbol{x}', \nabla_{\boldsymbol{x}'}\mathcal{L}_i \odot \boldsymbol{w}, \eta)$
end for
if $\boldsymbol{x}^{cf} =$ None **then** ▷ second attempt
  rerun the loop with $\boldsymbol{w} \leftarrow \mathbf{1}$, $\kappa \leftarrow 0$
end if
if $\boldsymbol{x}^{cf} =$ None **then**
  return failure
end if
for all $I \in$ SORTBYCHANGE$(\mathcal{I}, \boldsymbol{x}^{cf} - \boldsymbol{x})$ **do**
  $\widetilde{\boldsymbol{x}} \leftarrow \boldsymbol{x}^{cf}$; $\widetilde{\boldsymbol{x}}_I \leftarrow \boldsymbol{x}_I$
  if $\hat{y}(\widetilde{\boldsymbol{x}}) = t$ **then**
    $\boldsymbol{x}^{cf} \leftarrow \widetilde{\boldsymbol{x}}$; $\boldsymbol{w}_I \leftarrow 0$
  end if
end for

Phase 2: Consistency
$S \leftarrow \mathrm{supp}(\boldsymbol{x}^{cf} - \boldsymbol{x})$
while $m_t(\boldsymbol{x}^{cf}) < \gamma_t$ **do**
  $\boldsymbol{\delta} \leftarrow \arg\min_{\boldsymbol{\delta}} \frac{1}{2}\|\boldsymbol{\delta}\|_2^2 + \tau\|\nabla_j \boldsymbol{\delta}\|_1$ s.t. $m_t(\boldsymbol{x} + \boldsymbol{\delta}) \geq \gamma_t, \mathrm{supp}(\boldsymbol{\delta}) \subseteq S$
  if $m_t(\boldsymbol{x} + \boldsymbol{\delta}) \geq \gamma_t$ **then**
    $\boldsymbol{x}^{cf} \leftarrow \boldsymbol{x} + \boldsymbol{\delta}$; **break**
  end if
  $S \leftarrow S \cup$ NEXTSEGMENT$(\nabla_{\boldsymbol{x}} m_t)$ ▷ expand
end while
$\boldsymbol{x}^{cf} \leftarrow$ PRUNE$(\boldsymbol{x}^{cf},\ 0.8\, m_t(\boldsymbol{x}^{cf}))$ ▷ keep $\geq 80\%$
return $\boldsymbol{x}^{cf}$ ▷ phase-1 solution if γ_t unreachable

(ii) Targeted optimization and interval elimination. Within the admitted support $\boldsymbol{w}$, we initialize $\boldsymbol{x}'$ with small mask-restricted Gaussian jitter and minimize

$$\mathcal{L}(\boldsymbol{x}') = \lambda_1 \|\boldsymbol{x}' - \boldsymbol{x}\|_2 - \lambda_2 \log\big(f_{\theta,t}(\boldsymbol{x}') + \zeta\big) + \lambda_3\big(1 - \cos(\boldsymbol{x}' - \boldsymbol{x}, \bar{\boldsymbol{g}})\big)$$

using Adam (weights in Section B.6), where $\bar{\boldsymbol{g}}$ is the mean target-margin gradient of the transition subgroup, so the last term pulls the members of the subgroup toward a common edit direction, with every gradient projected through $\boldsymbol{w}$. An iterate is accepted only once $\hat{y}(\boldsymbol{x}') = t$ and $f_{\theta,t}(\boldsymbol{x}') \geq \kappa = 0.6$, and the search keeps the lowest-loss accepted iterate. A series that no admitted support flips gets a second attempt with the whole series editable and $\kappa = 0$, after which pruning shrinks the edit back (Section C.1.3). Starting from the lowest-change active interval, we then reset one whole interval at a time to its original values and keep the removal only if the classifier still predicts the target class; a final pass bisects the remaining edited runs likewise. This backward elimination keeps the counterfactual valid while shrinking the edited support. Phase 1 thus attains validity and fixes what the margin constraint of Problem 1 needs: the subgroup $G_{y,r\to t}$ and the starting support of each member.

Phase 2: Consistency. The iterate above is valid but lies near the decision boundary, so it is not yet a solution of Problem 1. Phase 2 keeps the subgroups and supports of phase 1, sets the reference margin γ_t to the median margin m_t of the series of class t (Section B.6), and solves the remaining constrained program

$$\min_{\{\boldsymbol{\delta}_i\}} \sum_{\boldsymbol{x}_i \in G} \left(\tfrac{1}{2}\|\boldsymbol{\delta}_i\|_2^2 + \tau \sum_j |\boldsymbol{\delta}_i[j{+}1] - \boldsymbol{\delta}_i[j]| \right) \quad \text{s.t.} \quad m_t(\boldsymbol{x}_i + \boldsymbol{\delta}_i) \geq \gamma_t,$$

with each $\boldsymbol{\delta}_i$ restricted to the phase-1 support of $\boldsymbol{x}_i$, solved with an augmented Lagrangian on the margin constraint. The total-variation weight τ keeps each refined perturbation contiguous. Besides γ_t (Problem 1), subgroup members share only the schedule of candidate supports, never a perturbation. Where the support of a member proves too small to reach γ_t, the support is expanded one segment at a time, in decreasing order of the gradient magnitude of that member, and members that still cannot attain the reference margin retain their phase-1 solution, so this phase can improve consistency but never forfeits validity. Support pruning then removes edits while keeping $\geq 80\%$ of the attained margin.

4.2 GALACTIC-G: TRANSITION SUMMARIES

Temporal perturbations are not enumerable, so selecting a few that induce $y \to t$ for the largest fraction of $X_{y \to t}$ needs a finite candidate pool.

Candidate-pool construction. For each transition $y \to t$, GALACTIC-L yields valid perturbations $\Delta_{y \to t}$, so every candidate is a valid, structure-respecting transition. Pools are built per directed transition and never mix sources or targets.

Algorithmic variants. We present four selection variants bounded by budget μ, plus a *Random* reference. **(1) Flat Optimal** (Appendix Algorithm 3) finds the exact minimizer over subsets of size at most μ. Since enumeration costs $\mathcal{O}(|\Delta_{y \to t}|^{\mu})$, we solve the same problem exactly as a mixed-integer linear program via branch-and-bound (Section B.8), yielding a true optimum at every reported scale. **(2) Flat Greedy** (Appendix Algorithm 4) iteratively selects the perturbation with maximal marginal MDL reduction, stopping when the best addition increases description length; under the admissibility condition in Section A, this reduction is monotone submodular, giving a $(1 - 1/e)$ approximation in $\mathcal{O}(\mu|\Delta_{y \to t}|)$ (Nemhauser et al., 1978; Hochba, 1997). **(3) Hierarchical Optimal** and **(4) Hierarchical Greedy** (Algorithm 2) first minimize MDL within each nonempty $G_{y,r \to t}$, under a budget proportional to its size, then select again from the union of the per-subgroup selections over all of $X_{y \to t}$, reducing the second-stage pool from $|\Delta_{y \to t}|$ to $|\widetilde{\Delta}_{y \to t}| \ll |\Delta_{y \to t}|$ at the price of global optimality.

Algorithm 2 GALACTIC-G: hierarchical

Input: $X_{y \to t}$, $\mathcal{G}_{y \to t}$, $\Delta_{y \to t}$, μ, selector $A \in \{\text{OPTIMAL}, \text{GREEDY}\}$
Output: transition summary $S_{y \to t}$
▷ SELECTMDL(Δ, X, μ, A) runs A (Appendix Algorithms 3 and 4) on (Δ, X, μ)
$\widetilde{\Delta} \leftarrow \emptyset$
for all $G \in \mathcal{G}_{y \to t}$ **do**
  $\Delta_G \leftarrow \{\boldsymbol{\delta} \in \Delta_{y \to t} : \exists \boldsymbol{x} \in G, \hat{y}(\boldsymbol{x} + \boldsymbol{\delta}) = t\}$
  $\mu_G \leftarrow \max\{1, \lceil |G|\mu / |X_{y \to t}| \rceil\}$
  $\widetilde{\Delta} \leftarrow \widetilde{\Delta} \cup$ SELECTMDL(Δ_G, G, μ_G, A)
end for
$S_{y \to t} \leftarrow$ SELECTMDL$(\widetilde{\Delta}, X_{y \to t}, \mu, A)$
return $S_{y \to t}$

5 EXPERIMENTAL EVALUATION

We evaluate GALACTIC through four questions. **RQ1** (Structural Contribution): Which mask strategy best restricts where GALACTIC-L edits a series while keeping the counterfactual valid? **RQ2** (Local Generators under Global Selection): GALACTIC-G accepts candidates from any local generator. How do the pools of different generators compare in effectiveness and MDL after selection, can the consistency phase of GALACTIC-L improve the pools of other generators, and how do the GALACTIC-G selectors differ? **RQ3** (Global Quality): Does transition-specific MDL selection outperform adapted global counterfactual methods? **RQ4** (Architecture Robustness): Do the conclusions of RQ2 and RQ3 hold across classifier architectures and training seeds?

Setup. We evaluate on 10 datasets from the UCR archive (Dau et al., 2019), selected to cover different domain types (sensor, image, spectroscopy, traffic, simulated), series lengths, class counts,

and dataset sizes. For each dataset, we train three classifiers: a ResNet (Wang et al., 2017), an LSTM-FCN (Karim et al., 2018) and a Transformer encoder; explanations are generated for the ResNet, and the other two are used in RQ4. Each experiment explains a seeded stratified 30% sample of the correctly classified instances of each class; Section C.1.4 compares this choice with MMD-Critic and with explaining every series. Section B describes the datasets, the preprocessing, and the training protocol. The source code is available in the **Anonymized GitHub Repository**.

Evaluation measurements. We report effectiveness (*Eff*) and average flipping cost (*Afc*) as defined in Section 3, with the cost normalized by the per-timestep value range of the dataset; average changed timesteps (*Act*), the fraction of the series edited; description length (*MDL*); and runtime (*RT*). Every (dataset, method) pair receives the same wall-clock budget and, for global methods, the same action budget $\mu = 4$ (Section B.2). Section B.9 gives the full definitions, the treatment of unreached instances, and the win/draw/loss protocol.

5.1 RQ1: Structural Contribution

RQ1 selects the mask strategy of GALACTIC-L: we compare the five strategies of Section 4 with *Unconstrained*, which uses no mask (Figure 2). Among the five masks and the *Unconstrained* control,

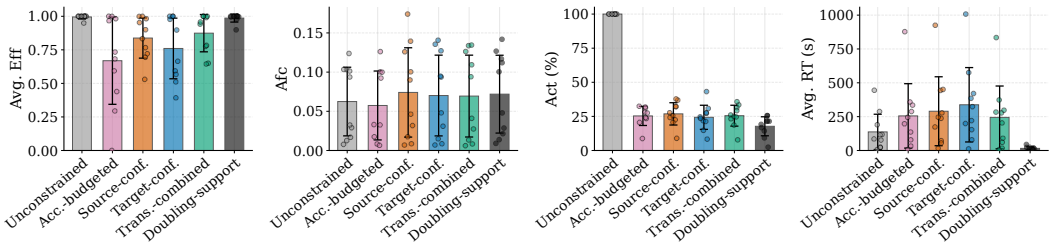


Figure 2: RQ1: effect of the mask that restricts where GALACTIC-L may edit. Each bar is one mask, or no mask (*Unconstrained*), with the full two-phase method; the panels show effectiveness (*Eff*, higher is better), flipping cost (*Afc*), fraction of edited timesteps (*Act*) and runtime (*RT*), all lower is better. Bars: mean over datasets; error bars: standard deviation; points: individual datasets.

doubling-support, the default in every later experiment, is the only one to reach full effectiveness while editing the smallest fraction of the series and running an order of magnitude faster than the rest. *Unconstrained* also reaches full effectiveness, but only because the edit may occur anywhere, and edits the whole series. A paired two-sided Wilcoxon signed-rank test over the ten datasets finds no difference in effectiveness between the two ($p = 1.00$), and doubling-support is never worse than any other variant (W/D/L $6/4/0$). The four masks built from a single signal reach lower effectiveness: a region the transition needs can fall outside a mask built from one importance measure, and reaching the target inside a smaller region costs more than *Unconstrained*.

5.2 RQ2: Local Generators under Global Selection

Local effectiveness alone is not decisive, since the global level needs candidates that transfer across series. We therefore compare GALACTIC-L with eight local baselines, NATIVE GUIDE (Delaney et al., 2021), SUB-SPACE (Refoyo & Luengo, 2024), MULTI-SPACE (Refoyo & Luengo, 2026), GLACIER (Wang et al., 2024), TSEVO (Höllig et al., 2022), CELS (Li et al., 2023), MASCOTS (Płudowski et al., 2026), and FASTPACE (Refoyo et al., 2026), by the global explanation that GALACTIC-G selects from the pool of each; Section B.2 describes each baseline and its configuration, and Sections C.2 and C.3 give the per-dataset local and local-under-global results and the coverage–description-length–edit-extent trade-offs.

Figure 3 applies the five GALACTIC-G selectors to the candidate pool of each local baseline and of GALACTIC-L. GALACTIC-G does not observe which generator produced a pool, so differences between the selectors within a source reflect the selection algorithm, and differences between sources under one selector reflect the candidate pool. Across sources, the pool of GALACTIC-L has the lowest average MDL cost under every selector. Section C.1.5 repeats the comparison restricted to

series that a source neither perturbs nor uses in selection; the ranking of the sources is unchanged, and GALACTIC-L retains the best effectiveness and MDL.

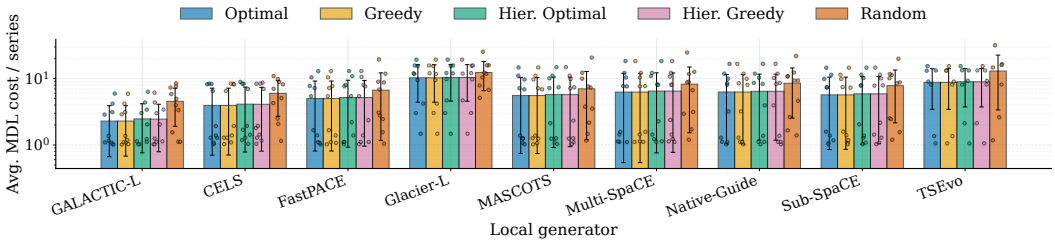


Figure 3: RQ2: description length (*MDL*, lower is better) of the explanation selected by each GALACTIC-G selector (colours) from the candidates of each local generator (groups). Bars: mean over datasets; error bars: standard deviation; points: individual datasets.

Table 1: RQ2: Phase 2 of GALACTIC-L applied to the candidates of other local generators. Effectiveness (higher is better) and *MDL* per series (lower is better) of the Flat Optimal selection, before and after Phase 2, and the change Δ.

