# LEARNING EVIDENCE-ADAPTIVE COMPOSITION FOR COMPOSITIONAL GENERALIZATION

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Modern vision systems increasingly combine pretrained models that provide complementary yet uncertain evidence about the same instance. However, existing approaches combine such evidence using fixed-parameter fusion functions or predefined composition rules, even when the appropriate interaction depends on the surrounding evidence. This limitation becomes more pronounced in multi-source settings, where an intermediate composition can change how subsequent evidence should be interpreted. We therefore ask whether a model should learn not only what evidence to combine, but also how that evidence should be composed. We introduce AdaCompose, a general framework for evidence-adaptive composition in which the complete evidence states being composed determine the parameters of the composition operator itself. The generated operator is applied to local evidence representations to produce pairwise relations and a new evidence state, which can then determine the operator for subsequent compositions. AdaCompose incorporates within-source permutation consistency, prediction-score gating, controlled low-rank operator adaptation, bounded relations, and a common state interface that enables recursive composition. We evaluate AdaCompose on video anomaly detection, compositional zero-shot recognition, and visual question answering, covering context-dependent recognition, unseen semantic combinations, and multimodal reasoning. Across these tasks, these evaluations assess generalization to unseen attribute-object combinations in CZSL and the integration of heterogeneous evidence in VAD and multimodal question answering. These results demonstrate the potential of making the composition operator itself dependent on the evidence being composed, providing a general mechanism for composing heterogeneous evidence beyond fixed composition rules.

## 1 INTRODUCTION

Modern vision systems combine evidence from pretrained models. A video may provide evidence about actions, objects, and scenes, while visual question answering may require jointly interpreting a question, textual context, and an image (Ding et al., 2025; Lu et al., 2022). Such sources provide complementary but uncertain evidence, and the relevance of one hypothesis can depend on the other evidence available during composition (Wang et al., 2023). The challenge is therefore not simply to combine more evidence, but to determine *how the evidence should be composed* (Schug et al., 2025). Existing adaptive mechanisms already allow computation to depend on its inputs. Attention adapts interactions between representations (Vaswani et al., 2017), feature modulation conditions intermediate features on context (Perez et al., 2018), and conditional parameterization can generate parameters from an input-dependent code (Ha et al., 2017). However, these mechanisms do not by themselves define a reusable composition abstraction in which the *complete evidence states being composed determine the operator used to compose them*. When the appropriate composition rule depends on the surrounding evidence, applying the same learned operator to a local hypothesis pair can be restrictive. This issue becomes more pronounced in multi-source settings: an intermediate composition changes the evidence available to later stages, and therefore can change how subsequent evidence should be interpreted. This motivates a fundamental question: *should a model learn not only what evidence to combine, but also how that evidence should be composed?*

We introduce AdaCompose, a framework for *evidence-adaptive composition*. Given two evidence states, AdaCompose uses their complete states to generate the parameters of the composition op-



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

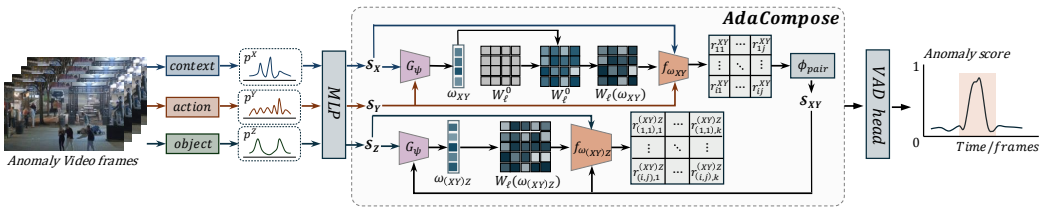



Figure 1: AdaCompose: the evidence determines the composition rule, and the composition determines the next rule. Illustrated for video anomaly detection (VAD) with context, action, and event evidence, each source $X$ is represented by an evidence state $\mathcal{S}_X = (\boldsymbol{H}_X, \boldsymbol{z}_X, \tau_X)$, containing local hypothesis states, an order-invariant global summary, and a type embedding. Given two states, $G_\psi$ generates an operator code $\omega_{XY}$ from their complete evidence, which modulates shared base weights to define a relation function shared across all hypothesis pairs. Score-gated pair states and their relations are aggregated into the composed state $\mathcal{S}_{XY}$. When a third source $\mathcal{S}_Z$ arrives, $\mathcal{S}_{XY}$ and $\mathcal{S}_Z$ generate a new code $\omega_{(XY)Z}$ and hence a new composition rule. Thus, the operator is shared across hypothesis pairs but regenerated from the current evidence at each step, enabling recursive composition. The VAD head produces window-level logits and frame-wise anomaly scores.

erator itself. The generated operator is then applied to local evidence representations to produce pairwise relations and a new evidence state. Thus, the evidence being combined determines the rule used to combine it. This is the central distinction of AdaCompose: rather than adapting representations within a fixed composition mechanism, we make the *composition mechanism itself evidence-dependent* (Schug et al., 2025). AdaCompose further treats each composition result as reusable evidence. Because the output state has the same structural interface as an input state, the same composition cell can be applied when another source arrives. A new operator is generated from the intermediate composition and the new evidence, so earlier compositions can change how later evidence is interpreted. This provides a recursive form of composition without introducing a separate architecture for each number of sources or composition depth. The composition is ordered: AdaCompose does not assume commutativity or associativity, so the chosen composition structure remains part of the reasoning process. To make evidence-adaptive composition structured and reusable, we incorporate the relevant properties directly into the architecture. Shared set processing makes the representation consistent to within-source hypothesis ordering, while multiplicative prediction-score gating gives the model an explicit mechanism for controlling local evidence strength. Evidence-conditioned low-rank parameter modulation provides a compact family of composition operators, and bounded relations together with a common state interface support recursive reuse. These properties follow from the parameterization of the composition cell rather than from auxiliary property-enforcing losses. We first isolate evidence-adaptive behavior in a controlled relation-learning setting, where a queried hypothesis pair is held fixed while the surrounding evidence changes. We then evaluate AdaCompose on compositional zero-shot recognition, video anomaly detection, and multimodal question answering, covering unseen semantic combinations, context-dependent prediction, and reasoning across heterogeneous visual and textual evidence.

Our main contributions are:

i. We formulate *evidence-adaptive composition*, in which the complete evidence states being composed determine the composition operator itself, rather than merely providing inputs to a fixed composition function.

ii. We develop a reusable composition cell with a common evidence-state interface, allowing each intermediate composition to become evidence for generating subsequent composition rules and thereby supporting recursive composition across varying numbers of evidence sources.

iii. We establish structural properties of the proposed cell and demonstrate its effectiveness through controlled contextual-intervention experiments and evaluations on compositional zero-shot recognition, video anomaly detection, and multimodal question answering.

We provide a detailed discussion of related work on compositional generalization, adaptive fusion and conditional parameterization, and set-structured and recursive reasoning in Appendix A. We next present AdaCompose and its evidence-adaptive composition mechanism.

# 2 METHOD

## 2.1 EVIDENCE-ADAPTIVE COMPOSITION

AdaCompose (see Fig. 1) makes the composition rule itself dependent on the evidence being composed. Each evidence source is represented as a structured state containing local hypothesis states and a summary of its complete evidence. Given two evidence states, their complete evidence generates an evidence-conditioned composition operator. The resulting operator is then shared across all hypothesis pairs between the two sources and determines how those pairs are related.

The key distinction from conventional fusion is therefore not simply that context is provided as an additional input. Instead, the context determines the *rule* used for composition:

$$\underbrace{\mathcal{S}_X, \mathcal{S}_Y}_{\text{complete evidence}} \longrightarrow \underbrace{\omega_{XY}}_{\text{composition rule}} \longrightarrow \underbrace{\boldsymbol{R}_{XY}}_{\text{pairwise relations}} . \tag{1}$$

Here, $\mathcal{S}_X$ and $\mathcal{S}_Y$ denote the two evidence states, while $\omega_{XY}$ parameterizes the composition rule for their current interaction. Because $\omega_{XY}$ is generated from the complete evidence states, changing the surrounding evidence can change how the same local hypothesis pair is interpreted.

**Composition as new evidence.** AdaCompose further treats the result of composition as new evidence. The pairwise relations are converted into a new evidence state with the same structural interface as its inputs: $(\mathcal{S}_X, \mathcal{S}_Y) \xrightarrow{\mathcal{C}_\theta} (\mathcal{S}_{XY}, \boldsymbol{R}_{XY})$. The resulting state $\mathcal{S}_{XY}$ can therefore be composed with another source using the same cell. Crucially, the operator for the next composition is generated again from the newly formed evidence state and the new source. An earlier composition can thus influence both the evidence available for subsequent composition and the rule used to interpret it. This formulation provides a reusable mechanism for composing heterogeneous evidence across variable numbers of hypotheses and sources. The composition cell developed next is designed around four structural requirements: the composition rule should depend on the complete evidence, prediction scores should directly control the contribution of individual hypotheses, operator adaptation should remain controlled, and every composition result should remain a valid evidence state for further composition.

## 2.2 ADACOMPOSE COMPOSITION CELL

AdaCompose implements evidence-adaptive composition through a reusable binary composition cell. Given two evidence states, the cell first summarizes their complete evidence, then generates a composition operator from these summaries, applies the resulting operator to all cross-source hypothesis pairs, and finally converts the pairwise relations into a new evidence state.

**Evidence-state representation.** We represent an evidence source $X$ by $\mathcal{S}_X = (\boldsymbol{H}_X, \boldsymbol{z}_X, \tau_X)$, $\boldsymbol{H}_X = [\boldsymbol{h}_1^X; \ldots; \boldsymbol{h}_{K_X}^X] \in \mathbb{R}^{K_X \times d_h}$, where $\boldsymbol{h}_i^X$ is the local state of hypothesis $i$, $\boldsymbol{z}_X$ is a permutation-invariant summary of the complete source evidence, and $\tau_X$ is a learned embedding identifying the source or composition type. The number of hypotheses $K_X$ may vary across sources and instances.

The input to the encoder consists of an evidence feature $\boldsymbol{e}_i^X$ and its associated prediction score $p_i^X \in [0, 1]$. Each feature-score pair is first mapped to a latent representation by a shared per-hypothesis MLP $\phi_{\text{slot}}$. We then aggregate these representations by mean pooling and apply a learned global MLP $\phi_{\text{global}}$ to obtain the source-level summary (Zaheer et al., 2017):

$$\boldsymbol{z}_X = \phi_{\text{global}} \left( \frac{1}{K_X} \sum_{i=1}^{K_X} \phi_{\text{slot}} \left( \begin{bmatrix} \boldsymbol{e}_i^X \\ p_i^X \end{bmatrix} \right) \right). \tag{2}$$

The resulting summary $\boldsymbol{z}_X$ captures information from the complete set of hypotheses and their prediction scores while remaining invariant to their ordering. Each hypothesis is then encoded together with the source-level summary: $\tilde{\boldsymbol{h}}_i^X = \phi_{\text{local}} \left( \begin{bmatrix} \boldsymbol{e}_i^X \\ p_i^X \\ \boldsymbol{z}_X \end{bmatrix} \right)$, $\boldsymbol{h}_i^X = p_i^X \tilde{\boldsymbol{h}}_i^X$, where $\phi_{\text{local}}$ is a shared local MLP that produces the hypothesis state conditioned on its feature-score pair and the source-level summary. The multiplicative gate makes the prediction score an explicit control on local evidence

strength, such that $p_i^X = 0$ gives $\boldsymbol{h}_i^X = \mathbf{0}$. Meanwhile, $p_i^X$ also contributes to $\boldsymbol{z}_X$ through Eq. 2, allowing prediction scores to influence both individual hypothesis states and, through the source summary, the composition rule generated for the source. The source-type embedding is obtained from a learned type table: $\tau_X = T_{\text{type}}[s_X]$, where $s_X$ identifies the source or composition type.

**Evidence-conditioned operator.** Given two evidence states $\mathcal{S}_X$ and $\mathcal{S}_Y$, AdaCompose first generates an operator code that parameterizes how the two states are composed. We use a shared operator generator $G_\psi$, implemented as an MLP with learnable parameters $\psi$, to map their global summaries and type embeddings to this code (Ha et al., 2017; Wang et al., 2023):

$$\omega_{XY} = \tanh\left(G_\psi\left(\begin{bmatrix}\boldsymbol{z}_X\\ \boldsymbol{z}_Y\\ \tau_X\\ \tau_Y\end{bmatrix}\right)\right), \tag{3}$$

where $\omega_{XY}$ is the operator code for the current composition of $X$ and $Y$. It is generated once from the source-level summaries and types and is shared across all hypothesis pairs between the two states. Thus, the same pair of local hypothesis states is processed according to a rule determined by the complete evidence available from both sources. The operator code is converted into an evidence-dependent composition function by modulating the weight matrix of each layer in the relation network. For layer $\ell$, let $\boldsymbol{W}_\ell^0$ denote its shared base weight matrix. We adapt this matrix using a low-rank update (Hu et al., 2021):

$$\boldsymbol{W}_\ell(\omega_{XY}) = \boldsymbol{W}_\ell^0 + \boldsymbol{U}_\ell \operatorname{diag}\left(\boldsymbol{g}_\ell(\omega_{XY})\right) \boldsymbol{V}_\ell^\top, \tag{4}$$

where $\boldsymbol{U}_\ell \in \mathbb{R}^{d_{\text{out}} \times r_{\text{mod}}}$ and $\boldsymbol{V}_\ell \in \mathbb{R}^{d_{\text{in}} \times r_{\text{mod}}}$ are learned factors that define the adaptation subspace, and $r_{\text{mod}}$ is the modulation rank. The corresponding modulation coefficients are given by $\boldsymbol{g}_\ell(\omega_{XY}) = \tanh\left(\boldsymbol{B}_\ell \omega_{XY} + \boldsymbol{b}_\ell\right) \in \mathbb{R}^{r_{\text{mod}}}$, where $\boldsymbol{B}_\ell$ and $\boldsymbol{b}_\ell$ are learned parameters that map the operator code to the modulation coefficients. The resulting weight matrix therefore consists of a shared base matrix plus an evidence-dependent update whose rank is at most $r_{\text{mod}}$:

$$\operatorname{rank}\left(\boldsymbol{W}_\ell(\omega_{XY}) - \boldsymbol{W}_\ell^0\right) \le r_{\text{mod}}. \tag{5}$$

The same operator code $\omega_{XY}$ is used across all layers and all hypothesis pairs in the current composition. Hence, the model does not learn a separate relation function for each pair; instead, the current evidence states determine a shared, composition-specific parameterization of the relation function.

**Relation field and composed state.** The generated operator is evaluated on every cross-source pair of local states (Santoro et al., 2017). For hypotheses $i$ and $j$ from $X$ and $Y$, respectively, we apply the evidence-conditioned relation network to their local states: $a_{ij}^{XY} = f_{\omega_{XY}}\left(\begin{bmatrix}\boldsymbol{h}_i^X\\ \boldsymbol{h}_j^Y\end{bmatrix}\right)$, $r_{ij}^{XY} = \sigma(a_{ij}^{XY})$, where $f_{\omega_{XY}}$ is the relation network formed by the evidence-conditioned layer weights $\boldsymbol{W}_\ell(\omega_{XY})$ in Eq. 4. The scalar $a_{ij}^{XY}$ is the relation logit, and $r_{ij}^{XY} \in (0,1)$ is the corresponding bounded relation score. Collecting all pairwise scores gives the relation field $\boldsymbol{R}_{XY} = [r_{ij}^{XY}]_{i,j} \in (0,1)^{K_X \times K_Y}$.

The relation score then controls the contribution of each hypothesis pair to the new local evidence state. Specifically, we first encode the pair of local states together with their relation score using a shared pairwise MLP $\phi_{\text{pair}}$: $\boldsymbol{h}_{ij}^{XY} = r_{ij}^{XY} \phi_{\text{pair}}\left(\begin{bmatrix}\boldsymbol{h}_i^X\\ \boldsymbol{h}_j^Y\\ r_{ij}^{XY}\end{bmatrix}\right)$. Stacking the pairwise states gives $\boldsymbol{H}_{XY} = [\boldsymbol{h}_{ij}^{XY}]_{i,j} \in \mathbb{R}^{(K_X K_Y) \times d_h}$. Thus, the relation field is not merely an auxiliary prediction: it directly controls how strongly each hypothesis pair contributes to the composed evidence state.

