Compositional Agentic Formulation Search for Open-Vocabulary Audio-Visual Event Localization

Beomkwon Cho [*1] **Sunchan Park** [*1] **Kyeongbo Kong** [1]

Abstract

Open-vocabulary audio-visual event localization requires models to temporally localize events and recognize categories that may be unseen during training, exposing a tension between task-specific adaptation and preservation of pretrained multimodal alignment. We recast this problem as policy-aware formulation discovery: instead of fine-tuning pretrained encoders, we keep audio, visual, and text representations frozen and compose them with an agent-discovered symbolic decision formulation and a video-specific neural policy. Our method, Cue2Rule, uses a multi-agent loop to search over executable decision formulations, jointly guided by oracle expressiveness and policy learnability. It then trains a lightweight policy network to predict per-video parameters for the selected formulation, replacing globally fixed thresholds with adaptive parameterization. This decomposition separates reusable decision structure from sample-specific adaptation. On OV-AVEBench, Cue2Rule improves over the best fine-tuning baseline, raising the total average score from 57.8 to 60.2 and unseen-category performance from 55.8 to 59.9, while reducing the seen–unseen gap from 7.1 to 1.2. These results suggest that agent-discovered symbolic structure with neural per-sample instantiation provides an effective decision-layer adaptation strategy for OV-AVEL with frozen encoders.

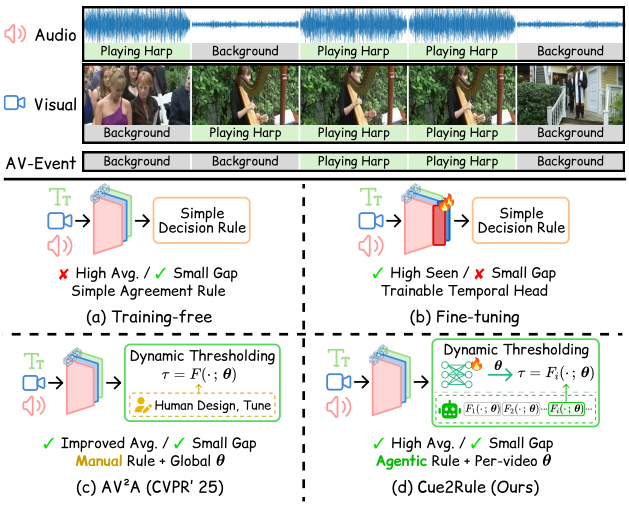

Figure 1. AVEL task and landscape of OV-AVEL decision-layer adaptation. *Top:* AVEL assigns a foreground event label to a segment only when the event is jointly supported by audio and visual evidence, and assigns background otherwise. *Bottom:* (a) Training-free methods keep encoders frozen and maintain a small seen–unseen gap, but suffer from low overall performance due to coarse agreement rules. (b) Fine-tuning methods train seen-class temporal heads and improve seen-category accuracy, but increase the seen–unseen gap under category shift. (c) AV^{2}A improves the frozen-encoder decision stage through dynamic thresholding, but relies on a manually designed rule with globally fixed parameters. (d) Cue2Rule discovers dynamic thresholding rules via report-guided agentic search and instantiates their parameters per video through a policy network.

1. Introduction

Audio-visual event perception is a central problem in multimodal understanding, requiring models to align what is heard, what is seen, and what event category is being expressed over time. Audio-Visual Event Localization (AVEL) (Tian et al., 2018) captures this challenge by asking a model to localize temporal event segments and assign semantic categories only when the event is jointly supported by audio and visual evidence, treating all other segments as background (top of Fig. 1). Recent work extends AVEL to the open-vocabulary setting (OV-AVEL) (Zhou et al., 2025a), where test videos may contain categories unseen during training. OV-AVEL is therefore a stringent test case for adapting multimodal foundation representations: the model must make task-specific temporal decisions while preserving the audio-visual-language alignment needed to recognize semantically novel categories.

This tension shifts the bottleneck from representation learning alone to the decision layer placed on top of frozen mul-

[*]Equal contribution [1]Pusan National University, Busan, South Korea. Correspondence to: Kyeongbo Kong <kbkong@pusan.ac.kr>.

Accepted to the 2nd Workshop on Compositional Learning at ICML 2026, Seoul, South Korea. Copyright 2026 by the author(s).

timodal cues. The bottom of Fig. 1 illustrates this design space. Training-free methods keep the encoders frozen and maintain a small seen–unseen gap, but rely on a simple top-category agreement rule that lacks temporal precision, leading to low overall performance (Fig. 1(a)). Fine-tuning methods train seen-class temporal layers and improve accuracy, but primarily on seen categories, increasing the seen–unseen gap under category shift (Fig. 1(b)). AV^{2}A (Shaar et al., 2025) improves the frozen-encoder decision stage through dynamic thresholding, but its formulation is manually designed and its globally tuned parameters are shared across all videos (Fig. 1(c)). This leaves the quadrant in Fig. 1(d): automatically discovering a dynamic decision formulation over frozen audio-text and visual-text similarity cues, and adapting its parameters per video. We cast this as policy-aware formulation discovery, where the symbolic formulation provides an interpretable scaffold over frozen multimodal cues, and a neural policy instantiates that scaffold with video-specific parameters.

In this paper, we propose **Cue2Rule**, an agentic decision-rule discovery framework for OV-AVEL. Cue2Rule separates the decision layer into two components: a symbolic formulation $F(\cdot;\theta)$ that maps frozen multimodal cues to audio-side and visual-side dynamic thresholds, and a lightweight policy network π_ϕ that predicts the formulation parameters θ for each input video. The LLM agents are used only during offline formulation discovery: the agentic loop searches for an interpretable formulation scaffold, while the policy network provides video-specific adaptation within the selected scaffold. At test time, Cue2Rule requires no LLM calls; inference consists of frozen feature extraction, one policy-network forward pass, and deterministic thresholding.

Cue2Rule searches the formulation space through a report-guided multi-agent loop. A *Plan Agent* summarizes prior results and sets the next search direction, a *Design Agent* proposes executable formulations under formal constraints, an *Oracle Agent* estimates each candidate's expressive ceiling through per-video parameter search, and a *Policy Agent* runs policy training and diagnoses the resulting policy-instantiated behavior. The resulting reports are accumulated in memory and guide subsequent iterations. This lets Cue2Rule refine formulations using two complementary signals: oracle capacity, which measures what a formulation can express under ideal per-video parameters, and policy-instantiated validation behavior, which measures whether the formulation remains usable when its parameters are predicted from frozen video embeddings.

We evaluate Cue2Rule on OV-AVEBench (Zhou et al., 2025a), comparing it with training-free, fine-tuning, and dynamic-thresholding baselines. The results show that Cue2Rule improves overall performance while maintaining a small seen–unseen gap, demonstrating that decision-layer adaptation can improve OV-AVEL without updating pretrained encoders. Our analysis further separates the roles of the discovered formulation and the adaptive policy: the formulation scaffold improves over manually designed thresholding rules, while the policy network replaces globally fixed parameters with video-specific calibration. We ablate the Plan, Design, Oracle, and Policy components and analyze oracle-policy gaps, showing that high oracle capacity alone does not guarantee policy-instantiated performance.

Our contributions are threefold:

- We formulate OV-AVEL adaptation as *policy-aware decision-rule discovery* over frozen multimodal cues, separating an interpretable symbolic formulation scaffold from video-specific neural parameterization.
- We introduce **Cue2Rule**, a report-guided multi-agent framework in which Plan, Design, Oracle, and Policy Agents propose, evaluate, and diagnose executable dynamic-thresholding formulations under formal constraints.
- We demonstrate on OV-AVEBench that combining an agent-discovered formulation scaffold with video-specific policy instantiation improves open-vocabulary audio-visual event localization without encoder fine-tuning, and we analyze the oracle-policy gap.

2. Related Work

Audio-visual event perception. Audio-visual event perception (AVEP) covers event understanding from paired audio-visual streams. AVEL localizes temporal segments where an event is both audible and visible (Tian et al., 2018); later closed-vocabulary methods improve fusion (Xu et al., 2020; Zhou et al., 2021; Duan et al., 2021; He et al., 2024; Lin et al., 2019; Wu et al., 2019; Zhou et al., 2025b), background discrimination (Xia & Zhao, 2022; Zhou et al., 2022), and temporal boundary localization (Yu et al., 2022; Mahmud & Marculescu, 2023; Wu et al., 2022). Other AVEP tasks include AVVP, which predicts modality-wise events (Tian et al., 2020; Chen et al., 2024; Gao et al., 2023; Zhao et al., 2025; Zhou et al., 2024a;b), and DAVEL, which targets dense events in longer videos (Geng et al., 2023; Li et al., 2026; Zhou et al., 2026). These methods largely assume fixed vocabularies learned from training categories.

Open-vocabulary audio-visual event localization. Open-vocabulary AVEL (OV-AVEL) (Zhou et al., 2025a) localizes and classifies events from both seen and unseen categories. OV-AVEBench (Zhou et al., 2025a) introduced this setting with segment-level annotations and open-vocabulary category splits. Recent methods use pretrained multimodal

encoders, such as ImageBind, LanguageBind, CLIP, and CLAP (Girdhar et al., 2023; Zhu et al., 2024; Radford et al., 2021; Elizalde et al., 2023), to compute audio-text and visual-text similarities over arbitrary text-defined categories. Existing approaches convert these frozen similarity scores into event predictions using hand-designed agreement rules, seen-class temporal modules, or validation-tuned dynamic thresholds (Zhou et al., 2025a; Shaar et al., 2025). Cue2Rule differs by searching for the decision formulation itself and instantiating its parameters per video.

LLM-guided program and formulation search. Large language models have been used to propose executable candidates, receive automatic feedback, and iteratively refine structured solutions. Prior work has explored this idea in mathematical discovery, program search, and symbolic regression for scientific equation discovery (Shojaee et al., 2025; Xia et al., 2026; Grayeli et al., 2024; Ma et al., 2024; Wang et al., 2025; Romera-Paredes et al., 2024). These methods show that LLMs can generate interpretable structures whose quality can be evaluated automatically. Cue2Rule applies this search paradigm to OV-AVEL, where the searched structures are decision formulations over multimodal similarity signals rather than equations or general-purpose programs.

3. Method

3.1. Problem Setup

As illustrated in Fig. 2, OV-AVEL (Zhou et al., 2025a) divides each video into T temporal segments and assigns to every segment a label from $C_{\text{seen}} \cup C_{\text{unseen}} \cup \{\text{background}\}$. Models are trained on segments whose foreground labels belong to C_{seen}, but at test time the ground-truth label of a segment may also belong to C_{unseen}, requiring generalization beyond the training categories. The pipeline consists of frozen encoders that extract audio-visual similarity cues, and a decision formulation that maps those cues to per-segment predictions.

Frozen Encoders. We extract all representations from frozen pretrained encoders. For a video with T segments and a category set of size C, we obtain per-segment audio and visual embeddings $\mathbf{e}^a, \mathbf{e}^v \in \mathbb{R}^{T\times D}$ and per-category text embeddings $\mathbf{t}^a, \mathbf{t}^v \in \mathbb{R}^{C\times D}$, where each text embedding is aligned to its corresponding modality space. When the encoder maps all modalities into a common space (e.g., ImageBind), $\mathbf{t}^a$ and $\mathbf{t}^v$ are identical. We denote by s^m_{tc} the cosine similarity between the embedding of modality $m \in \{a, v\}$ at segment t and the text embedding of category c, yielding similarity matrices $\mathbf{s}^a, \mathbf{s}^v \in \mathbb{R}^{T\times C}$.

Formulation Interface. A decision formulation $F(\cdot;\boldsymbol{\theta})$ takes frozen embeddings, similarity matrices $(\mathbf{s}^a, \mathbf{s}^v)$, and

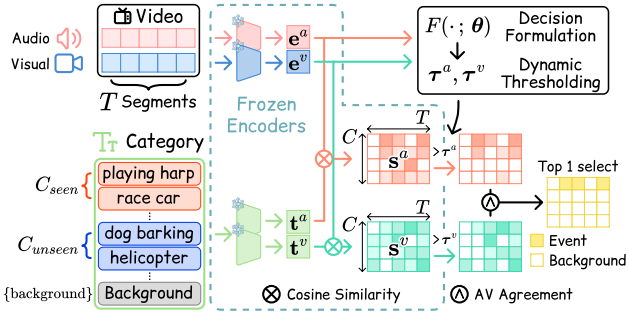

Figure 2. Overview of the OV-AVEL pipeline.

deterministic primitives derived from them as input, and produces per-segment, per-category thresholds $\boldsymbol{\tau}^a, \boldsymbol{\tau}^v \in \mathbb{R}^{T\times C}$. A segment t first filters out categories below the threshold in each modality, then selects the top-scoring category per modality among the remaining ones. The segment is assigned that category if both modalities agree, and labeled background otherwise. This decision procedure is fixed across all candidate formulations; only the threshold-producing function F varies during the agentic search. The parameter vector $\boldsymbol{\theta} \in \mathbb{R}^P$, where P is the number of parameters, controls the behavior of F.