Source	Avg. *Eff*			Avg. *MDL*/series		
	before	after	Δ	before	after	Δ
GLACIER	0.394	0.560	+0.166	9.94	7.62	-2.31
NATIVE GUIDE	0.661	0.762	+0.101	5.90	4.34	-1.56
TSEVO	0.492	0.555	+0.063	7.87	6.90	-0.98
FASTPACE	0.739	0.795	+0.057	4.69	3.87	-0.82
MULTI-SPACE	0.668	0.724	+0.056	5.79	4.98	-0.81
CELS	0.793	0.841	+0.048	3.95	3.29	-0.66
MASCOTS	0.721	0.767	+0.046	4.89	4.19	-0.70
SUB-SPACE	0.710	0.752	+0.041	5.04	4.46	-0.58
GALACTIC-L (native)	–	0.924	–	–	2.29	–

The native pool of GALACTIC-L attains the highest effectiveness and the shortest description of any source. The consistency phase improves every generator on both axes, with the largest gains for the pools initially furthest from margin consistency, so the consistency phase also applies to candidates we did not generate, but does not close the gap to GALACTIC-L.

Within each source, Flat Optimal (default) has the lowest cost, while Flat Greedy is close: a paired two-sided Wilcoxon signed-rank test over the 86 (dataset, source) pools with finite MDL costs finds no difference ($p = 0.59$), and greedy is exactly optimal on 41 pools. Hierarchical variants fall between the flat selectors and Random, the costliest on every pool. This ordering holds for every generator, so the global advantage of GALACTIC-L comes from the candidate pool, not the selector.

5.3 RQ3: GLOBAL QUALITY

We compare GALACTIC-G, using its default Flat Optimal selector over the GALACTIC-L pool, against six adapted global baselines: GLOBE-CE (Ley et al., 2023), GLANCE (Kavouras et al., 2026), FACEGROUP (Fragkathoulas et al., 2025), FAST ARES (Ley et al., 2022), CET (Kanamori et al., 2022), and T-CREX (Bewley et al., 2024). None targets time series: each treats a series as a vector of T timesteps (Section B.3). Table 2 reports the results, and Appendix Table 13 the per-dataset results.

GALACTIC-G has the shortest description on nine of ten datasets, well ahead of GLANCE$\star$. Effectiveness is closer: against the best competitor per dataset, GALACTIC-G wins four, ties two, and loses four, three of them to FACEGROUP$\star$ variants that edit every timestep of every series; the fourth, on ItalyPowerDemand, is a near-tie. CET$\star$, FAST ARES$\star$ and T-CREX$\star$ cover almost no series, as expected of rules and surrogates designed for tabular features, not hundreds of correlated

Table 2: RQ3: GALACTIC-G against six global baselines adapted to time series (⋆). #DS: datasets completed; ^{n†}: out of memory on n. Last row: Win/Draw/Lose vs. the best competitor per dataset.

Method	#DS	Avg. *MDL*/series	Avg. *Eff*	*Afc*	*Act*	Avg. *RT*
GALACTIC-G (Flat Optimal)	10	2.5	0.924	0.115	16.7%	73 s
CET⋆	10	15.6	0.042	0.096	16.7%	7039 s
FACEGROUP⋆ (cost)	$9^{\dagger}$	14.2	0.602	0.162	100.0%	3085 s
FACEGROUP⋆ (cov)	$9^{\dagger}$	14.3	0.757	0.195	100.0%	2989 s
FAST ARES⋆	$6^{4\dagger}$	15.0	0.004	0.011	2.7%	48 s
GLANCE⋆	10	6.5	0.657	0.127	8.5%	79 s
GLOBE-CE⋆	10	11.8	0.547	0.119	100.0%	414 s
T-CREX⋆	10	15.1	0.101	0.033	2.0%	12 s
W/D/L		9/0/1	4/2/4	3/0/7	2/0/8	3/0/7

timesteps. Against GLANCE⋆, the only other method with useful coverage that does not edit every series, GALACTIC-G leads on *Eff*, *Afc* and *MDL* at a comparable runtime; the one advantage of GLANCE⋆ is editing fewer timesteps, and no competitor with useful coverage beats GALACTIC-G on more than one axis. Selecting from margin-consistent candidates thus gives the shortest global explanation and highest average *Eff* without editing every timestep.

5.4 RQ4: ARCHITECTURE ROBUSTNESS

We repeat the local, local-under-global (Section 5.2) and global comparisons under LSTM-FCN and the Transformer, with every method run exactly as under ResNet (Table 3). The conclusions hold under the other architectures and all seeds. GALACTIC-L keeps the shortest description in the local-under-global comparison under every architecture, and GALACTIC-G leads globally on description length, losing at most two datasets. Effectiveness is closer, as under ResNet.

Table 3: RQ4: Win/Draw/Lose of our method against the best competitor on each dataset, in the local, local-under-global and global comparisons, for three architectures and three training seeds.

			Avg. MDL/series			*Avg. Eff*		
Comparison	Architecture	#DS	*Seed 42*	*Seed 43*	*Seed 44*	*Seed 42*	*Seed 43*	*Seed 44*
RQ2:	ResNet	10	–	–	–	2/8/0	1/8/1	2/7/1
Local	LSTM-FCN	10	–	–	–	3/7/0	2/8/0	2/6/2
	Transformer	10	–	–	–	0/9/1	2/8/0	1/9/0
RQ2:	ResNet	10	7/0/3	8/0/2	8/1/1	5/4/1	7/2/1	6/3/1
Local under Global	LSTM-FCN	10	8/0/2	7/1/2	8/0/2	6/3/1	6/3/1	6/3/1
	Transformer	10	6/0/4	6/0/4	6/0/4	4/4/2	4/4/2	4/3/3
RQ3:	ResNet	10	9/0/1	10/0/0	10/0/0	4/2/4	6/2/2	7/1/2
Global	LSTM-FCN	10	10/0/0	9/0/1	10/0/0	4/3/3	7/1/2	7/1/2
	Transformer	10	8/0/2	10/0/0	10/0/0	5/3/2	8/0/2	8/0/2

6 CONCLUSION

We presented GALACTIC, a local-to-global framework for counterfactual transitions in multiclass time-series classifiers. Since minimizing change to one series rewards no margin beyond it, such solutions are not built to transfer to others. GALACTIC-L instead drives each perturbation toward a reference margin shared by its structural subgroup, while GALACTIC-G selects a compact reusable set per directed class transition using an MDL objective agnostic to the candidate generator. Among established local generators, GALACTIC-L provides the best pool for global selection in coverage and description length while remaining locally valid on every attempted instance; against adapted global baselines, GALACTIC-G achieves the highest effectiveness and shortest description. The explanations are organized by instance, subgroup, transition, and classifier (Appendix Figure 6). Future work extends the formalism to multivariate series and MDL selection to other data domains.

Ethics statement

This work explains classifiers trained on public benchmark time series from the UCR archive (Dau et al., 2019), which contain no personal or identifying data, so no human-subjects or privacy concerns arise. Counterfactual explanations describe how the decision of a classifier could be changed rather than how a real process should be altered, and a global summary describes a model, not a population. We do not measure whether a counterfactual series is a plausible member of the target class or a physically realizable signal: the explanations describe the decision surface of the classifier, and adding manifold or domain constraints is left to future work. We therefore caution against reading a recourse action as a domain intervention without domain validation, and we report the edit extent of every method precisely so that implausible or degenerate edits remain visible rather than hidden behind a coverage number.

Reproducibility statement

The source code is available as an anonymized repository (Section 5). The repository provides the Conda environments, the ten datasets, the trained classifier checkpoints for all three architectures and seeds, the classifier training script, one runner per experiment, and one script per table and figure that builds it from the runner outputs. All sampling, training and selection is seeded, and the seeds are fixed in the released configurations. Section B.1 lists the datasets and the preprocessing; Section B.4 the hardware and the uniform per-method time budget; Sections B and B.5 the optimization settings and the classifier training protocol; Section B.3 what was adapted for each global baseline and why; and Section B.9 the measurement definitions and the treatment of populations a method never reached. Section A states every assumption and proves each proposition.

AI use statement

Generative AI tools were used only as a writing aid for editing and polishing prose. The research questions, the method, the experimental design, the implementation, and the analysis and interpretation of all results are the authors' own. The authors take responsibility for the final content of this work.

References

Omar Bahri, Soukaina Filali Boubrahimi, and Shah Muhammad Hamdi. Shapelet-based temporal association rule mining for multivariate time series classification. In *2022 IEEE International Conference on Big Data (Big Data)*, 2022a.

Omar Bahri, Peiyu Li, Soukaina Filali Boubrahimi, and Shah Muhammad Hamdi. Temporal rule-based counterfactual explanations for multivariate time series. In *2022 21st IEEE International Conference on Machine Learning and Applications (ICMLA)*, 2022b.

Tom Bewley, Salim I. Amoukou, Saumitra Mishra, Daniele Magazzeni, and Manuela Veloso. Counterfactual metarules for local and global recourse. In *Proceedings of the 41st International Conference on Machine Learning (ICML)*, volume 235 of *Proceedings of Machine Learning Research*, pp. 3707–3724. PMLR, 2024.

Hoang Anh Dau, Anthony Bagnall, Kaveh Kamgar, Chin-Chia Michael Yeh, Yan Zhu, Shaghayegh Gharghabi, Chotirat Ann Ratanamahatana, and Eamonn Keogh. The ucr time series archive. *IEEE/CAA Journal of Automatica Sinica*, 6(6):1293–1305, 2019.

Eoin Delaney, Derek Greene, and Mark T Keane. Instance-based counterfactual explanations for time series classification. In *International conference on case-based reasoning*, 2021.

Joseph Early, Gavin K. C. Cheung, Kurt Cutajar, Hanting Xie, Jas Kandola, and Niall Twomey. Inherently interpretable time series classification via multiple instance learning. In *International Conference on Learning Representations (ICLR)*, 2024.

Navid Mohammadi Foumani, Lynn Miller, Chang Wei Tan, Geoffrey I Webb, Germain Forestier, and Mahsa Salehi. Deep learning for time series classification and extrinsic regression: A current survey. *ACM Computing Surveys*, 56(9):1–45, 2024.

Christos Fragkathoulas, Vasiliki Papanikou, Evaggelia Pitoura, and Evimaria Terzi. Facegroup: Feasible and actionable counterfactual explanations for group fairness. In *ECML-PKDD*, 2025.

Christos Fragkathoulas, Eleni Psaroudaki, Themis Palpanas, and Evaggelia Pitoura. Galactica: An interactive system for local and global counterfactual explanations for time-series clustering. *Proceedings of the VLDB Endowment*, 19(12):4774–4777, 2026.

Ayanabha Ghosh, Rishi Jain, Shubham Parida, and Debasis Das. Ts-nuc: Nearest unlike cluster guided generative counterfactual estimation for time series classification. In *Lecture Notes in Computer Science*, pp. 392–406, 2024. doi: 10.1007/978-3-031-78395-1_26.

Mael Guillemé, Veronique Masson, Laurence Rozé, and Alexandre Termier. Agnostic local explanation for time series classification. In *IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, pp. 432–439, 2019.

Dorit S Hochba. Approximation algorithms for np-hard problems. *ACM Sigact News*, 1997.

Höllig et al. Tsevo: Evolutionary counterfactual explanations for time series classification. In *(ICMLA)*, 2022.

Pouya Hosseinzadeh, Peiyu Li, Omar Bahri, Soukaina Filali Boubrahimi, and Shah Muhammad Hamdi. Cactus: Cross-aligned counterfactual explanation for time series classification. In *2025 IEEE 12th International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 1–10, 2025. doi: 10.1109/DSAA65442.2025.11247987.

Qi Huang, Sofoklis Kitharidis, Thomas Bäck, and Niki van Stein. Tx-gen: Multi-objective optimization for sparse counterfactual explanations for time-series classification. In *Proceedings of the 1st International Conference on Explainable AI for Neural and Symbolic Methods*, pp. 62–74, 2024. doi: 10.5220/0013066400003886.

Kentaro Kanamori, Takuya Takagi, Ken Kobayashi, and Yuichi Ike. Counterfactual explanation trees: Transparent and consistent actionable recourse with decision trees. In *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 151 of *Proceedings of Machine Learning Research*, pp. 1846–1870. PMLR, 2022.

Fazle Karim, Somshubra Majumdar, Houshang Darabi, and Shun Chen. Lstm fully convolutional networks for time series classification. *IEEE Access*, 6:1662–1669, 2018. doi: 10.1109/ACCESS.2017.2779939.

Leonard Kaufman and Peter J Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. John Wiley & Sons, 1990.

Loukas Kavouras, Konstantinos Tsopelas, Giorgos Giannopoulos, Dimitris Sacharidis, Eleni Psaroudaki, Nikolaos Theologitis, Dimitrios Rontogiannis, Dimitris Fotakis, and Ioannis Emiris. Fairness aware counterfactuals for subgroups. *Advances in Neural Information Processing Systems*, 36:58246–58276, 2023.

Loukas Kavouras, Eleni Psaroudaki, Konstantinos Tsopelas, Dimitrios Rontogiannis, Nikolaos Theologitis, Dimitris Sacharidis, Giorgos Giannopoulos, Dimitrios Tomaras, Kleopatra Markou, Dimitrios Gunopulos, Dimitris Fotakis, and Ioannis Emiris. Glance: Global actions in a nutshell for counterfactual explainability. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 22545–22553, 2026. doi: 10.1609/aaai.v40i27.39414.

Been Kim, Rajiv Khanna, and Oluwasanmi O Koyejo. Examples are not enough, learn to criticize! criticism for interpretability. *Advances in neural information processing systems*, 2016.

Kin Kwan Leung, Clayton Rooke, Jonathan Smith, Saba Zuberi, and Maksims Volkovs. Temporal dependencies in feature importance for time series prediction. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=C0q9oBc3n4.

Dan Ley, Saumitra Mishra, and Daniele Magazzeni. Global Counterfactual Explanations: Investigations, Implementations and Improvements. In *ICLR 2022 Workshops: PAIR2Struct*, 2022. URL https://mlanthology.org/iclrw/2022/ley2022iclrw-global/. Introduces Fast AReS, an improvement of AReS (Rawal & Lakkaraju, 2020).

Dan Ley, Saumitra Mishra, and Daniele Magazzeni. Globe-ce: A translation based approach for global counterfactual explanations. In *International conference on machine learning*, pp. 19315–19342. PMLR, 2023.

Peiyu Li, Omar Bahri, Soukaïna Filali Boubrahimi, and Shah Muhammad Hamdi. Sg-cf: Shapelet-guided counterfactual explanation for time series classification. In *IEEE International Conference on Big Data (Big Data)*, 2022.

Peiyu Li, Omar Bahri, Soukaina Filali Boubrahimi, and Shah Muhammad Hamdi. Cels: Counterfactual explanations for time series data via learned saliency maps. In *2023 IEEE International Conference on Big Data (BigData)*, pp. 718–727, 2023. doi: 10.1109/BIGDATA59044.2023.10386229.

Peiyu Li, Omar Bahri, Soukaina Filali Boubrahimi, and Shah Muhammad Hamdi. M-cels: Counterfactual explanation for multivariate time series data guided by learned saliency maps. In *2024 International Conference on Machine Learning and Applications (ICMLA)*, pp. 713–718, 2024. doi: 10.1109/ICMLA61862.2024.00103.

Peiyu Li, Omar Bahri, Pouya Hosseinzadeh, Soukaina Filali Boubrahimi, and Shah Muhammad Hamdi. Diverse and plausible counterfactual explanations for time series via latent space optimization. In *2025 IEEE 12th International Conference on Data Science and Advanced Analytics (DSAA)*, pp. 1–10, 2025. doi: 10.1109/DSAA65442.2025.11248020.