In addition to pairwise states, AdaCompose summarizes the relation field by its mean relation score:

$$\bar{r}_{XY} = \frac{1}{K_X K_Y} \sum_{i=1}^{K_X} \sum_{j=1}^{K_Y} r_{ij}^{XY}. \tag{6}$$

We map this scalar to a relation-derived global summary using a learned affine transformation followed by a $\tanh$ activation:

$$\boldsymbol{z}_R = \tanh\left(\boldsymbol{W}_R \bar{r}_{XY} + \boldsymbol{b}_R\right), \tag{7}$$

where $\boldsymbol{W}_R$ and $\boldsymbol{b}_R$ are learned parameters. The resulting $\boldsymbol{z}_R$ provides a compact summary of the interaction between the two evidence states. We then combine the two input summaries and the relation-derived summary using learned memory weights:

$$\begin{bmatrix}\alpha_{XY,1}\\ \alpha_{XY,2}\\ \alpha_{XY,3}\end{bmatrix} = \mathrm{softmax}\left(\boldsymbol{M}_2\ \mathrm{GELU}\left(\boldsymbol{M}_1\begin{bmatrix}\boldsymbol{z}_X\\ \boldsymbol{z}_Y\\ \boldsymbol{z}_R\end{bmatrix}+\boldsymbol{b}_1\right)+\boldsymbol{b}_2\right), \tag{8}$$

where $\boldsymbol{M}_1, \boldsymbol{M}_2, \boldsymbol{b}_1, \boldsymbol{b}_2$ parameterize a shared two-layer memory MLP. The three nonnegative weights sum to one and determine the contributions of the two input summaries and the relation-derived summary, respectively. The global state of the composed evidence is therefore

$$\boldsymbol{z}_{XY} = \alpha_{XY,1}\boldsymbol{z}_X + \alpha_{XY,2}\boldsymbol{z}_Y + \alpha_{XY,3}\boldsymbol{z}_R. \tag{9}$$

Because the weights are produced by a softmax, $\boldsymbol{z}_{XY}$ is a convex combination of $\boldsymbol{z}_X$, $\boldsymbol{z}_Y$, and $\boldsymbol{z}_R$.

Finally, the composition cell returns

$$\boxed{(\mathcal{S}_X, \mathcal{S}_Y) \xrightarrow{\mathcal{C}_\theta} (\mathcal{S}_{XY}, \boldsymbol{R}_{XY}), \qquad \mathcal{S}_{XY} = (\boldsymbol{H}_{XY}, \boldsymbol{z}_{XY}, \tau_{XY}).} \tag{10}$$

Here, $\tau_{XY}$ is the learned type embedding associated with the composed state. The resulting state has the same structural interface as an input state, while its local states correspond to the $K_X K_Y$ cross-source hypothesis pairs. This closure allows the same composition cell to be reused recursively: when a new source $Z$ is introduced, a new operator code is generated from $\mathcal{S}_{XY}$ and $\mathcal{S}_Z$, rather than reusing $\omega_{XY}$. The resulting design supports evidence-adaptive and recursive composition. Its design rationale is discussed in Appendix B, while its formal guarantees are established in Section 2.3.

### 2.3 Structural Properties

The AdaCompose composition cell has four structural properties that follow directly from its parameterization, without requiring auxiliary property-enforcing losses. First, the shared set encoder produces permutation-invariant source summaries, so the generated composition operator is unchanged by reordering hypotheses within either source (Prop. 1). The corresponding relation field is permutation-equivariant, meaning that the rows and columns of the relation matrix are reordered consistently with the input hypotheses. Second, the softmax memory update represents the composed global state as a convex combination of the two input summaries and a bounded relation summary, which gives a depth-independent bound on the global state (Prop. 2). Third, the low-rank parameterization constrains evidence-dependent changes to the composition operator to a shared adaptation subspace (Prop. 3). Finally, for a fixed local hypothesis pair, the relation score varies Lipschitz-continuously with the generated operator code (Prop. 4). Formal statements and proofs are provided in Appendix C, while the scope and limitations are discussed in Appendix E.

## 3 Experiment

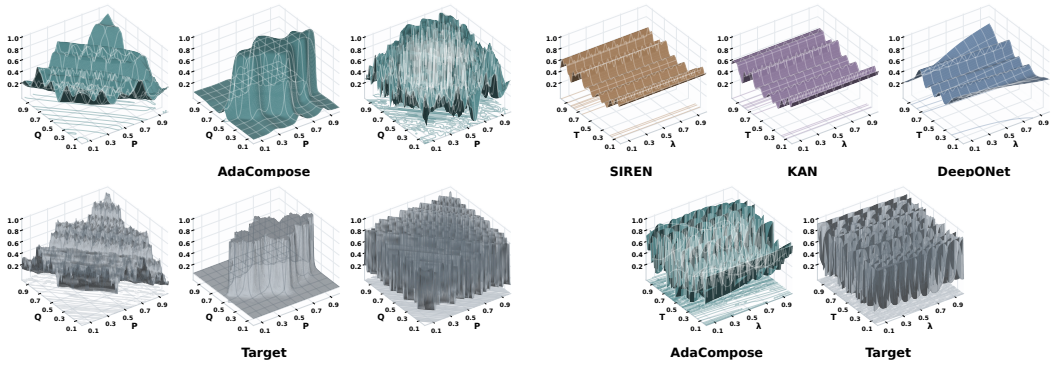
### 3.1 Experimental Setup

**Controlled relation learning.** We test whether the learned relation depends on surrounding evidence using Weierstrass (Elbrächter et al., 2021), Julia (Tjahjono et al., 2025), and Hardy $Z$ fields (Bui & Hall, 2023) over two unordered sets. Fixed targets depend only on queried coordinates, whereas contextual targets additionally warp them using set means and standard deviations. AdaCompose receives the complete sets; SIREN (Sitzmann et al., 2020) and KAN (Liu et al., 2025) receive query pairs, while DeepONet (Lu et al., 2021) also receives set statistics. Models are trained separately for each setting, and we report normalized root mean squared error (nRMSE; (Stekhoven & Bühlmann, 2012)), normalized by the target standard deviation, over 1,536 test fields per seed and averaged over three seeds. Full target definitions and training details are given in Appendix F.1.

**Compositional zero-shot recognition.** We evaluate unseen attribute-object combinations on MIT-States (Isola et al., 2015) and C-GQA (Naeem et al., 2021). Closed-world evaluation considers training and split-specific unseen pairs, while open-world evaluation considers the full attribute-object product with validation-selected feasibility filtering (Mancini et al., 2021; Huang et al., 2024). Frozen CLIP ViT-L/14 (Radford et al., 2021) supplies the evidence, and task modules are trained

Table 1: Controlled relation learning. nRMSE (↓) on fixed and context-dependent relations across Weierstrass, Julia, and Hardy $Z$ fields. AdaCompose achieves the lowest error across all contextual settings, where the relation depends on surrounding evidence, demonstrating its ability to adapt the composition rule to context. Mean ± population standard deviation over three seeds; fixed and contextual settings are trained separately. Bold denotes the lowest error in each column.

| | Fixed relations | | | Contextual relations | | |
|---|---|---|---|---|---|---|
| Method | Weierstrass | Julia | Hardy $Z$ | Weierstrass | Julia | Hardy $Z$ |
| SIREN | **0.1984 ± 0.0006** | **0.0313 ± 0.0011** | **0.0787 ± 0.0099** | 0.5289 ± 0.0003 | 0.3165 ± 0.0004 | 0.9682 ± 0.0004 |
| KAN | 0.1996 ± 0.0002 | 0.0712 ± 0.0007 | 0.3610 ± 0.0349 | 0.5291 ± 0.0005 | 0.3191 ± 0.0003 | 0.9686 ± 0.0007 |
| DeepONet | 0.4815 ± 0.0019 | 0.2568 ± 0.0291 | 0.9755 ± 0.0005 | 0.4921 ± 0.0043 | 0.2362 ± 0.0080 | 0.9690 ± 0.0048 |
| **AdaCompose** | 0.3094 ± 0.0027 | 0.1314 ± 0.0083 | 0.6145 ± 0.0360 | **0.2447 ± 0.0069** | **0.1837 ± 0.0076** | **0.5506 ± 0.0458** |




(a) Contextual relation fitting.

(b) Fixed-query context intervention.

Figure 2: AdaCompose adapts relations to surrounding evidence. (a) Target and AdaCompose surfaces for Weierstrass, Julia, and Hardy $Z$ fields (left to right), varying queried coordinates $p, q$ (axes $P, Q$) while holding the remaining set members fixed. (b) Hardy $Z$ predictions for fixed diagonal queries $(p, q) = (T, T)$ as the surrounding context varies with interpolation parameter $\lambda$. For each fixed queried coordinate pair, the target relation varies with the surrounding evidence, and AdaCompose tracks this contextual variation. Contextual targets recompute set statistics at each grid point; predictions are averaged over three seeds. See Appendix F.1 for more details.

with pair-label cross-entropy. We report seen/unseen accuracy (S/U), harmonic mean (HM), and calibration-curve area (AUC), following the standard CZSL protocol. Baseline and implementation details are given in Appendix F.2.

**Video anomaly detection.** We evaluate on MSAD Protocol II, with 480 training and 240 test videos and 80 training videos held out for validation (Zhu et al., 2024). Frozen Places365 ResNet-18 (Zhou et al., 2017), VideoMAE-base (Tong et al., 2022), and YOLOS-tiny (Fang et al., 2021) provide scene, action, and object evidence. Training and selection use video labels only, while frame annotations are used only for evaluation. We report pooled frame-level ROC-AUC and average precision (AP), overall and by anomaly type or scenario. Because our decoder-aligned frame set differs from the released benchmark alignment, published comparisons are treated as reference results rather than verified frame-matched rankings; details are provided in Appendix F.3.

**Multimodal question answering.** We evaluate on the full ScienceQA split of 12,726/4,241/4,241 train/validation/test examples (Lu et al., 2022). Frozen CLIP ViT-B/32 provides answer-choice distributions and semantic representations from questions, textual hints, and raw images; supplied captions, lectures, and solutions are excluded. Only the task head is trained, with validation accuracy used for model selection. We report overall accuracy and breakdowns by subject, context, and grade. Further implementation and baseline details are provided in Appendix F.4.

Table 2: CZSL performance and efficiency. Closed- and open-world results (%) on MIT-States and C-GQA, with each cell reporting S/U/HM/AUC (↑), together with parameter count and batch-one inference time on C-GQA (↓). AdaCompose achieves competitive compositional generalization with a compact 3.01M-parameter architecture, substantially smaller than DFSP and Troika. Task-parameter counts exclude frozen CLIP. C-GQA latency follows the separate architecture benchmark in Appendix F.2.

| | MIT-States | | C-GQA | | Efficiency | |
|---|---|---|---|---|---|---|
| Method | Closed-world | Open-world | Closed-world | Open-world | Task params (M) ↓ | ms/image ↓ |
| CLIP | 30.2/46.0/26.1/11.0 | 30.1/14.3/12.8/3.0 | 7.5/25.0/8.6/1.4 | 7.5/4.6/4.0/0.27 | None | $11.33 \pm 0.15$ |
| CoOp | 34.4/47.6/29.8/13.5 | 34.6/9.3/12.3/2.8 | 20.5/26.8/17.1/4.4 | 21.0/4.6/5.5/0.70 | 0.0023 | $11.15 \pm 0.06$ |
| CSP | 46.6/49.9/36.3/19.4 | 46.3/15.7/17.4/5.7 | 28.8/26.8/20.5/6.2 | 28.7/5.2/6.9/1.20 | 0.8348 | $11.24 \pm 0.21$ |
| DFSP (i2t) | 47.4/52.4/37.2/20.7 | 47.2/18.2/19.1/6.7 | 35.6/29.3/24.3/8.7 | 35.6/6.5/9.0/1.95 | 46.8528 | $106.55 \pm 0.36$ |
| DFSP (BiF) | 47.1/52.8/37.7/20.8 | 47.1/18.1/19.2/6.7 | 36.5/32.0/26.2/9.9 | 36.4/7.6/10.6/2.39 | 46.8528 | $107.17 \pm 4.15$ |
| DFSP (t2i) | 46.9/52.0/37.3/20.6 | 47.5/18.5/19.3/6.8 | 38.2/32.0/27.1/10.5 | 38.3/7.2/10.4/2.40 | 46.8528 | $14.23 \pm 0.34$ |
| Troika | 49.0/53.0/39.3/22.1 | 48.8/18.7/20.1/7.2 | 41.0/35.7/29.4/12.4 | 40.8/7.9/10.9/2.70 | 22.5400 | $24.08 \pm 0.23$ |
| **AdaCompose** | 44.08/48.18/34.34/17.58 | 43.91/16.57/17.10/5.38 | 34.37/29.28/24.67/8.57 | 33.43/5.51/7.30/1.43 | 3.0099 | $17.67 \pm 0.22$ |

| | Trainable params | Subject | | | Context | | Grade | | Overall |
|---|---|---|---|---|---|---|---|---|---|
| Method | (M) ↓ | NAT | SOC | LAN | IMG | NO | G1–6 | G7–12 | Avg |
| MCAN | 95 | 56.08 | 46.23 | 58.09 | 51.17 | 55.40 | 51.65 | 59.72 | 54.54 |
| Top-Down | 70 | 59.50 | 54.33 | 61.82 | 54.88 | 59.79 | 57.27 | 62.16 | 59.02 |
| BAN | 112 | 60.88 | 46.57 | **66.64** | 52.60 | **65.51** | 56.83 | 63.94 | 59.37 |
| DFAF | 74 | 64.03 | 48.82 | 63.55 | 54.49 | 64.11 | 57.12 | 67.17 | 60.72 |
| ViLT | 113 | 60.48 | 63.89 | 60.27 | 61.38 | 57.00 | 60.72 | 61.90 | 61.14 |
| Patch-TRM | 90 | **65.19** | 46.79 | 65.55 | 55.28 | 64.95 | 58.04 | 67.50 | 61.42 |
| VisualBERT | 111 | 59.33 | 69.18 | 61.18 | 62.17 | 58.54 | 62.96 | 59.92 | 61.87 |
| Mean | None | 53.20 | 35.55 | 48.45 | 44.82 | 47.53 | 50.92 | 43.51 | 48.27 |
| Product | None | 53.02 | 36.11 | 48.45 | 44.92 | 47.53 | 51.10 | 43.24 | 48.29 |
| **AdaCompose** | 0.158 | 64.74 | **76.94** | 61.45 | **71.19** | 59.51 | **65.01** | **69.02** | **66.45** |

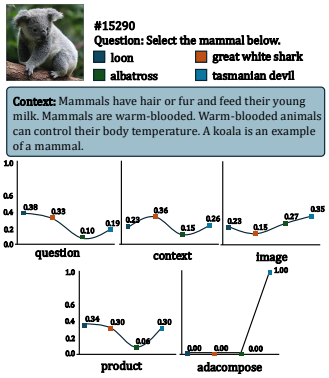



Table 3: Multimodal evidence composition on ScienceQA. *Left:* Test accuracy (%, ↑) across subject, context, grade, and overall, with trainable parameters (M, ↓). AdaCompose achieves the highest overall accuracy with only 0.158M trainable parameters. *Right:* Qualitative comparison with product aggregation on question #15290, illustrating the benefit of adaptive evidence composition.

**Roles of the real-world evaluations.** The three real-world tasks probe complementary capabilities: CZSL tests generalization to unseen semantic compositions, VAD tests heterogeneous multi-source composition and exposes intermediate evidence for anomaly localization, and ScienceQA tests multimodal reasoning over complementary visual and textual evidence.

### 3.2 Evidence-Adaptive Composition

Table 1 compares fixed and contextual relation learning on Weierstrass, Julia, and Hardy Z fields. SIREN achieves the lowest error on all fixed fields, whereas AdaCompose achieves the lowest error on all contextual fields. This pattern suggests a particular advantage for contextual relation learning rather than uniformly better function approximation. DeepONet also receives set-level statistics but remains less accurate on the contextual fields, supporting evidence-conditioned operator generation as an effective inductive bias within the tested models.

Figure 2 illustrates how AdaCompose captures context-dependent relations. Panel (a) compares predicted and target surfaces; panel (b) shows predictions tracking target changes as the surrounding context varies with the queried coordinates held fixed. The quantitative intervention results in Appendix F.1 further support this context responsiveness. Because context affects both local representations and generated operators, these results assess their combined response rather than isolate operator modulation. Appendix G.1 reports additional permutation and cardinality diagnostics.

### 3.3 Compositional Generalization

Table 2 reports closed- and open-world CZSL results on MIT-States and C-GQA. On C-GQA, AdaCompose achieves 24.67 HM and 8.57 AUC in the closed world and 7.30 HM and 1.43 AUC in the open world, exceeding CSP in both metrics while remaining below Troika. On MIT-States, it remains below CSP in both settings, indicating that the relative performance depends on the dataset.

The CZSL model combines recursive-composition, direct-pair, and marginal scores. The branch diagnostics in Table 7 show that full fusion improves over direct-pair scoring by $10.25/5.12$ HM/AUC points on C-GQA, compared with $0.31/0.15$ on MIT-States. These results indicate dataset-dependent benefits from score fusion, without isolating the contribution of recursion. On C-GQA, AdaCompose uses 3.01M task parameters, compared with 22.54M for Troika and 46.85M for DFSP, excluding frozen CLIP. This yields a smaller task-specific parameter budget than these models, although their recognition accuracy remains higher. The reported latency characterizes architecture costs under the protocol in Appendix F.2, rather than end-to-end costs of the reported accuracy pipelines.

