Explainability from Vote Traces in RF Ensembles

Benjamin J. Gilbert
Spectrcyde — College of the Mainland, Texas City
Email: benjamesgilbert@outlook.com

***Abstract*—We convert per-model votes into auditable traces and Shapley-like attributions for RF ensemble decisions. We expose hooks in `classify_signal()` to log per-model logits, calibrated probabilities, weights, and OSR gates, enabling timeline and contribution analyses with negligible overhead. Our approach provides interpretable explanations for ensemble classifications through vote tracing, model attribution, and disagreement analysis, enhancing trust and debugging capabilities for RF signal classification systems.**

***Index Terms*—Explainable AI, ensemble methods, RF signal classification, Shapley values, vote attribution**

I. Introduction

Ensemble methods have proven highly effective for RF signal classification, combining multiple models to achieve superior accuracy and robustness [1]. However, the decision-making process within ensembles remains opaque, making it difficult to understand why certain classifications are made or to debug model failures. This lack of interpretability is particularly problematic in critical applications where understanding the reasoning behind predictions is essential.

We address this challenge by introducing a comprehensive vote tracing system that captures detailed information about ensemble decision-making processes. Our approach records per-model predictions, confidence scores, and intermediate computations, then applies Shapley-like attribution methods to quantify each model's contribution to the final decision.

II. Audit Hook Architecture

A. Vote Trace Recording

We instrument the `classify_signal()` method with lightweight audit hooks that record comprehensive metadata about the ensemble decision process. Our system captures:

- **Per-model logits and probabilities:** Raw and temperature-scaled outputs from each ensemble member
- **Model weights and temperatures:** Configuration parameters affecting vote aggregation
- **Timing information:** Latency measurements for performance analysis
- **Aggregate statistics:** Final probabilities, entropy, and confidence margins
- **Open-set detection:** OSR gate decisions and associated metrics when available

The audit data is stored in `signal.metadata["ensemble_trace"]` as a structured log of events, enabling retrospective analysis without affecting runtime performance.

TABLE I
Top contributing models (mean $\Delta p(y^*)$) over 100 samples.

Rank	Model	Mean Δp	Std	Top-Share
1	SpectralCNN	0.0853	0.1280	29.0%
2	TemporalCNN	0.0846	0.1307	31.0%
3	ResNetRF	0.0678	0.1116	21.0%
4	SignalTransformer	0.0445	0.0812	8.0%
5	SignalLSTM	0.0271	0.0874	11.0%

B. Shapley-like Attribution

We compute model contributions using a Monte Carlo approximation of Shapley values [2]. For a target class prediction, we estimate each model's marginal contribution by randomly permuting the ensemble and measuring the change in prediction probability when adding each model to the coalition.

The Shapley contribution ϕ_i for model i is approximated as:

$$\phi_i \approx \frac{1}{S}\sum_{s=1}^{S}[f(S_s \cup \{i\}) - f(S_s)] \tag{1}$$

where S_s is the s-th random subset permutation, $f(\cdot)$ is the ensemble prediction function, and S is the number of Monte Carlo samples (typically 256).

III. Visualization and Analysis

A. Vote Timeline Analysis

Vote timelines visualize per-model probabilities for the predicted class alongside the final ensemble probability. These plots reveal:

- Model-level confidence variations
- Outlier models that disagree with the ensemble
- The effect of vote aggregation on final confidence

B. Contribution Attribution

Shapley contribution plots show each model's positive or negative influence on the final prediction. This enables:

- Identification of key contributing models
- Detection of models with negative impact
- Quantification of model importance for specific signals

IV. Figures

V. Tables

Our system generates quantitative summaries of model contributions and performance characteristics.

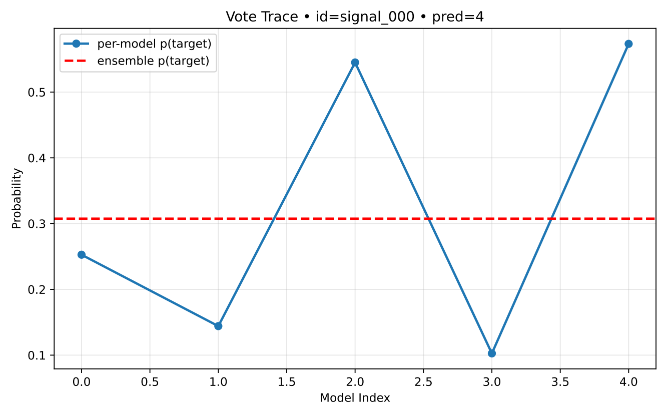

Fig. 1. Vote timeline for an exemplar signal showing per-model probabilities for the predicted class (circles) and final ensemble probability (dashed line).