Peiyu Li, Omar Bahri, Soukaina Filali Boubrahimi, and Shah Muhammad Hamdi. Counterfactual explanation bake-off: A review and experimental evaluation for time series classification. *Machine Learning*, 115(5), 2026. doi: 10.1007/s10994-026-07056-4.

George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions I. *Mathematical Programming*, 14(1):265–294, 1978.

Martin Pawelczyk, Klaus Broelemann, and Gjergji Kasneci. On counterfactual explanations under predictive multiplicity. In *Conference on Uncertainty in Artificial Intelligence*, pp. 809–818. PMLR, 2020.

Martin Pawelczyk, Teresa Datta, Johannes van den Heuvel, Gjergji Kasneci, and Himabindu Lakkaraju. Probabilistically robust recourse: Navigating the trade-offs between costs and robustness in algorithmic recourse. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net, 2023. URL https://openreview.net/forum?id=sC-PmTsiTB.

Dawid Płudowski, Francesco Spinnato, Piotr Wilczyński, Krzysztof Kotowski, Evridiki Vasileia Ntagiou, Riccardo Guidotti, and Przemysław Biecek. Mascots: Model-agnostic symbolic counterfactual explanations for time series. In *Machine Learning and Knowledge Discovery in Databases: ECML PKDD 2025*, volume 16016 of *Lecture Notes in Computer Science*, pp. 94–112, 2026. doi: 10.1007/978-3-032-06078-5_6.

Kaivalya Rawal and Himabindu Lakkaraju. Beyond individualized recourse: Interpretable and interactive summaries of actionable recourses. *Advances in Neural Information Processing Systems*, 33:12187–12198, 2020.

Mario Refoyo and David Luengo. Sub-space: Subsequence-based sparse counterfactual explanations for time series classification problems. In *XAI*, 2024.

Mario Refoyo and David Luengo. Multi-space: Multi-objective subsequence-based sparse counterfactual explanations for multivariate time series classification. *Neurocomputing*, 2026.

Mario Refoyo, Alejandro Boleas, and David Luengo. Fastpace: Fast planning of counterfactual explanations for time series classification. *Data Mining and Knowledge Discovery*, 2026.

Jorma Rissanen. Modeling by shortest data description. *Automatica*, 1978.

Torty Sivill and Peter Flach. Limesegment: Meaningful, realistic time series explanations. In *International Conference on Artificial Intelligence and Statistics*. PMLR, 2022.

Sohini Upadhyay, Shalmali Joshi, and Himabindu Lakkaraju. Towards robust and reliable algorithmic recourse. *Advances in Neural Information Processing Systems*, 34:16926–16937, 2021.

Sandra Wachter, Brent Mittelstadt, and Chris Russell. Counterfactual explanations without opening the black box: Automated decisions and the gdpr. *Harv. JL & Tech.*, 2017.

Zhendong Wang, Isak Samsten, Ioanna Miliou, Rami Mochaourab, and Panagiotis Papapetrou. Glacier: guided locally constrained counterfactual explanations for time series classification. *Machine Learning*, 2024.

Zhiguang Wang, Weizhong Yan, and Tim Oates. Time series classification from scratch with deep neural networks: A strong baseline. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 1578–1585, 2017. doi: 10.1109/IJCNN.2017.7966039.

Akihiro Yamaguchi, Ken Ueno, Ryusei Shingaki, and Hisashi Kashima. Learning counterfactual explanations with intervals for time-series classification. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, 2024.

Lexiang Ye and Eamonn Keogh. Time series shapelets: a new primitive for data mining. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 947–956, 2009.

Chin-Chia Michael Yeh, Yan Zhu, Liudmila Ulanova, Nurjahan Begum, Yifei Ding, Hoang Anh Dau, Diego Silva, Abdullah Mueen, and Eamonn Keogh. Matrix profile i: All pairs similarity joins for time series: A unifying view that includes motifs, discords and shapelets. In *2016 IEEE 16th International Conference on Data Mining (ICDM)*, pp. 1317–1322, 2016. doi: 10.1109/ICDM.2016.0179.

Linjiang Zhou, Chao Ma, Xiaochuan Shi, Dian Zhang, Wei Li, and Libing Wu. Salience-cam: Visual explanations from convolutional neural networks via salience score. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2021.

A Theoretical Foundations

This section proves three results. First, for the minimum-change formulation of Section 3.1 and a cost increasing along rays, every minimizing sequence approaches the decision boundary (Proposition 1), which motivates the margin-consistency requirement of Problem 1. Second, the transition cost $L(X, S)$ of Problem 2 is supermodular, equivalently its reduction $\mathcal{D}(S) = L(X, \emptyset) - L(X, S)$ is submodular, with no further assumption (Propositions 2 and 3). Third, under Assumption 1, $\mathcal{D}$ is also monotone on the cardinality-constrained domain, so greedy selection attains the standard $(1 - 1/e)$ approximation to the optimal selection (Proposition 4).

Fix a supported transition $y \to t$, write $X = X_{y \to t}$ and $\Delta = \Delta_{y \to t}$, and suppress the transition subscripts. For each $\boldsymbol{\delta} \in \Delta$, define its code cost

$$c_{\boldsymbol{\delta}} = \log_2(\|\boldsymbol{\delta}\|_0 + 1) + \log_2(\|\boldsymbol{\delta}\|_1 + 1),$$

and let $M = \max_{\boldsymbol{\delta} \in \Delta} c_{\boldsymbol{\delta}} + 2p_{sz}$ be the fixed outlier cost. As in MDL-based pattern mining, the two parts of $L(X, S)$ are code-length surrogates rather than a complete decodable code: the model code charges the size and the ℓ_1 magnitude of each perturbation, not its positions and values. Let $\gamma(\boldsymbol{\delta} \mid S)$ be the number of previously uncovered instances newly covered by adding $\boldsymbol{\delta}$ to S.

Assumption 1 (Budget-admissible marginal codes)**.** *For the candidate pool and budget μ, every addition considered in the feasible domain satisfies*

$$\gamma(\boldsymbol{\delta} \mid S)(M - p_{sz}) \geq c_{\boldsymbol{\delta}} + p_{sz} \quad \textit{for all } |S| < \mu,\ \boldsymbol{\delta} \notin S.$$

This condition is deliberately strong and testable. If it fails, the MDL reduction remains submodular but need not be monotone; we then describe greedy selection as a heuristic and do not invoke the $(1 - 1/e)$ guarantee.

That minimal-cost counterfactuals lie close to the decision boundary, and are therefore fragile, is known from algorithmic recourse (Pawelczyk et al., 2020; Upadhyay et al., 2021; Pawelczyk et al., 2023); the proposition states this formally for the target margin used here.

Proposition 1 (Minimal-cost counterfactuals have vanishing margin)**.** *Let z_θ be continuous, let $\boldsymbol{x}$ satisfy $\hat{y}(\boldsymbol{x}) = y$ with $y \neq t$, and recall the target margin $m_t(u) = z_{\theta,t}(u) - \max_{c \neq t} z_{\theta,c}(u)$ of Section 3, so that the validity constraint of Section 3.1 reads $m_t(\boldsymbol{x} + \boldsymbol{\delta}) > 0$. Write $c(\boldsymbol{\delta}) = d(\boldsymbol{x}, \boldsymbol{x} + \boldsymbol{\delta})$ and assume that (A1) c is strictly increasing along rays from $\boldsymbol{x}$, that is, $c(s\boldsymbol{\delta}) < c(\boldsymbol{\delta})$ for every $\boldsymbol{\delta} \neq \mathbf{0}$ and $s \in (0, 1)$, and (A2) c is continuous and coercive, that is, its sublevel sets are bounded. Every norm of $\boldsymbol{\delta}$ satisfies both, as does any continuous, strictly increasing, unbounded function of a norm. Let $F = \{\boldsymbol{\delta} : m_t(\boldsymbol{x} + \boldsymbol{\delta}) > 0\}$ be nonempty and let $r^* = \inf_{\boldsymbol{\delta} \in F} c(\boldsymbol{\delta})$. Then (i) no $\boldsymbol{\delta} \in F$ attains r^*, and (ii) every sequence $(\boldsymbol{\delta}_k) \subset F$ with $c(\boldsymbol{\delta}_k) \to r^*$ satisfies $m_t(\boldsymbol{x} + \boldsymbol{\delta}_k) \to 0$.*

Proof. (i) The map $\boldsymbol{\delta} \mapsto m_t(\boldsymbol{x} + \boldsymbol{\delta})$ is continuous, so F is open. Let $\boldsymbol{\delta} \in F$. Since $m_t(\boldsymbol{x}) \leq 0$, because $\boldsymbol{x}$ is classified as $y \neq t$, the zero perturbation is not in F, hence $\boldsymbol{\delta} \neq \mathbf{0}$. By continuity of m_t at $s = 1$ there is $s \in (0, 1)$ with $s\boldsymbol{\delta} \in F$, and by (A1) $c(s\boldsymbol{\delta}) < c(\boldsymbol{\delta})$. Hence $c(\boldsymbol{\delta}) > r^*$, and since $\boldsymbol{\delta} \in F$ was arbitrary, r^* is not attained.

(ii) Suppose the claim fails. Then there are $\varepsilon > 0$ and a subsequence with $m_t(\boldsymbol{x} + \boldsymbol{\delta}_k) \geq \varepsilon$. Its costs converge to r^*, so by coercivity the subsequence is bounded, and in $\mathbb{R}^T$ a further subsequence converges to some $\bar{\boldsymbol{\delta}}$. By continuity of c, $c(\bar{\boldsymbol{\delta}}) = r^*$, and by continuity of m_t, $m_t(\boldsymbol{x} + \bar{\boldsymbol{\delta}}) \geq \varepsilon > 0$. Then $\bar{\boldsymbol{\delta}} \in F$ attains r^*, contradicting (i). □

The margin constraint of Problem 1 addresses this directly. For a cost that grows along rays, the per-instance program of Section 3.1 has no minimizer, since its feasible set is open, and its minimizing sequences approach the decision boundary. A perturbation near $m_t = 0$ leaves no margin to absorb the difference between the series it was built for and another series, so reuse is unlikely. The result covers such costs only, not objectives that reward margin, and it does not describe any particular generator: heuristic generators need not follow a minimizing sequence, and Section 5.2 measures where their counterfactuals land.

Proposition 2 (Supermodularity of transition MDL)**.** *For fixed X and finite Δ, the transition MDL cost $L(X, S) = L(S) + L(X \mid S)$ is supermodular in $S \subseteq \Delta$.*

Proof. For $S \subseteq \Delta$, let

$$U(S) = X \setminus \mathrm{cov}(X, S)$$

be the instances not covered by any perturbation in S. The marginal coverage of $\boldsymbol{\delta} \notin S$ is

$$\begin{aligned} \gamma(\boldsymbol{\delta} \mid S) &= |\mathrm{cov}(X, S \cup \{\boldsymbol{\delta}\}) \setminus \mathrm{cov}(X, S)| \\ &= |\mathrm{cov}(X, \{\boldsymbol{\delta}\}) \cap U(S)| \\ &= |\mathrm{cov}(X, S \cup \{\boldsymbol{\delta}\})| - |\mathrm{cov}(X, S)|. \end{aligned} \tag{1}$$

If $A \subseteq B$, then $\mathrm{cov}(X, A) \subseteq \mathrm{cov}(X, B)$, and hence $U(B) \subseteq U(A)$. Therefore, for $\boldsymbol{\delta} \notin B$,

$$\gamma(\boldsymbol{\delta} \mid B) \leq \gamma(\boldsymbol{\delta} \mid A). \tag{2}$$

We next derive the complete marginal MDL cost. Expanding the model code before and after adding $\boldsymbol{\delta}$ gives

$$\begin{aligned} L(S \cup \{\boldsymbol{\delta}\}) &= \sum_{\boldsymbol{\delta}' \in S \cup \{\boldsymbol{\delta}\}} c_{\boldsymbol{\delta}'} + (|\mathrm{cov}(X, S \cup \{\boldsymbol{\delta}\})| + |S \cup \{\boldsymbol{\delta}\}|)\, p_{sz}, \\ L(S) &= \sum_{\boldsymbol{\delta}' \in S} c_{\boldsymbol{\delta}'} + (|\mathrm{cov}(X, S)| + |S|)\, p_{sz}. \end{aligned}$$

Subtracting the second expansion from the first, adding $\boldsymbol{\delta}$ contributes its code, one pointer for the new candidate, and one pointer for each newly covered instance:

$$\begin{aligned} L(S \cup \{\boldsymbol{\delta}\}) - L(S) &= c_{\boldsymbol{\delta}} + \big(|S \cup \{\boldsymbol{\delta}\}| - |S|\big)p_{sz} \\ &\quad + \big(|\mathrm{cov}(X, S \cup \{\boldsymbol{\delta}\})| - |\mathrm{cov}(X, S)|\big)\, p_{sz} \\ &= c_{\boldsymbol{\delta}} + p_{sz} + \gamma(\boldsymbol{\delta} \mid S)p_{sz}, \end{aligned} \tag{3}$$

where the last equality uses equation 1. The data code is $L(X \mid S) = |U(S)|M$. Since adding $\boldsymbol{\delta}$ removes exactly $\gamma(\boldsymbol{\delta} \mid S)$ instances from the uncovered set,

$$|U(S \cup \{\boldsymbol{\delta}\})| = |U(S)| - \gamma(\boldsymbol{\delta} \mid S). \tag{4}$$

It follows that

$$\begin{aligned} L(X \mid S \cup \{\boldsymbol{\delta}\}) - L(X \mid S) &\overset{equation\ 4}{=} \big(|U(S)| - \gamma(\boldsymbol{\delta} \mid S)\big)M - |U(S)|M \\ &= -\gamma(\boldsymbol{\delta} \mid S)M. \end{aligned} \tag{5}$$

Combining equation 3 and equation 5 gives

$$L(X, S \cup \{\boldsymbol{\delta}\}) - L(X, S) = c_{\boldsymbol{\delta}} + p_{sz} + \gamma(\boldsymbol{\delta} \mid S)(p_{sz} - M). \tag{6}$$

The fixed outlier code satisfies $M = \max_{\boldsymbol{\delta}' \in \Delta} c_{\boldsymbol{\delta}'} + 2p_{sz} \geq 2p_{sz} > p_{sz}$. To make the supermodularity comparison explicit, decompose the difference between the marginal increments at A and B into its model and data parts:

$$\begin{aligned} &[L(X, A \cup \{\boldsymbol{\delta}\}) - L(X, A)] - [L(X, B \cup \{\boldsymbol{\delta}\}) - L(X, B)] \\ &= [L(A \cup \{\boldsymbol{\delta}\}) - L(A) - L(B \cup \{\boldsymbol{\delta}\}) + L(B)] \\ &\quad + [L(X \mid A \cup \{\boldsymbol{\delta}\}) - L(X \mid A) - L(X \mid B \cup \{\boldsymbol{\delta}\}) + L(X \mid B)]\,. \end{aligned} \tag{7}$$