3.2. Cue2Rule Overview

Cue2Rule adapts frozen multimodal representations at the decision layer. It separates prediction into two components: a symbolic formulation $F(\cdot;\boldsymbol{\theta})$, which defines how frozen multimodal cues are converted into audio-side and visual-side thresholds, and a lightweight policy network π_ϕ, which predicts the formulation parameters $\boldsymbol{\theta}$ for each video from frozen audio-visual embeddings $(\mathbf{e}^a, \mathbf{e}^v)$. The formulation provides the decision-rule scaffold, while the policy supplies video-specific adaptation within that scaffold.

Fig. 3 shows the offline search procedure used to discover the formulation scaffold. Cue2Rule does not use LLM agents as test-time predictors. Instead, the agents are used only during formulation discovery. At each iteration, the framework performs four steps. First, the *Plan Agent* reads the accumulated memory and issues search directives. Second, the *Design Agent* proposes an executable formulation and parameter ranges under formal constraints. Third, the *Oracle Agent* evaluates the candidate's expressive capacity by searching for per-video oracle parameters and performing requested ablations. Fourth, the *Policy Agent* trains a policy network for the same candidate and diagnoses whether the formulation remains effective when its parameters are adaptively predicted from the audio-visual embeddings of each sample. The Oracle and Policy Agents write analysis reports, and all reports are appended to memory for the next iteration. Detailed agent inputs, outputs, including example prompts and reports, are provided in Appendix D.

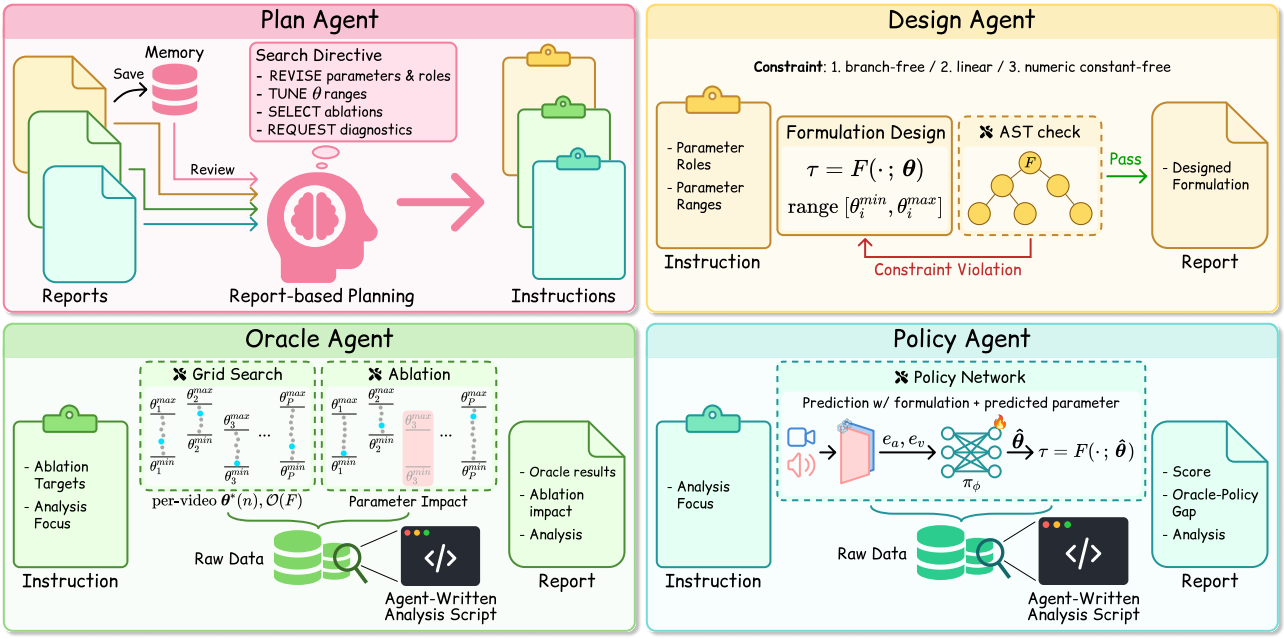

Figure 3. **Overview of Cue2Rule's report-guided agentic search.** At each iteration, the Plan Agent sets the search direction from accumulated reports, the Design Agent proposes an executable formulation verified against formal constraints, and the Oracle and Policy Agents analyze the candidate from complementary perspectives. The Oracle Agent estimates expressive capacity through per-video parameter search and ablations, while the Policy Agent trains and diagnoses a policy network for the candidate. All reports are appended to memory for the next iteration; no LLM agent is used at test time.

This makes Cue2Rule policy-aware, but not a full co-optimization procedure. For each candidate formulation F_i, Cue2Rule records two complementary signals:

$$q_i^{\text{oracle}} = \mathcal{O}(F_i), \qquad q_i^{\text{policy}} = \mathcal{V}(F_i, \pi_{\phi_i}),$$

where $\mathcal{O}(F_i)$ measures the best attainable performance under ideal per-video parameters, and $\mathcal{V}(F_i, \pi_{\phi_i})$ measures validation performance after training a policy to instantiate the formulation. The final formulation–policy pair is selected by validation performance, while the test split is held out from all agents and used only for final evaluation in Section 4. Appendix D provides an input–operation–output summary of the four agents and representative agent-visible prompts and reports.

3.3. Plan Agent

Search trajectory analysis. The Plan Agent begins each iteration by reviewing memory to assess the current state of the search. Memory stores per-iteration metrics, one-line agent summaries, and full per-agent reports accessed on demand. For example, a widening gap between the oracle ceiling and validation policy performance indicates a learnability problem: the policy fails to predict optimal parameters from context. In another case, when neither oracle nor validation performance improves, the formulation itself is the bottleneck, requiring an axis shift rather than incremental refinement.

Directive construction. Based on this diagnosis, the Plan Agent produces three directives: a natural-language strategy memo directed to the Design Agent, an evaluation protocol specifying parameter ablations paired with analysis intents for the Oracle Agent, and an analysis directive for the Policy Agent. The strategy memo assigns each parameter a conceptual role and an operating range. When the Oracle or Policy reports identify a parameter as inactive, the Plan Agent reallocates it to capture a different aspect of the audio-visual similarity structure. The evaluation protocol and analysis directive are grounded in the same parameter definitions as the strategy memo, so that oracle expressivity and policy learnability analyses test the same hypotheses.

3.4. Design Agent

Formulation proposal. Given the Plan Agent's directive, the Design Agent proposes a candidate formulation as a Python function together with a per-parameter range $[\theta_i^{\min}, \theta_i^{\max}]$ for each parameter, including zero to allow parameter deactivation. These ranges define both the search space for oracle evaluation and the output bounds for the policy network. The agent receives the interface specification—inputs including frozen embeddings, similarity matrices $(\mathbf{s}^a, \mathbf{s}^v)$, and deterministic primitives derived from them.

Formal constraints. Each candidate formulation must satisfy a set of structural constraints that together ensure the resulting threshold equations are interpretable as closed-form expressions. The threshold for each modality takes a fixed linear form: a bias plus a system-specified number of weighted terms. Each term is defined as a symbolic function over the available audio-visual inputs. To avoid non-linear parameter interactions that destabilize both Sobol oracle search and policy gradient learning, each parameter may only appear as an outer weight or bias in the threshold expression, never as an inner argument to a term. Numeric constants within the formulation are restricted to a small predefined set $\{0, \pm 0.5, \pm 1, \pm 2\}$ plus a numerical stability epsilon, preventing the Design Agent from inserting hidden hyperparameters.

Constraint verification. These constraints are specified as natural-language instructions in the prompt and verified via abstract syntax tree (AST) parsing, covering function count, per-function statement budget, numeric literals, inner-parameter usage, and control flow. The prohibition on control flow — no conditionals, loops, or comprehensions — requires all logic to be expressed as closed-form expressions, preserving the interpretability of the formulation. Candidates that fail any check are returned to the Design Agent with diagnostic feedback for revision.

3.5. Oracle Agent

Oracle ceiling. For each candidate formulation $F(\cdot\,;\,\boldsymbol{\theta})$, the Oracle Agent performs a per-video parameter search over the ranges $[\theta_i^{\min}, \theta_i^{\max}]$ defined by the Design Agent, using a Sobol sequence to provide more uniform coverage. For each training video n, the optimal parameters are obtained as $\boldsymbol{\theta}^\star(n) = \arg\max_{\boldsymbol{\theta}} \mathcal{R}(F, \boldsymbol{\theta}; n)$, where $\mathcal{R}$ is a per-video performance metric. The oracle ceiling is then defined as:

$$\mathcal{O}(F) = \mathbb{E}_n\big[\mathcal{R}(F, \boldsymbol{\theta}^\star(n); n)\big],$$

which measures the formulation's structural expressiveness independently of how well a policy can predict the optimal parameters in practice.

Parameter ablation. Following the evaluation protocol issued by the Plan Agent, each designated parameter subset is independently zeroed out and the remaining parameters are re-optimized via a separate Sobol grid to obtain per-video ablated optimal parameters $\boldsymbol{\theta}^\star_{\text{abl}}(n)$. The per-sample marginal contribution of each designated subset is then defined as:

$$\Delta(n) = \max\big(0,\, \mathcal{R}(F, \boldsymbol{\theta}^\star(n); n) - \mathcal{R}(F, \boldsymbol{\theta}^\star_{\text{abl}}(n); n)\big),$$

where the $\max(0, \cdot)$ clipping handles finite Sobol sampling noise: the ablated search space is a subset of the full one, so the inequality $\mathcal{R}(F, \boldsymbol{\theta}^\star_{\text{abl}}(n); n) \leq \mathcal{R}(F, \boldsymbol{\theta}^\star(n); n)$ holds in principle, but empirical optima from a finite grid may slightly violate it, yielding a distribution over training videos that quantifies how much each parameter group contributes to the oracle ceiling.

Analysis and report. Beyond the predefined oracle evaluation flow, the Oracle Agent writes and executes custom scripts against the raw per-sample data to conduct analyses at three levels. First, standard analyses covering parameter distribution statistics, modality balance, and per-ablation marginal patterns are conducted. Second, targeted analyses follow the Plan Agent's directives, examining specific hypotheses about parameter roles or ablation patterns. Third, the agent selects additional analyses based on what the results suggest, such as per-category marginal breakdowns or cross-ablation correlation. All findings are compiled into a natural-language report with a one-line summary, both appended to memory for the Plan Agent at the next iteration.

3.6. Policy Agent

Policy network. To provide video-specific parameter adaptation within the candidate formulation, the Policy Agent trains a lightweight policy network π_ϕ that takes a sequence of per-segment frozen embeddings $(\mathbf{e}^a, \mathbf{e}^v)$ as input and outputs a single parameter vector $\boldsymbol{\theta}$ per video. We implement π_ϕ with modality-specific self-attention layers followed by cross-attention between the two modalities. The resulting representations are mean-pooled over segments, concatenated, and passed to a linear head to produce a raw output $\mathbf{z} \in \mathbb{R}^P$. Each z_i is then squashed into the per-parameter range $[\theta_i^{\min}, \theta_i^{\max}]$ defined with the formulation via the logistic sigmoid:

$$\theta_i = \theta_i^{\min} + \sigma(z_i) \cdot (\theta_i^{\max} - \theta_i^{\min}).$$

These ranges, iteratively refined across iterations based on accumulated oracle and policy reports, serve as a hard prior on each parameter's operating region.

Policy gradient training. We cast per-video parameter prediction as a contextual bandit with state $(\mathbf{e}^a, \mathbf{e}^v)$, action $\boldsymbol{\theta}$, and reward $\mathcal{R}(F, \boldsymbol{\theta}; n)$. We train π_ϕ with policy gradient, as the decision procedure of Section 3.1 makes the reward non-differentiable and per-video oracle parameters do not provide a unique regression target. We use leave-one-out REINFORCE (Ahmadian et al., 2024) with a Gaussian policy (mean $\mathbf{z}$, fixed standard deviation), drawing K actions per video and using the mean reward of the remaining $K-1$ as an unbiased per-context baseline to reduce variance. At inference, the mean $\mathbf{z}$ is used directly. After training for a fixed number of epochs, the checkpoint with the highest validation performance is selected.

Analysis and report. The selected checkpoint is applied to the validation set to produce per-sample outputs including predicted parameter vectors, per-modality frame-level predictions, and performance metrics, all stored as raw data.

Table 1. **Performance comparison on OV-AVEBench.** Baseline scores are from (Zhou et al., 2025a) except AV^{2}A, which we adapt to ImageBind. AV^{2}A-Manual uses a manually selected global setting, while AV^{2}A-Grid is a test-tuned global ceiling for the AV^{2}A rule. Cue2Rule is selected by validation performance.