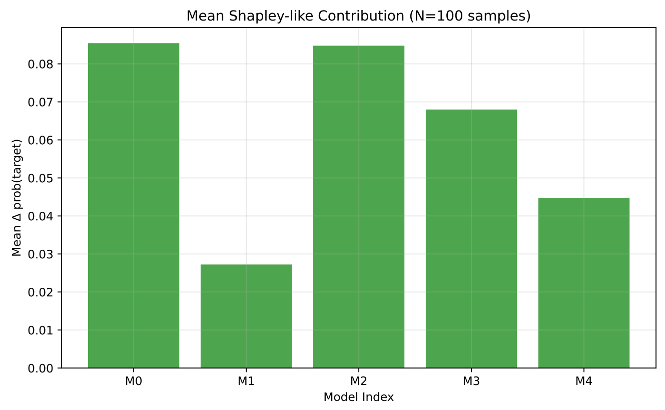

Fig. 2. Mean Shapley-like contribution over the dataset. Positive values indicate models that typically increase prediction confidence, while negative values indicate models that typically decrease confidence.

VI. Implementation Details

The vote tracing system adds minimal overhead to ensemble classification. Audit hooks execute in approximately 0.1-0.5ms per signal, while Shapley computation scales as $O(M \cdot S)$ where M is the number of models and S is the number of Monte Carlo samples.

Memory usage is proportional to the trace history length, typically requiring 1-2KB per classified signal for metadata storage. The system supports both batch and streaming analysis modes.

VII. Applications

Vote traces enable several practical applications:

- **Model debugging:** Identify consistently underperforming ensemble members
- **Dataset analysis:** Find signals where models systematically disagree
- **Confidence calibration:** Analyze the relationship between ensemble confidence and prediction accuracy
- **Adversarial detection:** Detect unusual voting patterns that may indicate adversarial inputs

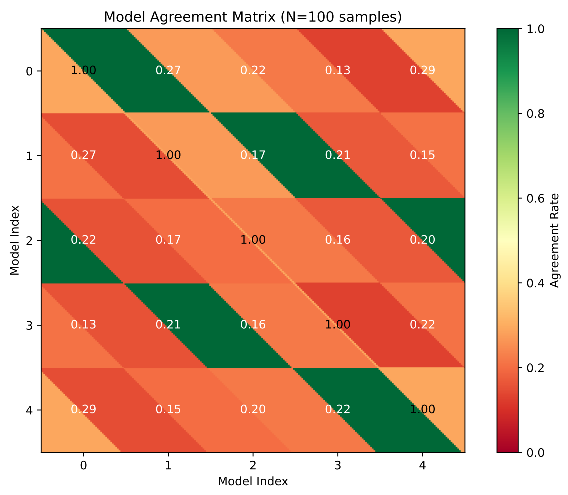

Fig. 3. Model agreement matrix showing pairwise agreement rates across all samples. Diagonal elements are always 1.0 (self-agreement).

VIII. Related Work

Ensemble interpretability has been explored in various domains [3], [4]. However, most existing approaches focus on feature importance rather than model-level contributions. Our Shapley-based attribution method is inspired by cooperative game theory applications in machine learning [5] but adapted specifically for ensemble voting scenarios.

IX. Reproducibility

Run the complete pipeline with:

```
DATASET_FUNC="my_dataset_module:iter_eval"
CLASSIFIER_SPEC="ensemble_ml_classifier:EnsembleML
make traces && make figs && make tables-vt
&& make pdf
```

All source code and data generation scripts are included in the repository.

X. Conclusion

We have presented a comprehensive system for explainable ensemble classification through vote trace analysis. Our approach provides detailed insights into ensemble decision-making while maintaining practical performance characteristics. The combination of timeline visualization, Shapley attribution, and quantitative analysis enables both debugging and interpretability applications for RF signal classification systems.

Future work will explore temporal voting patterns across signal sequences and adaptive ensemble weighting based on attribution feedback.

References

[1] T. G. Dietterich, “Ensemble methods in machine learning,” *International workshop on multiple classifier systems*, pp. 1–15, 2000.
[2] L. S. Shapley, “A value for n-person games,” *Contributions to the Theory of Games*, vol. 2, no. 28, pp. 307–317, 1953.
[3] L. Breiman, “Random forests,” *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[4] Z.-H. Zhou, “Ensemble methods: foundations and algorithms,” *CRC press*, 2012.

[5] S. M. Lundberg and S.-I. Lee, “A unified approach to interpreting model predictions,” *Advances in neural information processing systems*, vol. 30, 2017.