The model contribution is

$$\begin{aligned} &L(A \cup \{\boldsymbol{\delta}\}) - L(A) - L(B \cup \{\boldsymbol{\delta}\}) + L(B) \\ &\quad = \big(\gamma(\boldsymbol{\delta} \mid A) - \gamma(\boldsymbol{\delta} \mid B)\big)p_{sz}, \end{aligned}$$

by equation 3, while the data contribution is

$$\begin{aligned} &L(X \mid A \cup \{\boldsymbol{\delta}\}) - L(X \mid A) - L(X \mid B \cup \{\boldsymbol{\delta}\}) + L(X \mid B) \\ &\quad = -\big(\gamma(\boldsymbol{\delta} \mid A) - \gamma(\boldsymbol{\delta} \mid B)\big)M. \end{aligned}$$

by equation 5. Substitution into equation 7 yields

$$\begin{aligned} &[L(X, A \cup \{\boldsymbol{\delta}\}) - L(X, A)] - [L(X, B \cup \{\boldsymbol{\delta}\}) - L(X, B)] \\ &\quad = \big(\gamma(\boldsymbol{\delta} \mid A) - \gamma(\boldsymbol{\delta} \mid B)\big)(p_{sz} - M). \end{aligned}$$

The first factor on the right is nonnegative by equation 2, whereas the second is negative. Consequently,

$$L(X, A \cup \{\boldsymbol{\delta}\}) - L(X, A) \leq L(X, B \cup \{\boldsymbol{\delta}\}) - L(X, B),$$

which is the increasing-marginal-cost characterization of supermodularity. Notice that this argument does not invoke Assumption 1; that condition is needed only for monotonicity of the reduction. □

Proposition 3 (Submodular MDL reduction)**.** *The reduction $\mathcal{D}(S) = L(X, \emptyset) - L(X, S)$ is submodular. Under Assumption 1, it is also monotone over the feasible domain up to budget μ.*

Proof. First, consider submodularity. For $A \subseteq B \subseteq \Delta$ and $\boldsymbol{\delta} \notin B$,

$$\begin{aligned} \mathcal{D}(A \cup \{\boldsymbol{\delta}\}) - \mathcal{D}(A) &= L(X, A) - L(X, A \cup \{\boldsymbol{\delta}\}) \\ &\geq L(X, B) - L(X, B \cup \{\boldsymbol{\delta}\}) \\ &= \mathcal{D}(B \cup \{\boldsymbol{\delta}\}) - \mathcal{D}(B), \end{aligned}$$

where the inequality is the supermodularity inequality from Proposition 2 after multiplying by -1. Hence $\mathcal{D}$ is submodular.

For monotonicity, negate the marginal cost in equation 6:

$$\begin{aligned}\mathcal{D}(S \cup \{\boldsymbol{\delta}\}) - \mathcal{D}(S) &= L(X, S) - L(X, S \cup \{\boldsymbol{\delta}\}) \\ &= \gamma(\boldsymbol{\delta} \mid S)(M - p_{sz}) - (c_{\boldsymbol{\delta}} + p_{sz}). \end{aligned} \tag{8}$$

For every feasible $|S| < \mu$ and $\boldsymbol{\delta} \notin S$, Assumption 1 states exactly that the right-hand side of equation 8 is nonnegative. Therefore, adding any candidate within the stated feasible domain cannot decrease $\mathcal{D}$, establishing monotonicity there. Without this condition, submodularity still holds, but a candidate with insufficient new coverage can have a negative marginal reduction. □

Proposition 4 (Conditional greedy guarantee)**.** *If Assumption 1 holds over the cardinality-constrained feasible domain, greedy maximization of $\mathcal{D}$ returns S^{gr} satisfying*

$$\mathcal{D}(S^{\mathrm{gr}}) \geq (1 - 1/e) \max_{|S| \leq \mu} \mathcal{D}(S).$$

Proof. By construction, $\mathcal{D}(\emptyset) = 0$. By Proposition 3, $\mathcal{D}$ is submodular and, under Assumption 1, monotone on every set and addition considered up to cardinality μ. It is consequently nonnegative on this feasible domain. The standard greedy theorem for maximizing a normalized, nonnegative, monotone submodular function subject to a cardinality constraint therefore yields the stated $(1-1/e)$ bound. □

B Implementation, Datasets and Baselines

This section documents the datasets, classifiers, baselines and implementation details of GALACTIC needed to reproduce the experiments: the datasets and classifiers (Section B.1); the local and global baselines and how each was run or adapted (Sections B.2 and B.3); the compute environment (Section B.4); the segmentation, importance weighting and search settings of GALACTIC-L, with the two phases illustrated on one instance (Sections B.5, B.6 and B.6.1); the default GALACTIC-G selector, the selection algorithms, and the exact solver behind them (Sections B.7 and B.8); and the evaluation measurements with the semantics of targets and failures (Section B.9).

B.1 Datasets and Classifiers

Training uses the TRAIN and TEST partitions as the UCR archive ships them. The ten datasets span size, length, class count and domain type (sensor, image, spectroscopy, traffic, simulated). The ResNet (Wang et al., 2017) is the classifier explained in the main experiments; LSTM-FCN (Karim et al., 2018) and a Transformer encoder of our own implementation, which relies on self-attention rather than convolution or recurrence, are used in RQ4, so that the comparison covers three inductive biases. For each dataset and architecture, hyperparameters are selected by stratified 5-fold cross-validation within TRAIN over learning rate $\{5 \cdot 10^{-4}, 10^{-3}, 2 \cdot 10^{-3}\}$ and batch size $\{16, 32\}$, at 1500 epochs with early stopping at patience 120 in every configuration. Selection takes the highest mean cross-validation accuracy, with mean validation loss breaking ties. The selected configuration is refit on the full TRAIN partition under seeds 42, 43 and 44; seed 42 is the deployed model and the other two measure the spread across initializations, and which seed is deployed is fixed before any test score is computed. Appendix Table 4 reports the accuracy of the deployed model on TEST, and that model is the one explained.

Explanations are computed over the pooled population: TRAIN and TEST concatenated, duplicate rows removed, and every series z-normalized against its own mean and standard deviation. Counterfactuals are generated only for correctly classified instances, and each experiment draws a seeded stratified 30% sample of them. Explanations are therefore computed in part on series the classifier was fit on, which is deliberate: a transition population needs the members of both partitions to be large enough to summarize, and an explanation is a statement about the decision surface of a given model rather than a prediction to be generalized. Every method receives the same instances, preprocessing and classifier.

Table 4: Evaluation datasets: explanation population size N after concatenating the UCR TRAIN and TEST partitions and removing duplicates, series length T, class count $|C|$, and the accuracy of each deployed classifier on the held-out TEST partition, with the standard deviation over the three refit seeds.

Dataset	N	T	$\lvert C\rvert$	ResNet	LSTM-FCN	Transformer
Chinatown	363	24	2	0.974 ±0.010	0.974 ±0.002	0.965 ±0.022
Coffee	56	286	2	1.000 ±0.000	1.000 ±0.000	1.000 ±0.000
FreezerRegularTrain	2995	301	2	0.999 ±0.001	0.996 ±0.001	0.993 ±0.001
ItalyPowerDemand	1096	24	2	0.961 ±0.003	0.957 ±0.005	0.961 ±0.004
MixedShapesRegularTrain	2919	1024	5	0.973 ±0.003	0.946 ±0.002	0.913 ±0.006
SwedishLeaf	1122	128	15	0.965 ±0.002	0.970 ±0.002	0.944 ±0.022
SyntheticControl	600	60	6	1.000 ±0.002	0.990 ±0.003	0.973 ±0.009
Trace	200	275	4	1.000 ±0.000	1.000 ±0.000	1.000 ±0.000
TwoPatterns	5000	128	4	1.000 ±0.000	1.000 ±0.000	1.000 ±0.000
Wafer	7164	152	2	0.996 ±0.001	0.995 ±0.001	0.994 ±0.001

B.2 Local Baselines

The eight local baselines of Section 5.2 are time-series methods already, and each runs from the code released by its authors, with their defaults. Where a paper and its code disagree on a hyperparameter, we follow the code, since that is what their own experiments ran. Only three things are supplied from our side. *Classifier:* every method explains our trained classifier, never one it trains for itself; methods written for Keras receive a Keras copy of it with the trained weights transferred, not retrained, and checked against the original before any run. *Data:* every method draws its queries, and any reference or nearest-neighbour set it needs, from the same z-normalized population, labelled with the predictions of the classifier. *Scoring:* the edited series each method returns is re-scored by our own measurement code against the declared target $t(\boldsymbol{x})$, so no number depends on the internal reporting of a method. The only code changes are those needed for the released code to run at all under current library versions, none of which touches the search.

NATIVE GUIDE retrieves the nearest unlike neighbour of the query by dynamic time warping, then inserts a window of that neighbour into the query, centred on the region of highest class activation, and grows the window one timestep at a time until the prediction flips.

SUB-SPACE evolves a binary mask selecting which subsequences of the query to replace with the values of its nearest unlike neighbour, under a genetic algorithm seeded by Grad-CAM++ whose fitness trades validity against sparsity, the number of subsequences, and an autoencoder outlier score. Its authors tune the autoencoder per dataset and publish five configurations; we use those verbatim where a dataset overlaps, and elsewhere set the latent size to $0.42\,T$, the median latent-to-length ratio of those five configurations, which reproduces their published Coffee configuration exactly.

MULTI-SPACE is the multi-objective successor of SUB-SPACE: an NSGA-II search over the same subsequence-replacement masks, optimizing validity, sparsity, plausibility and subsequence count jointly. As in its own evaluation, the final Pareto front is reduced to a single counterfactual by its scalarized utility.

GLACIER runs gradient descent in the latent space of an autoencoder, minimizing a prediction-margin loss towards the target plus a step-weight penalty derived from local importance. Its reference script explains a classifier it trains itself; we pass ours in its place, and train its autoencoder with its own architecture and recipe on our data.

TSEVO is a multi-objective evolutionary search (NSGA-II) over perturbations drawn from instances of the opposing class, with mutation operators on authentic values, frequency bands and means.

CELS learns a per-timestep saliency map by gradient descent that blends the query with its nearest neighbour of the runner-up class, penalizing the target loss, the size of the map and its total variation, and then thresholds the map into a binary mask that replaces the selected timesteps outright.

MASCOTS is gradient-free: it maps the population to a bag of symbolic receptive fields, fits a small surrogate network to the predicted probabilities of the classifier, explains the surrogate with

DeepSHAP, and repeatedly shifts the symbolic pattern most supporting the current class towards the one most opposing it, until the prediction changes.

FASTPACE treats generation as planning over a binary edit mask: it starts from the full substitution of the nearest unlike neighbour, which is valid by construction, and a hierarchical cross-entropy planner removes substituted blocks, coarse to fine, while keeping the best valid candidate seen, under a reward that weighs validity, proximity, plausibility and sparsity.

TERCE (Bahri et al., 2022b) was also considered and excluded. It builds its counterfactuals from temporal rules mined by RULETRANSFORM (Bahri et al., 2022a), which pairs shapelets (Ye & Keogh, 2009) across channels; on univariate series both channels collapse to one and the enumeration is no longer pruned, reaching $115{,}584$ rules and 1.36 TiB on a single class of Coffee.

We summarize every pool, from the local baselines, GALACTIC-L, GALACTIC-G and every adapted global baseline, under the same budget $\mu = 4$, the value GLANCE (Kavouras et al., 2026) itself uses for its own number of global actions. Section C.1.1 varies μ from 1 to 8; the results move little beyond the first few values.

B.3 GLOBAL BASELINES

None of the six global baselines of Section 5.3 has a time-series formulation, so each runs in an adaptation we wrote, marked $\star$. Every adaptation starts from the released code of the method, never from a reimplementation of its paper, and changes the same three things. *Serialization:* each series becomes one tabular row with one continuous feature per timestep, so the categorical, ordinal and immutable-feature machinery of each method has nothing to act on and is dropped. *Classifier:* each method calls the batch prediction of our classifier directly, with no positive-class assumption. *Multi-class:* all six were written for binary recourse; the affected population becomes the correctly classified members of the source class, and success means reaching the declared target, as everywhere in the paper. Every method receives the same budget of $\mu = 4$ actions per transition (Section B.2), and the edited series it returns are re-scored by our own measurement code. The adaptations specific to one method are stated with it below.

GLOBE-CE samples random global translation directions, finds by bisection the smallest scaling of each that flips each instance, and greedily keeps a small diverse set of directions; we keep μ directions per source class.

GLANCE clusters the affected population, generates candidate counterfactuals for each cluster centroid, merges clusters whose centroids and counterfactuals are jointly closest until μ remain, and gives each cluster the candidate action that flips most of its members. Its default candidate generator is built around one-hot tabular features, so we use random sampling, the one of its four generators that needs only predictions.

FAST ARES bins every feature, mines frequent itemsets with apriori, and searches for a small two-level rule set of the form "if subgroup, then change these features". The itemset search is exponential in the number of simultaneously frequent bins; a series of hundreds of timesteps is far beyond the 23 features on which it has been run before, so we cap the search at the 9999 s that GLOBE-CE uses to declare it timed out.

CET partitions the population with a decision tree and assigns one action to each leaf, found by a mixed-integer program. That program is defined only for logistic regression, random forests and multilayer perceptrons; for any other classifier its authors fit a local linear surrogate (LIME), which is the path used here. It calls CPLEX, and where CPLEX is unavailable falls back to the HiGHS solver in SciPy, an exact solve of the same program.

T-CREX grows a surrogate tree on the predictions of the classifier, turns its nodes into candidate rules, keeps the rules that reliably reach another class, and assigns each instance its rule of lowest cost. Its rules describe regions rather than points, so to score an edited series we use its own prototype, the mean of the real series the rule contains, for the timesteps the rule constrains.

FACEGROUP builds a density-weighted feasibility graph over the data and states two group problems: a cost-constrained one that maximizes coverage under a cost limit, and a coverage-constrained one that minimizes the worst cost needed to reach a coverage level. Both take the number of selected points k as input, and both receive the same μ every other method does; these are the two FACE-

GROUP⋆ rows of Table 2. For the coverage-constrained problem we use its greedy heuristic, which its paper adopts, rather than its Gurobi program.

B.4 COMPUTE ENVIRONMENT

Every job, including classifier training, counterfactual generation and selection, was executed on the MareNostrum 5 accelerated partition, one NVIDIA H100 GPU and 20 CPU cores per job, so that the runtime of every method is measured on the same hardware class.

Each (dataset, method) pair is given a wall-clock budget of 7200 s, shared across all source classes of that dataset, including our own. Our method is terminated at the budget. The baselines check the elapsed time inside their own code and stop at the first check past the budget, so a method with long individual steps, such as CET⋆ and FACEGROUP⋆, can report a runtime above it. Every method is scored on everything it returned. A method that exhausts its budget returns the results completed so far and is recorded as a partial result; the population it did not reach is counted as failure rather than excluded (Section B.9).