### 3.4 Heterogeneous Evidence Composition

**Multimodal question answering.** Table 3 shows that AdaCompose achieves the best overall ScienceQA accuracy at $66.45\%$, improving over the strongest baseline, VisualBERT, by $4.58$ points. The gain is particularly pronounced for social science ($76.94\%$) and image questions ($71.19\%$), suggesting that adaptive composition is effective when the answer depends on complementary evidence sources rather than a single modality. It also consistently outperforms mean and product aggregation, showing that the improvement is not explained by simply combining the predicted probabilities. The example in Table 3 illustrates this distinction. For question #15290, product aggregation remains uncertain across the evidence sources, whereas AdaCompose suppresses the competing hypotheses and assigns the final probability to the correct answer. The result demonstrates the intended behavior of AdaCompose: the evidence sources are not merely averaged or multiplied, but jointly determine how their information should be composed.

**Video anomaly detection.** Table 4 (left) reports frame-level ROC-AUC/AP on MSAD Protocol II. AdaCompose achieves 82.7/55.5 overall and performs particularly well on explosion, people falling, robbery, traffic accident, and warehouse, showing that the same composition mechanism can accommodate substantially different anomaly types and scenes without anomaly-specific rules. The qualitative examples in Table 4 (right) show how this composition supports localization. In both people-falling and fire, the anomaly score increases in the anomalous window alongside changes in action, object, and context evidence. The fire example, for instance, shows a marked increase in fire-related action evidence. These intermediate states expose how heterogeneous evidence contributes to the final anomaly prediction.

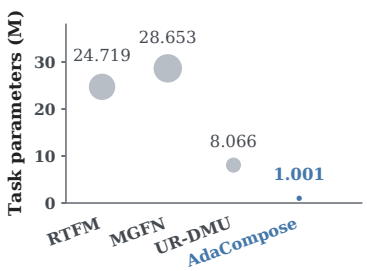



Figure 3: VAD task-parameter counts. AdaCompose uses 1.001M task parameters. Frozen evidence extractors are excluded from its count; circle area scales with parameter count.

Fig. 3 highlights the parameter efficiency: AdaCompose uses only 1.001M task parameters (frozen evidence extractors are excluded), substantially fewer than the compared methods, while retaining competitive detection performance. This supports the use of a shared, evidence-conditioned composition mechanism rather than task-specific parameter-heavy models.

### 3.5 Ablation and Further Analysis

**Interaction feedback is important for composition.** We ablate three interventions within the VAD pipeline. *Type-only operator codes* remove evidence-summary inputs to $G_\psi$ while retaining type conditioning. *Context-aware static factors* replace generated concept queries with a fixed-parameter network, while retaining operator-code inputs and the final rank gate. *Without interaction feedback* removes the first-stage interaction from the composed-state update but preserves its contribution to the final score. Implementation details are provided in Appendix F.3.

| Evaluation subset | RTFM | MGFN | UR-DMU | EGO | AdaCompose |
|---|---|---|---|---|---|
| *Category-level* | | | | | |
| Assault | **68.1/67.3** | 59.7/59.0 | 56.9/64.5 | 52.2/57.5 | 55.7/53.5 |
| Explosion | 46.8/60.4 | 64.5/71.9 | 67.9/74.5 | 57.6/74.4 | **73.6/81.0** |
| Fighting | **89.6**/93.0 | 89.4/**93.5** | 83.9/90.4 | 66.5/72.8 | 72.6/83.7 |
| Fire | 61.3/81.2 | **86.0/93.0** | 61.2/82.9 | 62.9/86.7 | 62.4/84.7 |
| Object falling | **94.7/96.7** | 90.9/94.8 | 92.1/95.8 | 92.3/94.8 | 65.1/75.0 |
| People falling | 56.5/50.4 | 52.7/47.8 | 42.5/43.7 | 35.4/43.8 | **67.4/59.9** |
| Robbery | 65.7/81.2 | 73.9/86.7 | 63.5/79.3 | 64.8/87.5 | **77.9/90.3** |
| Shooting | 78.2/84.7 | **86.8/88.5** | 81.4/87.8 | 68.6/78.4 | 79.0/86.4 |
| Traffic accident | 62.2/51.8 | 68.6/54.5 | 62.0/55.6 | 69.9/64.3 | **75.1/65.3** |
| Vandalism | 85.2/76.1 | 82.4/80.1 | 84.7/77.0 | **88.1/81.4** | 77.3/69.2 |
| Water incident | 96.3/99.1 | 85.5/97.0 | **98.5/99.5** | 81.9/95.4 | 94.3/98.8 |
| *Scenario-level* | | | | | |
| Frontdoor | 84.1/81.1 | **86.4/85.1** | 84.8/82.8 | 85.2/81.6 | 75.3/73.4 |
| Highway | 63.7/4.1 | 79.7/4.1 | 31.5/1.3 | **80.2/30.8** | —/— |
| Mall | 87.2/72.2 | 65.3/56.6 | 91.0/**83.8** | 82.3/73.4 | **92.3**/82.1 |
| Office | 78.1/68.8 | 75.1/62.4 | 77.8/67.3 | **80.0/71.7** | 71.1/58.2 |
| Park | 69.0/25.6 | 77.9/38.3 | 87.8/36.2 | **93.5/44.3** | —/— |
| Parkinglot | 74.4/35.9 | 68.1/14.5 | 91.4/53.9 | **96.8/75.2** | 85.4/41.8 |
| Pedestrian st. | 97.4/50.6 | 88.0/20.4 | 81.9/11.5 | **97.5**/52.0 | 94.9/**63.2** |
| Restaurant | **96.1/91.9** | 95.8/91.8 | 93.1/87.4 | 94.3/73.9 | 90.2/86.2 |
| Road | 54.0/16.8 | 77.9/49.7 | 83.0/64.4 | **89.8/64.6** | 73.8/47.7 |
| Shop | 80.6/**77.3** | **84.9**/77.2 | 81.3/64.5 | 83.4/72.2 | 80.5/66.5 |
| Sidewalk | 52.5/17.1 | 85.5/62.3 | 86.5/**64.1** | **87.1**/45.0 | 84.7/43.8 |
| Street highview | 43.3/12.3 | **87.6/40.7** | 85.0/37.7 | 28.2/10.1 | 77.8/32.5 |
| Train | 66.9/3.9 | 53.0/3.1 | 59.0/3.1 | **80.8/7.8** | 64.0/4.3 |
| Warehouse | 69.5/37.4 | 72.3/30.9 | 81.2/59.1 | 84.7/46.6 | **88.9/70.8** |
| Overall | 86.7/66.3 | 85.0/63.5 | 85.0/**68.3** | **87.3**/64.4 | 82.7/55.5 |

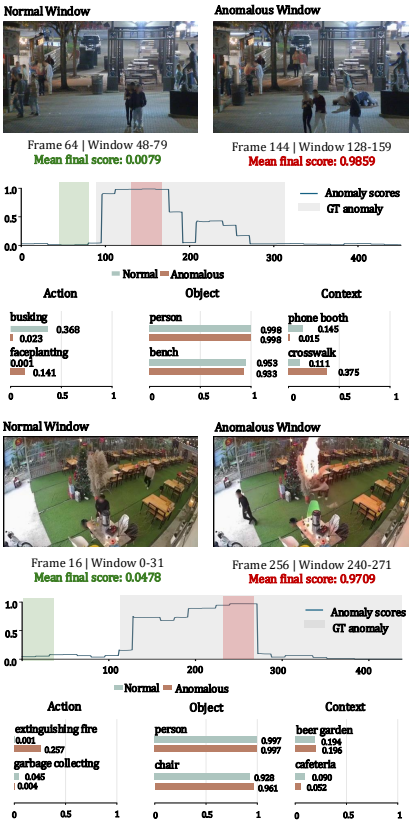



Table 4: Heterogeneous evidence for video anomaly detection. (*Left:*) Frame-level ROC-AUC / AP (%) across anomaly categories, scenarios, and overall; bold denotes the best value in each row. (*Right:*) People-falling and fire examples pair anomaly scores with action, object, and context evidence, illustrating changing semantic evidence alongside the final prediction.

Removing it reduces ROC-AUC by 1.17 points and AP by 0.66 points (Table 5), despite retaining the interaction in the final score. This indicates that its main additional benefit is updating the state that determines subsequent composition. Evidence-conditioned operator generation mainly improves ROC-AUC: full AdaCompose gains 0.28 and 0.79 points over type-only codes and static factors, respectively, although these alternatives achieve higher AP. The ablations therefore support the roles of both adaptive operators and interaction feedback, while not implying uniform gains across metrics. Further analyses and discussions are provided in Appendix G.

Table 5: Mechanism ablations on MSAD Protocol II (%). Each variant removes or restricts one component. Bold marks the best value.

| Variant | ROC-AUC ↑ | AP ↑ |
|---|---|---|
| Full AdaCompose | **82.739** | 55.468 |
| Type-only operator codes | 82.460 | **57.385** |
| Context-aware static factors | 81.953 | 56.584 |
| Without interaction feedback | 81.570 | 54.810 |

## 4 Conclusion

We introduced AdaCompose, a framework in which the evidence being composed determines the composition operator itself. Each composition produces a new evidence state that can determine how subsequent evidence is composed, enabling context-dependent and recursive composition of heterogeneous evidence. Controlled experiments demonstrate context-dependent relation learning, while downstream evaluations assess unseen attribute-object recognition and heterogeneous evidence integration. Experiments across four tasks demonstrate the effectiveness of this operator-level adaptation for compositional generalization.

## REFERENCES


Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Learning to compose neural networks for question answering. In *Proceedings of the 2016 conference of the North American chapter of the Association for Computational Linguistics: Human language technologies*, pp. 1545–1554, 2016a.

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. Neural module networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 39–48, 2016b.

Hung M Bui and Richard R Hall. On the derivatives of hardy's function z (t). *Bulletin of the London Mathematical Society*, 55(5):2304–2323, 2023.

Dexuan Ding, Lei Wang, Liyun Zhu, Tom Gedeon, and Piotr Koniusz. Learnable expansion of graph operators for multi-modal feature fusion. In *International Conference on Learning Representations*, volume 2025, pp. 74263–74285, 2025.

Dennis Elbrächter, Dmytro Perekrestenko, Philipp Grohs, and Helmut Bölcskei. Deep neural network approximation theory. *IEEE Transactions on Information Theory*, 67(5):2581–2623, 2021.

Yuxin Fang, Bencheng Liao, Xinggang Wang, Jiemin Fang, Jiyang Qi, Rui Wu, Jianwei Niu, and Wenyu Liu. You only look at one sequence: Rethinking transformer in vision through object detection. *Advances in neural information processing systems*, 34:26183–26197, 2021.

David Ha, Andrew M. Dai, and Quoc V. Le. Hypernetworks. In *International Conference on Learning Representations*, 2017. URL `https://openreview.net/forum?id=rkpACe1lx`.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Siteng Huang, Biao Gong, Yutong Feng, Min Zhang, Yiliang Lv, and Donglin Wang. Troika: Multi-path cross-modal traction for compositional zero-shot learning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 24005–24014. IEEE, 2024.

Drew A Hudson and Christopher D Manning. Compositional attention networks for machine reasoning. *arXiv preprint arXiv:1803.03067*, 2018.

Phillip Isola, Joseph J Lim, and Edward H Adelson. Discovering states and transformations in image collections. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1383–1391, 2015.

Xu Jia, Bert De Brabandere, Tinne Tuytelaars, and Luc V Gool. Dynamic filter networks. *Advances in neural information processing systems*, 29, 2016.

Jin-Hwa Kim, Jaehyun Jun, and Byoung-Tak Zhang. Bilinear attention networks. *Advances in neural information processing systems*, 31, 2018.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *International conference on machine learning*, pp. 3744–3753. PMLR, 2019.

Yun Li, Zhe Liu, Hang Chen, and Lina Yao. Context-based and diversity-driven specificity in compositional zero-shot learning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17037–17046. IEEE, 2024.

Ziming Liu, Yixuan Wang, Sachin Vaidya, Fabian Ruehle, James Halverson, Marin Soljacic, Thomas Hou, and Max Tegmark. Kan: Kolmogorov–arnold networks. In *International conference on learning representations*, volume 2025, pp. 70367–70413, 2025.

Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature machine intelligence*, 3(3):218–229, 2021.

Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. *Advances in neural information processing systems*, 35:2507–2521, 2022.

Xiaocheng Lu, Song Guo, Ziming Liu, and Jingcai Guo. Decomposed soft prompt guided fusion enhancing for compositional zero-shot learning. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 23560–23569. IEEE, 2023.

Massimiliano Mancini, Muhammad Ferjad Naeem, Yongqin Xian, and Zeynep Akata. Open world compositional zero-shot learning. *arXiv preprint arXiv:2101.12609*, 2021.

Jiayuan Mao, Chuang Gan, Pushmeet Kohli, Joshua B Tenenbaum, and Jiajun Wu. The neuro-symbolic concept learner: Interpreting scenes, words, and sentences from natural supervision. *arXiv preprint arXiv:1904.12584*, 2019.

Muhammad Ferjad Naeem, Yongqin Xian, Federico Tombari, and Zeynep Akata. Learning graph embeddings for compositional zero-shot learning. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 953–962. IEEE, 2021.

Nihal V Nayak, Peilin Yu, and Stephen H Bach. Learning to compose soft prompts for compositional zero-shot learning. *arXiv preprint arXiv:2204.03574*, 2022.

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PmLR, 2021.

Nasim Rahaman, Muhammad Waleed Gondal, Shruti Joshi, Peter Gehler, Yoshua Bengio, Francesco Locatello, and Bernhard Schölkopf. Dynamic inference with neural interpreters. *Advances in Neural Information Processing Systems*, 34:10985–10998, 2021.

Adam Santoro, David Raposo, David G Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Timothy Lillicrap. A simple neural network module for relational reasoning. *Advances in neural information processing systems*, 30, 2017.

Simon Schug, Seijin Kobayashi, Yassir Akram, João Sacramento, and Razvan Pascanu. Attention as a hypernetwork. In *International Conference on Learning Representations*, volume 2025, pp. 68744–68770, 2025.

Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in neural information processing systems*, 33:7462–7473, 2020.

Daniel J Stekhoven and Peter Bühlmann. Missforest—non-parametric missing value imputation for mixed-type data. *Bioinformatics*, 28(1):112–118, 2012.

Venansius Ryan Tjahjono, Samuel F Feng, Endah Rokhmati Merdika Putri, and Hadi Susanto. Learning with mandelbrot and julia: Classification of fractal sets using machine learning. *Nonlinear Dynamics*, 113(22):31787–31800, 2025.

Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *Advances in neural information processing systems*, 35:10078–10093, 2022.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Qingsheng Wang, Lingqiao Liu, Chenchen Jing, Hao Chen, Guoqiang Liang, Peng Wang, and Chunhua Shen. Learning conditional attributes for compositional zero-shot learning. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11197–11206. IEEE, 2023.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander Smola. Deep sets. *Advances in neural information processing systems*, 30, 2017.

Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1452–1464, 2017.

Liyun Zhu, Lei Wang, Arjun Raj, Tom Gedeon, and Chen Chen. Advancing video anomaly detection: A concise review and a new dataset. *Advances in Neural Information Processing Systems*, 37:89943–89977, 2024.

# APPENDIX CONTENTS

**A Related Work** **14**

**B Why AdaCompose Supports Compositional Reasoning** **15**

**C Formal Properties and Proofs** **16**

Proposition 1: Permutation-consistent recursive composition . . . . . 17

Proposition 2: Depth-independent global-state bound . . . . . 19

Proposition 3: Controlled operator family . . . . . 20

Proposition 4: Fixed-query sensitivity . . . . . 21

**D Recursive Composition and Task Instantiation** **22**

**E Scope and Limitations of the Formal Guarantees** **24**

**F Experimental Details** **24**

F.1 Controlled relation learning . . . . . 24

F.2 Compositional zero-shot recognition . . . . . 26

F.3 Video anomaly detection . . . . . 27

F.4 Multimodal question answering . . . . . 28

F.5 Uncertainty and selection history . . . . . 29

**G Supplementary Diagnostics and Sensitivity Analysis** **29**

G.1 Set Structure and Cardinality Transfer . . . . . 30

G.2 Task-Specific Evidence Diagnostics . . . . . 30

G.3 Configurations and Validation Sensitivity . . . . . 31

# A Related Work

**Compositional generalization and compositional zero-shot learning.** Compositional generalization asks models to combine familiar primitives into compositions that were not observed during training. Compositional zero-shot learning (CZSL) (Naeem et al., 2021; Mancini et al., 2021; Li et al., 2024) provides a direct visual setting, where models should recognize unseen attribute-object combinations. Earlier methods model attributes and objects separately and learn their compatibility (Naeem et al., 2021), while recent vision-language approaches improve compositionality through decomposed prompts and separate primitive and composition pathways. In particular, CSP learns composable state and object prompts (Nayak et al., 2022), DFSP decomposes visual and textual representations (Lu et al., 2023), and Troika introduces separate recognition paths for states, objects, and their compositions (Huang et al., 2024). Troika is especially relevant because it shows that explicitly modeling primitives can improve generalization to unseen compositions. Another closely related direction is CANet (Wang et al., 2023), which models context-dependent attribute recognition by using an attribute hyper-learner to generate parameters for an attribute learner conditioned on the recognized object and image.

