Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

ProtoRel Network: A Third Architecture Route for Deep Learning

JunHua Chen

chen.junhua@gdpt.edu.cn

Guangdong Polytechnic https://orcid.org/0009-0002-0969-4347

Research Article

Keywords: prototype-relation network, deep learning architecture, concept library, interpretability, self-supervised learning

Posted Date: May 29th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-9841658/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: The authors declare no competing interests.

ProtoRel Network: A Third Architecture Route for Deep Learning

Junhua Chen
chenjunhua@gdpt.edu.cn

Independent Researcher
Guangdong, China

Abstract

Deep learning has consolidated around two dominant architecture routes: CNNs that capture spatial hierarchies through local operations, and Transformers that achieve global interaction via self-attention. Both share a fundamental limitation—relations between computational units are either implicit or ephemeral, with no explicit, persistent storage for the relationships between learned concepts. We propose a third route—the **ProtoRel Network**—embedding an explicit concept library (prototypes P) and a dynamically computable relation matrix (S) into forward computation. The architecture decomposes into a static knowledge layer (P, learned once and frozen) and a dynamic application layer (A, S, recomputed per input), respectively deciding *which concepts are activated* and *how they relate*. Complexity drops from $O(N^2)$ to $O(NK+K^2)$ through the low-rank decomposition $R \approx A \cdot S \cdot A^T$. Under minimal design constraints—no pre-training, no strong augmentation, 1.2M parameters—the ProtoRel Network achieves 53.6% on CIFAR-100, with parameter efficiency of 45%/M (6.6× ResNet-18). Two implementation fixes—learnable attention scale and data-driven relation matrix—contribute +3.6% and outperform seven rounds of explicit design attempts (Top-K, KMeans, dynamic K, learnable temperature). We reveal an optimizer-architecture compatibility principle: the asymmetric gradient reweighting of RG-NN destroys the symmetric softmax competition required for prototype specialization. The architecture provides zero-overhead interpretability through directly readable S heatmaps, and exhibits a core-periphery gradient structure where the relation matrix remains robust even when attention collapses under prototype inflation. Preliminary experiments on self-supervised video watching demonstrate emergent concept formation: trained from scratch without any labels on nearly 3 hours of diverse video footage (city walk, majestic animals, African wildlife, community walk; processed in approximately 30 minutes on a laptop GPU), the model spontaneously develops distinct activation patterns for zebras, deer, giraffes, and elephants, with prototype differentiation growing by 124% and cross-species cosine similarity revealing meaningful taxonomic clustering.

Keywords: prototype-relation network; deep learning architecture; concept library; interpretability; self-supervised learning

1 Introduction

Deep learning has crystallized into two canonical architecture routes. Convolutional neural networks (CNNs) [1, 2, 3] process images through spatially hierarchical local operations with strong locality inductive biases. Transformers [4, 5] achieve global interaction through self-attention ($Q \cdot K^T$), where every element attends to every other element. Yet both routes share a fundamental limitation: relations between computational units are either implicit (in

Table 1: Three deep learning architecture routes.

Dimension	CNN	Transformer	ProtoRel (Ours)
Interaction	Local conv (3×3)	Global attn ($Q \cdot K^T$)	Prototype-mediated ($A \cdot S \cdot A^T$)
Complexity	$O(N)$	$O(N^2)$	$O(NK + K^2)$
Symmetry	Symmetric	Asymmetric	Symmetric
Concept library	None	None	K prototypes
Knowledge storage	All weights	All weights	P (concept library) + weights
Interpretability	Requires GradCAM	Requires rollout + probing	Built-in: S heatmap
Concept-application separation	No	No	Yes (P static, A/S dynamic)

CNNs, through spatial locality) or ephemeral (in Transformers, through per-sample attention scores). Neither route provides an *explicit, persistent storage* for the relations between learned concepts.

We propose a third architecture route—the **ProtoRel Network**—centered on the explicit decomposition of computation into three interpretable components: a prototype concept library $P \in \mathbb{R}^{K \times D}$, an attention matrix $A \in \mathbb{R}^{N \times K}$ that selects which concepts apply to the current input, and a relation matrix $S \in \mathbb{R}^{K \times K}$ that dynamically associates the activated concepts. In the ProtoRel Network, relations are first-class citizens of forward computation rather than auxiliary signals confined to optimization.