B.5 SEGMENTATION AND IMPORTANCE WEIGHTING

Let $P_I^{(b)}\boldsymbol{x}$ replace interval I of $\boldsymbol{x}$ with that interval from a random member of the same population. We compute signed confidence drops for the source subgroup and target class,

$$s_{\mathrm{src}}(I) = \mathbb{E}_{\boldsymbol{x}\in G_{y,r}}\left[f_{\theta,y}(\boldsymbol{x}) - \frac{1}{B}\sum_{b=1}^{B} f_{\theta,y}(P_I^{(b)}\boldsymbol{x})\right],$$

and analogously $s_{\mathrm{tgt}}(I)$ using $f_{\theta,t}$ over X_t. Direction-specific evidence uses the gradient of the target margin, restricted to the interval and averaged over the transition subgroup:

$$s_{y\to t}(I) = \mathbb{E}_{\boldsymbol{x}\in G_{y,r\to t}}\left[\frac{\|\nabla_{\boldsymbol{x}} m_t(\boldsymbol{x})_I\|_2}{\sqrt{|I|}}\right].$$

Positive scores identify intervals whose permutation weakens the class evidence or on which the target margin is sensitive; nonpositive scores are not admitted. Aligning all source and target breakpoints yields a set of aligned intervals $\mathcal{I}$, and each interval receives the maximum covering score among s_{src}, s_{tgt}, and $s_{y\to t}$, giving the aligned score s used below.

The five strategies of Section 4 are then:

- **Accuracy-budgeted**: $\boldsymbol{w} = \mathbb{1}[\,\mathrm{imp}(I) > q_{0.75}\,]$, where $\mathrm{imp}(I) = |a_y - \frac{1}{B}\sum_b a_{y,I}^{(b)}|$ is the permutation drop in subgroup accuracy a_y and $q_{0.75}$ is a fixed global quantile. This inherits the standard permutation-importance threshold and ignores the intended target.
- **Source-confidence**: $\boldsymbol{w} = \mathbb{1}[s_{\mathrm{src}} > 0]$, admitting only regions that support the current class y.
- **Target-confidence**: $\boldsymbol{w} = \mathbb{1}[s_{\mathrm{tgt}} > 0]$, admitting only regions required to reach t.
- **Transition-combined**: $\boldsymbol{w} = \mathbb{1}[s > 0]$ over the aligned intervals, so an interval is editable when it carries source, target, or transition evidence.
- **Doubling-support** (default): intervals are sorted by s and admitted whole while the cumulative admitted fraction stays below the coverage ceiling $\rho < 1$, giving a support that is never unrestricted even when scores tie. The schedule is computed once per (subgroup, target) pair and shared by its members. After optimization, intervals are reset one at a time in increasing order of change and each removal is kept only if the classifier still predicts the target class, which reduces the support while preserving validity.

Unconstrained ($\boldsymbol{w} = \mathbb{1}$) is reported only to show that the target is reachable when the edit is confined to no support at all, not as a candidate strategy.

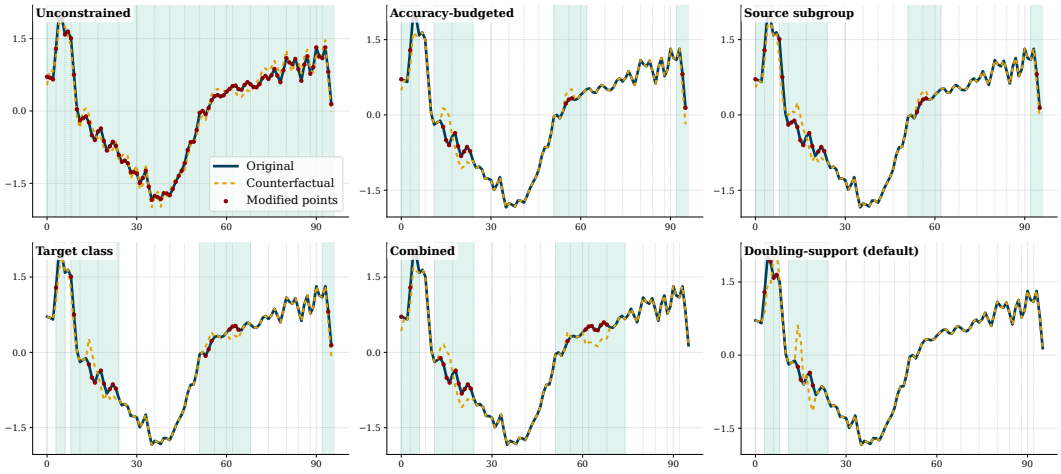


Figure 4: The counterfactual of one illustrative instance under each masking strategy of Section 4. Dotted lines mark the segments of the default strategy, used for every panel; a segment is shaded when the counterfactual changes at least one of its timesteps, and points mark the changed timesteps. *Doubling-support* is the default used throughout; *Unconstrained* is included only to show the target is reachable with no locality constraint, not as a candidate strategy, and edits nearly the whole series.

B.6 Local Search

Phase 1 (validity). The Adam optimizer uses step size 0.01, a convergence tolerance of 0.0001 with patience 2, a stagnation break after 40 non-improving steps, and a cap of 400 iterations, with loss weights $\lambda_1 = 0.2$ and $\lambda_2 = 1$ and $\zeta = 10^{-8}$. The coverage ceiling ρ admits at most 75% of timesteps into the mask, and segmentation is seeded with an initial-segment fraction of 0.5, with each structural segment split into 4 sub-segments. Importance is averaged over 10 repetitions. The transition score $s_{y \to t}$ is the gradient-magnitude signal of Section B.5, the same signal phase 2 uses to expand a support, and the direction term of the loss has weight $\lambda_3 = 0.5$. Source, target, and transition scores are combined on aligned intervals, and the default continuation mask admits the highest-scoring intervals up to the 75% ceiling, then applies whole-segment pruning that keeps the target class. Both phases read the same subgroups of Section 4, so they agree on subgroup membership and segment boundaries rather than each deriving its own. Those subgroups cluster the concatenation of the standardized segment-gap vector and the standardized target-margin gradient pooled into 20 equal-width temporal bins, with the two blocks equally weighted; K is chosen by silhouette score over $K \in \{2, \ldots, 6\}$, capped by the number of targets reachable from that source class.

Phase 2 (consistency). The reference margin γ_t is the 50th percentile (median) of the margin distribution of the correctly classified series of the target class, shared by every subgroup of the transition, so it is read off the data rather than tuned. The constrained solve runs for at most 800 iterations and stops early once the median margin of the series being solved has not improved for 100 consecutive iterations; beyond that point the margin is flat while the perturbation norm continues to grow, so stopping there is a cost reduction rather than an approximation. A total-variation penalty of 0.5 inside the solve keeps the resulting edits contiguous, and support pruning then shrinks each perturbation back toward validity while retaining 80% of the achieved margin. Each instance keeps its own perturbation throughout; the subgroup enters only through the shared γ_t and the shared schedule of candidate supports. Where the best single perturbation of a transition covers less than 90% of its own candidate series, the reference is raised to the 75th, 90th or 99th percentile, and the higher value is kept only if it raises that coverage. Across the ten datasets under ResNet, all 6421 explained series reach the target class, 89.5% end with $m_t \geq 0.8\gamma_t$, and phase 2 leaves 9.9% at their phase-1 solution.

Both phases run in a single process as one invocation of the method: phase 2 is not a post-processing script applied to the output of phase 1, but the second half of one solver for Problem 1. The search

of TSEVO is capped at a fixed generation budget shared across all datasets, chosen to keep runtime tractable across the benchmark without changing its validity.

A small number of series are not flipped by the masked search; those get one further attempt with the whole series editable and a bare-argmax stopping rule (Section C.1.3 reports how many, how large the resulting edits are, and their effect on the results).

B.6.1 THE TWO PHASES ON ONE INSTANCE

Appendix Figure 5 contrasts the two phases of Problem 1 on a single illustrative instance under the default strategy, with every segment that each phase edits shaded. Phase 1 keeps the lowest-loss accepted iterate and then removes every segment it can without losing the target, so it edits a small number of timesteps inside that support and leaves the counterfactual near the decision boundary. Phase 2 drives the same instance toward the reference margin γ_t, which takes both a larger amplitude and, where the initial support cannot reach γ_t, additional segments admitted by the expansion step. The extra edit extent visible between the panels is not incidental: it is precisely the *Act* cost reported in Section 5.2, and it is what allows the perturbation to carry over to a structural neighbour.

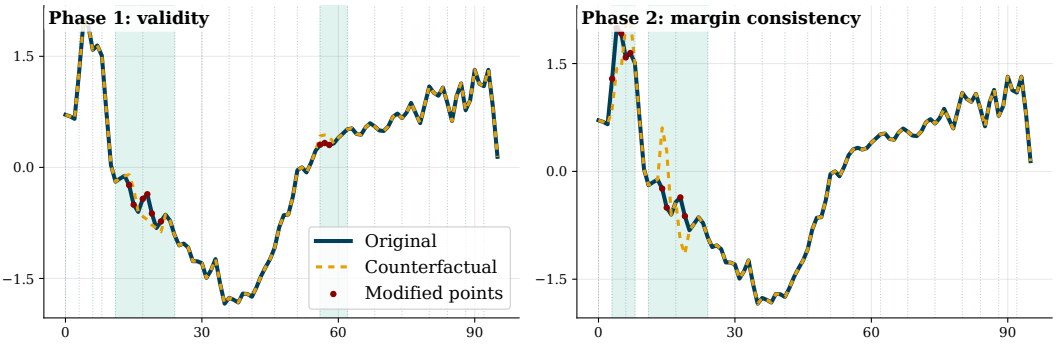


Figure 5: The two phases on the same illustrative instance under the default strategy. Shaded segments are the segments that phase edits; within them, only the marked points change.

B.6.2 EXPLANATIONS AT FOUR LEVELS

Appendix Figure 6 shows, for one directed transition of one evaluation dataset under ResNet, the four levels at which GALACTIC explains a classifier: the counterfactual of one series, the perturbations of the members of its structural subgroup, the perturbations GALACTIC-G selects for the transition with the number of series each covers, and the coverage of every directed transition of the classifier.

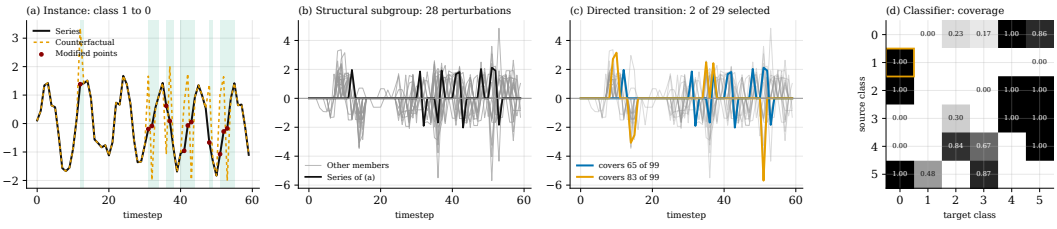


Figure 6: Four levels of one explanation: (a) a series, its counterfactual and the changed timesteps, (b) the perturbations of its structural subgroup, that of the series of (a) in black, (c) the perturbations selected for the transition and the series each covers, and (d) the coverage of the perturbations selected for each directed transition, each transition with its own selection; a cell at 0.00 is a transition whose few series, at most three here, receive at most one candidate, and an empty cell is a transition no series takes; the outlined cell is the transition of (a) to (c).

B.7 GALACTIC-G: FLAT SELECTION ALGORITHMS

Algorithm 2 in Section 4 calls a subroutine SELECTMDL(Δ, X, μ, A) that minimizes the objective of Problem 2 over a candidate pool Δ and population X under budget μ, using base algorithm $A \in \{\text{OPTIMAL}, \text{GREEDY}\}$; the hierarchical variants apply it once per subgroup and once more over the union of subgroup outputs, while the flat variants of Section 4 apply it directly to $(\Delta_{y\to t}, X_{y\to t}, \mu)$. Appendix Algorithms 3 and 4 give the two base algorithms.

Every experiment in the main paper uses Flat Optimal as the default GALACTIC-G selector; Section B.8 gives the exact solver behind it.

Algorithm 3 GALACTIC-G: Optimal

Input: Candidate pool Δ, population X, budget μ
Output: Selected perturbation set S^*

procedure OPTIMAL(Δ, X, μ)
  return $\arg\min_{S\subseteq\Delta,\,|S|\le\mu} L(S) + L(X \mid S)$ ▷ Exhaustive search over the power set, feasible for small $|\Delta|$
end procedure

Algorithm 4 GALACTIC-G: Greedy

Input: Candidate pool Δ, population X, budget μ
Output: Selected perturbation set S

procedure GREEDY(Δ, X, μ)
  $S \leftarrow \emptyset$
  while $|S| < \mu$ **do**
    $\boldsymbol{\delta}^* \leftarrow \arg\min_{\boldsymbol{\delta}\in\Delta\setminus S}\big(L(S\cup\{\boldsymbol{\delta}\}) + L(X \mid S\cup\{\boldsymbol{\delta}\})\big)$
    if $L(S\cup\{\boldsymbol{\delta}^*\}) + L(X \mid S\cup\{\boldsymbol{\delta}^*\}) > L(S) + L(X \mid S)$ **then**
      break ▷ Stop once the best addition would increase the joint description length
    end if
    $S \leftarrow S \cup \{\boldsymbol{\delta}^*\}$
  end while
  return S
end procedure

Under Assumption 1, the marginal MDL reduction is monotone submodular (Section A), so Appendix Algorithm 4 attains the standard $(1 - 1/e)$ approximation to Appendix Algorithm 3 (Nemhauser et al., 1978; Hochba, 1997). Otherwise Appendix Algorithm 4 is a fast heuristic that stops when no candidate decreases the objective.

B.8 EXACT SELECTION AS A MIXED-INTEGER PROGRAM

OPTIMAL selection minimizes the MDL cost over every subset of size at most μ. Exhaustive enumeration costs $\sum_{k\le\mu}\binom{n}{k}$ and is used only while that count stays within a fixed cap; beyond it the identical objective is solved as a mixed-integer linear program, so the selector never silently degrades to an approximation because a pool grew.

Let $z_{\boldsymbol{\delta}} \in \{0,1\}$ indicate that candidate $\boldsymbol{\delta}$ is selected and $u_g \in \{0,1\}$ that population group g is covered. Writing $c_{\boldsymbol{\delta}}$ for the code cost of a candidate, p_{sz} for the pointer width, M for the outlier cost and n_g for the size of group g, the program is

$$\min_{z,u} \sum_{\boldsymbol{\delta}} (c_{\boldsymbol{\delta}} + p_{sz})\, z_{\boldsymbol{\delta}} \;+\; \sum_{g} n_g\,(p_{sz} - M)\, u_g \quad \text{s.t.} \quad \sum_{\boldsymbol{\delta}} z_{\boldsymbol{\delta}} \le \mu, \qquad u_g \;\le \sum_{\boldsymbol{\delta}\ \text{covers}\ g} z_{\boldsymbol{\delta}},$$

with the constant $\sum_g n_g M$ dropped from the objective. The coverage rows allow $u_g = 1$ only when a selected candidate covers g. Since $M > p_{sz}$, the coefficient of u_g is negative and every optimal solution sets $u_g = 1$ for each covered group, so a covered group is charged $n_g p_{sz}$ and an uncovered one $n_g M$: exactly the pointer and outlier charges of the MDL cost. The cardinality row is the budget.

The program is solved by the HiGHS mixed-integer solver, a branch-and-cut method: branch and bound strengthened by cutting planes. Cuts tighten the relaxation and never remove a feasible integer solution, so the method is exact, not a heuristic. We run it with a relative optimality gap of zero (the absolute gap keeps the solver default of 10^{-6} bits) and a time limit of 600 s. If the limit is reached before optimality is proven, the run fails with an error rather than returning the best incumbent, so no reported selection is an unproven approximation; none of our runs hit the limit.