These works demonstrate the value of explicitly modeling the components and their composition, while recent approaches also introduce context-dependent adaptation for particular semantic factors. CANet, for example, generates parameters for an attribute learner in the attribute-object CZSL setting, while Troika constructs separate vision-language pathways for states, objects, and compositions (Huang et al., 2024). AdaCompose instead treats the *composition operator itself* as the object of adaptation: the complete evidence states being composed determine the rule used to compose them. This formulation is not tied to a particular semantic factor or task-specific recognition pathway. Moreover, because the resulting composition is returned as a new evidence state, it can participate in subsequent operator generation. AdaCompose therefore provides a reusable formulation for evidence-dependent composition of heterogeneous sources, including recursive multi-source composition.

**Adaptive fusion and conditional parameterization.** A large body of work makes neural computation dependent on its input. Attention dynamically determines interactions (Vaswani et al., 2017), bilinear models learn cross-source compatibility (Kim et al., 2018), and feature-wise modulation conditions representations on external information. FiLM (Perez et al., 2018), for example, conditions intermediate features through feature-wise affine transformations. Dynamic Filter Networks (Jia et al., 2016) generate filters conditioned on the input, while HyperNetworks (Ha et al., 2017; Schug et al., 2025) generate the parameters of one network from another network. These approaches establish general mechanisms for input-conditioned computation.

AdaCompose builds on this principle but differs in the *role and structure of the generated parameters*. Rather than introducing conditional parameters within a feature transformation or prediction architecture, AdaCompose uses them to define an explicit *composition operator* whose operands are heterogeneous evidence states. The operator is conditioned on the complete states of both operands, so the same local hypothesis pair can be interpreted differently under different surrounding evidence. The generated operator is shared across all hypothesis pairs between the two sources, providing a common composition rule rather than independent parameterization of individual interactions. Moreover, the resulting composition is returned as a new evidence state that can determine the operator for subsequent composition. Thus, AdaCompose uses established conditional-parameterization machinery to realize a distinct computational abstraction: a reusable, evidence-conditioned, and recursively composable operator for heterogeneous evidence.

**Set-structured and relational representation.** Evidence from pretrained models is naturally represented as a collection of hypotheses whose ordering is arbitrary. Deep Sets (Zaheer et al., 2017) establishes permutation-invariant set processing through shared element transformations and symmetric aggregation, while Set Transformer (Lee et al., 2019) extends set modeling with attention-based interactions. Relation Networks (Santoro et al., 2017) similarly apply a shared function to pairs of objects to model relational structure. These works provide the foundation for representing evidence as a structured set rather than forcing it into a fixed-order vector.

AdaCompose uses this set structure to construct an evidence-conditioned composition operator. A permutation-invariant summary of each complete evidence set determines the operator parameters, while shared pairwise processing makes the resulting relation field equivariant to independent per-

mutations of the two sources. Thus, set processing is not merely used to construct a representation: the complete evidence states jointly determine the *composition rule* applied consistently across their hypothesis pairs. AdaCompose additionally incorporates prediction scores into the evidence state and uses multiplicative gating to control the contribution of each local hypothesis. This yields a single composition cell that can operate across varying numbers of hypotheses without introducing cardinality-specific parameters.

**Recursive and modular reasoning.** Compositional reasoning has also been explored through recursive and modular architectures. Neural Module Networks construct task-specific networks by composing reusable neural modules according to the structure of a question (Andreas et al., 2016b). MAC repeatedly applies a shared reasoning cell while maintaining evolving control and memory (Hudson & Manning, 2018), while other compositional neural models recursively combine typed representations or intermediate denotations (Andreas et al., 2016a; Mao et al., 2019; Rahaman et al., 2021). These approaches demonstrate that reusing a common computational primitive across reasoning steps can provide a natural mechanism for multi-step composition.

AdaCompose shares the principle of recursive reuse but differs in what determines the computation at each step. In prior recursive and modular architectures, computation is typically defined by a shared module, a task-dependent module structure, or a reasoning controller. In AdaCompose, the *composition operator itself is determined by the current evidence*: each application generates a new operator code from its two current evidence states. The resulting intermediate composition therefore affects both the information available at the next step and the operator used to process the next source. Because the composed state has the same evidence-state interface as an input state, the same composition cell can be reused across different numbers of sources without introducing a depth-specific architecture.

**Position of AdaCompose.** The closest prior directions establish mechanisms that are relevant to AdaCompose: CZSL demonstrates the importance of generalizing to unseen compositions; CANet shows that contextual information can generate parameters for a component learner; dynamic filters and hypernetworks establish conditional parameter generation; set-based and relational models provide permutation-consistent processing of unordered evidence; and recursive or modular reasoning demonstrates reuse across multiple reasoning steps. AdaCompose does not claim novelty for these individual mechanisms. Instead, it organizes them around a distinct computational abstraction for evidence composition.

Specifically, AdaCompose treats the *composition operator itself as evidence-dependent*. The complete evidence states being composed determine the operator, the operator produces a new evidence state, and that state can determine the operator for the next composition. Thus, the result of one composition is not only an output of the current computation but also evidence that can change how subsequent evidence is composed. This yields a reusable interface for heterogeneous evidence that is simultaneously context-dependent, set-structured, and recursively composable. The formulation is not restricted to a particular semantic factorization such as attribute-object composition, nor does it require a predefined library of operators or composition-specific modules. This abstraction enables the same core composition cell to be evaluated in controlled relation learning, compositional recognition, video anomaly detection, and multimodal question answering.

## B Why AdaCompose Supports Compositional Reasoning

The AdaCompose cell is designed around four properties that make evidence-adaptive composition possible: complete-evidence conditioning, prediction-score gating, controlled operator adaptation, and recursive closure. These properties arise from the architecture itself rather than from auxiliary objectives. They govern how the composition rule depends on the surrounding evidence, how individual hypotheses contribute to composition, and how the result of one composition can condition the next.

**Complete evidence.** The composition rule should depend on the evidence configuration in which a hypothesis occurs, rather than only on an isolated hypothesis pair. AdaCompose achieves this by generating $\omega_{XY}$ from the permutation-invariant summaries $\boldsymbol{z}_X$ and $\boldsymbol{z}_Y$. Consequently, changing a hypothesis or its prediction score can change the operator applied to all hypothesis pairs between the two sources. The same local pair can therefore receive different relations under different surrounding

evidence configurations. This is distinct from simply providing context as an additional input to a fixed relation function: in AdaCompose, the surrounding evidence determines the function used for the composition itself. Because the summaries use shared encoding followed by symmetric aggregation, they are independent of the arbitrary ordering of hypotheses within each source. The resulting relation field is therefore permutation consistent, changing only by the corresponding row and column reordering.

**Prediction-score gating.** Prediction scores play two complementary roles in the cell. They contribute to the source summary and can therefore influence the generated composition rule, while also directly scaling the corresponding local state: $p_i^X = 0 \implies \boldsymbol{h}_i^X = \mathbf{0}$. The prediction score is therefore not merely an additional input feature; it directly modulates the strength of the corresponding local evidence. At the same time, because prediction scores also affect the source summary, changes in evidence strength can influence the composition rule at the source level. We do not assume that the scores are calibrated or that the resulting relations are monotonic in the scores.

**Controlled adaptation.** Evidence-dependent composition should not require an unrelated network for each evidence configuration. AdaCompose instead retains shared base parameters and adapts them through the low-rank modulation defined in Eq. 4. For each layer, the evidence-dependent update lies in the span of at most $r_{\mathrm{mod}}$ learned rank-one directions (Eq. 5). The model therefore learns a shared family of composition rules whose behavior changes with the current evidence, while constraining the form of that change through shared adaptation directions. The resulting operator is adaptive at the function level, while its parameter variation remains structured and shared across evidence configurations.

**Recursive closure.** For multi-source composition, the result of one composition should remain a valid input to the same composition mechanism. AdaCompose explicitly returns $\mathcal{S}_{XY} = (\boldsymbol{H}_{XY}, \boldsymbol{z}_{XY}, \tau_{XY})$, which has the same structural interface as an input evidence state. Consequently, the cell can be reused directly:

$$(\mathcal{S}_{XY}, \mathcal{S}_Z) \xrightarrow{\mathcal{C}_\theta} (\mathcal{S}_{XYZ}, \boldsymbol{R}_{(XY)Z}). \tag{11}$$

Crucially, the operator for this second composition is generated anew from $\mathcal{S}_{XY}$ and $\mathcal{S}_Z$ rather than reusing $\omega_{XY}$. An earlier composition therefore affects both the evidence available at the next step and the rule used to interpret the next source. This recursive closure allows the same composition mechanism to operate across different numbers of sources without introducing depth-specific parameters.

## C Formal Properties and Proofs

The AdaCompose cell derives several structural properties directly from its parameterization, without auxiliary property-enforcing losses. We establish four properties: permutation consistency, depth-independent global-state boundedness, controlled low-rank operator adaptation, and fixed-query sensitivity to the evidence-generated operator code. These guarantees characterize the composition mechanism itself and are distinct from empirical claims such as compositional generalization. Detailed derivations are provided in Appendix C.

**Permutation-consistent composition.** A set of hypotheses has no intrinsic ordering, so a composition mechanism should not change its interpretation when the same hypotheses are presented in a different order. Consider two evidence states $\mathcal{S}_X$ and $\mathcal{S}_Y$ containing $K_X$ and $K_Y$ hypotheses, respectively. Let $\boldsymbol{P} \in \{0,1\}^{K_X \times K_X}$, $\boldsymbol{Q} \in \{0,1\}^{K_Y \times K_Y}$ be permutation matrices that reorder the hypotheses of $X$ and $Y$. Their actions on the corresponding evidence states are $\boldsymbol{P}\cdot\mathcal{S}_X = (\boldsymbol{P}\boldsymbol{H}_X, \boldsymbol{z}_X, \tau_X)$, $\boldsymbol{Q}\cdot\mathcal{S}_Y = (\boldsymbol{Q}\boldsymbol{H}_Y, \boldsymbol{z}_Y, \tau_Y)$, where the associated prediction scores are reordered consistently with the local states. Because the source summary uses shared element-wise encoding followed by mean aggregation, reordering the hypotheses does not change $\boldsymbol{z}_X$ or $\boldsymbol{z}_Y$. Hence, the operator code generated from these summaries is also unchanged: $\omega_{XY}(\boldsymbol{P}\cdot\mathcal{S}_X, \boldsymbol{Q}\cdot\mathcal{S}_Y) = \omega_{XY}(\mathcal{S}_X, \mathcal{S}_Y)$. The same operator is then applied to the corresponding cross-source hypothesis pairs. Therefore, permutations of the two inputs only permute the rows and columns of the relation field:

$$\boxed{\boldsymbol{R}_{XY}(\boldsymbol{P}\cdot\mathcal{S}_X, \boldsymbol{Q}\cdot\mathcal{S}_Y) = \boldsymbol{P}\,\boldsymbol{R}_{XY}(\mathcal{S}_X, \mathcal{S}_Y)\,\boldsymbol{Q}^\top.} \tag{12}$$

Under the fixed row-major ordering of hypothesis pairs, the composed local states transform according to the corresponding Cartesian-product permutation:

$$\boxed{\boldsymbol{H}_{XY}(\boldsymbol{P}\cdot\mathcal{S}_X, \boldsymbol{Q}\cdot\mathcal{S}_Y) = (\boldsymbol{P}\otimes\boldsymbol{Q})\,\boldsymbol{H}_{XY}(\mathcal{S}_X, \mathcal{S}_Y).} \tag{13}$$

**Proposition 1** (Permutation-consistent recursive composition)**.** *For any fixed ordered binary composition tree, independently permuting the hypotheses within any leaf source changes only the ordering of the corresponding relation fields and local hypothesis states at every descendant node. The global summaries, generated operator codes, type embeddings, and memory weights remain invariant.*

We prove Proposition 1 by first establishing permutation consistency for a single composition step and then extending the result to an arbitrary fixed ordered binary composition tree.

*Proof.* Consider two evidence states $\mathcal{S}_X = (\boldsymbol{H}_X, \boldsymbol{z}_X, \tau_X)$, $\mathcal{S}_Y = (\boldsymbol{H}_Y, \boldsymbol{z}_Y, \tau_Y)$, with $\boldsymbol{H}_X = [\boldsymbol{h}_1^X; \ldots; \boldsymbol{h}_{K_X}^X] \in \mathbb{R}^{K_X \times d_h}$, $\boldsymbol{H}_Y = [\boldsymbol{h}_1^Y; \ldots; \boldsymbol{h}_{K_Y}^Y] \in \mathbb{R}^{K_Y \times d_h}$. Let $\boldsymbol{P} \in \{0,1\}^{K_X \times K_X}$, $\boldsymbol{Q} \in \{0,1\}^{K_Y \times K_Y}$ be permutation matrices acting on the hypotheses of $X$ and $Y$, respectively. The corresponding evidence features and prediction scores are permuted consistently with their local hypotheses.

**Invariance of the source summaries.** The global summary of $X$ is

$$\boldsymbol{z}_X = \phi_{\text{global}}\left(\frac{1}{K_X}\sum_{i=1}^{K_X}\phi_{\text{slot}}\left(\begin{bmatrix}\boldsymbol{e}_i^X\\ p_i^X\end{bmatrix}\right)\right).$$

Let $\pi$ denote the permutation induced by $\boldsymbol{P}$. After applying the permutation, the summary becomes

$$\begin{aligned}\boldsymbol{z}'_X &= \phi_{\text{global}}\left(\frac{1}{K_X}\sum_{i=1}^{K_X}\phi_{\text{slot}}\left(\begin{bmatrix}\boldsymbol{e}_{\pi(i)}^X\\ p_{\pi(i)}^X\end{bmatrix}\right)\right)\\ &= \phi_{\text{global}}\left(\frac{1}{K_X}\sum_{j=1}^{K_X}\phi_{\text{slot}}\left(\begin{bmatrix}\boldsymbol{e}_j^X\\ p_j^X\end{bmatrix}\right)\right) = \boldsymbol{z}_X,\end{aligned}$$

because the permutation only reorders the terms in the sum. Thus, $\boldsymbol{z}'_X = \boldsymbol{z}_X$. The same argument gives $\boldsymbol{z}'_Y = \boldsymbol{z}_Y$. The type embeddings $\tau_X$ and $\tau_Y$ depend only on the corresponding source or composition types and are therefore unchanged by permutations within the sources.

**Invariance of the generated operator.** The operator code is $\omega_{XY} = \tanh\left(G_\psi\left(\begin{bmatrix}\boldsymbol{z}_X\\ \boldsymbol{z}_Y\\ \tau_X\\ \tau_Y\end{bmatrix}\right)\right)$.

Since $\boldsymbol{z}_X$, $\boldsymbol{z}_Y$, $\tau_X$, and $\tau_Y$ are invariant to the within-source permutations, $\omega'_{XY} = \omega_{XY}$. Thus, reordering the hypotheses within either source does not change the composition operator.

**Equivariance of the relation field.** For a hypothesis pair $(i,j)$, the relation score is $r_{ij}^{XY} = \sigma\left(f_{\omega_{XY}}\left(\begin{bmatrix}\boldsymbol{h}_i^X\\ \boldsymbol{h}_j^Y\end{bmatrix}\right)\right)$. Let $\pi$ and $\rho$ denote the permutations induced by $\boldsymbol{P}$ and $\boldsymbol{Q}$, respectively. Because the operator code is invariant,

$$\begin{aligned}r_{ij}^{XY\,\prime} &= \sigma\left(f_{\omega_{XY}}\left(\begin{bmatrix}\boldsymbol{h}_{\pi(i)}^X\\ \boldsymbol{h}_{\rho(j)}^Y\end{bmatrix}\right)\right)\\ &= r_{\pi(i),\rho(j)}^{XY}.\end{aligned}$$

Hence, the relation field is equivariant to the two within-source permutations:

$$\boldsymbol{R}'_{XY} = \boldsymbol{P}\boldsymbol{R}_{XY}\boldsymbol{Q}^\top. \tag{14}$$

**Equivariance of the composed local states.** The local state associated with pair $(i, j)$ is $\boldsymbol{h}_{ij}^{XY} = r_{ij}^{XY}\,\phi_{\text{pair}}\left(\begin{bmatrix}\boldsymbol{h}_i^X\\ \boldsymbol{h}_j^Y\\ r_{ij}^{XY}\end{bmatrix}\right)$. Therefore,

$$\begin{aligned}\boldsymbol{h}_{ij}^{XY\,\prime} &= r_{\pi(i),\rho(j)}^{XY}\,\phi_{\text{pair}}\left(\begin{bmatrix}\boldsymbol{h}_{\pi(i)}^X\\ \boldsymbol{h}_{\rho(j)}^Y\\ r_{\pi(i),\rho(j)}^{XY}\end{bmatrix}\right)\\ &= \boldsymbol{h}_{\pi(i),\rho(j)}^{XY}.\end{aligned}$$

Under the fixed row-major ordering of Cartesian-product pairs, this gives $\boldsymbol{H}'_{XY} = (\boldsymbol{P}\otimes\boldsymbol{Q})\boldsymbol{H}_{XY}$.