	Seen				Unseen				Total			
Method	Acc.	Seg.	Eve.	**Avg.**	Acc.	Seg.	Eve.	**Avg.**	Acc.	Seg.	Eve.	**Avg.**
Training-free												
Video-LLaMA2	50.1	40.6	32.0	40.9	48.5	38.5	29.0	38.6	48.9	39.1	29.8	39.3
CLIP&CLAP	51.4	41.4	31.9	41.6	51.6	42.2	31.6	41.8	51.5	41.9	31.7	41.7
ImageBind-TF	57.5	45.0	34.0	45.5	59.8	47.3	34.0	47.0	59.2	46.7	34.0	46.6
Fine-tuning												
CMRA	65.2	58.8	54.3	59.4	36.0	31.0	26.3	31.1	44.3	38.9	34.3	39.2
AVE	**76.6**	63.6	56.0	65.4	44.6	33.2	24.0	34.0	53.8	41.9	33.2	42.9
PSP	75.4	66.8	61.0	67.7	33.7	28.2	24.2	28.7	45.6	39.3	34.7	39.9
MM-Pyramid	76.5	**66.9**	**62.3**	**68.6**	36.8	29.0	23.8	29.9	48.4	40.2	35.2	41.2
ImageBind-FT	72.5	61.8	54.5	62.9	64.9	55.0	47.5	55.8	67.1	56.9	49.5	57.8
Dynamic Thresholding												
AV^{2}A-Manual	53.0	46.1	42.7	47.3	59.5	51.1	46.4	52.3	57.6	49.7	45.4	50.9
AV^{2}A-Grid	59.9	53.1	49.1	54.1	63.4	55.6	50.9	56.6	62.4	54.9	50.4	55.9
Cue2Rule (ours)	68.6	60.1	54.6	61.1	**68.5**	**59.1**	**52.0**	**59.9**	**68.6**	**59.4**	**52.7**	**60.2**

The Policy Agent then writes and executes custom scripts against this data following the same three-level structure as the Oracle Agent — standard, directed, and custom — with a focus on learnability. The findings are then compiled into a report and a one-line summary appended to memory for the next iteration.

4. Experiments

4.1. Setup

Evaluation. We evaluate on OV-AVEBench (Zhou et al., 2025a): 24,800 10-second videos across 67 categories (46 seen, 21 unseen). Training uses seen-only videos; validation and test include both seen and unseen classes. We use frozen ImageBind (Girdhar et al., 2023) as the shared multimodal encoder for audio, visual, and text modalities. Following (Zhou et al., 2025a), we report segment-level accuracy (Acc.), segment-level F1-score (Seg.), event-level F1-score at Intersection over Union (IoU) ≥ 0.5 (Eve.), and their average (Avg.). Baselines include the OV-AVEBench training-free and fine-tuning models (Zhou et al., 2025a) and AV^{2}A (Shaar et al., 2025).

Implementation. The agentic search runs 3 isolated sessions of 20 iterations each, yielding 60 candidate formulations. All agents use gpt-5.4 (Singh et al., 2026) in OpenAI Codex (v0.125.0, April 2026). Each formulation defines $P{=}10$ parameters with ranges $[\theta_i^{\min}, \theta_i^{\max}]$. The oracle uses a Sobol sequence of 2^{17} $\boldsymbol{\theta}$ samples per video. All π_ϕ share the same hyperparameters: dim 256, 4 heads, FFN 512, dropout 0.1, $\sigma{=}0.3$; trained with AdamW (Loshchilov & Hutter, 2019) (lr $1{\times}10^{-4}$, batch 32, 20 epochs) on a single RTX 3090, selecting the best validation Acc. checkpoint. The leave-one-out REINFORCE (Ahmadian et al., 2024) objective draws $K{=}4$ samples with video-level Acc. as reward. The best-validation candidate across sessions is taken as $F^\star$ with policy $\pi_\phi^\star$. Each session takes 9.8 hours on average.

4.2. Quantitative Results

Overall performance. We organize baselines into three groups: training-free, fine-tuning, and dynamic thresholding methods. Since AV^{2}A has no published ImageBind configuration, we evaluate two variants: AV^{2}A-Manual, with parameters manually selected via validation inspection, and AV^{2}A-Grid, which selects a single global hyperparameter setting by grid search on the test set. Thus, AV^{2}A-Grid should be read as an empirical upper-bound estimate for a globally tuned AV^{2}A rule on benchmark, rather than as a validation-selected baseline. Cue2Rule is selected solely by validation performance. Cue2Rule achieves the highest Total Avg. of 60.2 among all baselines (Table 1). Within dynamic thresholding group, Cue2Rule surpasses AV^{2}A-Grid by $+4.3$ Avg. despite AV^{2}A-Grid having access to globally optimized test-set parameters, demonstrating that per-video adaptive parameterization captures structure that global optimization cannot. Cue2Rule also outperforms the best fine-tuning baseline, ImageBind-FT, by $+2.4$ Avg. while keeping the encoder frozen.

Seen–unseen balance. Training-free methods tend to show relatively balanced Seen and Unseen performance, though their absolute accuracy remains limited. Fine-tuning methods (AVE, PSP, MM-Pyramid) achieve strong Seen performance but suffer severe degradation on Unseen categories, with gaps exceeding 30 points—a clear sign of seen-class overfitting. ImageBind-FT, which freezes the encoder and

Table 2. **Ablation study.** *Oracle*: accuracy obtained with per-video oracle parameters on the training set. *Test*: evaluation results on the test set across three metrics (Acc., Seg., Eve.) and their average (Avg.).

Variant	Oracle	Test			
	Acc.	Acc.	Seg.	Eve.	Avg.
Cue2Rule (full)	89.6	**68.6**	**59.4**	**52.7**	**60.2**
w/o policy agent	**94.1**	67.1	57.1	49.2	57.8
w/o oracle agent	91.4	66.2	55.3	46.3	55.9
w/o feedback	90.0	64.7	53.3	44.4	54.1
global threshold	62.6	63.5	53.3	45.2	54.0

trains only additional temporal layers, narrows the gap to 7.1 points. Dynamic thresholding methods achieve higher overall performance than training-free baselines while maintaining better Seen–Unseen balance than fine-tuning methods. Cue2Rule reaches the highest Total Avg. with a small Seen–Unseen gap of 1.2 points, confirming that per-video parameterization adapts to each video without relying on seen-class supervision.

4.3. Ablation Study

Policy agent. To isolate the effect of learnability feedback, we remove the Policy Agent from the search loop (Table 2). Without this signal, the search drives formulation expressiveness substantially higher (Oracle Acc. 94.1), yet this gain does not transfer to test performance, with Total Avg. dropping by 2.4 points. This indicates that expressiveness alone is not a sufficient search signal; policy-side diagnostics help steer the search toward formulations that remain usable after policy instantiation.

Oracle agent. Examining the role of expressiveness feedback, we remove the Oracle Agent from the search loop. Guided solely by policy feedback, the search tends to stay within learnable formulations without exploring sufficiently expressive directions—despite a similar oracle accuracy level (91.4), Total Avg. drops by 4.3 points, a larger drop than removing the Policy Agent. This suggests that the Oracle Agent's structural signal is essential for guiding the search toward expressive formulations.

No feedback. Removing both agent feedbacks entirely reduces the search to blind exploration over the symbolic form space, with the Plan Agent seeing only the history of prior formulations. This yields the largest Total Avg. drop among search variants at 6.1 points, demonstrating that the iterative feedback loop is the primary source of search quality, not the agent's intrinsic priors over good formulations. The gap relative to each single-agent removal suggests that both feedback signals each contribute to search quality.

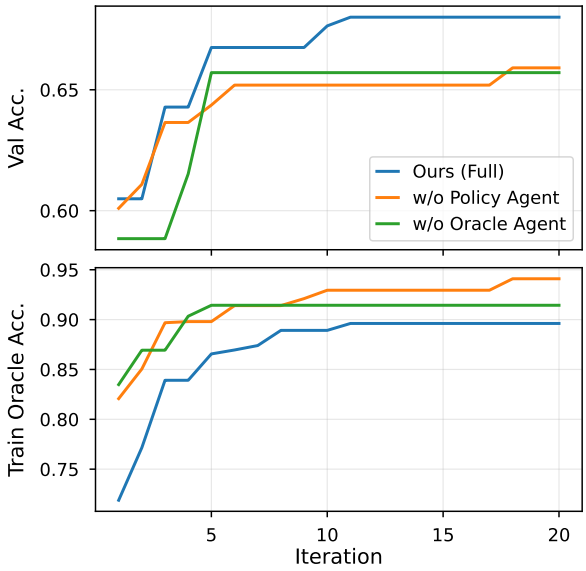

Figure 4. **Best score trajectory.** Guided by joint oracle–policy feedback, Cue2Rule attains the highest Val Acc. despite the lowest oracle ceiling.

Global threshold. For this variant, we replace the per-video policy with a single global $\boldsymbol{\theta}$ found by oracle search on the training set, with the best candidate selected by validation Acc. under this fixed parameter set. Applying the selected formulation to the test data yields a Total Avg. of 54.0, close to AV^{2}A-Grid, suggesting that the agentic search reliably discovers competitive formulations under globally fixed parameters. The 6.2-point improvement from the full model then shows that π_ϕ meaningfully advances performance beyond what global optimization can achieve.

4.4. Analysis

Search trajectory. Fig. 4 traces the running-best Val Acc. and Train Oracle Acc. across the 20 search iterations for the full Cue2Rule and the two single-agent ablations. By the final iteration, Cue2Rule reaches the lowest Train Oracle Acc. among the three variants yet attains the highest Val Acc. Without the Policy Agent, late-iteration oracle gains do not transfer to Val Acc., indicating expressive structures that the per-video policy cannot realize. Without the Oracle Agent, both metrics plateau early, as the search lacks the structural signal needed to escape learnable but expressively limited regions. The full search instead rises more slowly in Train Oracle Acc. while continuing to improve Val Acc. through later iterations. This supports our framing of Cue2Rule as *policy-aware formulation discovery*: the dual feedback steers the search toward formulations whose expressiveness is realizable by the per-video policy, rather than toward the highest oracle ceiling.

Qualitative analysis. Fig. 5 illustrates Cue2Rule's per-segment adaptation. In (a), audio and visual co-occur, and

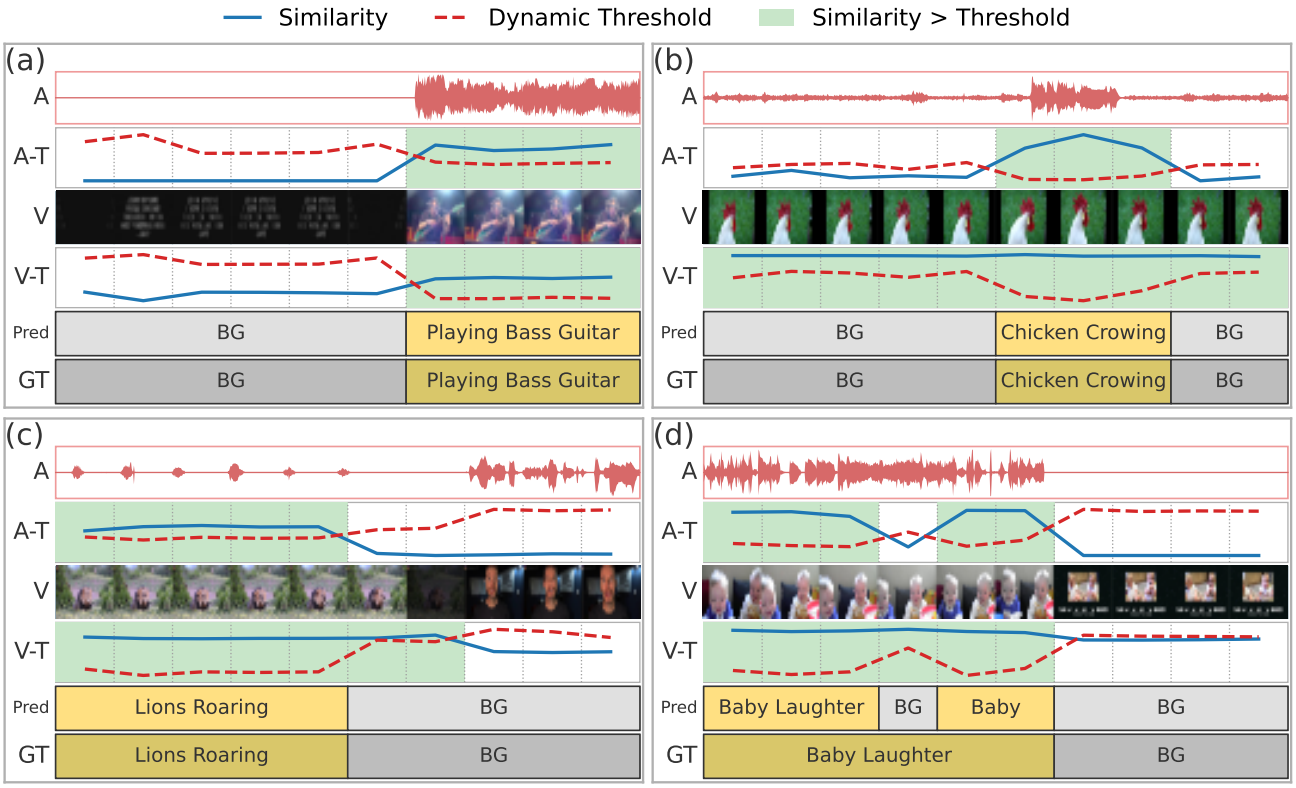

Figure 5. **Per-video adaptive thresholds across diverse event types.** Each panel shows audio (A) and visual (V) signals, their similarities to the target category (A-T, V-T), and Cue2Rule's per-segment dynamic thresholds (red dashed). Green regions denote similarity > threshold; Pred and GT bars below give predicted and ground-truth labels.

the threshold drops cleanly over the event. In (b), the visual is constant while the audio is sparse; the threshold lowers within audio segments and rises elsewhere, cleanly identifying the event window. In (c), the audio fades before the visual: both thresholds rise gradually, with A-T similarity crossing below its threshold first to mark the audio side as background; once the visual also fades, V-T's threshold rises further to firmly partition the visual side as well. Panel (d) is a failure: when A-T similarity drops within the event due to encoder noise, the threshold rises instead of compensating. Across panels, threshold movements in one modality often track the other, reflecting that supervision is provided only at the AV-event level.