Two reductions are applied first, and neither can change the optimum. Candidates with identical coverage patterns are interchangeable, so only the candidate of lowest code cost in each group is retained; and population members covered by exactly the same candidates are always covered or uncovered together, so they are grouped and weighted by n_g rather than represented individually.

B.9 Evaluation Measurements

Five quantities are reported: ***Eff***, ***Afc***, ***Act***, ***MDL*** cost per series and ***RT***.

Effectiveness (***Eff***) is the fraction of explained series that reach the target class (Section 3). Average flipping cost (***Afc***) is the Euclidean distance of the successful counterfactual divided by the per-timestep value range of the dataset, $\|\max_i \boldsymbol{x}_i - \min_i \boldsymbol{x}_i\|_2$, clipped to $[0, 1]$. Average changed timesteps (***Act***) is the edited-timestep fraction. Description length (***MDL***) is the total cost of encoding the selected perturbation set plus the population it fails to cover, divided by the population size so it stays comparable across datasets of different sizes rather than growing with the population. Runtime (***RT***) is wall-clock on the hardware of Section B.4. A method marked $^{n+}$ reached its time budget on n datasets before reaching every source class; the classes it reached are scored as usual and the rest count as failures, and a run that reached the budget before completing any source class returned no result. Six (dataset, method) pairs exceed the host memory of a job, 160 GB and still at 320 GB: both FACEGroup$\star$ objectives on MixedShapesRegularTrain, and Fast AReS$\star$ on MixedShapesRegularTrain, Trace, FreezerRegularTrain and Wafer. They are excluded under every architecture, not imputed, and marked ^{n†}. Every mean is over the datasets on which a method returned a result, and #DS is that denominator. Win/draw/loss compares our method on each dataset with the best competitor on that metric, after rounding both to three decimals; equal values are a draw. A dataset counts when our method and at least one competitor have a result on it, and as a win when every competitor falls below the effectiveness floor. That floor excludes from the best competitor any method scoring below 25% effectiveness on that dataset, since a method that explains almost nothing of the population could otherwise register a win on cost or runtime: a method that flips fewer than a quarter of the series is measured on cost, edit extent and runtime over those few, typically the easiest, so without the floor the W/D/L rows would compare it with methods that explain most of the population. Appendix Table 5 repeats the W/D/L row of Table 2 at floors of 0, 10, 25 and 50%: description length and effectiveness are unchanged, while at low floors the flipping cost, edit extent and runtime are decided by methods that explain almost none of the population.

Table 5: Win/Draw/Lose row of Table 2 under four effectiveness floors: a competitor below the floor on a dataset is excluded from the best competitor there; 25% is the default.

Floor	Avg. *MDL*/series	Avg. *Eff*	*Afc*	*Act*	Avg. *RT*
0%	9/0/1	4/2/4	1/0/9	1/0/9	0/0/10
10%	9/0/1	4/2/4	2/0/8	1/0/9	3/0/7
25% (default)	9/0/1	4/2/4	3/0/7	2/0/8	3/0/7
50%	9/0/1	4/2/4	5/0/5	5/0/5	3/0/7

The default target is the highest-scoring class other than the correct predicted class. A local run succeeds only when the classifier predicts the target class.

C Additional Experiments

This section reports the experiments and analyses that support the main text: the sensitivity of the results to the action budget (Section C.1.1), to the pointer width (Section C.1.2), and to the second attempt for unflipped series (Section C.1.3); the choice of seeded series (Section C.1.4); held-out

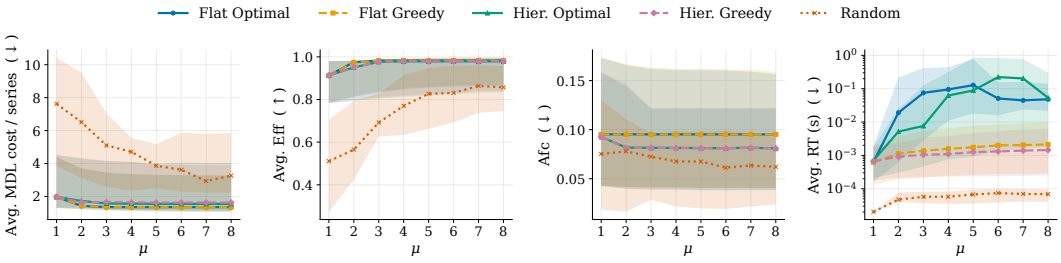


Figure 7: Sensitivity to the action budget μ (Section C.1.1): Avg. *MDL*/series, Avg. *Eff*, *Afc* and Avg. *RT* (log-scaled) for all five GALACTIC-G selectors, $\mu = 1$ to 8, on all ten datasets; *Eff* is higher-is-better, the other three lower-is-better. Faint lines are individual datasets, bold lines their mean; colour identifies the selector; solid lines are the exact selectors, dashed the greedy ones.

transfer (Section C.1.5); the Pareto analysis with the local-under-global summary (Section C.2); and the complete per-dataset comparison tables (Section C.3).

C.1 HYPERPARAMETER SENSITIVITY

Every ablation in this subsection varies one hyperparameter at a time, with every other setting at its default.

C.1.1 ACTION BUDGET

Every experiment in the main paper fixes the per-transition action budget at $\mu = 4$; Appendix Figure 7 varies μ from 1 to 8.

The budget μ does not need to be tuned to its exact value: it is only an upper bound on how many actions a transition may use, and the MDL objective already charges each selected perturbation its own code cost, so it stops adding actions once a further one no longer reduces the description length, regardless of how much higher μ is set. At $\mu = 4$ the exact selector already picks a mean of only 1.88 actions, and only 25% of transitions use the whole budget; raising μ to 6 or 8 leaves this mean nearly unchanged, because the objective, not the cap, is what decides how many actions are used. Random, which spends the same budget without any selection rule, stays far worse on both MDL cost and effectiveness at every μ, confirming that the other four variants owe their advantage to what they select rather than to how many actions they are allowed.

Runtime is the panel that separates the variants, and it does so by three orders of magnitude rather than by growth rate. At $\mu = 1$ every selector is negligibly cheap (≈ 3 ms), because a single action admits no combinatorial choice. From $\mu = 2$ the exact selectors incur the cost of branch-and-cut: Flat Optimal rises to 3.3 s and Hierarchical Optimal to 0.29 s, while the greedy variants grow linearly and stay near ten milliseconds throughout, ending at 13 ms and 12 ms at $\mu = 8$. This gap, and not the shape of either curve, is why the greedy variants are preferable on large transitions; it is the cost of the exact solution relative to greedy selection, whose $(1 - 1/e)$ bound holds only under Assumption 1.

What the exact selectors do *not* do is blow up. Flat Optimal peaks near $\mu = 3$ and then stops rising: its mean is flat from there to $\mu = 8$ (6.1, 5.8, 5.8, 5.6, 4.5, 5.6 s) while its median stays below 0.12 s, and Hierarchical Optimal grows more slowly, to 2.3 s at $\mu = 8$, still well below the flat exact selector. This is the same self-limiting behaviour the first three panels show, observed in the solver rather than in the objective: once μ exceeds the number of actions the MDL objective wants, the budget constraint stops binding, the relaxation at the root node is already near-integral, and the search terminates almost immediately. A budget cap set well above the operating point therefore costs little even under exact selection, so μ need not be tuned tightly in either direction.

C.1.2 POINTER WIDTH

Each covered series is charged p_{sz} bits to point to the one perturbation, among the selected set, that covers it; the main experiments fix $p_{sz} = 1$, the width of an exact index into a summary of two

perturbations, close to the 1.88 the exact selector chooses on average at $\mu = 4$ (Section C.1.1). A larger width raises this cost in proportion to how many series are covered, so Appendix Table 6 repeats the selection of Section 5.2 at three widths: $p_{sz} = 1$, the default; $p_{sz} = 2$, enough to index any summary of at most $\mu = 4$ perturbations; and $p_{sz} = 64$, the width of a machine address, an upper bound well beyond what indexing the selection itself requires. Selection and scoring both use the width under test, and the outlier cost ($M = \max_{\boldsymbol{\delta}} c_{\boldsymbol{\delta}} + 2p_{sz}$, Section A) is rebuilt for each width in turn.

Table 6: Pointer-width sensitivity (Section C.1.2): Avg. *Eff* and Avg. *MDL*/series of the pool of every source, selected by Flat Optimal at $\mu = 4$ with pointer widths p_{sz} of 1 (default), 2 and 64 bits, averaged over the ten datasets; *Eff* is shown once because the width changes it by at most 0.001.

		Mdl/series		
Source	Eff	$p_{sz} = 1$	$p_{sz} = 2$	$p_{sz} = 64$
GALACTIC-L	0.924	2.29	3.40	72.17
CELS	0.793	3.95	5.18	81.48
FASTPACE	0.739	4.68	5.97	85.48
MASCOTS	0.721	4.90	6.19	86.52
SUB-SPACE	0.710	5.05	6.36	87.66
MULTI-SPACE	0.668	5.79	7.15	91.09
NATIVE GUIDE	0.661	5.91	7.27	91.68
TSEVO	0.345	10.58	12.26	116.02
GLACIER	0.355	10.70	12.37	116.19
GALACTIC-L lowest on		7/10	7/10	7/10

The order of the nine sources by description length is the same at every width, effectiveness changes by at most 0.001, and GALACTIC-L has the lowest description length on seven of the ten datasets at every width. Description length grows with the width because every covered series incurs it, and the gap between GALACTIC-L and the next source widens, from 1.66 bits per series at $p_{sz} = 1$ to 1.78 at 2 and 9.31 at 64, since GALACTIC-L covers the most series.

C.1.3 SECOND ATTEMPT FOR UNFLIPPED SERIES

When no admitted support flips a series under the $\kappa = 0.6$ stopping rule, the search is retried once with the whole series editable and $\kappa = 0$ (Algorithm 1); the backward elimination of Section 4 then still applies to this retried edit exactly as to any other, resetting every admitted segment whose removal from the explanation still leaves the classifier predicting the target class, so the retry starts from the whole series but is pruned back down like every other candidate. The success check still requires the prediction of the classifier to equal the declared target class, so this step changes which series are solved, never how success is judged. This affects 23 of 6421 explained series (0.36%), on TwoPatterns, Wafer, ItalyPowerDemand and SyntheticControl only. After pruning, the edits on these series cover 28% of the timesteps on average against 14% for the rest, still well short of the whole series they started from. Appendix Table 7 shows the effect: adding the second attempt raises effectiveness from 0.997 to 1.000 at almost no cost in flipping cost or edited fraction. At 0.36% of series, this step is too small a share of any pool to change what a reader takes away from the actions selected by GALACTIC-G.

Table 7: GALACTIC-L under three settings of κ and the second attempt (Section C.1.3); *Eff* is higher-is-better, *Afc* and *Act* lower-is-better. The default is $\kappa = 0.6$ with the second attempt.

Setting	Eff	Afc	Act
$\kappa = 0.9$, no second attempt	0.987	0.0719	17.7%
$\kappa = 0.6$, no second attempt	0.997	0.0744	17.9%
$\kappa = 0.6$ with second attempt (default)	1.000	0.0748	18.1%

C.1.4 WHICH SERIES TO PERTURB

GALACTIC-L perturbs only a subset of each class, so the pool the selector receives depends on which series are perturbed. Appendix Figure 8 compares three choices; MMD-Critic (Kim et al., 2016) picks series that represent the class well together with series that the class represents poorly.

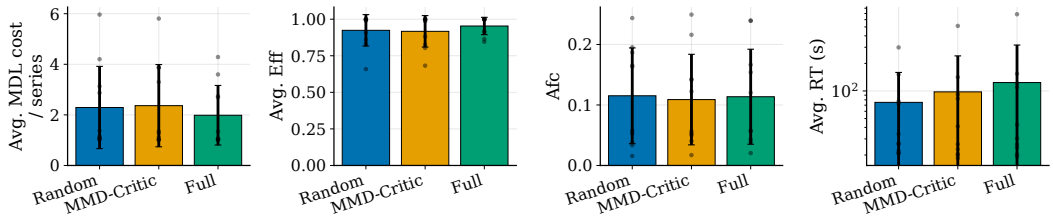


Figure 8: Which series to perturb: Avg. *MDL*/series, Avg. *Eff*, *Afc* and Avg. *RT* (log-scaled) when GALACTIC-L perturbs a random 30% of each class (default), the same number chosen by MMD-Critic, or every series, each selected by Flat Optimal at $\mu = 4$ and scored on the same population. *Eff* is higher-is-better, the other three lower-is-better. Bars are the cross-dataset mean, error bars the standard deviation, and points each dataset.

MMD-Critic does not differ from random on effectiveness (0.917 against 0.924) and yields a longer description than random on three of the ten datasets, at a higher runtime (98 s against 75 s) once its own seed-selection cost is counted. Perturbing every series improves effectiveness to 0.953 and shortens the description to 1.99 bits per series on nearly every dataset, at over 1.6× the selection runtime (124 s against 75 s), with the largest gains where the 30% strategies are least effective (SwedishLeaf: 5.96 to 4.29 bits). The choice of seed series is therefore a trade-off between compute and quality, and a selection criterion such as MMD-Critic does not improve on random sampling.

Three controls keep the comparison about the pool and nothing else. Each pool is built independently inside each structural subgroup $G_{y,r\to t}$, so no candidate crosses a source, target, or subgroup boundary and the three strategies differ only in which instances within a subgroup are perturbed. Every pool is then handed to the same downstream selector and scored against the same evaluation population, so a difference in the figure is a difference in candidate quality rather than in how the candidates were used.

C.1.5 HELD-OUT TRANSFER

For a subgroup G, GALACTIC-L builds the schedule of candidate supports from every series in G, including the series later scored. This raises a fair question: could the coverage of another series by a candidate in the same subgroup come from that shared structure, rather than the perturbation genuinely transferring? A weaker version of the same question applies to every source, not only ours: the main comparison of Section 5.2 scores every source on a population that includes the series its own candidates were built from, and a candidate trivially covers the series it was built for.

To test this, Appendix Table 8 re-scores the same cached candidate pools of all nine sources, without regenerating any candidate, under three increasingly strict protocols. *Full* is the protocol every other table in this paper uses: every series of the transition is scored. *Excl.* removes, from every transition, every series that generated a candidate for any source, so every source is scored on one shared population that built none of the candidates. *Held-out* goes further, testing the specific worry above directly: this remaining population is split at random into two halves, the selector picks its at most $\mu = 4$ actions using only the first half, and coverage and description length are computed on the second half alone, so no series informs both the choice of actions and their score. This split is repeated for three seeds, 42, 43 and 44, the same three used to train the classifiers, and the reported *Held-out* numbers are the average of the three splits; *Full* and *Excl.* involve no random split and are each a single deterministic run, so they have no seeds to average over. For GALACTIC-L specifically, γ_t and the schedule are built from the first half only, so nothing about the second half can have contributed to its own candidates. Transitions left with fewer than four series after removing the generating series are dropped from all three columns for comparability, which is why *Full* here differs very slightly from the main comparison (they hold 0.5% of all series).