**Invariance of the relation-derived summary.** The relation-derived scalar summary is $\bar{r}_{XY} = \frac{1}{K_X K_Y}\sum_{i=1}^{K_X}\sum_{j=1}^{K_Y} r_{ij}^{XY}$. Because Eq. 14 only reorders the entries of $\boldsymbol{R}_{XY}$, its mean remains unchanged: $\bar{r}'_{XY} = \bar{r}_{XY}$. Consequently, $\boldsymbol{z}'_R = \tanh\left(\boldsymbol{W}_R\bar{r}'_{XY} + \boldsymbol{b}_R\right) = \boldsymbol{z}_R$.

**Invariance of the global memory.** The memory weights are $\boldsymbol{a}_{XY} = \text{softmax}\left(\boldsymbol{M}_2\ \text{GELU}\left(\boldsymbol{M}_1\begin{bmatrix}\boldsymbol{z}_X\\ \boldsymbol{z}_Y\\ \boldsymbol{z}_R\end{bmatrix} + \boldsymbol{b}_1\right) + \boldsymbol{b}_2\right)$. All three inputs are invariant under the within-source permutations. Therefore, $\boldsymbol{a}'_{XY} = \boldsymbol{a}_{XY}$. It follows that

$$\begin{aligned}\boldsymbol{z}'_{XY} &= \alpha'_{XY,1}\boldsymbol{z}'_X + \alpha'_{XY,2}\boldsymbol{z}'_Y + \alpha'_{XY,3}\boldsymbol{z}'_R\\ &= \alpha_{XY,1}\boldsymbol{z}_X + \alpha_{XY,2}\boldsymbol{z}_Y + \alpha_{XY,3}\boldsymbol{z}_R\\ &= \boldsymbol{z}_{XY}.\end{aligned}$$

Thus, $\boldsymbol{z}'_{XY} = \boldsymbol{z}_{XY}$. The composed type embedding $\tau_{XY}$ depends only on the composition type and is independent of hypothesis ordering. Hence, $\tau'_{XY} = \tau_{XY}$.

Combining these results, a single composition step satisfies

$$\mathcal{C}_\theta\left(\boldsymbol{P}\cdot\mathcal{S}_X, \boldsymbol{Q}\cdot\mathcal{S}_Y\right) = \left(((\boldsymbol{P}\otimes\boldsymbol{Q})\boldsymbol{H}_{XY}, \boldsymbol{z}_{XY}, \tau_{XY}), \boldsymbol{P}\boldsymbol{R}_{XY}\boldsymbol{Q}^\top\right).$$

**Recursive composition.** We now extend the result to a fixed ordered binary composition tree. The claim holds at every leaf because a within-source permutation changes only the ordering of its local hypotheses, while leaving its global summary and type embedding invariant.

Consider an arbitrary internal node $v$ with children $v_l$ and $v_r$. Assume that the claim holds for both children. Their global summaries and type embeddings are therefore invariant, while their local states transform according to the corresponding Cartesian-product permutations. Applying the single-step result to the composition at node $v$ leaves the generated operator code and global state invariant, while transforming the relation field and local states according to the corresponding row-column and Cartesian-product permutations. Thus, the claim also holds at node $v$. Induction over the internal nodes of the fixed binary tree establishes the result for every descendant and, in particular, for the root. This proves Proposition 1.

□

This property is deliberately restricted to permutations *within* each source. It does not imply that the two sources can be exchanged freely, nor does it imply commutativity or associativity of the composition operator.

**Depth-independent global-state bound.** Recursive composition repeatedly feeds a composed state back into the same cell. Without a constraint on the global memory update, its magnitude could potentially grow with composition depth. AdaCompose instead constructs the new global summary as a convex combination of the two input summaries and a bounded relation-derived summary:

$$\boldsymbol{z}_{XY} = \alpha_1\boldsymbol{z}_X + \alpha_2\boldsymbol{z}_Y + \alpha_3\boldsymbol{z}_R, \qquad \alpha_k \geq 0, \qquad \sum_{k=1}^{3}\alpha_k = 1, \tag{15}$$

where $\boldsymbol{z}_R = \tanh\left(\boldsymbol{W}_R\bar{r}_{XY} + \boldsymbol{b}_R\right)$. Thus, every component of $\boldsymbol{z}_R$ lies in $(-1, 1)$, and the update cannot increase the infinity norm beyond the largest norm already present in the two input summaries or the unit bound induced by $\boldsymbol{z}_R$.

**Proposition 2** (Depth-independent global-state bound)**.** *For any finite ordered binary composition tree and any node $v$,*

$$\boxed{\|\boldsymbol{z}_v\|_\infty \le \max\left\{1, \max_{m\in\text{leaves}(v)} \|\boldsymbol{z}_m\|_\infty\right\}.} \tag{16}$$

*Consequently, if every leaf satisfies $\|\boldsymbol{z}_m\|_\infty \le B$, then every intermediate global state satisfies $\|\boldsymbol{z}_v\|_\infty \le \max\{B, 1\}$, independently of the composition depth or tree shape.*

We next prove Proposition 2, which establishes the depth-independent global-state bound.

*Proof.* At an internal node $v$ with children $v_l$ and $v_r$, AdaCompose updates the global state as

$$\boldsymbol{z}_v = \alpha_1 \boldsymbol{z}_{v_l} + \alpha_2 \boldsymbol{z}_{v_r} + \alpha_3 \boldsymbol{z}_R, \qquad \alpha_k \ge 0, \qquad \sum_{k=1}^{3} \alpha_k = 1, \tag{17}$$

where the coefficients are produced by the softmax memory mechanism. The relation-derived summary is $\boldsymbol{z}_R = \tanh\left(\boldsymbol{W}_R \bar{r} + \boldsymbol{b}_R\right)$ ($\bar{r}$ for simplicity), and therefore $\|\boldsymbol{z}_R\|_\infty \le 1$. Applying the triangle inequality to Eq. 17 gives

$$\begin{aligned}\|\boldsymbol{z}_v\|_\infty &\le \alpha_1 \|\boldsymbol{z}_{v_l}\|_\infty + \alpha_2 \|\boldsymbol{z}_{v_r}\|_\infty + \alpha_3 \|\boldsymbol{z}_R\|_\infty \\ &\le \alpha_1 \|\boldsymbol{z}_{v_l}\|_\infty + \alpha_2 \|\boldsymbol{z}_{v_r}\|_\infty + \alpha_3.\end{aligned}$$

Since $\alpha_1, \alpha_2, \alpha_3$ are nonnegative and sum to one, the right-hand side is a convex combination of $\|\boldsymbol{z}_{v_l}\|_\infty$, $\|\boldsymbol{z}_{v_r}\|_\infty$, and 1. Hence,

$$\|\boldsymbol{z}_v\|_\infty \le \max\left\{\|\boldsymbol{z}_{v_l}\|_\infty, \|\boldsymbol{z}_{v_r}\|_\infty, 1\right\}. \tag{18}$$

We now prove the tree-level bound by induction over the composition tree. For a leaf node $m$, $\|\boldsymbol{z}_m\|_\infty \le \max\left\{1, \max_{u\in\text{leaves}(m)} \|\boldsymbol{z}_u\|_\infty\right\}$, which holds immediately because $m$ is its only leaf. Now consider an internal node $v$ and assume that the claimed bound holds for both children $v_l$ and $v_r$. By Eq. 18,

$$\begin{aligned}\|\boldsymbol{z}_v\|_\infty &\le \max\left\{\|\boldsymbol{z}_{v_l}\|_\infty, \|\boldsymbol{z}_{v_r}\|_\infty, 1\right\} \\ &\le \max\left\{1, \max_{u\in\text{leaves}(v_l)} \|\boldsymbol{z}_u\|_\infty, \max_{u\in\text{leaves}(v_r)} \|\boldsymbol{z}_u\|_\infty\right\}.\end{aligned}$$

Because the leaves under $v$ are exactly the union of the leaves under $v_l$ and $v_r$, $\text{leaves}(v) = \text{leaves}(v_l) \cup \text{leaves}(v_r)$, we obtain $\|\boldsymbol{z}_v\|_\infty \le \max\left\{1, \max_{u\in\text{leaves}(v)} \|\boldsymbol{z}_u\|_\infty\right\}$. Therefore, for every node $v$ in the composition tree,

$$\boxed{\|\boldsymbol{z}_v\|_\infty \le \max\left\{1, \max_{u\in\text{leaves}(v)} \|\boldsymbol{z}_u\|_\infty\right\}.} \tag{19}$$

If every leaf satisfies $\|\boldsymbol{z}_u\|_\infty \le B$, then Eq. 19 immediately gives $\|\boldsymbol{z}_v\|_\infty \le \max\{B, 1\}$ for every node $v$, independently of the depth or shape of the composition tree. This proves Proposition 2.

□

The result concerns the global summary only. It does not bound the local hypothesis states, imply contractivity of the complete composition operator, or guarantee lossless information propagation through arbitrarily deep compositions.

**Controlled low-rank operator adaptation.** Evidence-adaptive composition requires the composition rule to change with the evidence, but an unconstrained input-specific network would make it difficult to characterize what is shared across evidence configurations. AdaCompose instead adapts a shared base relation operator through a low-rank modulation (Eq. 4). The evidence therefore does not select an unrelated network for every input. Instead, it changes the shared operator through a finite set of learned adaptation directions. Let the modulation rank be $r_{\text{mod}}$. For simplicity, we write $r$ and $\omega$ instead of $r_{ij}^{XY}$ and $\omega_{XY}$ when the context is clear. Then

$$\boxed{\text{rank}\left(\boldsymbol{W}_\ell(\omega) - \boldsymbol{W}_\ell^0\right) \le r_{\text{mod}}.} \tag{20}$$

Moreover, because each component of $\boldsymbol{g}_\ell(\omega_{XY})$ lies in $(-1, 1)$, the adaptive update itself is bounded in spectral norm:

$$\left\|\boldsymbol{W}_\ell(\omega) - \boldsymbol{W}_\ell^0\right\|_2 \le \|\boldsymbol{U}_\ell\|_2 \|\boldsymbol{V}_\ell\|_2. \tag{21}$$

**Proposition 3** (Controlled operator family)**.** *The evidence-conditioned relation operators form a shared continuous family whose layer-wise deviations from the learned base operators have rank at most $r_{\mathrm{mod}}$. Consequently, evidence can change the composition rule, but only through a restricted set of learned adaptation directions.*

We next prove Proposition 3, which establishes the controlled low-rank operator property.

*Proof.* For simplicity, we write $r$ and $\omega$ instead of $r_{ij}^{XY}$ and $\omega_{XY}$ when the context is clear. For layer $\ell$, the evidence-conditioned weight matrix is $\boldsymbol{W}_\ell(\omega) = \boldsymbol{W}_\ell^0 + \boldsymbol{U}_\ell \operatorname{diag}(\boldsymbol{g}_\ell(\omega)) \boldsymbol{V}_\ell^\top$, $\boldsymbol{g}_\ell(\omega) = \tanh(\boldsymbol{B}_\ell \omega + \boldsymbol{b}_\ell)$, where $\boldsymbol{U}_\ell \in \mathbb{R}^{d_{\mathrm{out}} \times r_{\mathrm{mod}}}$, $\boldsymbol{V}_\ell \in \mathbb{R}^{d_{\mathrm{in}} \times r_{\mathrm{mod}}}$, and $r_{\mathrm{mod}}$ is the modulation rank.

**Rank bound.** Subtracting the shared base matrix gives $\boldsymbol{W}_\ell(\omega) - \boldsymbol{W}_\ell^0 = \boldsymbol{U}_\ell \operatorname{diag}(\boldsymbol{g}_\ell(\omega)) \boldsymbol{V}_\ell^\top$. Because the middle matrix is $r_{\mathrm{mod}} \times r_{\mathrm{mod}}$,

$$\begin{aligned}\operatorname{rank}\left(\boldsymbol{W}_\ell(\omega) - \boldsymbol{W}_\ell^0\right) &\le \min\left\{\operatorname{rank}(\boldsymbol{U}_\ell), r_{\mathrm{mod}}, \operatorname{rank}(\boldsymbol{V}_\ell^\top)\right\} \\ &\le r_{\mathrm{mod}}.\end{aligned}$$

Equivalently,

$$\boldsymbol{W}_\ell(\omega) - \boldsymbol{W}_\ell^0 = \sum_{k=1}^{r_{\mathrm{mod}}} g_{\ell,k}(\omega) \boldsymbol{U}_{\ell,:,k} \boldsymbol{V}_{\ell,:,k}^\top.$$

Thus, the evidence-dependent matrix update lies in the span of at most $r_{\mathrm{mod}}$ learned rank-one directions. Hence,

$$\boxed{\operatorname{rank}\left(\boldsymbol{W}_\ell(\omega) - \boldsymbol{W}_\ell^0\right) \le r_{\mathrm{mod}}.}$$

**Bound on the adaptive update.** Because $\tanh$ maps real values into $(-1, 1)$, $\|\operatorname{diag}(\boldsymbol{g}_\ell(\omega))\|_2 = \|\boldsymbol{g}_\ell(\omega)\|_\infty \le 1$. Therefore, by submultiplicativity of the spectral norm,

$$\begin{aligned}\left\|\boldsymbol{W}_\ell(\omega) - \boldsymbol{W}_\ell^0\right\|_2 &\le \|\boldsymbol{U}_\ell\|_2 \left\|\operatorname{diag}(\boldsymbol{g}_\ell(\omega))\right\|_2 \|\boldsymbol{V}_\ell^\top\|_2 \\ &\le \|\boldsymbol{U}_\ell\|_2 \|\boldsymbol{V}_\ell\|_2.\end{aligned}$$

Since $\|\boldsymbol{V}_\ell^\top\|_2 = \|\boldsymbol{V}_\ell\|_2$, we obtain

$$\boxed{\left\|\boldsymbol{W}_\ell(\omega) - \boldsymbol{W}_\ell^0\right\|_2 \le \|\boldsymbol{U}_\ell\|_2 \|\boldsymbol{V}_\ell\|_2.}$$

**Lipschitz dependence on the operator code.** For two operator codes $\omega$ and $\omega'$, the 1-Lipschitz property of $\tanh$ gives

$$\begin{aligned}\|\boldsymbol{g}_\ell(\omega) - \boldsymbol{g}_\ell(\omega')\|_2 &\le \|\boldsymbol{B}_\ell(\omega - \omega')\|_2 \\ &\le \|\boldsymbol{B}_\ell\|_2 \|\omega - \omega'\|_2.\end{aligned}$$

Because the spectral norm of a diagonal matrix equals the infinity norm of its diagonal, and

$$\|\boldsymbol{g}_\ell(\omega) - \boldsymbol{g}_\ell(\omega')\|_\infty \le \|\boldsymbol{g}_\ell(\omega) - \boldsymbol{g}_\ell(\omega')\|_2,$$

we have

$$\begin{aligned}\|\boldsymbol{W}_\ell(\omega) - \boldsymbol{W}_\ell(\omega')\|_2 &\le \|\boldsymbol{U}_\ell\|_2 \|\boldsymbol{V}_\ell\|_2 \|\boldsymbol{g}_\ell(\omega) - \boldsymbol{g}_\ell(\omega')\|_\infty \\ &\le \|\boldsymbol{U}_\ell\|_2 \|\boldsymbol{V}_\ell\|_2 \|\boldsymbol{B}_\ell\|_2 \|\omega - \omega'\|_2.\end{aligned}$$

Hence,

$$\boxed{\|\boldsymbol{W}_\ell(\omega) - \boldsymbol{W}_\ell(\omega')\|_2 \le \|\boldsymbol{U}_\ell\|_2 \|\boldsymbol{V}_\ell\|_2 \|\boldsymbol{B}_\ell\|_2 \|\omega - \omega'\|_2.} \tag{22}$$

The evidence-conditioned operator therefore varies Lipschitz-continuously with the operator code, while its deviation from the shared base weights has rank at most $r_{\mathrm{mod}}$. This proves Proposition 3.

□

Hence, AdaCompose learns a structured family of composition rules sharing a common base, rather than an independent unconstrained relation network for every evidence configuration.

This distinction is central to the formulation: the model is adaptive at the *operator level*, while the space of possible adaptations remains parameterized by shared learned directions.