5. Limitations and Future Directions

Cue2Rule has three main limitations. First, our task-level evaluation is limited to OV-AVEL, so the results support decision-layer adaptation for this setting but do not establish generality across arbitrary multimodal tasks. Second, formulation discovery is an offline search procedure that requires LLM-agent calls, oracle evaluation, and policy training for multiple candidates. Although no LLM is used at test time, this search cost is higher than training a single fixed model. Third, the current agents reason over textual reports and aggregate statistics rather than directly inspecting raw audio-visual samples. Future work should evaluate Cue2Rule on additional audio-visual tasks, reduce search cost through more efficient candidate selection, study open or smaller LLM backbones, and investigate multimodal agent feedback.

6. Conclusion

We introduced Cue2Rule, which frames OV-AVEL as policy-aware formulation discovery over frozen multimodal cues: a multi-agent loop searches for a symbolic decision formulation, while a lightweight neural policy supplies parameters specific to each video within that scaffold. On OV-AVEBench, Cue2Rule outperforms prior baselines across training-free, fine-tuning, and dynamic thresholding while maintaining a small seen–unseen gap. Our analysis further shows that the agentic search is most effective when guided jointly by oracle capacity and policy learnability. This paradigm may extend to other frozen-encoder adaptation tasks; we leave such extensions to future work.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (No. RS-2024-00456152), the Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (No. RS-2025-25437541), and the "Advanced GPU Utilization Support Program" funded by the Government of the Republic of Korea (Ministry of Science and ICT), and the authors gratefully acknowledge the Cluster Server for Computational Science at Pusan National University for providing computational resources.

References

Ahmadian, A., Cremer, C., Gallé, M., Fadaee, M., Kreutzer, J., Pietquin, O., Üstün, A., and Hooker, S. Back to basics: Revisiting reinforce style optimization for learning from human feedback in llms. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 12248–12267, 2024.

Chen, Y., Guo, R., Liu, X., Wu, P., Li, G., Li, Z., and Wang, W. Cm-pie: Cross-modal perception for interactive-enhanced audio-visual video parsing. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 8421–8425. IEEE, 2024.

Duan, B., Tang, H., Wang, W., Zong, Z., Yang, G., and Yan, Y. Audio-visual event localization via recursive fusion by joint co-attention. In *IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 4013–4022, 2021.

Elizalde, B., Deshmukh, S., Al Ismail, M., and Wang, H. Clap: Learning audio concepts from natural language supervision. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 1–5. IEEE, 2023.

Gao, J., Chen, M., and Xu, C. Collecting cross-modal presence-absence evidence for weakly-supervised audio-visual event perception. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 18827–18836, 2023.

Geng, T., Wang, T., Duan, J., Cong, R., and Zheng, F. Dense-localizing audio-visual events in untrimmed videos: A large-scale benchmark and baseline. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22942–22951, 2023.

Girdhar, R., El-Nouby, A., Liu, Z., Singh, M., Alwala, K. V., Joulin, A., and Misra, I. Imagebind: One embedding space to bind them all. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15180–15190, 2023.

Grayeli, A., Sehgal, A., Costilla-Reyes, O., Cranmer, M., and Chaudhuri, S. Symbolic regression with a learned concept library. *Advances in Neural Information Processing Systems*, 37:44678–44709, 2024.

He, X., Liu, X., Li, Y., Zhao, D., Shen, G., Kong, Q., Yang, X., and Zeng, Y. Cace-net: Co-guidance attention and contrastive enhancement for effective audio-visual event localization. In *ACM International Conference on Multimedia*, pp. 985–993, 2024.

Li, H., Dang, Y., Xing, Y., Wang, Y., and Yin, J. Esg-net: Event-aware semantic guided network for dense audio-visual event localization. *IEEE Transactions on Multimedia*, 2026.

Lin, Y.-B., Li, Y.-J., and Wang, Y.-C. F. Dual-modality seq2seq network for audio-visual event localization. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 2002–2006. IEEE, 2019.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. *International Conference on Learning Representations*, 2019.

Ma, P., Wang, T.-H., Guo, M., Sun, Z., Tenenbaum, J. B., Rus, D., Gan, C., and Matusik, W. Llm and simulation as bilevel optimizers: A new paradigm to advance physical scientific discovery. In *International Conference on Machine Learning*, pp. 33940–33962, 2024.

Mahmud, T. and Marculescu, D. Ave-clip: Audioclip-based multi-window temporal transformer for audio-visual event localization. In *IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 5158–5167, 2023.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pp. 8748–8763. PMLR, 2021.

Romera-Paredes, B., Barekatain, M., Novikov, A., Balog, M., Kumar, M. P., Dupont, E., Ruiz, F. J., Ellenberg, J. S., Wang, P., Fawzi, O., et al. Mathematical discoveries from program search with large language models. *Nature*, 625 (7995):468–475, 2024.

Shaar, E., Shaulov, A., Chechik, G., and Wolf, L. Adapting to the unknown: Training-free audio-visual event perception with dynamic thresholds. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3142–3151, 2025.

Shojaee, P., Meidani, K., Gupta, S., Farimani, A. B., and Reddy, C. K. Llm-sr: Scientific equation discovery via programming with large language models. *International Conference on Learning Representations*, 2025.

Singh, A., Fry, A., Perelman, A., Tart, A., Ganesh, A., El-Kishky, A., McLaughlin, A., Low, A., Ostrow, A., Ananthram, A., et al. Openai gpt-5 system card. *arXiv preprint arXiv:2601.03267*, 2026.

Tian, Y., Shi, J., Li, B., Duan, Z., and Xu, C. Audio-visual event localization in unconstrained videos. In *European Conference on Computer Vision*, pp. 247–263, 2018.

Tian, Y., Li, D., and Xu, C. Unified multisensory perception: Weakly-supervised audio-visual video parsing. In *European Conference on Computer Vision*, pp. 436–454. Springer, 2020.

Wang, R., Wang, B., Li, K., Zhang, Y., and Cheng, J. Drsr: Llm-based scientific equation discovery with dual reasoning from data and experience. *arXiv preprint arXiv:2506.04282*, 2025.

Wu, Y., Zhu, L., Yan, Y., and Yang, Y. Dual attention matching for audio-visual event localization. In *IEEE/CVF International Conference on Computer Vision*, pp. 6292–6300, 2019.

Wu, Y., Zhang, X., Wang, Y., and Huang, Q. Span-based audio-visual localization. In *ACM International Conference on Multimedia*, pp. 1252–1260, 2022.

Xia, S., Sun, Y., and Liu, P. Sr-scientist: Scientific equation discovery with agentic ai. *International Conference on Learning Representations*, 2026.

Xia, Y. and Zhao, Z. Cross-modal background suppression for audio-visual event localization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19989–19998, 2022.

Xu, H., Zeng, R., Wu, Q., Tan, M., and Gan, C. Cross-modal relation-aware networks for audio-visual event localization. In *ACM International Conference on Multimedia*, pp. 3893–3901, 2020.

Yu, J., Cheng, Y., Zhao, R.-W., Feng, R., and Zhang, Y. Mm-pyramid: Multimodal pyramid attentional network for audio-visual event localization and video parsing. In *ACM International Conference on Multimedia*, pp. 6241–6249, 2022.

Zhao, P., Zhou, J., Zhao, Y., Guo, D., and Chen, Y. Multimodal class-aware semantic enhancement network for audio-visual video parsing. In *AAAI Conference on Artificial Intelligence*, volume 39, pp. 10448–10456, 2025.

Zhou, J., Zheng, L., Zhong, Y., Hao, S., and Wang, M. Positive sample propagation along the audio-visual event line. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8436–8444, 2021.

Zhou, J., Guo, D., and Wang, M. Contrastive positive sample propagation along the audio-visual event line. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(6):7239–7257, 2022.

Zhou, J., Guo, D., Mao, Y., Zhong, Y., Chang, X., and Wang, M. Label-anticipated event disentanglement for audio-visual video parsing. In *European Conference on Computer Vision*, pp. 35–51. Springer, 2024a.

Zhou, J., Guo, D., Zhong, Y., and Wang, M. Advancing weakly-supervised audio-visual video parsing via segment-wise pseudo labeling. *International Journal of Computer Vision*, 132(11):5308–5329, 2024b.

Zhou, J., Guo, D., Guo, R., Mao, Y., Hu, J., Zhong, Y., Chang, X., and Wang, M. Towards open-vocabulary audio-visual event localization. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8362–8371, 2025a.

Zhou, J., Zhou, Z., Zhou, Y., Mao, Y., Duan, Z., and Guo, D. Clasp: Cross-modal salient anchor-based semantic propagation for weakly-supervised dense audio-visual event localization. In *AAAI Conference on Artificial Intelligence*, volume 40, pp. 13674–13682, 2026.

Zhou, Z., Zhou, J., Qian, W., Tang, S., Chang, X., and Guo, D. Dense audio-visual event localization under cross-modal consistency and multi-temporal granularity collaboration. In *AAAI Conference on Artificial Intelligence*, volume 39, pp. 10905–10913, 2025b.

Zhu, B., Lin, B., Ning, M., Yan, Y., Cui, J., Wang, H., Pang, Y., Jiang, W., Zhang, J., Li, Z., et al. Languagebind: Extending video-language pretraining to n-modality by language-based semantic alignment. *International Conference on Learning Representations*, 2024.

Appendix

Contents

A Adaptive Parameterization Details 12
A.1 Policy Architecture 12
A.2 Policy Learning 13
B Additional Experimental Details . 13
C Computational Cost 14
D Agent Details 14
D.1 Plan Agent 15
D.2 Design Agent 19
D.3 Oracle Agent 22
D.4 Policy Agent 25
E Discovered Formulation 28
F Additional Qualitative Results ... 29

A. Adaptive Parameterization Details

The policy network predicts video-specific parameters for a fixed symbolic formulation. It does not update the pretrained encoders or alter the final audio–visual decision rule. Instead, for each video, it outputs a bounded parameter vector $\boldsymbol{\theta}$ that is consumed by the selected formulation.

A.1. Policy Architecture

Input representation. For each video, the policy receives the frozen per-segment audio and visual embeddings

$$\mathbf{e}^a \in \mathbb{R}^{T\times d_a}, \qquad \mathbf{e}^v \in \mathbb{R}^{T\times d_v},$$

as defined in Section 3.1. The policy does not take category text embeddings as input; text embeddings are used by the symbolic formulation and the final threshold-based decision rule. This keeps the policy architecture independent of the category vocabulary and allows the same policy design to be used across different open-vocabulary category sets.

Architecture. The audio and visual embeddings are first projected into a shared hidden dimension d_h using modality-specific linear layers:

$$\tilde{\mathbf{e}}^a = \mathrm{Linear}_a(\mathbf{e}^a), \qquad \tilde{\mathbf{e}}^v = \mathrm{Linear}_v(\mathbf{e}^v).$$

We use $d_h = 256$ in all experiments.

Each modality is then processed by a modality-specific Transformer-style self-attention block:

$$\mathbf{h}^a = \mathrm{SelfAttn}_a(\tilde{\mathbf{e}}^a), \qquad \mathbf{h}^v = \mathrm{SelfAttn}_v(\tilde{\mathbf{e}}^v).$$

We use one self-attention block per modality. Each block consists of multi-head attention, residual connection, layer normalization, and a feed-forward network with GELU activation. All attention blocks use 4 heads, a 512-dimensional feed-forward layer, and dropout 0.1.

To exchange information between modalities, we apply bidirectional cross-attention:

$$\hat{\mathbf{h}}^v = \mathrm{CrossAttn}_v(\mathbf{h}^v, \mathbf{h}^a), \qquad \hat{\mathbf{h}}^a = \mathrm{CrossAttn}_a(\mathbf{h}^a, \mathbf{h}^v),$$

where the first argument provides the query sequence and the second argument provides keys and values. Thus, the visual stream attends to the audio stream, and the audio stream attends to the visual stream through separate cross-attention blocks.

The resulting temporal features are mean-pooled over segments:

$$\bar{\mathbf{h}}^a = \frac{1}{T}\sum_{t=1}^{T} \hat{\mathbf{h}}^a_t, \qquad \bar{\mathbf{h}}^v = \frac{1}{T}\sum_{t=1}^{T} \hat{\mathbf{h}}^v_t.$$

The pooled modality features are concatenated and fused:

$$\mathbf{h} = \mathrm{Dropout}\left(\mathrm{GELU}\left(W_f[\bar{\mathbf{h}}^a; \bar{\mathbf{h}}^v] + \mathbf{b}_f\right)\right).$$

A linear head predicts the mean of a Gaussian action distribution,

$$\boldsymbol{\mu} = W_\mu \mathbf{h} + \mathbf{b}_\mu \in \mathbb{R}^P,$$

where P is the number of parameters defined by the selected formulation.