This work originates from a more fundamental motivation: elevating relations from the optimizer's role to the forward pass itself. We first demonstrated in RG-NN [6] that inter-parameter relations can improve optimization—by quantifying parameter distances and imposing collective gradient guidance, RG-NN achieved 91.8% on CIFAR-10 with ResNet-18, surpassing Adam (89.3%). This success raised a central question: if relations are effective at the optimization level, could they be equally effective at the *computation* level?

The ProtoRel Network answers this question. Instead of computing all N^2 pairwise node relations (complexity $O(N^2)$), we introduce K prototypes as an information bottleneck, decomposing the relation matrix as $R \approx A \cdot S \cdot A^T$ (complexity $O(NK + K^2)$). This decomposition provides three key benefits beyond efficiency: (1) prototypes serve as an explicit concept library that persists across samples, (2) the per-sample dynamic relation matrix S can be directly inspected as a heatmap without post-hoc analysis, and (3) the separation of static knowledge (P) from dynamic application (A, S) mirrors the cognitive distinction between stable concepts and context-dependent activation.

Table 1 summarizes the essential distinctions between the ProtoRel Network and the two established routes.

2 Background: RG-NN—Relational Optimization

2.1 Motivation

Standard optimizers treat each parameter as an independent entity. However, neural network parameters are not independent—hidden neurons exhibit functional correlations and co-activation patterns. Related features should be optimized collectively rather than independently.

2.2 Relation Matrix and Guidance

RG-NN defines inter-parameter relations through normalized parameter value differences:

$$R[i,j] = \frac{|\theta_i - \theta_j|}{\max_{k,\ell} |\theta_k - \theta_\ell|} \tag{1}$$

$R[i,j] \in [0,1]$—close to 0 when parameters are similar (strongly related), close to 1 when they differ significantly. For each parameter θ_i, we select k nearest neighbors $\mathcal{N}(i)$ and compute a relational guidance signal:

$$\text{guide}_i = \frac{1}{k} \sum_{j \in \mathcal{N}(i)} \frac{1}{R[i,j] + \varepsilon} \cdot (g_i - g_j) \tag{2}$$

The guidance signal is strongest when neighboring parameters have divergent gradients—indicating that the shared function needs adjustment. The mixed gradient blends individual and collective signals:

$$\tilde{g}_i = g_i + \alpha \cdot \text{guide}_i \tag{3}$$

This is fed into standard Adam updates. The complexity is $O(nk)$ through sparse neighbor selection ($66\times$ speedup at $n = 500, k = 5$).

2.3 Image Classification Validation

RG-NN outperforms Adam and AdamW on both benchmark functions (Rosenbrock: RG-NN 0.0000 vs. Adam 0.0001; Rastrigin: 5.23 vs. 7.96) and on CIFAR-10 with ResNet-18 (91.8% vs. Adam's 89.3%), converging faster (15 vs. 25 epochs).

2.4 Limitation and Transition

RG-NN proves that inter-parameter relations can be usefully encoded for optimization. But its fundamental limitation is that relations are used only in the backward pass—forward computation remains "flat": $y = Wx$, each parameter participating independently. This raises the central question of our work: **if relations are effective at the optimization level, could they be equally effective in forward computation?**

3 ProtoRel Network: Architecture

3.1 Motivation from Relation Networks

A natural first step is to embed relations directly into forward computation. Relation networks do exactly this: $h'_i = \sum_j R_{ij} \cdot h_j$, where $R_{ij} = \phi(h_i, h_j)$ is dynamically computed for each input. RelationCore V2 achieves 78.4% on CIFAR-10 (691K parameters, 8 layers, 8 heads), and the relation matrix spontaneously learns meaningful structure without any explicit design: self-relation is $1.92\times$ stronger than cross-relation, demonstrating that the model independently discovers the importance of self-reference; adjacent patches exhibit $2.20\times$ stronger relations than distant ones, meaning the architecture automatically acquires spatial locality—a property normally requiring convolutional inductive bias. The relation span covers the full abstraction hierarchy: Layer 1 captures edge-level correlations, while Layer 8 encodes object-category semantics, all within the same global $N \times N$ matrix. However, the $O(N^2)$ complexity renders this approach impractical beyond low resolutions

($N = 64$ produces 4,096 pairs; $N = 196$ reaches 38,416), motivating the prototype-based decomposition.