**Fixed-query sensitivity.** The previous property characterizes how the composition operator changes with its conditioning code. We next ask whether such operator changes lead to controlled changes in the resulting relation. To isolate this effect, hold a local hypothesis pair fixed: $\boldsymbol{q}_{ij}^{XY} = \begin{bmatrix} \boldsymbol{h}_i^X \\ \boldsymbol{h}_j^Y \end{bmatrix}$. For two operator codes $\omega$ and $\omega'$, the low-rank construction provides the layer-wise bound

$$\|\boldsymbol{W}_\ell(\omega) - \boldsymbol{W}_\ell(\omega')\|_2 \le \|\boldsymbol{U}_\ell\|_2 \|\boldsymbol{V}_\ell\|_2 \|\boldsymbol{B}_\ell\|_2 \|\omega - \omega'\|_2. \tag{23}$$

This follows from the 1-Lipschitz property of $\tanh$ and the submultiplicativity of the spectral norm. For a fixed and bounded hypothesis pair, let $\boldsymbol{q} = \begin{bmatrix} \boldsymbol{h}_i^X \\ \boldsymbol{h}_j^Y \end{bmatrix}$. Let $r(\boldsymbol{q}; \omega)$ denote the relation score obtained by evaluating this pair with operator code $\omega$. Then there exists a finite constant $C_{\mathrm{rel}}$, determined by the learned network parameters and the bound on $\boldsymbol{q}$, such that

$$\boxed{|r(\boldsymbol{q}; \omega) - r(\boldsymbol{q}; \omega')| \le C_{\mathrm{rel}} \|\omega - \omega'\|_2.} \tag{24}$$

**Proposition 4** (Fixed-query sensitivity)**.** *For any bounded fixed local hypothesis pair, the relation score is Lipschitz continuous with respect to the evidence-generated operator code.*

We prove Proposition 4 by first establishing the Lipschitz dependence of the dynamic weight matrices on the operator code and then propagating this dependence through the finite relation network.

*Proof.* Fix a local hypothesis pair $\boldsymbol{q} = \begin{bmatrix} \boldsymbol{h}_i^X \\ \boldsymbol{h}_j^Y \end{bmatrix}$, $\|\boldsymbol{q}\|_2 \le B_q$. Consider the relation network with $L$ dynamic affine layers. Let $\boldsymbol{u}_0(\omega) = \boldsymbol{q}$, and for $\ell = 1, \dots, L-1$ define the hidden states by

$$\boldsymbol{u}_\ell(\omega) = \phi_\ell\left(\boldsymbol{W}_\ell(\omega)\boldsymbol{u}_{\ell-1}(\omega) + \boldsymbol{b}_\ell\right),$$

where $\phi_\ell$ is the GELU activation. The final layer produces the relation logit

$$a(\boldsymbol{q}; \omega) = \boldsymbol{W}_L(\omega)\boldsymbol{u}_{L-1}(\omega) + \boldsymbol{b}_L, \tag{25}$$

and the relation score is

$$r(\boldsymbol{q}; \omega) = \sigma\big(a(\boldsymbol{q}; \omega)\big).$$

Let $L_\ell < \infty$ denote a global Lipschitz constant of the activation $\phi_\ell$ at hidden layer $\ell$. From Eq. 22 in the proof of the low-rank operator property,

$$\|\boldsymbol{W}_\ell(\omega) - \boldsymbol{W}_\ell(\omega')\|_2 \le \|\boldsymbol{U}_\ell\|_2 \|\boldsymbol{V}_\ell\|_2 \|\boldsymbol{B}_\ell\|_2 \|\omega - \omega'\|_2. \tag{26}$$

The bounded low-rank update also gives

$$\|\boldsymbol{W}_\ell(\omega)\|_2 \le \|\boldsymbol{W}_\ell^0\|_2 + \|\boldsymbol{U}_\ell\|_2 \|\boldsymbol{V}_\ell\|_2 =: W_\ell. \tag{27}$$

Because the query is bounded and the network has finitely many layers, its hidden states are bounded for finite parameters. In particular, define $Q_0 = B_q$, $Q_\ell = L_\ell\left(W_\ell Q_{\ell-1} + \|\boldsymbol{b}_\ell\|_2\right)$, $\ell = 1, \dots, L-1$. Then

$$\|\boldsymbol{u}_\ell(\omega)\|_2 \le Q_\ell, \qquad \ell = 0, \dots, L-1. \tag{28}$$

Now consider two operator codes $\omega$ and $\omega'$ and define $D_\ell = \|\boldsymbol{u}_\ell(\omega) - \boldsymbol{u}_\ell(\omega')\|_2$. Since the local query is fixed, $D_0 = 0$. For each hidden layer $\ell = 1, \dots, L-1$, the Lipschitz property of $\phi_\ell$ gives

$$\begin{aligned} D_\ell &\le L_\ell \left\|\boldsymbol{W}_\ell(\omega)\boldsymbol{u}_{\ell-1}(\omega) - \boldsymbol{W}_\ell(\omega')\boldsymbol{u}_{\ell-1}(\omega')\right\|_2 \\ &\le L_\ell\Big(\left\|\boldsymbol{W}_\ell(\omega) - \boldsymbol{W}_\ell(\omega')\right\|_2 \left\|\boldsymbol{u}_{\ell-1}(\omega)\right\|_2 + \left\|\boldsymbol{W}_\ell(\omega')\right\|_2 D_{\ell-1}\Big). \end{aligned}$$

Using Eqs. 26, 27, and 28, we obtain

$$D_\ell \le A_\ell D_{\ell-1} + B_\ell \|\omega - \omega'\|_2, \tag{29}$$

where $A_\ell = L_\ell W_\ell$, $B_\ell = L_\ell \|\boldsymbol{U}_\ell\|_2 \|\boldsymbol{V}_\ell\|_2 \|\boldsymbol{B}_\ell\|_2 Q_{\ell-1}$. Since $D_0 = 0$, repeated application of Eq. 29 gives $D_{L-1} \le C_{L-1}\|\omega - \omega'\|_2$, where $C_{L-1} = \sum_{\ell=1}^{L-1} B_\ell \prod_{t=\ell+1}^{L-1} A_t$, with the empty product defined as 1. Since the network has finitely many layers and all parameters are finite, $C_{L-1} < \infty$. It remains to bound the perturbation of the final relation logit. From Eq. 25,

$$\begin{aligned}
|a(\boldsymbol{q};\omega) - a(\boldsymbol{q};\omega')| &\le \|\boldsymbol{W}_L(\omega)\boldsymbol{u}_{L-1}(\omega) - \boldsymbol{W}_L(\omega')\boldsymbol{u}_{L-1}(\omega')\|_2 \\
&\le \|\boldsymbol{W}_L(\omega) - \boldsymbol{W}_L(\omega')\|_2 \, \|\boldsymbol{u}_{L-1}(\omega)\|_2 \\
&\quad + \|\boldsymbol{W}_L(\omega')\|_2 \|\boldsymbol{u}_{L-1}(\omega) - \boldsymbol{u}_{L-1}(\omega')\|_2 \\
&\le \left(\|\boldsymbol{U}_L\|_2 \|\boldsymbol{V}_L\|_2 \|\boldsymbol{B}_L\|_2 Q_{L-1} + W_L C_{L-1}\right) \|\omega - \omega'\|_2.
\end{aligned}$$

Define $C_a = \|\boldsymbol{U}_L\|_2 \|\boldsymbol{V}_L\|_2 \|\boldsymbol{B}_L\|_2 Q_{L-1} + W_L C_{L-1}$. Then $|a(\boldsymbol{q};\omega) - a(\boldsymbol{q};\omega')| \le C_a \|\omega - \omega'\|_2$.

Finally, the sigmoid function is $1/4$-Lipschitz. Therefore,

$$\begin{aligned}
|r(\boldsymbol{q};\omega) - r(\boldsymbol{q};\omega')| &= \left|\sigma\big(a(\boldsymbol{q};\omega)\big) - \sigma\big(a(\boldsymbol{q};\omega')\big)\right| \\
&\le \frac{1}{4} \left|a(\boldsymbol{q};\omega) - a(\boldsymbol{q};\omega')\right| \\
&\le \frac{C_a}{4} \|\omega - \omega'\|_2.
\end{aligned}$$

Setting $C_{\mathrm{rel}} = \frac{C_a}{4}$, we obtain

$$\boxed{|r(\boldsymbol{q};\omega) - r(\boldsymbol{q};\omega')| \le C_{\mathrm{rel}} \|\omega - \omega'\|_2.} \tag{30}$$

This proves Proposition 4.

□

The fixed-query condition is essential to the result. A change in the underlying evidence can simultaneously alter the local hypothesis states and the generated operator code. Therefore, Eq. 30 characterizes the sensitivity of the relation score to the evidence-generated operator code for a fixed local query; it does not constitute a robustness guarantee for arbitrary perturbations of the raw evidence.

**Scope.** The formal guarantees above apply to the generic AdaCompose cell described in Sec. 2.2. Their scope and limitations are discussed in Appendix E.

## D Recursive Composition and Task Instantiation

The binary composition cell in Section 2.2 provides the basic reusable unit of AdaCompose. Because its input and output share the same evidence-state interface, the cell can be recursively applied to multiple heterogeneous sources. Crucially, each intermediate composition becomes part of the evidence used to generate the rule for the next composition. Thus, AdaCompose does not simply accumulate representations: the outcome of one composition can change how subsequent evidence is interpreted.

**Multi-source recursion.** Let $\mathcal{S}_1, \ldots, \mathcal{S}_M$ denote $M$ evidence states. A left-associated composition is defined recursively as

$$\mathcal{S}_{1:1} = \mathcal{S}_1, \qquad (\mathcal{S}_{1:m}, \boldsymbol{R}_{(1:m-1),m}) = \mathcal{C}_\theta\left(\mathcal{S}_{1:m-1}, \mathcal{S}_m\right), \quad m = 2, \ldots, M, \tag{31}$$

where $\mathcal{C}_\theta$ denotes the shared AdaCompose composition cell. At each step, a new operator code is generated from the global summaries and type embeddings of the two current states:

$$\omega_{(1:m-1),m} = \tanh G_\psi \left( \begin{bmatrix} \boldsymbol{z}_{1:m-1} \\ \boldsymbol{z}_m \\ \tau_{1:m-1} \\ \tau_m \end{bmatrix} \right). \tag{32}$$

Therefore, the operator at step $m$ depends on the evidence accumulated in $\mathcal{S}_{1:m-1}$. In particular, after forming $\mathcal{S}_{1:2}$, the cell generates a new operator from $\mathcal{S}_{1:2}$ and $\mathcal{S}_3$ rather than reusing the operator generated for the first composition. The same process continues at each subsequent step.

Thus, an intermediate composition is not merely an output representation; it becomes evidence that determines how the next source is composed.

More generally, composition can be organized by any fixed ordered binary tree. For an internal node $v$ with children $v_l$ and $v_r$,

$$(S_v, R_v) = \mathcal{C}_\theta(S_{v_l}, S_{v_r}). \tag{33}$$

The same parameters $\theta$ are shared across all internal nodes, while the operator code is regenerated separately at each node from the corresponding input states. The tree therefore specifies the computational decomposition, not a separate set of learned composition parameters. AdaCompose does not assume commutativity or associativity; when a task imposes a particular order of evidence integration, that order is represented by the chosen composition tree.

The generic Cartesian interaction evaluates all hypothesis pairs between two input states. If the states contain $K_X$ and $K_Y$ hypotheses, respectively, one composition step evaluates $K_X K_Y$ pairwise relations. Under left-associated recursion, the total number of pairwise evaluations is

$$N_{\text{pair}} = \sum_{m=2}^{M} \prod_{j=1}^{m} K_j, \tag{34}$$

under the explicit Cartesian construction. Parameter sharing is independent of this computational cost: reusing the same composition cell does not by itself eliminate the cost of evaluating the required hypothesis pairs. When the task provides a structured candidate space, the interaction set can instead be restricted or factorized to reduce this cost, as described below.

**Task-specific realization.** Section 2.2 defines the generic Cartesian composition cell. The downstream models adapt its evidence-conditioned composition principle to their candidate spaces and prediction objectives. The following descriptions distinguish the generic construction from task-specific interaction structures and readouts. The formal guarantees concern the generic cell and their applicability to specialized implementations depends on the corresponding computational structure (Appendix E).

*Video anomaly detection.* Each stream first combines action and object evidence. The resulting intermediate state is then combined with scene evidence to condition the second operator. The VAD realization uses evidence-weighted concept factors and low-rank interactions to produce window-level scores. Interaction feedback carries the first-stage interaction into the intermediate state used for subsequent composition. The full system combines two composition streams and a normal-action reference through a learned temporal fusion head. Implementation and ablation details are provided in Appendix F.3.

*Compositional zero-shot recognition.* For compositional zero-shot recognition, the evidence states are aligned with candidate semantic components and their possible compositions. AdaCompose recursively combines the corresponding evidence while retaining candidate structure in the intermediate state. The resulting candidate-level evidence therefore contributes to the operator that determines relations between individual components. Where the candidate space admits structured restrictions or factorization, the corresponding interaction set can be reduced without changing the underlying evidence-conditioned composition principle.

*Visual question answering.* AdaCompose first composes question and textual-context states. The resulting state and the visual state condition a second relation operator. Masked averaging of its relation logits over valid question-context hypothesis pairs yields one logit per answer choice, optimized with cross-entropy. The final stage reads out relation logits without constructing another composed state.

Across these tasks, the composition cell and its evidence-conditioned parameterization remain shared at the framework level, while the evidence organization and interaction structure are adapted to each task. The key principle is that composition is itself an evidence-generating operation: the current evidence determines the composition rule, and the resulting composed state becomes the evidence that determines subsequent composition. This recursive reuse extends AdaCompose beyond adaptive pairwise fusion to compositional reasoning over multiple heterogeneous evidence sources.

# E Scope and Limitations of the Formal Guarantees

The formal properties established in Section 2.3 characterize structural guarantees of the generic AdaCompose composition cell. They should be distinguished from empirical claims about downstream performance or compositional generalization. In particular, the construction does not imply the following properties:

i. **Commutativity or associativity.** The composition operator is ordered and does not guarantee $\mathcal{C}_\theta(\mathcal{S}_X, \mathcal{S}_Y) = \mathcal{C}_\theta(\mathcal{S}_Y, \mathcal{S}_X)$ or invariance to different parenthesizations.

ii. **Monotonicity or calibration.** Although prediction scores directly gate local states, the architecture does not guarantee that relation scores are monotonic in the input prediction scores or that the scores are calibrated probabilities.

iii. **Bounded local states.** Sec. 2.3 bounds the global summary $\boldsymbol{z}$ under recursive composition, but does not establish a corresponding bound on the local hypothesis states in $\boldsymbol{H}$.

iv. **Contractivity or lossless recursion.** The bounded global-state update does not imply that the complete composition operator is contractive, nor that recursive composition preserves all information from its inputs.

v. **Robustness to arbitrary evidence perturbations.** The fixed-query sensitivity result concerns changes in the generated operator code while holding the local hypothesis pair fixed. It therefore does not provide a general robustness guarantee for perturbations of the underlying evidence.

The use of variable hypothesis cardinality likewise means that the same parameters can be reused across different numbers of hypotheses; it does not imply cardinality-independent computational cost. The generic Cartesian composition explicitly evaluates the constructed hypothesis pairs, yielding $K_X K_Y$ pairwise evaluations for a single composition step. Under left-associated recursion, this leads to the complexity characterized in Eq. 34. Task-specific candidate restrictions or factorizations can reduce this cost, but such specialized interaction structures require their own corresponding permutation analysis.

Finally, the formal guarantees apply directly to the generic Cartesian composition cell defined in Section 2.2. When a task uses a restricted or factorized interaction structure, the relevant properties should be re-established for that interaction structure rather than assumed automatically from the generic construction.

# F Experimental Details

## F.1 Controlled relation learning

We distinguish fitting a fixed pairwise function from learning a relation whose value also depends on the surrounding evidence. The contextual targets use permutation-invariant set statistics to shift the evaluation coordinates. Thus, the same queried pair can have different targets in different contexts.

**Context-dependent coordinates.** Let $P = \{p_i\}_{i=1}^{K}$ and $Q = \{q_j\}_{j=1}^{L}$ contain coordinates in $[0, 1]$. Their means $\mu_P, \mu_Q$ and population standard deviations $s_P, s_Q$ include the queried elements. We define

$$\begin{aligned} c &= 0.075 \tanh\big(2(\mu_P - \mu_Q) + 1.5(s_P - s_Q)\big), \\ \delta_x &= 0.10 \tanh\big(2(\mu_P - 0.5) + 1.25(s_Q - 0.25)\big), \\ \delta_y &= 0.10 \tanh\big(2(\mu_Q - 0.5) - 1.25(s_P - 0.25)\big), \\ u_{ij} &= \frac{p_i + q_j}{2} + c. \end{aligned} \tag{35}$$

These shifts modify where each function is evaluated, rather than adding an offset to its output. Throughout this subsection, $\sigma$ denotes the sigmoid.