Range-bounded parameterization. The policy operates in an unconstrained raw action space. For each parameter dimension i, the selected formulation specifies a valid range $[\theta_i^{\min}, \theta_i^{\max}]$. A raw action z_i is mapped into this range by

$$\theta_i = \theta_i^{\min} + \sigma(z_i)\left(\theta_i^{\max} - \theta_i^{\min}\right).$$

This guarantees that the predicted parameters remain inside the formulation-defined bounds.

During training, raw actions are sampled independently for each parameter dimension from a Gaussian centered at $\boldsymbol{\mu}$, as described in Appendix A.2. During evaluation, no sampling is used; the deterministic mean action $\boldsymbol{\mu}$ is mapped directly to $\boldsymbol{\theta}$ and passed to the selected formulation.

A.2. Policy Learning

We train the policy as a contextual bandit. For a video n, the context is the frozen audio–visual embedding sequence $(\mathbf{e}_n^a, \mathbf{e}_n^v)$, the action is the formulation parameter vector $\boldsymbol{\theta}$, and the reward is the task score obtained after applying the fixed formulation and the final audio–visual decision rule:

$$r_{n,k} = \mathcal{R}(F, \boldsymbol{\theta}_{n,k}; n).$$

In our implementation, $\mathcal{R}$ is per-video segment accuracy, chosen because it is fast to compute for each sampled action in the policy-gradient inner loop. Final results are reported using the full OV-AVEL metric suite.

Action sampling. The policy predicts the mean $\boldsymbol{\mu}_n \in \mathbb{R}^P$ in the raw action space. During training, we sample K raw action vectors $\mathbf{z}_{n,k} \in \mathbb{R}^P$. Each parameter dimension is sampled independently as

$$z_{n,k,i} \sim \mathcal{N}(\mu_{n,i}, \sigma_a^2), \qquad k = 1, \ldots, K, \quad i = 1, \ldots, P,$$

with fixed $\sigma_a = 0.3$. Each raw action vector is mapped to the formulation-defined range by the sigmoid transformation in Appendix A.1, producing $\boldsymbol{\theta}_{n,k}$.

Leave-one-out REINFORCE. Because the thresholding and agreement rule is non-differentiable, gradients do not flow through the reward. We therefore optimize the policy with REINFORCE. For each sampled raw action vector, we use the average reward of the other samples from the same video as a leave-one-out baseline:

$$b_{n,k} = \frac{1}{K-1} \sum_{j \neq k} r_{n,j}.$$

The resulting advantage is

$$\hat{A}_{n,k} = r_{n,k} - b_{n,k}.$$

The policy-gradient loss over a minibatch $\mathcal{B}$ is

$$\mathcal{L}_{\mathrm{PG}} = -\frac{1}{|\mathcal{B}|K} \sum_{n \in \mathcal{B}} \sum_{k=1}^{K} \hat{A}_{n,k} \sum_{i=1}^{P} \log \mathcal{N}(z_{n,k,i}; \mu_{n,i}, \sigma_a^2).$$

In implementation, the advantages are treated as constants, so gradients are taken only through the Gaussian log-probability terms. This objective directly optimizes the task reward induced by the same hard decision rule used at evaluation time, without requiring gradients through the thresholding and agreement rule.

Training details. We use $K = 4$ samples per video, AdamW with learning rate 1×10^{-4}, batch size 32, and train for 20 epochs. For each candidate formulation, the checkpoint with the highest validation accuracy is selected.

B. Additional Experimental Details

This section describes how we produced the two AV^{2}A baseline variants in Table 1. The dataset split, evaluation metrics, encoder choice, and Cue2Rule training protocol are given in the main paper and Appendix A; here we specify only the AV^{2}A adaptation and hyperparameter selection procedure.

Why we re-evaluate AV^{2}A. We re-evaluate AV^{2}A for two reasons. First, the original AV^{2}A paper does not report results on OV-AVEBench, so there is no directly comparable number for the benchmark used in Table 1. Second, our main experiments use ImageBind, whereas the original paper reports CLIP+CLAP and LanguageBind only. We tune the AV^{2}A global hyperparameters using metrics on the full OV-AVEBench test split used for Table 1.

AV^{2}A-Manual. AV^{2}A-Manual uses the original AV^{2}A decision rule with one manually selected ImageBind hyperparameter setting. We used the settings reported in the original AV^{2}A paper for CLIP+CLAP and LanguageBind as references:

$$\text{CLIP+CLAP: } (\alpha, \tau^0, \tau_r, \tau_f, \lambda) = (0.45, 0.75, 0.75, 0.50, 1.0),$$
$$\text{LanguageBind: } (\alpha, \tau^0, \tau_r, \tau_f, \lambda) = (0.50, 0.75, 0.75, 0.55, 2.5).$$

Starting from these values, we manually selected one ImageBind setting by monitoring Avg. on the validation set, and then kept the selected values fixed for all videos and categories when reporting the corresponding metrics. The selected hyperparameters are

$$\alpha = 0.45, \quad \tau^0 = 0.75, \quad \tau_r = 0.75, \quad \tau_f = 0.60, \quad \lambda = 3.0.$$

AV^{2}A-Grid. AV^{2}A-Grid uses the same AV^{2}A decision rule, but selects its global hyperparameters through a two-stage grid search on the test set. Since the AV^{2}A decision procedure is relatively costly to rerun for each hyperparameter combination, we use a coarse-to-refined search rather than a dense exhaustive search. The first stage evaluates a coarse grid over all hyperparameters, and the second stage evaluates a refined grid around promising regions from the first stage. We select the grid result by test Avg., and then keep the selected values fixed for all videos and categories when reporting test metrics.

Table 3. Search space and hyperparameters selected on the OV-AVEBench test split for AV^{2}A-Grid.

Hyperparameter	Stage-1 search values	Stage-2 search values	Selected value
α	$\{0.25, 0.50, 0.75\}$	$\{0.35, 0.45, 0.50, 0.60\}$	0.35
τ^0	$\{0.25, 0.50, 0.75\}$	$\{0.70, 0.75, 0.85, 0.95\}$	0.95
τ_r	$\{0.25, 0.50, 0.75\}$	$\{0.05, 0.25, 0.95\}$	0.05
τ_f	$\{0.25, 0.50, 0.75\}$	$\{0.25, 0.35, 0.50, 0.75, 0.95\}$	0.25
λ	$\{1, 2, 3, 4\}$	$\{2, 3, 4\}$	2.0

Interpretation of the two variants. AV^{2}A-Manual and AV^{2}A-Grid use the same decision rule, but differ in how the global hyperparameters are selected. The grid-selected setting differs most clearly from the manually selected one in τ^0 (0.75 vs. 0.95), τ_r (0.75 vs. 0.05), and τ_f (0.60 vs. 0.25), including near-boundary values such as 0.95 and 0.05.

C. Computational Cost

Table 4 reports the per-session wall-clock breakdown of the agentic search, averaged over the 3 search sessions in our main experiments. Each session runs 20 iterations on a single RTX 3090, with LLM calls (Plan, Design, Oracle, Policy agents) issued through gpt-5.4 in OpenAI Codex (v0.125.0, April 2026) at high reasoning effort.

Table 4. Per-session wall-clock breakdown of the agentic search (20 iterations), averaged over 3 sessions on a single RTX 3090.

Component	Time (h)	Share
Agent (LLM inference)	4.0	41%
Oracle search	1.8	18%
Policy training & evaluation	4.0	41%
Total	9.8	100%

The oracle search remains efficient despite using a Sobol sequence of 2^{17} samples per video, as ImageBind embeddings are precomputed and cached, and the per-sample formulation evaluation is parallelized on GPU.

D. Agent Details

This section documents the agentic search loop used to discover the symbolic formulation. The goal is not to prescribe a particular natural language style, but to make clear what information each agent receives, what computation it is allowed to perform, and what artifact is returned to the shared memory. We therefore describe each agent through three parts: *prompt input*, *operation*, and *output*.

Reporting convention. The examples below are representative excerpts from the actual search logs. Long file paths, repeated environment boilerplate, and non-essential token-level traces are omitted for readability. When a log contains private chain-of-thought or provider-specific hidden reasoning, we report only the agent-visible plan, executed commands, code snippets, and final analysis text. This makes the agent-visible inputs and outputs auditable without relying on unrecoverable internal reasoning.

Table 5. Input–operation–output view of one Cue2Rule search iteration. The LLM agents are used only during offline formulation discovery; test-time prediction uses the selected formulation and policy network.

Agent	Receives	Performs	Writes to memory
Plan	Prior candidate summaries, per-candidate records (formulations, parameter ranges, oracle scores, policy validation scores, agent reports).	Diagnoses the search state; decides whether to refine, abandon, or redirect formulation families; assigns parameter roles, range changes, ablation targets, and analysis requests.	Design directive, oracle ablation protocol, policy analysis protocol, and one-line planning summary.
Design	Plan directive, formulation interface, symbolic-form constraints, and prior formulation code patterns.	Proposes an executable formulation $F_i(\cdot;\theta)$; declares parameter roles and ranges Θ_i; revises the code until AST checks pass.	Candidate code, parameter ranges, parameter role descriptions, validity status, and design summary.
Oracle	Verified formulation F_i, parameter ranges Θ_i, training split, and requested ablations.	Searches per-video oracle parameters; estimates expressive capacity; repeats per-video oracle search for each ablation; analyzes parameter distributions, boundary saturation, and component impact.	Oracle score $O(F_i)$, ablation statistics, parameter diagnostics, raw-analysis artifacts, and oracle report.
Policy	Verified formulation F_i, parameter ranges Θ_i, trained policy outputs, validation metrics, and requested diagnostics.	Trains and evaluates π_{ϕ_i} for the candidate; analyzes policy-instantiated behavior, predicted parameter distributions, and failure modes.	Validation score $V(F_i, \pi_{\phi_i})$, predicted-parameter statistics, learnability diagnostics, and policy report.

D.1. Plan Agent

Prompt input. At the beginning of each search iteration, the Plan Agent receives the current iteration index and a compact memory of previous iterations. The memory contains the candidate identifier, validation and oracle metrics when available, parameter ranges, one-line reports from the Design, Oracle, and Policy Agents, and pointers to detailed reports. The prompt also states the fixed formulation interface and asks the Plan Agent to produce a directive for the next candidate rather than a formulation itself.

Operation. The Plan Agent does not execute model training or modify the formulation code directly. Instead, it reads the accumulated memory and identifies which aspects of the search should be changed in the next iteration. In particular, it can (i) retain structural motifs that have repeatedly improved oracle or validation performance, (ii) request that ineffective parameters be removed, repurposed, or range-adjusted, (iii) specify which parameters should be ablated by the oracle evaluation script, and (iv) ask the Oracle or Policy Agent to inspect specific failure modes in the next analysis. This makes the Plan Agent the controller of the search direction, while leaving executable formulation construction and empirical analysis to the specialized agents.

Output. The Plan Agent outputs a short directive consumed by the Design Agent and the later analysis agents. The directive typically includes the intended role of each parameter, the desired parameter ranges or range changes, the requested oracle ablations, and analysis questions to answer after the candidate is evaluated.

```
You are the Plan agent in a 4-agent loop
(Plan / Design / Oracle / Policy) that searches for
parameterized threshold formulations for OV-AVEL.

You do not write formulations yourself. Each iteration you produce
three artifacts upfront:
1. a natural-language strategy memo for the Design agent,
2. an evaluation protocol (JSON) controlling extra Sobol ablations,
3. a policy protocol (JSON) suggesting what the Policy agent should
   analyze after a 20-epoch training run.

Two evaluation tracks per iteration:
1. Oracle search: per-sample Sobol search inside PARAM_RANGES.
   train_oracle_frame_acc is the mean per-sample best frame_acc and
   represents the expressive ceiling on train.
2. Policy training: a sample-conditioned transformer policy is trained
   with REINFORCE for 20 epochs. The best epoch is selected by
   val_frame_acc and represents policy-learnable performance.

Read /work/history.jsonl first. Pay attention to:
oracle_delta_3iter, val_best_streak, val_oracle_gap, and summaries from
Plan, Oracle, and Policy. Then inspect 1--2 recent oracle and policy
reports to identify the proximate cause of stagnation or failure.

Modality balance is a prerequisite. If reports flag modality bypass
risk, e.g. visual_active_rate > 0.9 and visual_independence_index < 0.15,
the next memo should target the structural cause. If oracle balance is
healthy but learned policy balance collapses, treat the problem as
learnability rather than formulation expressivity.

Design contract:
- 10 learned params total.
- audio threshold uses params[0..4], visual uses params[5..9].
- each threshold is bias + sum(weight * primitive).
- params are outer weights only; no learned inner shifts, scales, or
  temperatures inside primitives.
- no control flow; use tensor operations.
- allowed inputs: a_sim, v_sim, a_emb, v_emb, a_t_emb, v_t_emb.
- asymmetric audio/visual feature sets are allowed.

The memo must name the move, justify it using history plus at least one
oracle and one policy report, assign a conceptual role and suggested
range to every slot params[0] ... params[9], and state what to avoid.
The final line must be:
<!-- summary: <one short sentence> -->

Write:
/work/strategy_memo/<output_file_memo>
/work/eval_protocol/<output_file_eval_protocol>
/work/policy_protocol/<output_file_policy_protocol>
```

