Adapting the Gini Impurity Criterion for Splitting and Merging Strategies in Extended Automata Learning

Using CART-inspired impurity scores inside the FlexFringe framework

Tobiasz Zalewski[1] Dr.ir. S.E. S. Verwer[2]

[1] Computer Science Bachelor student, Technische Universiteit Delft

[2] Responsible professor, Algorithmics group, Technische Universiteit Delft

CSE3000 Research Project, Q4 2026

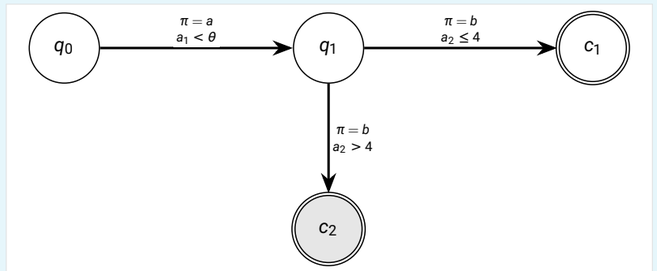
1 Background and Research Question

EDFA: sequential model with data guards

An Extended Deterministic Finite Automaton reads traces of symbol–attribute pairs

$$(\pi_1, \mathbf{a}_1), \ldots, (\pi_n, \mathbf{a}_n),$$

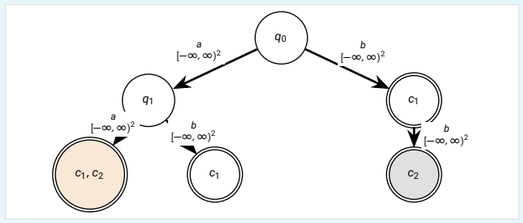
follows guarded transitions, and predicts the class from the final state. A transition guard can test numeric attributes, so the model captures both event order and data-dependent behaviour.



The learned automaton is inspectable: paths explain sequence structure, while guards explain which attribute values trigger different futures.

From APTA to guarded splits

The APTA is built by inserting each training trace into a prefix tree: traces with the same event prefix share the same path, branching only when the next symbol differs. Initially all guards are full-domain $[-\infty, \infty)^d$, so attributes do not separate traces and a shared node can contain mixed classes.



Splits can replace full-domain guards using thresholds that separate mixed classes.

CART/Gini intuition

With class fraction $p_c(A)$ at a node,

$$G(A) = \sum_{c \in C} p_c(A)(1 - p_c(A)) = 1 - \sum_{c \in C} p_c(A)^2.$$

Intuition: chance of misclassification when guessing from the node's class mix; 0 for pure nodes, higher for mixed classes.

Problem statement

FlexFringe repeatedly chooses a refinement: merge states, split a guarded transition, or extend the red core. **My focus:** a Gini-inspired heuristic for refinement scoring.

Research question

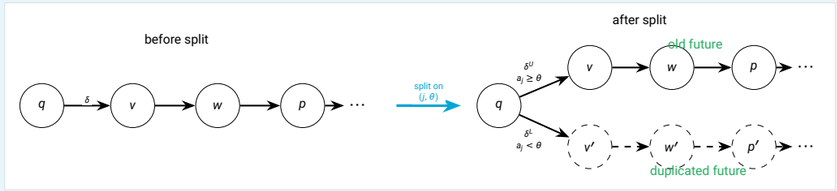
Can a Gini-based refinement evaluation heuristic improve EDFA learning compared to RTI while maintaining interpretable automaton structure?

- compare accuracy against RTI and CART;
- measure EDFA size and final-state consistency;
- test whether weighting future tails helps.

2 Methodology: Gini scoring for refinements

A transition split duplicates the future

A candidate split chooses an attribute j and threshold θ. Tails with $a_j < \theta$ are routed to a new successor; the previous future subtree is duplicated so that both groups keep their continuation structure.



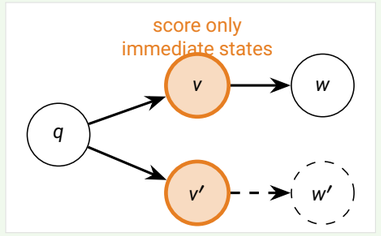
Gini score for candidate refinements

Reward a decrease in weighted impurity:

Split: $\Delta G = G_{\text{before}}(v) - \text{weighted } G_{\text{after}}(\text{split nodes})$,
Merge: $\Delta G = \text{weighted } G_{\text{before}}(\text{merge nodes}) - G_{\text{after}}(q^*)$,
Extend: $\Delta G = 0$.