Table 8: Held-out transfer (Section C.1.5): Avg. *Eff* and Avg. *MDL*/series of the pool of every source, averaged over the ten datasets, under three protocols: *Full* scores every series of the transition; *Excl.* removes every series that generated a candidate for any source; *Held-out* also selects on one random half and scores on the other, over three seeds.

	Avg. *Eff*			Avg. *MDL*/series		
Source	Full	Excl.	Held-out	Full	Excl.	Held-out
GALACTIC-L	0.926	0.910	0.893	2.24	2.45	2.89
CELS	0.797	0.785	0.758	3.93	4.08	4.60
FASTPACE	0.743	0.737	0.713	4.66	4.78	5.28
MASCOTS	0.725	0.721	0.708	4.88	4.98	5.30
SUB-SPACE	0.715	0.707	0.673	5.02	5.14	5.77
MULTI-SPACE	0.672	0.664	0.640	5.77	5.89	6.35
NATIVE GUIDE	0.665	0.653	0.629	5.88	6.07	6.56
TSEVO	0.345	0.339	0.327	10.59	10.67	10.96
GLACIER	0.356	0.330	0.325	10.69	10.91	11.09
W/D/L	6/3/1	6/2/2	6/2/2	8/0/2	8/0/2	8/0/2

The lead of GALACTIC-L over the best other source, CELS, does not shrink under the strictest protocol: 0.135 in effectiveness under *Held-out* (0.893 against 0.758 for CELS) against 0.129 under *Full*, and a 1.71-bit description-length lead against 1.69 bits. Effectiveness does drop a little for GALACTIC-L under the stricter protocols, from 0.926 to 0.893, but every other source drops by a comparable or larger amount (0.017 to 0.042), and the ranking of all nine sources is unchanged throughout: this drop is a property of the stricter test itself (the selector sees only half the population when picking its actions, which can miss a perturbation that would have covered the other half well), not something specific to GALACTIC-L. Building γ_t and the schedule from the generating half only, rather than the whole class, also costs nothing: it reaches $\mathrm{Eff} = 0.893$, while building them from the whole class instead, which would let the held-out half leak into the shared structure, reaches only 0.884 and 2.99 bits per series. So the result is not an artifact of how the evaluation population overlaps with the candidates: the lead holds under the strictest of the three protocols.

C.2 PARETO STRUCTURE OF THE OBJECTIVE AXES

Appendix Table 9 gives the domination analysis referenced in Section 5.2, computed on the cross-dataset means of Appendix Table 10. Description length is excluded from the axes because it is the proposed scalarization *of* them, and including it would beg the question. The frontier does not rank the sources, which is why the selection objective must supply an exchange rate between the axes rather than leave the choice to the reader.

Table 9: Pareto structure of the local-under-global results (cross-dataset means of Appendix Table 10) on three axes: Avg. *Eff* (higher is better), *Afc* and *Act* (lower is better). A source is Pareto-optimal when no other source is at least as good on all three axes and strictly better on one; #Dominated by counts the sources that dominate it.

Source	Avg. Eff ↑	Afc ↓	Act ↓	#Dominated by	Pareto-optimal
GALACTIC-L	0.924	0.115	16.7	0	yes
CELS	0.793	0.089	9.8	0	yes
FASTPACE	0.727	0.122	26.9	2	no
GLACIER	0.387	0.092	100.0	1	no
MASCOTS	0.693	0.223	26.9	3	no
MULTI-SPACE	0.652	0.094	14.9	1	no
NATIVE GUIDE	0.648	0.139	34.2	5	no
SUB-SPACE	0.683	0.101	20.2	1	no
TSEVO	0.430	0.163	29.2	5	no

Only two of the nine sources are non-dominated, GALACTIC-L and CELS; each of the other seven is dominated by between one and five sources, and GALACTIC-L is dominated by none. The two non-

dominated sources sit at opposite ends of the frontier: GALACTIC-L has the highest effectiveness (0.924), and CELS edits the least of the series (9.8%) and has the lowest flipping cost (0.089) at lower effectiveness (0.793). Neither dominates the other, and description length is the scalarization that orders them, at 2.29 against 3.95 bits per series. This is the time-series form of the difficulty Kavouras et al. (2026) identify for tabular recourse.

Appendix Figure 9 shows the three axes together with description length. A 2-D projection would favour whichever two axes are chosen; in parallel coordinates, a line that crosses its neighbours between two axes is a source paying for one objective with the other. No line is above every other line everywhere, the same fact Appendix Table 9 states numerically: GALACTIC-L is highest on effectiveness and description length, and behind CELS on flipping cost and edit extent.

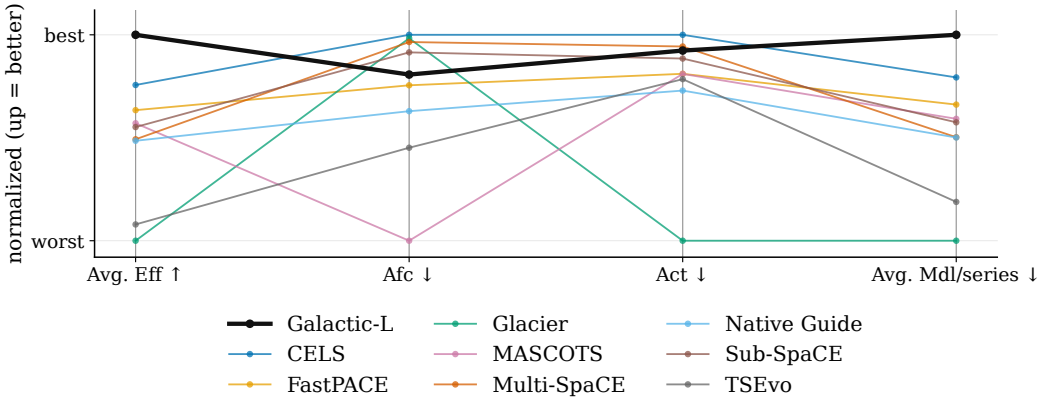


Figure 9: Parallel coordinates of Avg. *Eff*, *Afc*, *Act* and Avg. *MDL*/series for the pool of every local generator under global selection. Each axis is min–max normalized and oriented so that up is better; GALACTIC-L is drawn in black, and a crossing between two adjacent axes is a trade-off.

Why the pools differ. With the selector fixed, the pool is what varies. The per-instance objective of Section 3.1 asks only that each perturbation flip its own series, so the margin a pool attains is incidental rather than built for transfer, and such pools cover a small fraction of the population under the shared budget. The consistency phase of GALACTIC-L instead drives every series of a transition to the shared margin γ_t, and the pool of GALACTIC-L ranks first of the nine sources on both effectiveness (0.924 against 0.793 for the next source, CELS) and description length (2.29 against 3.95 bits per series).

What consistency costs. A perturbation that must reach γ_t is larger than the per-instance minimum for its series: the consistency phase raises its amplitude and, where needed, expands its support. GALACTIC-L therefore edits more of the series than CELS and MULTI-SPACE (16.7% against 9.8% and 14.9%), and this is the price of the coverage that the consistency phase adds to every pool (Table 1).

Table 10: Local generators under global selection (RQ2): the pool of each local generator selected by Flat Optimal at $\mu = 4$, averaged over the datasets in #DS. *Eff* is higher-is-better, the other metrics lower-is-better; $^{n+}$: time budget reached on n datasets. Last row: Win/Draw/Lose of GALACTIC-L vs. the best other source per dataset.

Source	#DS	Avg. *MDL*/series	Avg. *Eff*	*Afc*	*Act*	Avg. *RT*
GALACTIC-L	10	2.292	0.924	0.115	16.7%	229 s
CELS	10	3.951	0.793	0.089	9.8%	434 s
FASTPACE	10^{+}	5.008	0.727	0.122	26.9%	2262 s
GLACIER	9^{4+}	10.305	0.387	0.092	100.0%	3989 s
MASCOTS	10^{2+}	5.561	0.693	0.223	26.9%	3584 s
MULTI-SPACE	10^{2+}	6.278	0.652	0.094	14.9%	3913 s
NATIVE GUIDE	10^{2+}	6.289	0.648	0.139	34.2%	2998 s
SUB-SPACE	10^{2+}	5.697	0.683	0.101	20.2%	4188 s
TSEVO	7^{8+}	8.795	0.430	0.163	29.2%	6443 s
W/D/L		7/0/3	5/4/1	2/0/8	1/0/9	6/0/4

C.3 FULL COMPARISON TABLES

This section gives the complete measurements behind the summaries of Section 5 under ResNet: every dataset, every source and every metric.

Appendix Table 11 shows the measurements from the local baselines and GALACTIC-L. GALACTIC-L moves each series toward the reference margin γ_t of its target class, so its edits span several segments, and reaches Eff $= 1.000$, averaged over the ten datasets; the only two sources with both lower *Afc* and lower *Act*, CELS (0.071, 9.4%) and MULTI-SPACE (0.073, 14.2%), do so at Eff $= 0.668$ and 0.702, and at a higher *RT* (330 s and 3885 s against 154 s). Every other source is higher than GALACTIC-L on *Afc*, *Act* and *RT* as well as lower on *Eff*.

Appendix Table 12 shows the same sources scored under the Flat Optimal selector of GALACTIC-G instead of on their own untouched output (Section 5.2). GALACTIC-L reaches Eff $= 0.924$, averaged over the ten datasets; the same two sources that undercut it on *Afc* and *Act* at the local scope, CELS (0.089, 9.8%) and MULTI-SPACE (0.094, 14.9%), do so again here, at Eff $= 0.793$ and 0.652 and a higher *RT* (434 s and 3913 s against 229 s). GLACIER and SUB-SPACE have a lower *Afc* (0.092 and 0.101) but a higher *Act* (100.0% and 20.2%), and both a higher *RT* and a lower *Eff* (0.387 and 0.683). The remaining four, FASTPACE, MASCOTS, NATIVE GUIDE and TSEVO, are higher on *Afc*, *Act* and *RT*, and lower on *Eff*.

Appendix Table 13 shows the measurements from the global baselines and GALACTIC-G. GALACTIC-G reaches Eff $= 0.924$ at Afc $= 0.115$ and Act $= 16.7\%$, averaged over the ten datasets. GLANCE alone edits less of the series (Act $= 8.5\%$) at a comparable runtime (79 s against 73 s), but at a higher *Afc* (0.127) and a lower *Eff* (0.657). CET has a lower *Afc* (0.096) and the same *Act* (16.7%), at a much higher *RT* (7039 s) and a lower *Eff* (0.042). FAST ARES and T-CREX have both a lower *Afc* and a lower *Act* (0.011/2.7% and 0.033/2.0%) at a lower *RT* (48 s and 12 s), but at Eff $= 0.004$ and 0.101. The remaining three, FACEGROUP under both objectives and GLOBE-CE, are higher on *Afc*, *Act* and *RT*, and lower on *Eff*.

Table 11: Local counterfactual comparison (RQ2), per dataset: each local generator scored on its own counterfactuals, under ResNet.

Dataset	Method	*Eff*	*Afc*	*Act*	*RT*
Chinatown	GALACTIC-L	1.000	0.113	24.4%	14 s
	CELS	0.402	0.081	25.0%	20 s
	FASTPACE	1.000	0.098	38.2%	153 s
	GLACIER	0.505	0.156	100.0%	1072 s
	MASCOTS	0.822	0.274	18.9%	168 s
	MULTI-SPACE	1.000	0.095	35.8%	293 s
	NATIVE GUIDE	1.000	0.111	57.4%	96 s
	SUB-SPACE	1.000	0.097	37.4%	1024 s
	TSEVO	0.981	0.196	39.7%	6339 s

Dataset	Method	*Eff*	*Afc*	*Act*	*RT*
Coffee	GALACTIC-L	1.000	0.116	15.7%	11 s
	CELS	1.000	0.129	6.2%	14 s
	FASTPACE	1.000	0.172	23.4%	49 s
	GLACIER	0.611	0.275	100.0%	395 s
	MASCOTS	0.889	0.801	29.8%	83 s
	MULTI-SPACE	1.000	0.142	8.1%	299 s
	NATIVE GUIDE	1.000	0.176	21.2%	63 s
	SUB-SPACE	1.000	0.152	10.9%	212 s
	TSEVO	0.889	0.272	28.4%	2622 s
FreezerRegularTrain	GALACTIC-L	1.000	0.021	2.4%	97 s
	CELS	0.945	0.009	2.4%	596 s
	FASTPACE	1.000	0.011	9.7%	1595 s
	MASCOTS	0.581	0.112	27.8%	5897 s
	MULTI-SPACE	0.507	0.007	1.4%	7195 s
	NATIVE GUIDE	0.999	0.008	2.1%	5164 s
	SUB-SPACE	0.683	0.009	2.3%	7198 s
ItalyPowerDemand	GALACTIC-L	1.000	0.129	24.0%	25 s
	CELS	0.612	0.064	18.6%	59 s
	FASTPACE	0.997	0.091	30.7%	409 s
	GLACIER	0.281	0.067	100.0%	3544 s
	MASCOTS	0.808	0.235	43.3%	618 s
	MULTI-SPACE	1.000	0.089	26.8%	861 s
	NATIVE GUIDE	1.000	0.112	38.8%	295 s
	SUB-SPACE	1.000	0.092	27.9%	3577 s
	TSEVO$^+$	0.165	0.257	44.2%	7209 s
MixedShapesRegularTrain	GALACTIC-L	1.000	0.009	19.2%	618 s
	CELS	0.239	0.031	4.5%	820 s
	FASTPACE$^+$	0.335	0.064	32.2%	5731 s
	GLACIER$^+$	0.065	0.004	100.0%	6705 s
	MASCOTS$^+$	0.052	0.110	29.9%	7191 s
	MULTI-SPACE$^+$	0.040	0.021	5.1%	7220 s
	NATIVE GUIDE$^+$	0.058	0.129	41.2%	7196 s
	SUB-SPACE$^+$	0.115	0.090	51.4%	7212 s
SwedishLeaf	GALACTIC-L	1.000	0.029	22.4%	251 s
	CELS	0.540	0.042	9.0%	134 s
	FASTPACE	0.997	0.048	34.5%	3659 s
	GLACIER	0.840	0.025	100.0%	5150 s
	MASCOTS	0.276	0.103	17.1%	5582 s
	MULTI-SPACE	1.000	0.039	22.5%	6529 s
	NATIVE GUIDE	0.709	0.075	42.3%	1895 s
	SUB-SPACE	1.000	0.047	29.3%	5426 s
	TSEVO$^+$	0.061	0.097	46.2%	5753 s
SyntheticControl	GALACTIC-L	1.000	0.144	25.6%	109 s
	CELS	0.567	0.141	13.7%	55 s
	FASTPACE	1.000	0.121	24.3%	362 s
	GLACIER	0.178	0.094	100.0%	2454 s
	MASCOTS	0.400	0.132	38.7%	1154 s
	MULTI-SPACE	1.000	0.115	20.3%	892 s
	NATIVE GUIDE	0.922	0.144	31.6%	312 s
	SUB-SPACE	1.000	0.121	21.1%	1718 s
	TSEVO$^+$	0.294	0.202	47.0%	7180 s
Trace	GALACTIC-L	1.000	0.046	20.4%	44 s
	CELS	0.717	0.056	4.1%	41 s
	FASTPACE	1.000	0.065	13.8%	264 s
	GLACIER	0.400	0.036	100.0%	1371 s
	MASCOTS	0.433	0.124	17.3%	374 s
	MULTI-SPACE	1.000	0.064	8.4%	1164 s
	NATIVE GUIDE	0.917	0.095	24.4%	265 s
	SUB-SPACE	1.000	0.063	10.6%	830 s
	TSEVO$^+$	0.326	0.139	26.9%	6961 s
TwoPatterns	GALACTIC-L	1.000	0.128	17.1%	220 s
	CELS	0.751	0.120	8.3%	577 s
	FASTPACE	0.999	0.131	13.8%	4835 s
	GLACIER$^+$	0.006	0.078	100.0%	7106 s
	MASCOTS$^+$	0.085	n/a	47.0%	6501 s
	MULTI-SPACE$^+$	0.228	0.128	10.0%	7196 s
	NATIVE GUIDE$^+$	0.327	0.217	30.4%	7198 s
	SUB-SPACE$^+$	0.216	0.126	9.6%	7195 s
Wafer	GALACTIC-L	1.000	0.014	9.3%	150 s
	CELS	0.906	0.040	2.3%	988 s
	FASTPACE	1.000	0.087	24.7%	3663 s
	GLACIER$^+$	0.006	0.018	100.0%	3767 s

Dataset	Method	*Eff*	*Afc*	*Act*	*RT*
	MASCOTS	0.658	n/a	4.2%	7183 s
	MULTI-SPACE	0.247	0.030	3.1%	7197 s
	NATIVE GUIDE	1.000	0.067	13.3%	7175 s
	SUB-SPACE	0.314	0.047	6.1%	7195 s
	TSEVO^{+}	0.011	0.038	18.4%	7218 s

Table 12: Local generators under global selection (RQ2), per dataset: the pool of each local generator selected by Flat Optimal at $\mu = 4$, under ResNet.