Figure 6. Condensed input prompt for the Plan Agent.

```
## Sustained Support vs Burst Suppression

Iter 10 is the best visible policy run so far
(`val_frame_acc=0.6764`), so this is not the moment for a wholesale
reset: `val_best_streak` returned to `0`, and both Evaluator and Policy
reports say modality balance is healthy. But the current backbone has
stopped opening new headroom: `oracle_delta_3iter` is `-0.0023`,
effectively flat versus iter 7, and `val_oracle_gap` is still `0.1953`,
almost unchanged from iter 5. The latest reports point to a
slot-allocation problem, not a balance failure. Oracle says
`crosscheck_off` is still the strongest ablation (`mean marginal 0.095`),
`local_competition_off` is nearly dead (`0.004`), and the persistence
family is real but secondary. Policy agrees on the dead local terms, but
the new visual persistence pair mostly merges (`corr(v_own,v_shared)=0.925`),
and hard samples still lean on cross-check pressure instead of receiving
stronger realized temporal support.

Designer guidance: keep one simple cross-check term per modality and one
same-frame contender term per modality, but use the freed capacity to
separate sustained support from burst suppression. The temporal slots
should not be two views of the same clip prior. One slot should reward
categories whose support persists against clip-level rivals; the other
should punish categories whose current-frame spike outruns that support.

- `params[0]`: audio bias offset; baseline audio strictness. Suggested
  range `[-0.1, 0.5]`.
- `params[1]`: audio same-frame contender penalty; categories trailing
  the strongest audio/shared rival in the frame should pay. Suggested
  range `[0.0, 2.0]`.
        ...
- `params[9]`: visual cross-check penalty; raise visual threshold when
  audio does not selectively support the same class. Suggested range
  `[0.0, 1.8]`.

Avoid four failures. Do not spend slots `1` or `6` on raw local-rival
copies again; iter 10 says that route is dead. Do not make slots `7` and
`8` alternate versions of the same persistence statistic; the policy
already merged that family. Do not bring back normalized salience, frame
ambiguity, or raw clip means as near-constant offsets. Do not reopen
permissive low-bias or negative-relief behavior.

<!— summary: Keep balanced cross-check core, but replace dead local and
merged persistence slots with sustained-vs-burst roles. —>
```

Figure 7. Condensed Strategy directive produced by the Plan Agent for the Design Agent.

```
{
  "ablations": [
    {
      "name": "crosscheck_off",
      "params_zero": [2, 9],
      "intent": "Test whether one simple corroboration-deficit slot per
      modality remains the broad oracle core after the temporal family is
      split into sustained-support and burst-suppression roles."
    },
    {
      "name": "frame_contender_off",
      "params_zero": [1, 6],
      "intent": "Measure whether the replacement for dead raw local
      rivalry now adds real same-frame category discrimination instead of
      collapsing back into a negligible tail."
    },
    {
      "name": "sustain_off",
      "params_zero": [3, 7],
      "intent": "Isolate the value of clip-level sustained-support priors
      apart from burst suppression; these slots should reward repeated
      evidence against clip-level competitors."
    },
    {
      "name": "burst_off",
      "params_zero": [4, 8],
      "intent": "Check whether anti-burst penalties hit a distinct hard
      subset where transient peaks outrun persistent support, rather than
      duplicating the sustained-support slots."
    },
    {
      "name": "audio_temporal_off",
      "params_zero": [3, 4],
      "intent": "Measure whether audio gains from the full
      sustained-support plus burst-suppression pair, not just the retained
      same-frame and cross-check core."
    },
    {
      "name": "visual_temporal_off",
      "params_zero": [7, 8],
      "intent": "Focus on whether the visual temporal pair is now
      structurally meaningful and more distinct than the merged
      persistence pair from iteration 10."
    }
  ],
  "candidates_per_ablation": 819
}
```


Figure 8. Oracle ablation protocol produced by the Plan Agent.

```
{
  "analyses": [
    {
      "name": "burst_vs_sustain_usage",
      "intent": "Check whether slots 3/4 and 7/8 learn distinct
      sample-conditioned behavior rather than merging into near-duplicate
      temporal weights, especially on the visual side."
    },
    {
      "name": "hard_sample_temporal_realization",
      "intent": "For low-accuracy samples, compare realized winner-side
      sustained-support and burst-suppression contributions to the raw
      learned parameters to see whether the new temporal features actually
      fire where hard cases need them."
    },
    {
      "name": "crosscheck_dependence_shift",
      "intent": "Verify whether hard samples still fall back to unusually
      large cross-check barriers in slots 2/9, or whether load has shifted
      into the new temporal-shape slots as intended."
    },
    {
      "name": "balance_guard",
      "intent": "Confirm that learned modality balance stays close to the
      oracle pattern and that the new burst-suppression terms do not
      recreate a permissive visual branch or an overly silent audio branch."
    }
  ]
}
```

Figure 9. Policy analysis protocol produced by the Plan Agent.

D.2. Design Agent

Prompt input. The Design Agent receives the Plan Agent's strategy memo, the formal formulation interface, previous formulation files as code-pattern references, and the symbolic-form constraints. It does not receive the full search history or oracle/policy reports directly; those signals are summarized by the Plan Agent. The interface specifies the available inputs—frozen audio embeddings, visual embeddings, audio text embeddings, and visual text embeddings—and requires the function to return audio and visual threshold matrices. The prompt also states the final decoding rule: categories are filtered by modality-specific thresholds, the top surviving category is selected in each modality, and the segment is assigned a foreground category only when the two modalities agree.

Operation. The Design Agent writes an executable Python implementation of the candidate formulation together with a bounded range for every parameter. Each candidate is checked by an AST-based validator before evaluation. The validator enforces the symbolic-form constraints used in the main paper: the formulation must be branch-free, linear in the parameter vector, and free of arbitrary numeric constants outside the allowed structural set. If the validator rejects the candidate, the diagnostic message is returned to the Design Agent, which revises the function within the same iteration until a valid candidate is produced or the iteration fails.

Output. The Design Agent outputs a Python formulation file, a parameter range table, parameter names, and a short natural-language rationale for the candidate. The rationale records why each term was introduced and which Plan Agent request it is intended to address. The submitted code, rather than the rationale, is the object checked by the AST validator and evaluated by the Oracle and Policy stages.

```
You are designing parameterized threshold formulations for OV-AVEL.

The Design Agent's job is to implement the Plan Agent's strategy memo
faithfully under a hard symbolic-form contract. It does not read the full
search history or oracle results directly; those are summarized by the
Plan Agent in strategy_memo.md. Prior formulation files may be read only
as code-pattern references, not copied verbatim.

Evaluation protocol:
- A per-sample oracle search is run on the train split.
- Roughly 131K Sobol parameter candidates are sampled inside the
  bounding box declared by PARAM_RANGES.
    ...

The generated Python file must define:
NAME, DESCRIPTION, NUM_PARAMS, PARAM_RANGES, PARAM_NAMES,
params_to_thresholds(...), and params_to_thresholds_batch(...).

The formulation receives:
a_emb, v_emb, a_t_emb, v_t_emb, a_sim, v_sim, params.
It returns audio and visual threshold matrices. The external decoder then
lets each modality vote for its top category above threshold, and predicts
a foreground category only when both modalities vote for the same class.

Symbolic-form constraints:
- NUM_PARAMS = 10.
- Audio threshold uses params[0..4]; visual threshold uses params[5..9].
- Each modality must be bias + at most four outer-weighted primitive
  terms.
- Learned params may only appear as outer weights or biases; they cannot
  be passed into primitives as shifts, scales, or temperatures.
- At most five helper primitives are allowed.
- Helper primitives and main functions have strict statement-count caps.
- No control flow, comprehensions, lambdas, or in-place operations.
    ...

Available feature axes include per-frame category statistics, joint
audio-visual signals, same-frame competitive gaps, cross-modal coupling,
frame-level scalars, per-category temporal aggregates, and smooth
standardized or sharpened surrogates.

Avoid common failure modes: one-sided threshold motion, uniform broadcast
features dominating the category-specific decision, modality rubber-stamp
behavior, and redundant slots assigned to near-identical primitives.

Inputs come from separate audio, visual, and text embedding spaces.
Do not take dot products across non-aligned spaces. Use a_sim and v_sim
as the default evidence signals.

Write the formulation to output/<output_file>. Before finishing, check
that python -m py_compile succeeds and that the batched function matches
the single-sample function on random parameters.
```

Figure 10. Condensed input prompt for the Design Agent.

```python
NAME = "Blended Contenders With Sustained Burst Control"
DESCRIPTION = "Each modality keeps a simple opposite-modality cross-check,
but replaces raw same-frame self-rivalry with a contender term on blended
own-plus-shared support. The temporal slots are separated into clip-level
sustained-support rivalry and a signed burstiness term that penalizes
frame spikes unsupported by persistent evidence."

NUM_PARAMS = 10
PARAM_RANGES = [
    (-0.1, 0.5), (0.0, 2.0), (0.0, 2.0), (0.0, 1.8), (0.0, 1.8),
    (0.0, 0.8), (0.0, 2.1), (0.0, 2.0), (0.0, 2.0), (0.0, 1.8),
]
PARAM_NAMES = [
    "a_bias",
    "a_blended_contender_penalty",
    "a_visual_cross_check_penalty",
    "a_sustained_support_penalty",
    "a_burstiness_penalty",
    "v_bias",
    "v_blended_contender_penalty",
    "v_sustained_support_penalty",
    "v_burstiness_penalty",
    "v_audio_cross_check_penalty",
]

def _shared_support(a_sim, v_sim):
    return torch.minimum(a_sim, v_sim)

def _rival_gap(sim):
    top2 = sim.topk(2, dim=-1)
    top1 = top2.values[..., :1]
    top2v = top2.values[..., 1:2]
    winner = torch.nn.functional.one_hot(
        top2.indices[..., 0], num_classes=sim.shape[-1]
    ).to(sim.dtype)
    return top1 + (top2v - top1) * winner - sim

def _temporal_rival(sim):
    clip = sim.mean(dim=-2, keepdim=True)
    return _rival_gap(clip)

def _burst_gap(sim, support):
    sustain = support.mean(dim=-2, keepdim=True)
    return sim - sustain

def params_to_thresholds(...):
    joint = _shared_support(a_sim, v_sim)
    a_support = 0.5 * (a_sim + joint)
    v_support = 0.5 * (v_sim + joint)
    a_contend = _rival_gap(a_support)
    a_cross = _rival_gap(v_sim)
    a_sustain = _temporal_rival(a_support)
    a_burst = _burst_gap(a_sim, a_support)
    v_contend = _rival_gap(v_support)
    v_sustain = _temporal_rival(v_support)
    v_burst = _burst_gap(v_sim, v_support)
    v_cross = _rival_gap(a_sim)
    a_thresh = params[0] + params[1] * a_contend + params[2] * a_cross \
        + params[3] * a_sustain + params[4] * a_burst
    v_thresh = params[5] + params[6] * v_contend + params[7] * v_sustain \
        + params[8] * v_burst + params[9] * v_cros
    return a_thresh, v_thresh
```

Figure 11. Condensed formulation code produced by the Design Agent.

D.3. Oracle Agent

Prompt input. The Oracle Agent receives the verified formulation, its parameter ranges, the Plan Agent's requested ablations, and the raw outputs of the oracle search. For every training video, the oracle search records the best parameter vector found by Sobol sampling and the corresponding per-video reward. For requested ablations, it also records the performance obtained after zeroing or disabling designated parameters and re-optimizing the remaining parameters.

Operation. The Oracle Agent analyzes the expressive capacity of the formulation under ideal per-video parameters. It is allowed to write and execute custom analysis scripts over the oracle search outputs, for example to compute parameter distribution statistics, identify boundary-pinned parameters, measure per-parameter ablation effects, inspect correlations between parameters and video-level performance, and compare successful and failed videos. These scripts are not part of the final model; they serve only to produce diagnostic evidence for the search memory.