Variant 1: local split impurity

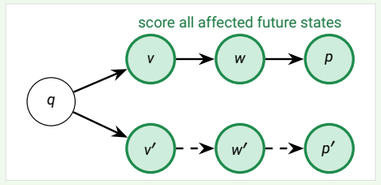
Scores only the two immediate successor states after the split:



$$h = G_{pre}(v) - r_v G(v) - r_{v'} G(v').$$

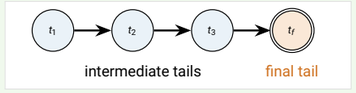
Compact and fast, but ignores deeper future behaviour.

Variant 2: future-aware impurity



Scores all affected future states with equal tail mass, $w(t) = 1$. In every state both continuing and final traces are considered equally.

Variant 3: λ-weighted future-aware impurity



Uses the same future-aware score, but changes each tail's contribution. In each state, a trace is assigned weight based on whether it is ending in that state or continuing further:

$$w_\lambda(t) = \lambda \mathbf{1}[\text{Final}(t)] + \frac{1 - \lambda}{L_\delta(\text{Tr}(t))}.$$

Higher λ: final labels matter more; length term limits long traces.

Merge scoring and selection

Merges use the same before–after idea in each variant, but in reverse: before a merge, the candidate states are treated like separate futures. Their weighted impurity is calculated before the operation, and the impurity of a merged state is calculated after.

3 Results and conclusions

Evaluation setup

Experiments used nine aeon/UCR time-series classification datasets and the CTU-13 network traffic dataset. Hyperparameters were selected on validation data; final performance was measured on test data.

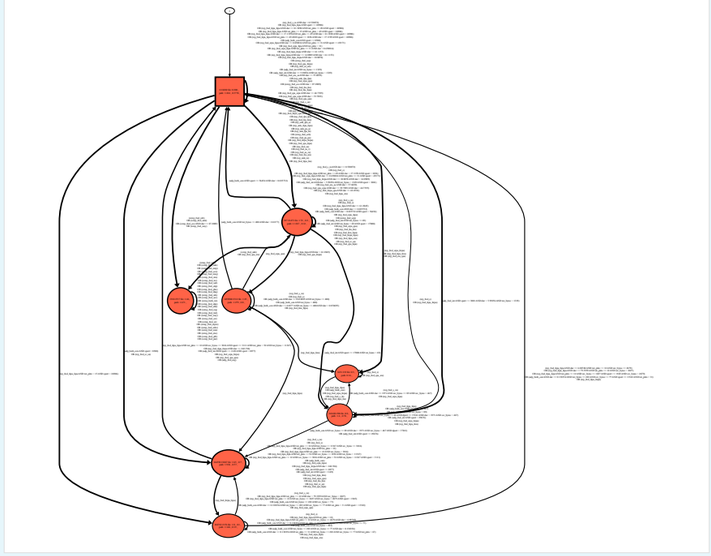
Metrics: accuracy, specificity, EDFA size, and final-state consistency. Consistency is the ratio of the number of non-mixed final states to the number of all final states.

Average test results

Method	Accuracy [%]	States	Consistency
Gini V1	60.27	**4.00**	0.73
Gini V2	62.95	12.30	0.78
Gini V3	**67.04**	41.20	**0.94**
RTI	63.86	303.10	0.53
CART	80.22	–	–

Variant 3 vs. RTI matched/outperformed on 8 of 10 datasets
Variant 3 vs. Variant 2 matched/outperformed on 9 of 10 datasets

Example learned EDFA



Variant 1 learned a compact EDFA for CTU-13, where it achieved the best test accuracy among all evaluated methods.

Key findings

- Gini variants produced much smaller automata than RTI.
- Variant 1 that ignores the future affected states performed the worst on average
- Variant 3 achieved the best average EDFA accuracy and highest final-state consistency.
- CART remained the strongest interpretable classifier for the selected problems
- When interpretability is the highest priority, Variant 1 or 2 may be preferable due to smaller automata.

Conclusion

The Gini-based evaluation function can improve EDFA learning, but not absolutely. It can produce smaller EDFAs than RTI with a competitive accuracy, but does not always perform better.

Gini impurity is a promising basis for EDFA refinement scoring. The λ-weighted variant gives the best accuracy–structure trade-off. Nevertheless, Variant 1 & 2 can still prove useful in certain scenarios.

TU Delft Delft University of Technology