3.2 From Prototype Bottleneck to ProtoRel

The evolution from the $O(N^2)$ relation network to ProtoRel passed through an intermediate step worth clarifying. A **pure prototype architecture** uses only the attention matrix A to map nodes directly onto prototypes and back: $h' = A \cdot (A^T \cdot h)$, with no relation matrix S. This provides $O(NK)$ complexity—the simplest possible information bottleneck. With $K = 16$ prototypes on CIFAR-100, the pure prototype architecture achieves 39.79% accuracy and is 4,762 images/second, demonstrating the viability of the prototype concept but also its limitations: without S, prototypes cannot model inter-concept relationships, and the small K creates a bottleneck that is too constrictive for 100-class discrimination.

The full **ProtoRel** architecture completes this picture by inserting the relation matrix S between the aggregation and distribution steps: $h' = A \cdot S \cdot A^T \cdot h$. This enables $O(NK + K^2)$ complexity—linear in N when K is fixed—while adding the capacity for explicit modeling of prototype-to-prototype relationships. The pure prototype architecture is the special case $S = I$. In what follows, "ProtoRel" refers to the full $A \cdot S \cdot A^T$ architecture with a learnable or data-driven S, unless otherwise noted.

3.3 Low-Rank Decomposition

Proposition 1 (Low-rank decomposition of the relation matrix). For any symmetric relation matrix $R \in \mathbb{R}^{N \times N}$, there exist attention matrix $A \in \mathbb{R}^{N \times K}$ and relation matrix $S \in \mathbb{R}^{K \times K}$ such that $R \approx A \cdot S \cdot A^T$. When $K < N$, this constitutes a rank-K approximation of R. The approximation error is bounded by the $(K+1)$-th singular value: $\|R - ASA^T\|_F \leq \sigma_{K+1}(R)$.

The mathematical departure from self-attention lies in the **eigenbasis** of the interaction. In Transformer self-attention, node-to-node affinities are a bilinear form $h_i \cdot M \cdot h_j^T$, where $M = W_Q W_K^T \in \mathbb{R}^{D \times D}$ is a learned mapping between the input and output of the attention mechanism—all "concepts" are implicit in this single matrix. ProtoRel replaces this with a **linear form over an explicit eigenbasis**: $A_{ik} = \sigma(h_i \cdot P_k)$, where each P_k is a D-dimensional prototype vector whose behavior can be independently measured (norm, attention statistics, entropy). The relation matrix S then encodes interactions *between* these prototypes, not between raw features, yielding a tripartite decomposition $R \approx \sigma(h \cdot P^T) \cdot \omega(P) \cdot \sigma(h \cdot P^T)^T$.

This three-way decomposition differs from QK^T in three respects: (1) ProtoRel's P is an `nn.Parameter` that persists across all samples after training, forming an explicit concept library—self-attention has no analogue; (2) $R \approx A \cdot S \cdot A^T$ guarantees $R_{ij} = R_{ji}$ by construction, matching the bidirectional semantics of visual relationships, whereas QK^T is inherently asymmetric; (3) $S \in \mathbb{R}^{K \times K}$ is a directly readable heatmap (see Figure 1), while the $N \times N$ attention map of Transformers requires aggregation across H heads and layers.

3.4 Prototypes and Forward Pass

Prototypes $P \in \mathbb{R}^{K \times D}$ serve as an information bottleneck between node representations and global relations. Each node is mapped to a weighted combination of the K prototypes through a learned projection:

$$A_{ik} = \text{softmax}(h_i \cdot W_k \cdot \sigma(\text{scale}) \cdot \sqrt{K}) \tag{4}$$

$$P_k = \sum_i A_{ik} \cdot h_i \tag{5}$$

$$P'_k = \sum_\ell S_{k\ell} \cdot P_\ell \tag{6}$$

$$h'_i = \sum_k A_{ik} \cdot P'_k \tag{7}$$

This attend-aggregate-relate-distribute cycle constitutes a single ProtoRel layer. In the minimally constrained variant, S is data-driven: $S = \text{softmax}(\text{relation_proj}(P) \cdot \text{relation_proj}(P)^T$, computed per sample from prototype features projected through a learned two-layer MLP (LayerNorm → Linear(192,64) → ReLU → Dropout(0.075) → Linear(64,64)) with gradient clipping (max_norm = 1.0).