Output. The Oracle Agent returns a report summarizing what the oracle results imply about the formulation's structure. The report distinguishes between high-level performance, parameter-level findings, and diagnostic signals used by the next Plan Agent. A one-line summary is appended to the compact memory, while the full report remains available for later inspection.

```
You are the Oracle agent in the multi-agent loop. Your job is to read
this iteration's oracle results, including raw per-sample marginal-value
arrays, and write a concise report for the next Plan Agent. Diagnose
patterns from the data; do not propose new formulations.

Available files:
- /work/results.json: formulation metadata, train oracle metrics,
  parameter statistics, and ablation summaries.
- /work/raw.npz: raw arrays including full_per_sample_scores,
  category_ids, ablation_per_sample_scores, ablation_marginal,
  oracle_audio_votes, and oracle_visual_votes.
- /work/formulation.py: evaluated formulation code.
- /work/strategy_memo.md: Plan Agent strategy and slot mapping.
- /work/eval_protocol.json: ablation hypotheses.

Workflow:
1. Read results.json, eval_protocol.json, strategy_memo.md, and
   formulation.py.
2. Load raw.npz and always run the modality-balance check.
3. Run at most three additional custom analyses over the raw arrays,
   such as per-category marginal breakdown, cross-ablation correlation,
   top-k affected-sample overlap, or distribution-shape checks.
4. Use Python computations for every claim beyond pre-computed summary
   statistics.

Mandatory oracle modality-balance quantities:
- audio_active_rate = mean(audio vote is not -1)
- visual_active_rate = mean(visual vote is not -1)
- both_active_rate = mean(both modalities vote)
- agreement_when_both_active = agreement rate conditional on both voting
- visual_independence_index = among visual votes, fraction where visual
  differs from audio or audio is silent

The report must contain:
1. Header copied from the strategy memo.
2. Baseline oracle result and a concise reading of parameter statistics.
3. One paragraph per ablation: hypothesis, marginal distribution
   (mean, median, p10, p90, fraction_zero), and verdict.
4. Mandatory modality-balance section.
5. Up to three custom analyses with method, numerical result, and
   interpretation.
6. Optional cross-cutting patterns.

Do not recommend what to try next, speculate about validation or test
generalization, restate the whole formulation code, or invent metrics.
The final line must be a summary marker
<!-- summary: <one short sentence> -->
```

Figure 12. Condensed input prompt for the Oracle Agent.

```
### Sustained Support vs Burst Suppression

Full-grid train oracle ceiling is `frame_acc=0.896`
(`seg_f1=0.856`, `eve_f1=0.833`). The two bias slots are the tightest
(`a_bias std=0.107`, `v_bias std=0.255`), while all eight penalty slots
stay broad (`std≈0.45-0.65`) and most run from near zero to near their
upper bounds, so the oracle is still using sample-specific penalty
mixtures rather than a single fixed regime.

#### `crosscheck_off`
This tests whether the simple opposite-modality corroboration deficit is
still the broad core after splitting temporal roles. Marginal value is
large but heterogeneous: mean `0.080`, median `0.000`, `p90=0.300`,
`fraction_zero=0.772`. Verdict: strongly supported. The effect is zero
on most samples, but the active tail is substantial enough to account for
the largest oracle drop.

...

### Custom analyses

Positive-tail shape: `crosscheck_off` has `3001/13182` positive samples;
among positives its median marginal is `0.200`, `p90=0.800`, `p99=1.000`.
`frame_contender_off` has only `748/13182` positive samples, with
positive-sample median `0.100` and `p99=0.200`. Thus, cross-check is
active on about four times as many samples and reaches much larger
per-sample gains.

Temporal alignment: `burst_off` vs. `audio_temporal_off` has Pearson
`0.848` and top-10% Jaccard `0.569`, while `sustain_off` vs.
`visual_temporal_off` has Pearson `0.706` and Jaccard `0.422`. Thus, the
audio temporal pair is driven mostly by the burst subset, whereas the
visual temporal pair aligns more with sustained-support losses.

Per-category marginal values: category `20` is high in every major
ablation (`crosscheck_off=0.240`, `burst_off=0.110`,
`audio_temporal_off=0.117`, `visual_temporal_off=0.065`,
`sustain_off=0.031`), and categories `15` and `38` are elevated in four
of the five major ablations. The effects are therefore not confined to
one-off sample noise.

### Cross-cutting patterns

Cross-check remains the only broad, high-amplitude term family. The
temporal split is real, but its realized weight is asymmetric: audio-side
temporal value tracks burst suppression most closely, while visual-side
temporal value looks closer to sustained support and remains weaker
overall.

<!— summary: Cross-check is still the broad core; temporal value is real
but split asymmetrically across audio burst and weaker visual sustain. —>
```

Figure 13. Condensed output report of the Oracle Agent.

D.4. Policy Agent

Prompt input. The Policy Agent receives the verified formulation, its parameter ranges, the trained policy checkpoint selected by validation accuracy, and the stored per-video outputs produced by that checkpoint. These outputs include predicted parameter vectors, segment-level predictions, per-video rewards, and aggregate validation metrics. When oracle results are available for the same candidate, the prompt may also include pointers to the oracle parameter statistics so that the Policy Agent can compare what the policy predicts against what the oracle prefers.

Operation. The Policy Agent analyzes whether the formulation is learnable by the policy network, rather than merely expressive under oracle parameters. It may write and execute scripts to inspect the distribution of predicted parameters, detect collapsed or boundary-saturated policy outputs, measure the gap between oracle and policy performance, identify videos where the policy improves or fails, and summarize modality-specific prediction patterns. The resulting diagnostics are used to determine whether a high-oracle formulation is practically useful after policy training.

Output. The Policy Agent returns a report focused on policy learnability. The report states whether the predicted parameters use the available ranges, whether the learned policy collapses to a narrow region of the parameter space, which learnability failure modes dominate, and which diagnostic signals should be passed to the next planning step. As with the Oracle Agent, the full report is stored for inspection and a one-line summary is appended to the compact memory.

```
You are the Policy agent in the 4-agent loop. Your role is learnability
diagnosis, complementary to the Oracle Agent's expressivity diagnosis.
The Oracle Agent has analyzed the oracle ceiling and ablation marginals;
your job is to read the actual policy-training outputs and answer whether
the sample-conditioned policy captured what the oracle found expressive.

Important: the policy is sample-conditioned. A transformer over the video
context outputs a different parameter vector for every validation sample.
There is no single shared parameter vector. Any parameter mean, standard
deviation, clipping rate, or histogram must be computed from raw
per-sample arrays.

Available files:
- /work/policy_results.json: best validation epoch, aggregate validation
  metrics, close/open group metrics, per-epoch validation curve, parameter
  names, and parameter ranges.
...

Workflow:
1. Read policy_results.json, policy_protocol.json, oracle_report.md,
   strategy_memo.md, and formulation.py.
2. Load policy_raw.npz, and raw.npz if oracle comparison is needed.
3. Address every requested analysis in policy_protocol.json, or choose
   up to three analyses if the protocol is empty.
4. Cover three layers when possible: training trajectory, per-slot
   learnability, and realized policy behavior.
5. Run Python computations for every quantitative claim beyond values
   already present in policy_results.json.

Required learned-modality-balance quantities:
- audio_active_rate and visual_active_rate from per-sample validation
  votes,
- both_active_rate,
- agreement_when_both_active,
- visual_independence_index.
Compare these to the oracle balance from train raw arrays at the level of
distributional shape, not exact sample matching.

The report must contain:
1. Header copied from the strategy memo.
2. Training trajectory: best epoch, best validation metrics, close/open
   splits, and convergence shape.
3. Oracle-vs-policy gap: train oracle frame_acc versus validation
   per-sample policy frame_acc mean, with the split mismatch stated.
4. Mandatory learned modality balance.
5. One subsection per requested policy-protocol analysis.
6. Optional custom analyses and cross-cutting patterns.

Do not recommend new formulations, speculate about test generalization,
restate the JSON verbatim, or invent metrics. The final line must be:
<!-- summary: <one short sentence> -->
```

Figure 14. Condensed input prompt for the Policy Agent.

```
## Sustained Support vs Burst Suppression

**Training trajectory**
Best val epoch was `14`, with `frame_acc=0.680`, `seg_f1=0.585`,
`eve_f1=0.516`, `avg=0.594`. Close/open stayed matched at the selected
checkpoint (`frame_acc 0.678/0.681`, `avg 0.596/0.593`). `val_frame_acc`
climbed from `0.648` to `0.680` by epoch 14, then slid to `0.668` at
epoch 20, so this run looks like a mid-run plateau rather than an
unfinished upward trend.

**Oracle vs trained gap**
Oracle train ceiling is `0.896` frame accuracy, while the learned policy
averages `0.680` per-sample frame accuracy on val; the split-mismatched
gap is `0.216`. The val distribution is wide: median `0.800`, `p10=0.000`,
`p90=1.000`.

**Learned modality balance**
Learned policy rates are `audio_active=0.435`, `visual_active=0.658`,
`both_active=0.380`, `agreement_when_both=0.941`,
`visual_independence=0.457`. Oracle shape on train was
`0.530/0.525/0.316/0.952/0.427`. Verdict: not a full bypass, but balance
drifted toward a quieter audio branch and a more permissive visual branch
than the oracle.

...

**Custom analysis: sample-conditioning vs collapse**
Method: compare policy val std to oracle per-sample std slot by slot.
Result: `a_sustain`, `a_burst`, and `v_sustain` retain `0.794x`, `0.936x`,
and `0.862x` of oracle dispersion, but `a_visual_cross_check` drops to
`0.119x` and `v_burst` to `0.352x`. Verdict: sample-conditioning is real,
but selective collapse remains in one broad core slot and one visual
temporal slot.

**Cross-cutting patterns**
This matches the Oracle Agent's asymmetry: cross-check is still the core,
audio temporal structure is learnable, and the main learned failure is
not temporal duplication but temporal under-realization on the visual
burst branch plus near-saturation of audio cross-check.

<!-- summary: Policy is sample-conditioned, but it over-saturates audio
cross-check and under-learns visual burst suppression. -->
```

Figure 15. Condensed output report of the Policy Agent.

E. Discovered Formulation

This section gives the final threshold-producing formulation selected by the search procedure. We use the notation from Section 3.1: s_{tc}^m denotes the cosine similarity between modality $m \in \{a, v\}$ at segment t and category text embedding c. The formulation produces thresholds $\boldsymbol{\tau}^a$ and $\boldsymbol{\tau}^v$; the subsequent filtering, top-category selection, and audio–visual agreement rule are exactly the fixed decision procedure defined in Section 3.1.

The formulation first computes, for each segment-category pair, the lower of the audio-text and visual-text similarity scores:

$$q_{tc} = \min(s_{tc}^a, s_{tc}^v),$$

and averages this value with each modality's original similarity score:

$$b_{tc}^a = \frac{1}{2}(s_{tc}^a + q_{tc}),$$
$$b_{tc}^v = \frac{1}{2}(s_{tc}^v + q_{tc}).$$

Here q_{tc} is high only when both modalities assign a high similarity to category c at segment t, and is otherwise limited by the lower of the two modality scores. Equivalently, the lower of the two modality scores is unchanged, while the higher score is replaced by the average of the two scores.

We also compute the temporal average of these scores for each category:

$$\bar{b}_c^m = \frac{1}{T}\sum_{t=1}^{T} b_{tc}^m, \qquad m \in \{a, v\}.$$

We use $\mathcal{G}$ as shorthand for comparing category c with the strongest category other than c. For the segment-level score tables used below, namely $\mathbf{b}^a$, $\mathbf{b}^v$, $\mathbf{s}^a$, and $\mathbf{s}^v$, this operation is

$$\mathcal{G}(\mathbf{u})_{tc} = \max_{c' \neq c} u_{tc'} - u_{tc}.$$

Here $\mathbf{u}$ denotes whichever one of these score tables is being used. For the clip-level averaged scores, we apply the same category comparison directly to $\bar{\mathbf{b}}^a$ and $\bar{\mathbf{b}}^v$:

$$\mathcal{G}(\bar{\mathbf{b}}^m)_c = \max_{c' \neq c} \bar{b}_{c'}^m - \bar{b}_c^m, \qquad m \in \{a, v\}.$$

The value is positive when category c is below some competing category, and negative when category c is higher than all competing categories.

We also define the difference between the current modality-specific similarity score and the corresponding clip-level averaged score:

$$\beta_{tc}^a = s_{tc}^a - \bar{b}_c^a,$$
$$\beta_{tc}^v = s_{tc}^v - \bar{b}_c^v.$$

The audio and visual thresholds are

$$\tau_{tc}^a = \theta_0 + \theta_1 \mathcal{G}(\mathbf{s}^v)_{tc} + \theta_2 \mathcal{G}(\mathbf{b}^a)_{tc} + \theta_3 \mathcal{G}(\bar{\mathbf{b}}^a)_c + \theta_4 \beta_{tc}^a, \tag{1}$$
$$\tau_{tc}^v = \theta_5 + \theta_6 \mathcal{G}(\mathbf{s}^a)_{tc} + \theta_7 \mathcal{G}(\mathbf{b}^v)_{tc} + \theta_8 \mathcal{G}(\bar{\mathbf{b}}^v)_c + \theta_9 \beta_{tc}^v, \tag{2}$$

Table 6 lists the parameter ranges assigned by the Design Agent.

Each threshold consists of a bias and four score-dependent quantities. The terms $\mathcal{G}(\mathbf{s}^v)_{tc}$ in the audio threshold and $\mathcal{G}(\mathbf{s}^a)_{tc}$ in the visual threshold use the opposite modality's category scores at the same segment. If the opposite modality scores category c above its competitors, the quantity is negative and the threshold is reduced. If the opposite modality scores another category higher, the threshold is increased.

Table 6. Parameter ranges for the discovered formulation.

Parameter	Multiplied term	Range
θ_0	audio bias	$[-0.1,\ 0.5]$
θ_1	$\mathcal{G}(\mathbf{s}^v)_{tc}$	$[0.0,\ 2.0]$
θ_2	$\mathcal{G}(\mathbf{b}^a)_{tc}$	$[0.0,\ 2.0]$
θ_3	$\mathcal{G}(\bar{\mathbf{b}}^a)_c$	$[0.0,\ 1.8]$
θ_4	β^a_{tc}	$[0.0,\ 1.8]$
θ_5	visual bias	$[0.0,\ 0.8]$
θ_6	$\mathcal{G}(\mathbf{s}^a)_{tc}$	$[0.0,\ 1.8]$
θ_7	$\mathcal{G}(\mathbf{b}^v)_{tc}$	$[0.0,\ 2.1]$
θ_8	$\mathcal{G}(\bar{\mathbf{b}}^v)_c$	$[0.0,\ 2.0]$
θ_9	β^v_{tc}	$[0.0,\ 2.0]$

The terms $\mathcal{G}(\mathbf{b}^a)_{tc}$ and $\mathcal{G}(\mathbf{b}^v)_{tc}$ compare category c with the strongest competing category at the same segment after the modality score has been averaged with q_{tc}. The Design Agent assigned non-negative ranges to the corresponding coefficients, so this part raises the threshold when a competing category is larger under these averaged scores and lowers the threshold when category c is larger.