Dataset	Source	*MDL*/series	*Eff*	*Afc*	*Act*	*RT*
Chinatown	GALACTIC-L	1.032	1.000	0.187	20.3%	35 s
	CELS	1.286	0.972	0.128	26.2%	34 s
	FASTPACE	1.108	0.994	0.133	32.8%	167 s
	GLACIER	1.477	0.955	0.126	100.0%	1088 s
	MASCOTS	1.077	0.994	0.161	10.7%	183 s
	MULTI-SPACE	1.113	0.994	0.148	38.9%	309 s
	NATIVE GUIDE	1.115	0.994	0.129	57.9%	102 s
	SUB-SPACE	1.085	0.997	0.138	44.1%	1036 s
	TSEVO	1.059	1.000	0.142	29.4%	6355 s
Coffee	GALACTIC-L	1.369	1.000	0.244	17.8%	34 s
	CELS	1.287	1.000	0.167	7.1%	21 s
	FASTPACE	1.671	0.982	0.252	30.3%	63 s
	GLACIER	4.217	0.839	0.295	100.0%	413 s
	MASCOTS	1.381	1.000	0.649	17.6%	97 s
	MULTI-SPACE	1.574	0.982	0.159	9.9%	314 s
	NATIVE GUIDE	3.211	0.893	0.179	25.9%	68 s
	SUB-SPACE	1.588	0.982	0.186	11.9%	224 s
	TSEVO	1.360	1.000	0.275	15.3%	2630 s
FreezerRegularTrain	GALACTIC-L	1.015	0.999	0.057	1.0%	174 s
	CELS	1.033	0.998	0.021	2.3%	623 s
	FASTPACE	1.022	0.999	0.047	6.3%	1631 s
	MASCOTS	1.021	0.999	0.107	11.1%	5923 s
	MULTI-SPACE	1.131	0.992	0.013	2.7%	7219 s
	NATIVE GUIDE	1.021	0.999	0.049	3.7%	5191 s
	SUB-SPACE	1.030	0.999	0.011	2.6%	7226 s
ItalyPowerDemand	GALACTIC-L	1.085	0.995	0.195	24.8%	52 s
	CELS	1.932	0.914	0.127	18.7%	68 s
	FASTPACE	1.074	0.995	0.200	61.4%	432 s
	GLACIER	3.036	0.811	0.119	100.0%	3557 s
	MASCOTS	1.263	0.979	0.324	41.8%	645 s
	MULTI-SPACE	1.109	0.993	0.178	26.8%	877 s
	NATIVE GUIDE	1.111	0.993	0.206	57.3%	305 s
	SUB-SPACE	1.176	0.987	0.177	35.1%	3593 s
	TSEVO^{+}	8.147	0.323	0.236	25.0%	7214 s
MixedShapesRegularTrain	GALACTIC-L	2.872	0.907	0.016	16.7%	769 s
	CELS	8.335	0.577	0.026	2.6%	858 s
	FASTPACE^{+}	12.947	0.302	0.073	36.0%	7238 s
	GLACIER^{+}	19.318	0.010	0.003	100.0%	7233 s
	MASCOTS^{+}	14.757	0.221	0.114	31.0%	7218 s
	MULTI-SPACE^{+}	18.413	0.053	0.026	5.9%	7231 s
	NATIVE GUIDE^{+}	16.687	0.125	0.129	48.7%	7206 s
	SUB-SPACE^{+}	14.598	0.229	0.059	34.5%	7230 s
SwedishLeaf	GALACTIC-L	5.965	0.659	0.038	26.3%	285 s
	CELS	8.403	0.419	0.039	9.6%	203 s
	FASTPACE	10.462	0.261	0.035	28.8%	3734 s
	GLACIER	9.539	0.401	0.026	100.0%	5256 s
	MASCOTS	9.730	0.309	0.089	16.2%	5659 s
	MULTI-SPACE	10.429	0.259	0.030	18.7%	6616 s
	NATIVE GUIDE	9.962	0.305	0.065	43.7%	1930 s
	SUB-SPACE	9.895	0.307	0.036	24.0%	5488 s
	TSEVO^{+}	13.298	0.070	0.108	47.1%	7230 s
SyntheticControl	GALACTIC-L	3.131	0.860	0.163	23.1%	136 s
	CELS	6.531	0.543	0.142	14.0%	89 s
	FASTPACE	7.226	0.480	0.137	27.5%	395 s
	GLACIER	11.780	0.102	0.098	100.0%	2490 s
	MASCOTS	6.902	0.523	0.192	46.6%	1189 s
	MULTI-SPACE	7.649	0.443	0.133	23.9%	925 s
	NATIVE GUIDE	8.076	0.405	0.160	32.0%	325 s
	SUB-SPACE	7.315	0.470	0.139	23.3%	1743 s

Dataset	Source	*MDL*/series	*Eff*	*Afc*	*Act*	*RT*
	TSEvo$^+$	9.042	0.349	0.201	44.6%	7209 s
Trace	Galactic-L	4.201	0.830	0.053	19.9%	65 s
	CELS	8.228	0.535	0.069	4.2%	68 s
	FastPACE	7.747	0.575	0.063	13.7%	283 s
	Glacier	12.065	0.300	0.028	100.0%	1402 s
	MASCOTS	7.167	0.605	0.167	29.6%	401 s
	Multi-SpaCE	7.730	0.580	0.061	9.8%	1189 s
	Native Guide	8.534	0.525	0.109	19.2%	274 s
	Sub-SpaCE	7.455	0.595	0.061	11.6%	850 s
	TSEvo$^+$	13.189	0.216	0.134	28.1%	7230 s
TwoPatterns	Galactic-L	1.093	0.998	0.165	12.8%	292 s
	CELS	1.390	0.978	0.122	9.3%	664 s
	FastPACE	5.428	0.712	0.160	17.6%	4871 s
	Glacier$^+$	15.524	0.038	0.104	100.0%	7231 s
	MASCOTS$^+$	11.069	0.319	0.373	53.0%	7233 s
	Multi-SpaCE$^+$	12.200	0.247	0.132	9.9%	7220 s
	Native Guide$^+$	11.888	0.257	0.250	32.9%	7213 s
	Sub-SpaCE$^+$	11.444	0.294	0.126	9.1%	7213 s
Wafer	Galactic-L	1.157	0.990	0.034	4.5%	449 s
	CELS	1.082	0.995	0.053	4.2%	1715 s
	FastPACE	1.394	0.974	0.123	14.7%	3802 s
	Glacier$^+$	15.792	0.029	0.026	100.0%	7232 s
	MASCOTS	1.245	0.984	0.053	11.8%	7294 s
	Multi-SpaCE	1.432	0.971	0.060	3.0%	7229 s
	Native Guide	1.287	0.981	0.113	21.3%	7368 s
	Sub-SpaCE	1.386	0.974	0.075	6.1%	7276 s
	TSEvo$^+$	15.467	0.051	0.043	15.3%	7234 s

Table 13: Global counterfactual comparison (RQ3), per dataset: Galactic-G (Flat Optimal) and the six adapted global baselines ($\star$), under ResNet.

Dataset	Method	*MDL*/series	*Eff*	*Afc*	*Act*	*RT*
Chinatown	Galactic-G (Flat Optimal)	1.0	1.000	0.187	20.3%	21 s
	CET$\star$	13.0	0.028	0.078	16.7%	3525 s
	FACEGroup$\star$ (cost)	9.2	0.975	0.167	100.0%	3 s
	FACEGroup$\star$ (cov)	9.3	1.000	0.210	100.0%	3 s
	Fast AReS$\star$	13.2	0.008	0.014	4.2%	8 s
	GLANCE$\star$	1.1	1.000	0.222	11.5%	6 s
	GLOBE-CE$\star$	6.7	0.997	0.081	100.0%	33 s
	T-CREx$\star$	12.6	0.073	0.044	4.2%	0 s
Coffee	Galactic-G (Flat Optimal)	1.4	1.000	0.244	17.8%	19 s
	CET$\star$	16.8	0.000	n/a	n/a	4331 s
	FACEGroup$\star$ (cost)	15.2	0.804	0.253	100.0%	0 s
	FACEGroup$\star$ (cov)	15.0	1.000	0.279	100.0%	0 s
	Fast AReS$\star$	16.8	0.000	n/a	n/a	8 s
	GLANCE$\star$	1.9	1.000	0.215	16.2%	38 s
	GLOBE-CE$\star$	15.7	0.643	0.520	100.0%	44 s
	T-CREx$\star$	16.8	0.000	n/a	n/a	0 s
FreezerRegularTrain	Galactic-G (Flat Optimal)	1.0	0.999	0.057	1.0%	79 s
	CET$\star$	18.6	0.000	n/a	n/a	9750 s
	FACEGroup$\star$ (cost)	15.6	0.991	0.093	100.0%	4552 s
	FACEGroup$\star$ (cov)	16.6	1.000	0.147	100.0%	4519 s
	GLANCE$\star$	1.1	0.997	0.023	0.5%	47 s
	GLOBE-CE$\star$	5.9	0.855	0.050	100.0%	337 s
	T-CREx$\star$	16.5	0.140	0.008	0.3%	3 s
ItalyPowerDemand	Galactic-G (Flat Optimal)	1.1	0.995	0.195	24.8%	24 s
	CET$\star$	8.5	0.389	0.114	16.6%	7390 s
	FACEGroup$\star$ (cost)	10.1	0.800	0.104	100.0%	28 s
	FACEGroup$\star$ (cov)	10.0	1.000	0.185	100.0%	28 s
	Fast AReS$\star$	13.3	0.003	0.014	4.2%	22 s
	GLANCE$\star$	1.0	1.000	0.179	13.2%	8 s
	GLOBE-CE$\star$	7.4	0.843	0.103	100.0%	54 s
	T-CREx$\star$	8.4	0.532	0.080	4.2%	1 s
MixedShapesRegularTrain	Galactic-G (Flat Optimal)	2.9	0.907	0.016	16.7%	153 s
	CET$\star$	18.7	0.000	n/a	n/a	9652 s
	GLANCE$\star$	13.6	0.287	0.040	0.6%	205 s
	GLOBE-CE$\star$	16.9	0.276	0.003	100.0%	2032 s
	T-CREx$\star$	18.6	0.001	0.006	0.1%	47 s

Dataset	Method	*MDL*/series	*Eff*	*Afc*	*Act*	*RT*
SwedishLeaf	GALACTIC-G (Flat Optimal)	7.1	0.659	0.038	26.3%	31 s
	CET⋆	16.0	0.000	n/a	n/a	7374 s
	FACEGROUP⋆ (cost)	15.4	0.272	0.069	100.0%	1231 s
	FACEGROUP⋆ (cov)	15.9	0.061	0.150	100.0%	1232 s
	FAST ARES⋆	15.8	0.010	0.005	0.9%	114 s
	GLANCE⋆	14.5	0.105	0.082	1.5%	105 s
	GLOBE-CE⋆	14.8	0.302	0.027	100.0%	249 s
	T-CREX⋆	16.0	0.000	n/a	n/a	3 s
SyntheticControl	GALACTIC-G (Flat Optimal)	3.5	0.860	0.163	23.1%	24 s
	CET⋆	14.6	0.000	n/a	n/a	5687 s
	FACEGROUP⋆ (cost)	14.0	0.528	0.218	100.0%	16 s
	FACEGROUP⋆ (cov)	14.0	0.488	0.219	100.0%	16 s
	FAST ARES⋆	14.6	0.002	0.010	1.7%	2 s
	GLANCE⋆	11.0	0.278	0.180	10.2%	36 s
	GLOBE-CE⋆	13.5	0.412	0.166	100.0%	90 s
	T-CREX⋆	14.6	0.000	n/a	n/a	1 s
Trace	GALACTIC-G (Flat Optimal)	4.4	0.830	0.053	19.9%	22 s
	CET⋆	17.0	0.000	n/a	n/a	5305 s
	FACEGROUP⋆ (cost)	15.7	0.845	0.100	100.0%	1 s
	FACEGROUP⋆ (cov)	15.7	0.890	0.106	100.0%	1 s
	GLANCE⋆	11.8	0.340	0.090	10.0%	154 s
	GLOBE-CE⋆	16.2	0.215	0.019	100.0%	180 s
	T-CREX⋆	17.0	0.000	n/a	n/a	0 s
TwoPatterns	GALACTIC-G (Flat Optimal)	1.1	0.998	0.165	12.8%	71 s
	CET⋆	16.4	0.000	n/a	n/a	9043 s
	FACEGROUP⋆ (cost)	16.2	0.199	0.291	100.0%	1085 s
	FACEGROUP⋆ (cov)	16.2	0.378	0.320	100.0%	1018 s
	FAST ARES⋆	16.4	0.000	n/a	n/a	136 s
	GLANCE⋆	6.8	0.628	0.185	20.0%	157 s
	GLOBE-CE⋆	16.3	0.012	0.196	100.0%	343 s
	T-CREX⋆	16.4	0.000	n/a	n/a	35 s
Wafer	GALACTIC-G (Flat Optimal)	1.2	0.990	0.034	4.5%	287 s
	CET⋆	16.8	0.000	n/a	n/a	8335 s
	FACEGROUP⋆ (cost)	16.8	0.000	n/a	n/a	20852 s
	FACEGROUP⋆ (cov)	15.8	1.000	0.140	100.0%	20083 s
	GLANCE⋆	2.0	0.938	0.058	1.4%	36 s
	GLOBE-CE⋆	4.8	0.921	0.022	100.0%	779 s
	T-CREX⋆	13.8	0.266	0.025	1.3%	33 s