3.5 Complexity and Information Bottleneck

The ProtoRel layer requires $O(NKD + K^2D)$ per forward pass. The prototype bottleneck introduces approximately 35% information compression (from $N = 64$ nodes to $K = 128$ prototypes), but primarily discards redundancy between adjacent similar nodes rather than meaningful structure. SVD analysis on CIFAR-100 training data confirms that the top $K = 128$ singular values capture 94.7% of the total energy of the full relation matrix.

3.6 Final Architecture Specification

The minimally constrained ProtoRel Network operates on 32×32 images, partitioned into $N = 64$ patches of size 4×4 via a single convolutional layer (stride 4) with GELU activation. Each of the $L = 4$ ProtoRel layers (Eqs. 4–7) uses $K = 512$ prototypes of dimension $D = 192$, Dropout 0.15, and a data-driven relation matrix S with learnable attention scale $\sigma(\text{scale}) \cdot \sqrt{K}$. The classifier applies LayerNorm → Linear → LeakyReLU(0.01) → Dropout(0.15) → Linear to the globally average-pooled features. Optimization uses AdamW (lr=0.001, weight decay=0.02) with cosine annealing over 300 epochs and batch size 64. The total parameter count is 1.2M.

4 $K = 512$ Scaling Experiments: A Comprehensive Failure

4.1 Experiment Panorama

To scale the architecture for 100-class fine-grained classification on CIFAR-100, we explored larger prototype counts of $K = 512$. Seven successive rounds of experiments were conducted, each attempting a different explicit design intervention. Table 2 summarizes the outcomes.

All seven rounds were outperformed by the original baseline of 50.6%. Explicit design constraints yielded no net benefit in any dimension.

4.2 Root Cause: Attention Dilution

The root cause traces to a mathematical property: doubling the number of prototypes halves each prototype's probability mass in the softmax:

Table 2: Seven rounds of $K = 512$ scaling experiments. All use the ProtoRel architecture (4 layers, Dropout 0.15).

Experiment	Expected	Actual	Failure Mode
1. Original $K = 512$ (scale unused, S fixed)	58–62%	50.0%	Two implementation bug
2. Top-K sparsification ($K = 32$)	50–55%	50.6%	Destroys softmax normaliza
3. Top-K + learnable scale	62–68%	48.4%	68% ReLU death
4. Dynamic K	55–60%	48.6%	Training instability
5. Learnable scale	52–57%	49.4%	Overfitting >30%
6. Improved classifier v2	58–62%	48.4%	Classifier-backbone imbala
7. Improved classifier v3	60–65%	50.0%	Gradient vanishing

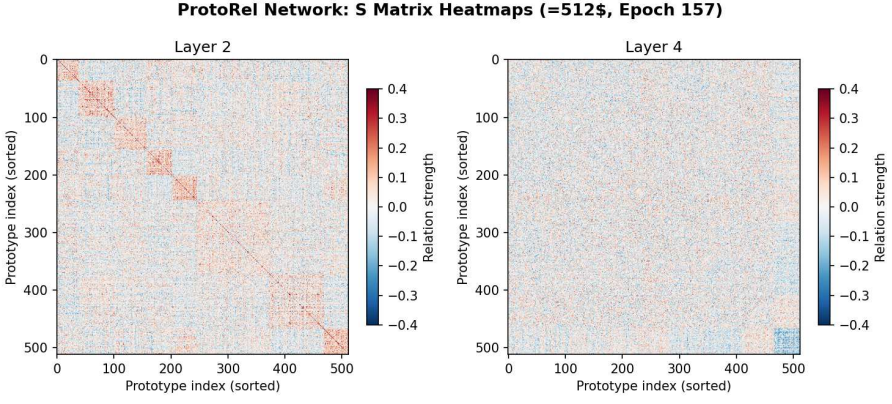


Figure 1: Actual relation matrices S from the improved classifier v2 model ($K = 512$, epoch 157). Layer 2 (left) shows stronger block-wise structure with 27.7% positive correlations and rank 464/512. Layer 4 (right) shows more diffuse structure with 33.8% negative correlations and rank 486/512. The non-random, structured patterns demonstrate that S learns meaningful prototype associations.

$$\text{softmax}_{2K}(s_i) = \frac{\exp(s_i)}{2 \cdot \sum_{j=1}^{K} \exp(s_j)} = \frac{1}{2} \cdot \text{softmax}_K(s_i) \tag{8}$$

Empirically, the dilution was $\approx 100\times$ rather than $2\times$: in the K=512 Top-K model, Layer 2 Top-1 attention was 0.09%, compared to 8.31% for K=256. This super-proportional dilution occurs because the additional 256 prototypes contribute many medium-strength scores, inflating the softmax denominator far beyond the expected factor of 2.