**Weierstrass target.** A seven-scale cosine series converts the shared coordinate shift into phase changes at multiple frequencies:

$$Y_{ij}^{\mathrm{W}} = \sigma\left(\frac{3(1-0.65)}{1-0.65^7}\sum_{n=0}^{6} 0.65^n \cos\bigl(2\pi 3^n u_{ij} + 0.35n\bigr)\right). \tag{36}$$

**Julia target.** Context shifts the initial condition of a complex quadratic iteration:

$$\begin{aligned} z_{ij}^{(0)} &= 3(p_i - 0.5 + \delta_x) + 3\mathrm{i}(q_j - 0.5 + \delta_y), \\ z_{ij}^{(r+1)} &= \bigl(z_{ij}^{(r)}\bigr)^2 - 0.745 + 0.113\mathrm{i}, \qquad r = 0, \ldots, 63. \end{aligned} \tag{37}$$

Here, i is the imaginary unit. Escape is checked after each update. Define

$$\begin{aligned} r_{ij}^{\star} &= \min\left(\left\{r \in \{1,\ldots,64\} : |z_{ij}^{(r)}| > 2\right\} \cup \{\infty\}\right), \\ \rho_{ij} &= \begin{cases} 2^{-r_{ij}^{\star}} \log|z_{ij}^{(r_{ij}^{\star})}|, & r_{ij}^{\star} < \infty, \\ 0, & r_{ij}^{\star} = \infty. \end{cases} \end{aligned} \tag{38}$$

The resulting target is

$$Y_{ij}^{\mathrm{J}} = \sigma\left(3\left[1 - 2\tanh\left(5\sqrt{\rho_{ij}}\right)\right]\right). \tag{39}$$

**Hardy target.** Context shifts the evaluation point along an oscillatory Hardy $Z$ function:

$$\begin{aligned} t_{ij} &= 100 + 150\min\{1, \max\{0, u_{ij}\}\}, \\ Y_{ij}^{\mathrm{H}} &= \sigma\left(2.5\tanh\left(\frac{\operatorname{asinh}(\widehat{Z}(t_{ij}))}{1.5}\right)\right). \end{aligned} \tag{40}$$

The approximation $\widehat{Z}$ linearly interpolates 2,049 uniformly spaced values on $[100, 250]$, computed with 30 decimal digits of precision.

**Sampling and model inputs.** The fixed setting uses $c = \delta_x = \delta_y = 0$ and independent coordinates from $\mathcal{U}[0,1]$. For each set in the contextual setting, we independently draw a center from $\mathcal{U}[0.25, 0.75]$ and a half-width from $\mathcal{U}[0.15, 0.40]$, sample uniformly within the resulting interval, and clip coordinates to $[0,1]$. The two settings therefore differ in both target construction and input distribution; their comparison is not an isolated test of coordinate warping.

AdaCompose represents coordinate $p$ as

$$e(p) = \bigl[p,\ p^2,\ \sin(2\pi p),\ \cos(2\pi p)\bigr], \tag{41}$$

and also places $p$ in the confidence channel. This synthetic use of the channel does not evaluate confidence calibration. SIREN and KAN receive only $(p_i, q_j)$. DeepONet additionally receives $(\mu_P, \mu_Q, s_P, s_Q)$. The synthetic AdaCompose model uses the encoder, operator generator, and relation field, without recursive composition.

**Optimization and error reporting.** Parameter counts are approximately matched: 20,657 for AdaCompose, 19,009 for SIREN, 19,001 for KAN, and 17,153 for DeepONet. KAN uses compact cubic B-splines. All models use Adam, batch size 64, 6,000 updates, gradient-norm clipping at 5, and cosine learning-rate decay to one tenth of the initial value. The initial learning rate is $2 \times 10^{-3}$ for AdaCompose and $10^{-3}$ for the other models.

Validation MSE on 256 fields selects the checkpoint. Validation is performed after the first update, every 250 updates, and after the final update. For each seed, all methods share 1,536 independent test fields, containing 98,304 scalar entries. Normalized error is computed over all test entries:

$$\mathrm{nRMSE} = \sqrt{\frac{\frac{1}{N}\sum_{n=1}^{N}(\widehat{y}_n - y_n)^2}{\frac{1}{N}\sum_{n=1}^{N}(y_n - \bar{y})^2}}, \qquad \bar{y} = \frac{1}{N}\sum_{n=1}^{N} y_n. \tag{42}$$

Table 1 reports the mean and population standard deviation of the per-seed nRMSE for seeds 0, 1, and 2. These are errors of independently trained models, not of an ensemble prediction.

**Fixed-query context interventions.** To isolate sensitivity to surrounding evidence, we generate 1,024 paired contexts per seed while keeping the queried coordinates fixed. The query is $(0.42, 0.58)$ for Weierstrass and Hardy $Z$, and $(0.16, 0.64)$ for Julia. The Julia anchor is selected for nonzero target sensitivity, without consulting model predictions.

Let $\Delta y_m$ and $\Delta \widehat{y}_m$ denote the target and predicted changes for context pair $m$. We measure

$$E_{\text{context}} = \sqrt{\frac{\sum_{m=1}^{M} (\Delta \widehat{y}_m - \Delta y_m)^2}{\sum_{m=1}^{M} (\Delta y_m)^2}}, \qquad M = 1{,}024. \tag{43}$$

In Weierstrass/Julia/Hardy order, AdaCompose obtains $0.351/0.200/0.391$, compared with $0.787/0.413/0.887$ for DeepONet. The pointwise models obtain 1: their inputs remain unchanged, so they cannot respond to the context intervention.

Figure 2 provides a separate visual diagnostic. The left panel varies the query over a $161 \times 161$ grid while holding other set members fixed; the right panel uses a $101 \times 101$ query-context grid. Set statistics are recomputed at each grid point. Predictions are averaged across the three trained seeds, without smoothing.

**Permutation and cardinality checks.** For permutation matrices $\Pi_P$ and $\Pi_Q$, we measure

$$E_{\text{perm}} = \frac{\left\| R(\Pi_P P, \Pi_Q Q) - \Pi_P R(P, Q) \Pi_Q^\top \right\|_{\text{F}}}{\| R(P, Q) \|_{\text{F}}}. \tag{44}$$

Here, permutations act jointly on embeddings and confidences in the encoded evidence sets. Task-wise mean errors across three seeds range from $1.06 \times 10^{-7}$ to $1.24 \times 10^{-6}$, consistent with Proposition 1.

A Julia model trained at cardinality 8 obtains nRMSE $0.1848/0.1837/0.1824$ at cardinalities $4/8/16$ without retraining, using 256 fields per seed at each new cardinality. These experiments assess numerical equivariance and cardinality transfer, not extrapolation to greater recursive depth.

### F.2 Compositional zero-shot recognition

**Datasets and candidate sets.** MIT-States contains 30,338/10,420/12,995 train/validation/test images, with 115 attributes and 245 objects. C-GQA contains 26,920/7,280/5,098 images, with 413 attributes and 674 objects.

Let $\mathcal{A}$ and $\mathcal{O}$ denote the attribute and object vocabularies. For evaluation split $s$, the candidate sets are

$$\begin{aligned} \mathcal{C}_s^{\text{closed}} &= \mathcal{C}_{\text{train}} \cup \mathcal{C}_{s,\text{unseen}}, \\ \mathcal{C}^{\text{open}} &= \mathcal{A} \times \mathcal{O}. \end{aligned} \tag{45}$$

Closed-world test evaluation uses 1,662 candidates for MIT-States and 6,515 for C-GQA. The corresponding open-world vocabularies contain 28,175 and 278,362 pairs.

Following Troika-style feasibility filtering, we select a threshold from 50 evenly spaced candidates using validation AUC, with HM as the tie-breaker. The threshold is fixed before test evaluation. Evidence probabilities and states are computed over the complete candidate vocabulary; processing chunks do not introduce separate normalizations.

**Calibration metrics.** For calibration bias $b$, let $S(b)$ and $U(b)$ denote seen and unseen accuracy. We report

$$\begin{aligned} S &= \max_b S(b), \\ U &= \max_b U(b), \\ \text{HM} &= \max_b \frac{2 S(b) U(b)}{S(b) + U(b)}. \end{aligned} \tag{46}$$

AUC is the area under the seen-unseen accuracy curve, not ROC-AUC. The reported S and U can occur at different biases. The test bias sweep defines these metrics; it does not select model weights or the feasibility threshold.

The streaming evaluator preserves canonical candidate-index tie-breaking and is checked against both dense evaluation and the reference evaluator. Published baseline rows follow Troika's reporting conventions and are not newly retrained comparisons.

**Model and adaptation.** The frozen CLIP ViT-L/14 backbone uses a four-template text ensemble to construct attribute, object, and direct-composition evidence. The task model combines recursive-composition scores, direct-pair scores, and factorized marginals, with learned feature adapters and primitive embedding residuals. Encoder, state, operator, and field widths are 256, 192, 96, and 256, respectively. The rank is 32 and the gate has 256 hidden units. Adapter ranks are 64 for MIT-States and 32 for C-GQA.

Training uses staged pair-label cross-entropy. For each dataset, the same validation-selected checkpoint is used in closed- and open-world evaluation. Because the model combines several branches, full-model results do not isolate the contribution of recursive composition alone.

**Architecture efficiency.** Efficiency is measured on one RTX 4090 with batch size 1, FP16 autocast, and 6,515 C-GQA candidates. Five warm-up passes precede 30 timed repeats. We report mean latency and standard deviation in milliseconds per image.

Timing includes image encoding, candidate scoring, and answer selection. It excludes preprocessing, host-to-device transfer, and cached text processing. Parameter counts exclude frozen CLIP; DFSP counts include registered branches that are unused by some variants. Equivalent caching and vectorization are documented in the benchmark code.

Baseline task modules are instantiated without C-GQA-trained weights. AdaCompose uses its trained head with direct CLIP image features for timing, rather than the reconstructed-feature cache used for accuracy evaluation. These measurements therefore characterize architecture costs, not the end-to-end costs of the reported accuracy pipelines.

### F.3 Video anomaly detection

**Data and model selection.** The development split uses seed 4040 and contains 400 training videos and 80 validation videos: 300/60 normal and 100/20 anomalous, respectively. The test split contains 120 normal and 120 anomalous videos.

Validation video-level BCE selects the fusion configuration and a 37-epoch refit duration, with video AUC as a secondary criterion. Only the 995-parameter temporal head is then refitted on all 480 development videos. The two composition streams and the normal-reference statistic remain frozen. The latter is estimated from normal development-training videos. Frame annotations and anomaly-category labels are not used for optimization or checkpoint selection.

**Evidence and temporal prediction.** Places365 ResNet-18 provides 365 scene probabilities, VideoMAE-base provides 400 action probabilities, and YOLOS-tiny provides 80 object confidences. Scene and action outputs are normalized distributions; object scores are multi-label.

Windows contain 32 frames with stride 16. VideoMAE receives 16 sampled frames per window. Each composition stream first combines action and object evidence, then incorporates scene evidence. The temporal head has hidden width 20, one kernel-7 block, dropout 0.35, and a minimum action-reference weight of 0.225. Mean pooling over the top $22.5\%$ of window logits produces the video logit for BCE. Frame scores are the maximum over overlapping window scores, without additional smoothing.

**Fine-grained metrics.** All subsets reuse the same frame predictions. Each anomaly-category row pools all frames from test videos of that category, including their normal segments. Normal-only videos are not added to every category. Scenario rows pool normal and anomalous videos assigned to the same scenario; the overall result pools all 240 test videos.

ROC-AUC and AP are computed over pooled frames rather than averaged over per-video metrics. No category- or scenario-specific fitting, threshold selection, or calibration is applied. The current Highway and Park groups contain no positive frames, so their discrimination metrics are not reported.

**Frame alignment and comparison scope.** Our decoder-aligned evaluation contains 146,015 frames, including 33,739 positives. The released ground-truth array contains 143,920 entries, including 33,160 positives. Matching video IDs therefore does not imply matched frame-level evaluation. The current scenario mapping also differs from the published partitions.

On decoder-aligned frames, AdaCompose obtains $82.7393\%$ ROC-AUC and $55.4675\%$ AP; trapezoidal PR-AUC is $55.4556\%$. The MSAD paper names its second metric AP, whereas the released RTFM/MGFN evaluators use trapezoidal integration of the precision-recall curve. Published values are retained as reported. They provide reference comparisons across differing frame alignments and metric implementations.

**Semantic examples.** The people-falling example uses windows 48–79 and 128–159 from `People falling 42`, represented by frames 64 and 144. The fire example uses windows 0–31 and 240–271 from `Fire 22`, represented by frames 16 and 256.

Displayed anomaly scores are window means of the frozen final frame predictions. Semantic evidence comes from the corresponding extractor outputs; category and scenario labels are dataset metadata, not model predictions. The examples were selected after evaluation to illustrate model behavior and do not estimate its prevalence.

**Mechanism ablations.** Table 5 evaluates three interventions in the VAD pipeline. *Type-only operator codes* remove evidence-summary inputs to $G_\psi$ while retaining type conditioning. *Context-aware static factors* replace generated concept queries with a fixed-parameter network that receives the operator code, while retaining the code-dependent final rank gate. Thus, this variant changes the factor-generation mechanism while preserving contextual conditioning. *Without interaction feedback* removes the first-stage interaction from the composed-state update while preserving its direct contribution to the final score. Each intervention is applied to both composition streams. All variants retain the normal-action reference and temporal fusion head. The streams are retrained, followed by temporal-head training and refitting for a validation-selected duration. Results use a single seed (2026), with no across-seed uncertainty estimate.

### F.4 Multimodal question answering

**Answer-conditioned evidence.** Each source produces evidence over the valid answer choices. Let $a_k$ be the normalized CLIP embedding of choice $k$, and let $v_s$ be the normalized embedding of source $s$: question, textual hint, or image. For a present source, we compute

$$p_{s,k} = \frac{\exp(a_k^\top v_s/0.07)}{\sum_{\ell\in\mathcal{K}} \exp(a_\ell^\top v_s/0.07)}, \qquad e_{s,k} = \frac{a_k + 0.35 v_s}{\|a_k + 0.35 v_s\|_2}. \tag{47}$$

Here, $\mathcal{K}$ is the set of valid choices. A missing source uses a uniform distribution over $\mathcal{K}$ and retains $a_k$ as its representation. Padded choices are masked.

All 10,332 supplied images across the official splits are encoded from raw pixels. Ground-truth answers are used only for supervision. Lectures, solutions, and supplied captions are excluded from the input evidence.

**Composition and training.** The first composition stage combines question and textual-context evidence. Its output state conditions a second operator that incorporates visual evidence. Masked averaging over the composed-state axis of the final relation logits produces one logit per valid answer. Cross-entropy trains the head; softmax produces the displayed answer scores.

Encoder, state, operator, and field widths are 96, 64, 32, and 64, respectively. The rank and type dimension are both 8. The head contains exactly 158,020 trainable parameters.

Training uses AdamW with batch size 512, learning rate $2 \times 10^{-3}$, weight decay $3 \times 10^{-4}$, and seed 2026. The training budget and cosine-schedule horizon are 160 epochs. Early stopping uses a minimum of 40 epochs and patience 20. Maximum validation accuracy selects the checkpoint, with validation cross-entropy as the tie-breaker.

The selected checkpoint, from epoch 136, answers 2,798 of 4,241 validation questions correctly ($65.98\%$). It answers 2,818 of 4,241 test questions correctly ($66.45\%$).

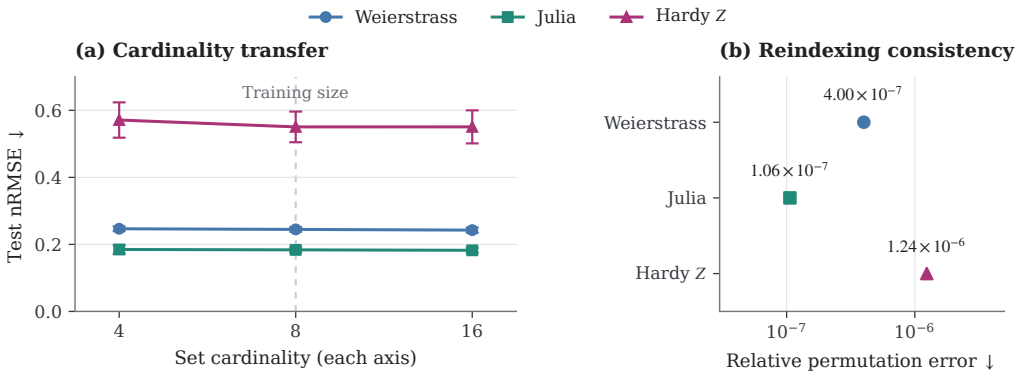



Figure 4: **Set-size transfer and permutation consistency.** (a) Test nRMSE across cardinalities; markers and error bars show means and population SDs over three seeds. The dashed line marks the training cardinality, eight; no test-time retraining. (b) Mean relative Frobenius discrepancy after aligning permuted indices (logarithmic axis); this is not prediction error.

**Baselines and reporting.** Mean and Product aggregate the same extracted probabilities without task-specific training. AdaCompose additionally receives semantic representations. This comparison evaluates the complete trained head against fixed probability fusion; it does not isolate the operator alone.

Published supervised baseline accuracies follow ScienceQA (Lu et al., 2022), with rounded trainable parameter counts from the official leaderboard. AdaCompose's parameter count excludes frozen CLIP and measures trainable adaptation, not total system size.

NAT, SOC, and LAN denote natural science, social science, and language science. IMG includes all 2,017 image-containing test questions, including those with a hint. NO contains the 1,435 questions with neither image nor hint. G1–6 and G7–12 follow the supplied grade metadata.