The terms $\mathcal{G}(\bar{\mathbf{b}}^a)_c$ and $\mathcal{G}(\bar{\mathbf{b}}^v)_c$ apply the same category comparison after averaging b^m_{tc} over all temporal segments of the video. This part therefore adjusts the threshold using category competition at the clip level rather than only at the current segment.

Finally, β^a_{tc} and β^v_{tc} compare the current modality-specific similarity score with the corresponding score averaged over all temporal segments of the video, $\bar{b}^m_c$. Since the Design Agent also assigned non-negative ranges to these coefficients, this part raises the threshold when the current modality-specific score is above this temporal average, and lowers it when the current score is below it.

F. Additional Qualitative Results

To complement Fig. 5, we provide twelve further successful predictions covering temporal patterns not represented in the main figure: post-event modality fade, persistent visual with sparse audio, mid-segment modality switch, late-onset audio, single-pulse audio, multi-event sequences with short gaps, full-clip continuous events, and correct background classification including AV-agreement rejection of single-modality activations. Across all panels, Cue2Rule's per-segment thresholds adjust to each modality's local temporal signature, and the AV-agreement rule resolves modality disagreements consistently with the ground truth. *Conversely*, we also illustrate four failure modes that surface during qualitative inspection: a missed within-event background gap, delayed event boundaries, semantically close category confusion, and under-detection of repeated short events.

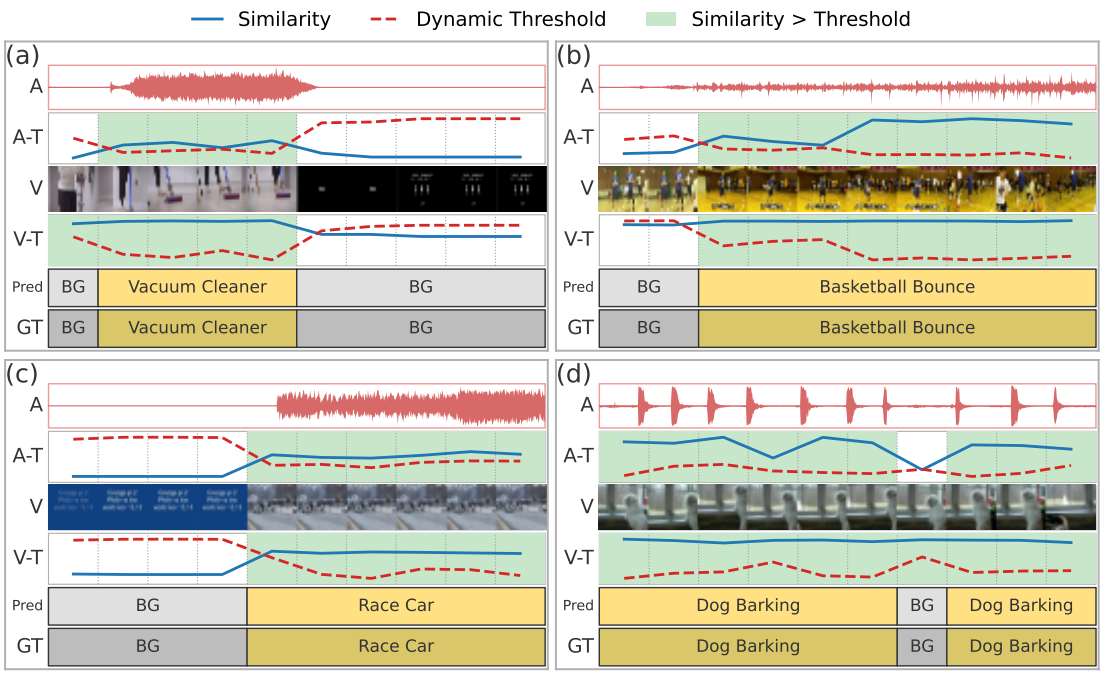

Figure 16. **Additional qualitative results (set 1).** (a) *Vacuum Cleaner*: both modalities fade after the event; thresholds rise to mark the post-event background. (b) *Basketball Bounce*: visual stays consistent while audio builds up from silence; the audio threshold lowers accordingly, capturing the onset. (c) *Race Car*: the visual switches to the target as audio fades in mid-clip; both thresholds drop, marking the event window. (d) *Dog Barking*: a brief silent gap within the event is detected as background as the audio threshold rises above A-T there.

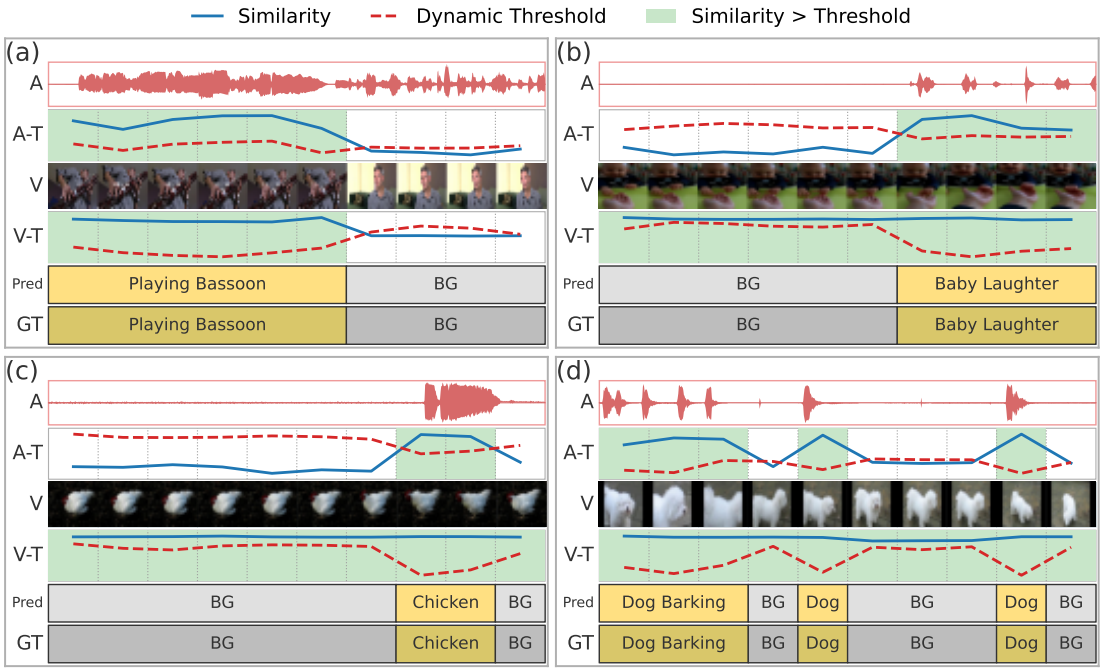

Figure 17. **Additional qualitative results (set 2).** (a) *Playing Bassoon*: strong audio at start fades out together with the visual; thresholds rise jointly afterward. (b) *Baby Laughter*: audio onset is delayed; thresholds drop only when both modalities support the event. (c) *Chicken*: a brief late audio pulse aligns with a target-consistent visual; the threshold isolates the short event window precisely. (d) *Dog Barking*: multiple disjoint events with a brief background gap; thresholds track audio pulses while the persistent visual stays above its threshold.

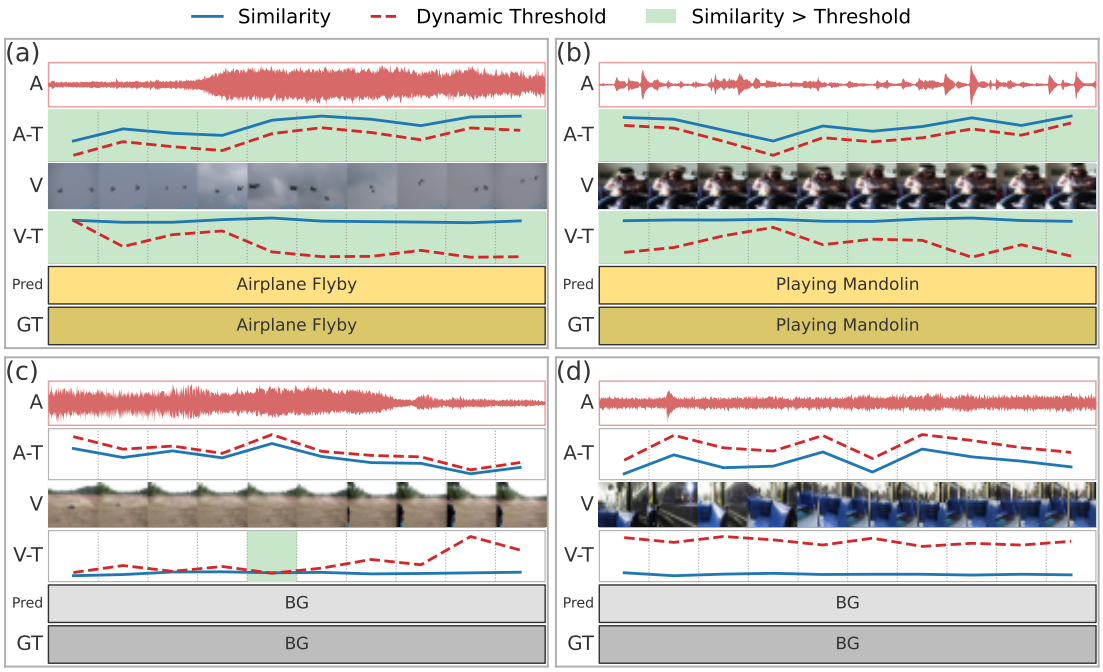

Figure 18. **Additional qualitative results (set 3).** (a) *Airplane Flyby*: a full-clip event with gradual audio buildup; both thresholds stay below A-T and V-T throughout, sustaining continuous detection. (b) *Playing Mandolin*: a stable full-clip event with clean cues; both similarities remain above their thresholds throughout, with no spurious flips. (c) *Background*: a brief V-T spike does not trigger a false positive; A-T stays below its threshold, so the AV-agreement rule rejects the segment. (d) *Background*: both A-T and V-T stay below their thresholds throughout, keeping the entire clip correctly classified as background.

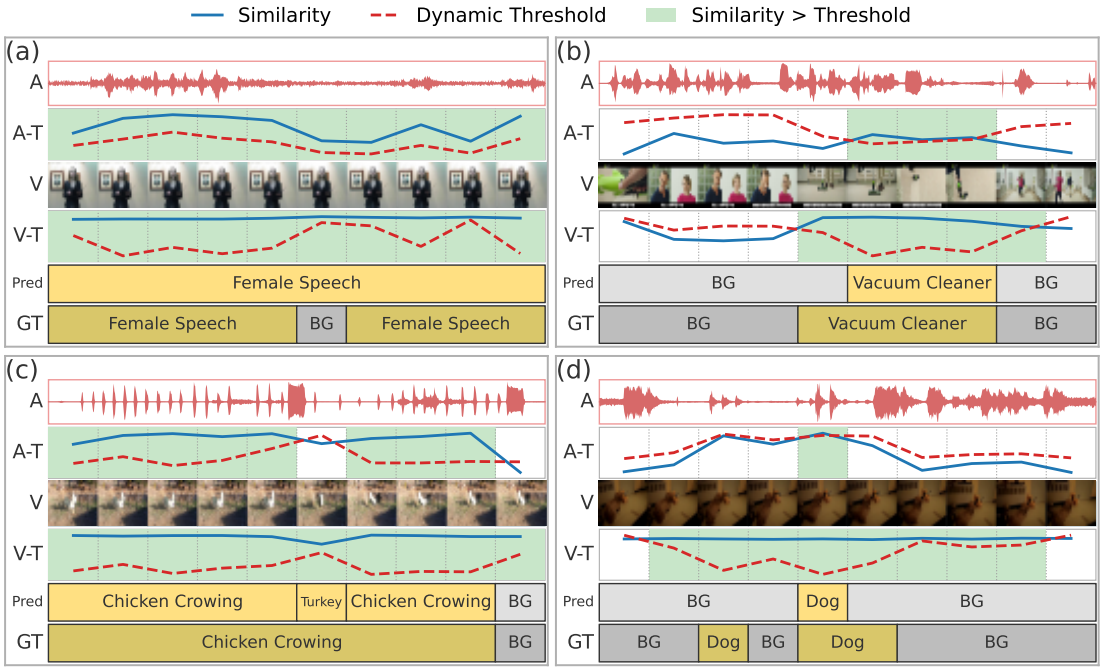

Figure 19. **Failure cases.** (a) *Female Speech*: a brief background gap within the speech is missed; the audio threshold tracks the dipping A-T instead of rising, so neither modality flips. (b) *Vacuum Cleaner*: the event window is detected with delayed onset and early offset; both thresholds drop only once the audio is fully established. (c) *Chicken Crowing*: a mid-clip segment is briefly misclassified as a semantically close category (*Turkey*); the encoder's similarity favors the wrong class within that window. (d) *Dog Barking*: a few boundary segments around the event are missed; the audio threshold tracks A-T closely and crosses above it at minor dips.