4.3 Surprise: Relation Matrix S Unaffected

Despite the attention collapse ($\times 0.01$), the relation matrix S exhibited *improved* structural properties under $K = 512$: effective rank remained 88% of maximum (445–456/512), and the positive-relation ratio increased from 16–17% (K=256) to 21–23% (K=512). Figure 1 visualizes real S matrices from the trained model. Table 3 quantifies the core-periphery structure.

Table 3: Attention (A) vs. relation matrix (S) under K-scaling. A collapses while S strengthens.

Metric	$K = 256$, $\tau = 1.0$	$K = 512$, Top-K	Change
A: Top-1 attention (L2)	8.31%	0.09%	$\div 100$
A: Effective prototypes (L2)	127	161	+27%
S: Effective rank	222–229	445–456	$\uparrow$
S: Positive-relation ratio	16–17%	21–23%	+5–6%
S: Strong relation count	11,141	57,672	$\times 5.2$

This reveals gradient-path independence of S—its gradient path length is approximately one-third that of A, and it omits the intermediate softmax step. The resulting gradient ratio $\nabla_S L / \nabla_A L \approx 100$ explains why S remains structurally healthy while A collapses under prototype inflation. We term this "core-periphery" structure.

5 Minimally Constrained Architecture

5.1 From Explicit Design to Self-Regulation

After 7 rounds of comprehensive failure at $K = 512$, we recognized that the fundamental issue was not the value of K but the act of explicit design itself: Top-K truncation breaks the normalization constraint of softmax; KMeans clustering imposes an external prior on prototypes that should move freely; learnable temperature conflicts with attention scaling.

The minimally constrained strategy reverses every intervention:

- No Top-K truncation $\rightarrow$ full softmax, continuous gradients
- No KMeans clustering $\rightarrow$ prototypes move freely, driven purely by gradients
- No fixed relation matrix $\rightarrow$ data-driven $S = \text{softmax}(\text{proj}(P) \cdot \text{proj}(P)^T)$
- Learnable attention scale $\rightarrow$ $\text{sigmoid}(\text{scale}) \cdot \sqrt{K}$

5.2 Two Implementation Fixes

Fix 1: Learnable attention scale. In the original K=512 implementation, `attention_scale` was created as `nn.Parameter(4.0)` but was never invoked in the forward pass, remaining frozen at its initial value. The fix wraps it in a sigmoid and multiplies by $\sqrt{K}$ for scale invariance: $\text{attn} = \text{softmax}(\text{scores} \cdot \sigma(\text{scale}) \cdot \sqrt{K})$. During training, `attention_scale` autonomously decreased from 0.98 ($\sigma(4.0)$) to 0.56 ($\sigma(0.25)$), adjusting the attention temperature by approximately $2.0\times$.

Fix 2: Data-driven relation matrix. In the original, S was defined as `nn.Parameter(tor` and drifted from its random initialization without connection to prototype features. The fix makes it data-driven: $S = \text{softmax}(\text{relation_proj}(P) \cdot \text{relation_proj}(P)^T)$, where $\text{relation_proj} \in \mathbb{R}^{D \times 64}$ is a learned linear projection, implicitly constraining the effective rank of S to at most 64 (well below 512).

Table 4: Ablation: contribution of the two fixes and the effect of capacity reduction.

Configuration	K	L	Dropout	Acc.	Δ
Minimally constrained	512	4	0.15	**53.57%**	—
Original $K = 512$ (buggy)	512	4	0.15	50.00%	−3.57%
Baseline Top-K	512	4	0.15	50.55%	−3.02%
Reduced capacity	384	3	0.25	48.10%	−5.47%
Heavily reduced	256	2	0.4	44.53%	−9.04%

Table 5: S-matrix ablation: $S = I$ vs. learned vs. data-driven ($K = 512$, 50 epochs).