**Qualitative example.** Question #15290 illustrates a successful prediction under disagreement between sources. The correct option is *Tasmanian devil*; the supplied koala image illustrates the mammal concept rather than depicting an answer candidate. Question, text, and image evidence favor *loon*, *great white shark*, and *Tasmanian devil*, respectively.

Product aggregation assigns $34.21\%$ to *loon* and $29.61\%$ to the correct option. AdaCompose selects the correct option with a normalized score of $99.9928\%$. This score is not calibrated confidence. Lines connecting answer-choice scores in the figure are visual guides between discrete categories, not a continuous probability density.

### F.5 Uncertainty and Selection History

Synthetic results summarize three independently trained seeds. Downstream experiments use a single seed, 2026; we do not infer confidence intervals or statistical significance from these runs. Published accuracy rows are reference results, not matched-budget retraining experiments.

Earlier exploratory development included inspection of test results. Although final archived configurations use validation-based selection, the overall development process was not fully test-blind.

## G Supplementary Diagnostics and Sensitivity Analysis

We examine structural consistency, the contribution of different evidence scores, and sensitivity to selected training parameters. These diagnostics complement the retrained mechanism ablations: standalone branch scores describe a fixed model, whereas parameter sweeps compare separately trained configurations. Synthetic results summarize three seeds; downstream diagnostics use one seed and support comparisons within the tested settings.

Table 6: VAD full-system and ablation settings. Configuration summary, not a hyperparameter sweep.

| Setting | Base core | Larger core | Temporal head |
|---|---|---|---|
| Window / stride (frames) | 32 / 16 | 32 / 16 | / |
| Relation rank | 8 | 32 | / |
| State / factor width | 64 / 64 | 128 / 128 | / |
| Learning rate | $3 \times 10^{-4}$ | $3 \times 10^{-4}$ | $3 \times 10^{-5}$ |
| Weight decay | $10^{-4}$ | $10^{-4}$ | $10^{-2}$ |
| Videos / step | 2 | 2 | 4 |
| Historical budget (epochs) | 100 | 100 | 100 |
| Ablation budget (epochs) | 30 | 30 | 100 |
| Pre-refit cosine horizon (epochs) | 100 | 100 | 100 |
| Early stopping: min. / patience | / | / | 25 / 15 |

Table 7: Archived CZSL branch diagnostics. Test HM/AUC (%) from fixed-checkpoint scores, without branch retraining.

| Score used | MIT-States | C-GQA |
|---|---|---|
| Base attribute-object | 0.40 / 0.06 | 3.61 / 0.41 |
| Recursive score | 4.48 / 0.33 | 6.34 / 0.67 |
| Direct pair | 34.03 / 17.43 | 14.42 / 3.45 |
| Marginal product | 25.92 / 10.69 | 10.47 / 1.88 |
| Full fusion | 34.34 / 17.58 | 24.67 / 8.57 |

### G.1 Set Structure and Cardinality Transfer

**Set-size transfer and reindexing consistency.** Figure 4(a) shows that predictive accuracy largely transfers beyond the training cardinality. Models trained on $8 \times 8$ fields are evaluated without retraining at cardinalities 4, 8, and 16, using 1,536 fields per seed at size 8 and 256 at each other size. Across these cardinalities, nRMSE varies by only 0.0024 for Julia and 0.0040 for Weierstrass. Hardy $Z$ is more sensitive to smaller sets: its error increases from 0.5506 at size 8 to 0.5711 at size 4. Transfer therefore depends on the target function, even when the same operator accepts different set sizes.

Figure 4(b) tests a separate property: consistency under reindexing. Relative permutation errors range from $1.06 \times 10^{-7}$ to $1.24 \times 10^{-6}$, consistent with numerical equivariance. Together, the panels distinguish empirical accuracy across set sizes from structural consistency under permutation; neither evaluates extrapolation to greater recursive depth.

### G.2 Task-Specific Evidence Diagnostics

**Dataset-dependent reliance on score fusion.** Table 7 shows markedly different fusion benefits across datasets. On MIT-States, full fusion closely tracks the direct-pair branch, improving HM/AUC by only 0.31/0.15 points. On C-GQA, the gains are 10.25/5.12 points. The test-average direct-pair mixture weights follow the same pattern: $98.08\%$ on MIT-States versus $51.28\%$ on C-GQA. Thus, the archived MIT-States model is predominantly direct-pair driven, while C-GQA benefits more from combining scores.

These results use the MIT-States round-8 and C-GQA round-5 checkpoints with the archived Troika-style finite bias sweep, separately from the main paired-world evaluation. Branches are scored without retraining. Because full fusion includes both recursive and marginal terms, its gains do not isolate recursion; likewise, low standalone recursive accuracy does not establish that the branch is unnecessary within the fused model.

**Learned fusion improves beyond fixed aggregation.** Figure 5(c) shows almost no difference between Mean and Product overall: $48.27\%$ versus $48.29\%$. AdaCompose reaches $66.45\%$, improving over Product by 18.16 percentage points. On image-containing questions (Figure 5(a)), it exceeds

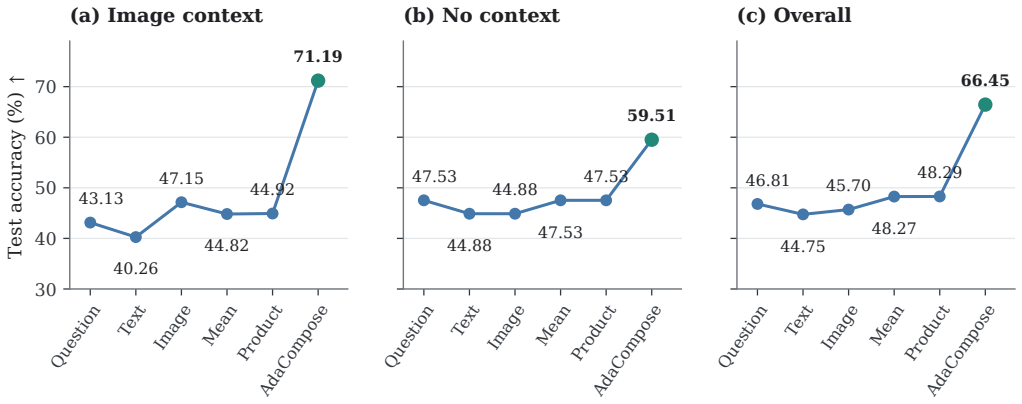



Figure 5: **ScienceQA evidence and aggregation diagnostics.** Test accuracy (%, ↑) for image-containing questions, questions with no context, and the full test split; IMG and NO are not complementary. All methods share extracted probabilities; AdaCompose additionally uses semantic features and task-specific training. Lines connect categorical prediction rules, not a continuous parameter sweep.

the strongest individual source, Image, by 24.04 points (71.19% versus 47.15%). The gain also persists without contextual inputs: Figure 5(b) shows an 11.98-point improvement over both fixed aggregation rules. The benefit is therefore not confined to image-containing questions.

Individual-source predictions use the archived uniform fallback for absent modalities. AdaCompose uses a 158,020-parameter head trained with learning rate 0.002 and weight decay 0.0003, selected at epoch 136 by validation. Unlike Mean and Product, it also receives semantic representations. The comparison supports the complete trained semantic head over these fixed probability rules, while leaving the effects of semantic inputs, training, and dynamic parameterization coupled.

### G.3 Configurations and Validation Sensitivity

**Configuration separates temporal resolution from capacity.** Table 6 summarizes model settings, rather than controlled sensitivity experiments. Both cores use 32-frame windows with stride 16, but differ jointly in relation rank and representation width. Their coexistence therefore does not isolate a rank effect. VideoMAE samples 16 frames per window, while the temporal head applies a kernel-7 block across window representations; these specify different stages of temporal processing.

Both cores use log-mean-exp pooling with temperature 0.5. The head uses width 20, dropout 0.35, top-22.5% mean pooling, and normal-reference weight floor 0.225. Minimum validation video BCE selects core epochs 23/6 and head epoch 37 for the full system. The head is then refitted on all 480 development videos, with both duration and cosine horizon set to 37 epochs. Each ablation selects its own refit duration by the same rule. The 30-epoch core ablations retain the original 100-epoch learning-rate schedule. Slashes in the table denote inapplicable or unused settings.

**Pooling favors different settings under BCE and AUC.** Figures 6(a,b) show that increasing the pooling ratio improves discrimination without monotonically improving the selection loss. Raising the ratio from 0.175 to 0.25 reduces validation BCE from 0.177588 to 0.176362. At 0.275, AUC reaches its highest tested value, 97.42%, but BCE rises to 0.177113. Thus, the best tested ratio is 0.25 under BCE and 0.275 under AUC: improved ranking need not imply lower probabilistic prediction loss.

**Regularization has smaller local effects.** At pooling ratio 0.225, Figure 6(c) shows that increasing dropout from 0.25 to 0.45 lowers BCE by 0.000094. Figure 6(d) shows that increasing weight decay from 0.003 to 0.03 raises BCE by 0.000040. Both changes are smaller than the observed variation across the pooling sweep, although they are measured at a different pooling ratio.

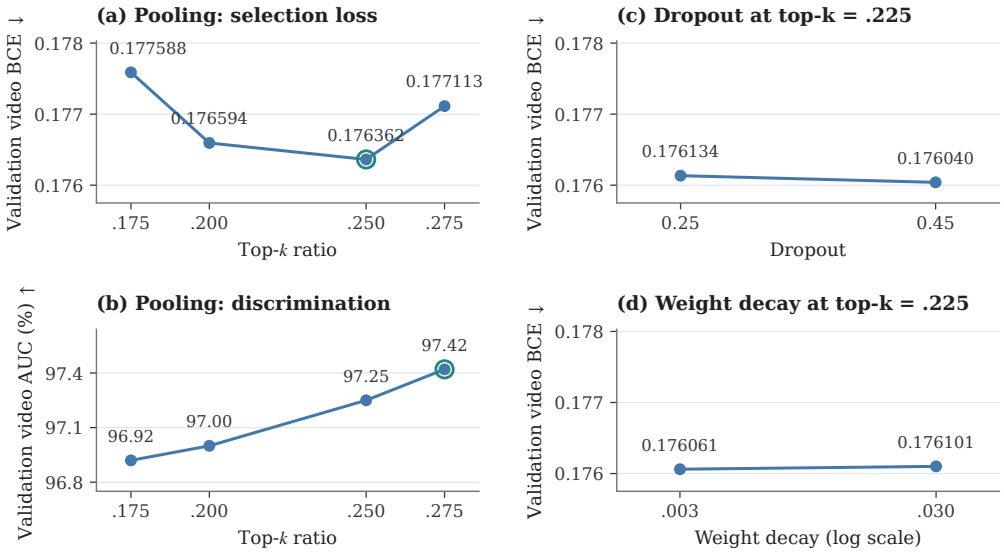



Figure 6: **VAD temporal-head validation sensitivity.** (a,b) Pooling-ratio changes at dropout 0.35 and WD 0.01. (c) Dropout changes at ratio 0.225 and WD 0.01. (d) WD changes at ratio 0.225 and dropout 0.35. BCE panels share a scale; open rings mark the best observed pooling configuration for each metric. Single seed; lines connect tested settings only. The central $0.225/0.35/0.01$ configuration was not evaluated under this budget.

This historical sweep freezes both cores and the normal reference. The head uses width 20, one kernel-7 block, learning rate $3 \times 10^{-5}$, four videos per step, and reference-weight floor 0.225. With seed 2026, minimum validation video BCE selects checkpoints under a 120-epoch cosine horizon and early-stopping minimum/patience 30/20. This differs from the final 100-epoch, 25/15 recipe. The joint setting $(0.225, 0.35, 0.01)$ was not evaluated under the sweep protocol, so these results characterize local head sensitivity rather than establish the final configuration as optimal or demonstrate improved frame localization.

**Learning rate improves accuracy with a loss trade-off.** Figure 7(a) shows consistent accuracy gains when the learning rate increases from 0.001 to 0.002 at rank 8. Accuracy rises by 3.09 points at weight decay 0.0003 (62.89% to 65.98%) and by 2.47 points at weight decay 0.003 (62.56% to 65.03%). However, Figure 7(b) shows that cross-entropy also increases, from approximately 1.96 to 4.165 and 5.425, respectively. The higher-accuracy checkpoints therefore incur greater average probability loss; their accuracy gains do not establish improved calibration.

**Higher rank does not improve the selection metric.** At matched learning rate 0.0015 and weight decay 0.001, Figure 7(c) shows that rank 8 outperforms rank 16 in accuracy (65.93% versus 65.48%). The ordering reverses for cross-entropy: rank 16 achieves 2.927, compared with 4.183 for rank 8 (Figure 7(d)). Additional capacity therefore changes the accuracy-loss trade-off rather than improving both metrics. Among the displayed configurations, rank 8 with learning rate 0.002 and weight decay 0.0003 has the highest validation accuracy, supporting its selection within the tested settings.

All configurations use frozen CLIP ViT-B/32 raw-image evidence, seed 2026, batch size 512, a 160-epoch cosine horizon, and early-stopping minimum/patience 40/20. Maximum validation accuracy selects checkpoints, with cross-entropy as the tie-breaker. Both plotted metrics are evaluated at the same selected checkpoint.

## AI USE STATEMENT

In this work, we used generative AI tools to provide feedback on methodological and mathematical exposition and assist with interpreting reported results. We did not use these tools to originate the

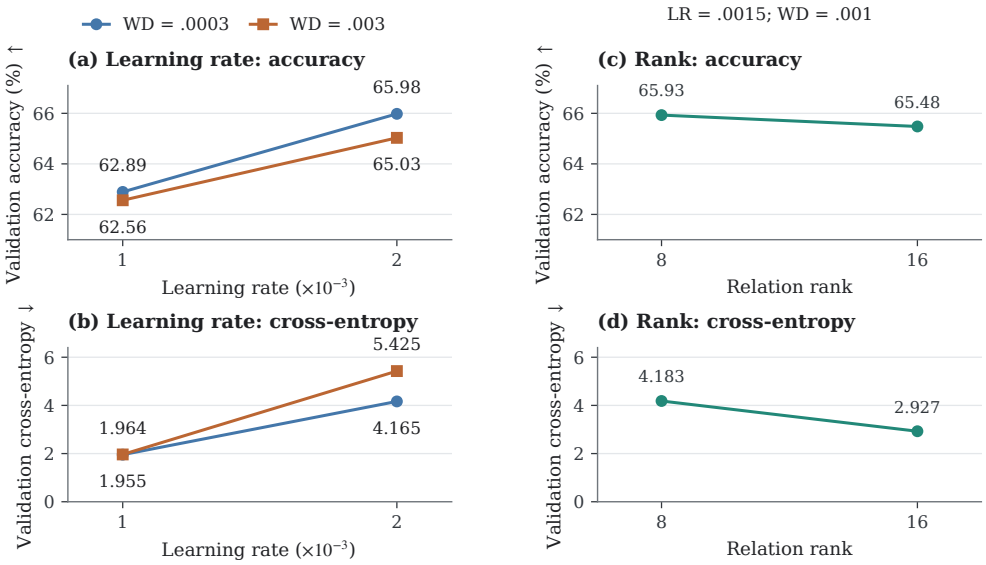



Figure 7: **ScienceQA optimization and rank sensitivity.** (a,b) Learning-rate comparisons at rank eight for two weight-decay values. (c,d) Rank comparison at LR 0.0015 and WD 0.001. Accuracy and cross-entropy are reported at the same validation-selected checkpoint. Lines connect tested configurations; single seed, without across-seed uncertainty estimates.

core research ideas or design the model architecture and experiments. Additionally, we used AI for grammar checking, language polishing, literature discovery, LaTeX formatting, and programming assistance. The authors designed the code structure and manually implemented the core algorithm; the coding assistant supported selected programming tasks. The authors reviewed AI-assisted revisions and code suggestions for consistency with the intended methodology and checked factual and numerical statements against the underlying materials. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

## Ethics Statement

This work uses synthetic data and publicly released research benchmarks: MIT-States, C-GQA, MSAD, and ScienceQA. The research ideas, methodology, and experimental design were developed by the authors. Public availability does not eliminate privacy concerns or biases inherited from datasets and pretrained models. Our evaluations do not establish demographic fairness or deployment safety. In particular, anomaly scores should not be interpreted as evidence of intent or wrongdoing, and displayed semantic evidence consists of model outputs rather than verified explanations. Practical deployment would require appropriate privacy safeguards, human oversight, and assessment of false alarms and missed events.

## Reproducibility Statement

We provide methodological and experimental documentation to support independent reproduction. Section 2 defines the composition cell and its structural properties. Appendix C presents formal statements and proofs, with their scope clarified in Appendix E. Appendix D describes recursive composition and task-specific realizations. Appendix F documents dataset splits, evidence construction, training settings, model selection, and evaluation procedures across the four tasks. Appendix F.5 reports uncertainty and selection history, while Appendix G provides additional diagnostics, configurations, and sensitivity analyses.