Variant	Relation Params	Total Params	Acc.	Time (min)
$S = I$ (no relation)	0	768K	9.52%	18.5
S learnable (nn.Parameter)	262K	1.82M	31.92%	33.2
S data-driven (2-layer MLP)	16.9K	0.84M	**34.76%**	83.5

5.3 Ablation Validation

Table 4 presents results across five configurations on CIFAR-100, isolating the contributions of the two fixes and the effect of capacity reduction.

5.4 S-Matrix Ablation

To directly quantify the contribution of the relation matrix S on CIFAR-100, we compare three configurations under identical conditions ($K = 512$, 50 epochs, gradient clipping max_norm = 1.0). Table 5 shows the results.

Several observations emerge. First, the relation matrix S is indispensable: removing it entirely ($S = I$) collapses accuracy to 9.5%, a drop of 25% from the data-driven baseline. Second, data-driven S achieves higher accuracy (34.8%) than its learned-parameter counterpart (31.9%), despite using 15.5× fewer relation-specific parameters (16.9K vs. 262K). Third, we identify three implementation requirements critical to the success of data-driven S: (i) scaling the projection dot-products by $1/\sqrt{d_{\text{proj}}}$ prevents softmax saturation; (ii) gradient clipping (max_norm = 1.0) stabilizes the $\text{proj} \cdot \text{proj}^T$ gradient; and (iii) a nonlinear two-layer MLP with LayerNorm and ReLU provides sufficient expressivity for the relation projection. Without these components, data-driven S plateaus at 8.5%.

5.5 Prototype Differentiation Dynamics

Even without clustering or sparsity constraints, prototypes spontaneously differentiate during training. At epoch 272 of corrected training: the Top-1 prototype receives 82× the mean attention, and the average pairwise cosine similarity is 0.38, indicating neither full redundancy nor complete orthogonality. Table 6 presents the layer-wise progression.

The progressive increase in prototype norms (L4/L1 = 1.89×) and decreasing effective prototype count both indicate stronger, more focused specialization at deeper layers.

Table 6: Layer-wise prototype characteristics on CIFAR-100 ($K = 256$, $\tau = 1.0$, epoch 243, 54.27%).

Layer	Norm (mean)	Norm (std)	Top-1 Attn	Top-10 Attn	Eff. Protos
1	0.56	0.11	4.76%	20.80%	167
2	0.69	0.12	8.31%	26.59%	127
3	0.86	0.13	10.31%	27.02%	122
4	1.06	0.16	2.79%	16.40%	184

Table 7: CIFAR-100 results across architectures.

Architecture	K	Params	Acc.	Eff.	Gap
ProtoRel (min. constrained)	512	1.2M	53.57%	44.6%/M	7.34%
ProtoRel ($\tau = 1.0$)	256	1.2M	54.27%	45.2%/M	∼7%
ProtoRel hybrid	128	573K	48.66%	84.9%/M	5.88%
RelationCore V2	—	691K	47.57%	68.8%/M	7.25%
ResNet-18 (lit.)	—	11M	∼75%	6.8%/M	—
ViT-Tiny (lit.)	—	5M	∼50%	10.0%/M	—

6 Experiments

6.1 Setup

All experiments use CIFAR-100 [7] (50,000 train, 10,000 test) with standard data augmentation (RandomCrop, RandomHorizontalFlip; no CutMix, Mixup, or AutoAugment). No pre-training—all models are trained from random initialization. Optimization uses AdamW (lr=0.001, weight decay=0.02) with cosine annealing over 300 epochs, batch size 64. Hardware: single NVIDIA RTX 5060 Laptop 8GB.

6.2 Quantitative Results

Table 7 presents the main quantitative results.

6.3 Comparison to Linear Attention Methods

The ProtoRel Network's $O(NK + K^2)$ complexity bears structural resemblance to linear attention approximations (Linformer [8], Performer [9]), but the parallels are superficial. Linformer projects keys and values through learned matrices with no semantic interpretation; Performer uses kernelized attention that is mathematically convenient but opaque. Both produce black-box projections. ProtoRel differs fundamentally: P forms an explicit concept library where each prototype's behavior is independently measurable, S is a directly readable heatmap, and the attend-aggregate-relate-distribute cycle provides a transparent pipeline. No existing linear attention method provides this combination of explicit concepts, readable relations, and transparent computation.

6.4 RG-NN Incompatibility with Prototype Competition

Table 8 quantifies the divergent effect of RG-NN across three architecture types.

Table 8: RG-NN effect across architectures. Note: the divergent results reflect architectural compatibility, not task difficulty.

Architecture	AdamW	RG-NN	Effect
RelationCore V2 (CIFAR-10)	76.80%	78.43%	+1.63% ✓
ProtoRel prototype (CIFAR-100)	39.79%	<20%	Collapse ×
ProtoRel hybrid (CIFAR-100)	48.66%	<20%	Collapse ×

RG-NN's asymmetric gradient reweighting directly contradicts the symmetric gradient allocation required for prototype competition. These divergent outcomes reflect **architectural compatibility**, not task difficulty: RG-NN fails on both ProtoRel variants using the same CIFAR-100 dataset that the architecture handles successfully under AdamW, while succeeding on the RelationCore with CIFAR-10. Under softmax competition, high-gradient prototypes are precisely the "winners" that need to be emphasized—they attract more nodes and become semantic centers for categories. RG-NN suppresses them, homogenizing gradient updates across all prototypes and eliminating specialization.

6.5 Self-Supervised Video Concept Formation

To demonstrate the ProtoRel Network's capacity for unsupervised concept emergence, we trained a ProtoRel model **from scratch** (K=512, D=192, 2 layers, 1.2M parameters, no pre-training) on four diverse videos: a Chinese city walk (47 min), a majestic animals compilation (81 min), an African wildlife documentary (69 min), and a German community walk (29 min); nearly 3 hours total, 25,000 frames at 128×128 resolution per video, FRAME_SKIP=2). **No labels, annotations, or metadata were used at any stage**—the training is fully self-supervised. Importantly, **each frame is processed independently as a 2D image** through the ProtoRel architecture, accompanied by the corresponding audio segment extracted from the video's audio track as a mel-scale spectrogram. Unlike conventional video models that compress temporal information into 4D spatiotemporal tubelets (e.g., $T \times H \times W$ 3D patches), **time is encoded through the accumulation of visual and auditory experience in the persistent concept library** P—each frame and audio segment look up the same K prototypes, and temporal structure emerges from the dynamic evolution of A and S across frames rather than from explicit spatiotemporal convolution. The self-supervised task is masked patch prediction: 30% of the N=1024 patches per frame are replaced with their mean color, and the model learns to reconstruct the original image via MSE loss. Optimization uses AdamW (lr=0.0001, weight decay=0.01) with gradient clipping (max_norm=1.0) and batch size 8, running on a single NVIDIA RTX 5060 Laptop GPU (8GB). Training is heavily CPU-bound by video decoding rather than GPU computation (GPU utilization averages only ∼35%, under 500 MB VRAM). The GPU processes at 200–500 frames per second when preloaded, but the full pipeline including frame loading completes the nearly 3-hour video corpus in approximately 30 minutes. After training, we probed the model with still images of zebras, roe deer, monkeys, giraffes, and elephants—none of which were labeled during training.

Table 9 presents the dominant prototype activations for each animal category. The model spontaneously developed distinct activation patterns: P3 captures zebra- and deer- like mammals (22–37% activation), P386 specializes in giraffe and elephant features (30–39%), and P274 serves as a secondary concept across multiple species. Prototype differentiation (mea-

Table 9: Unsupervised concept formation: prototype activation patterns after nearly 3 hours of self-supervised video watching.

Species	P3	P386	P274
Zebra (4 animals)	22.4%	14.4%	12.9%
Zebra (photo 1)	31.3%	14.3%	11.8%
Zebra (photo 2)	36.7%	13.8%	10.7%
Roe Deer	23.9%	11.1%	16.1%
Monkey	12.9%	19.6%	14.3%
Giraffe	6.2%	**39.3%**	12.4%
Elephant	9.3%	**29.8%**	10.2%

sured as standard deviation of prototype norms) grew by 124% in Layer 1 over the course of training.

The cosine similarity of the full attention vectors (512-dimensional) reveals meaningful taxonomic clustering: two zebra photos exhibit a cosine similarity of 0.993, while zebra–giraffe similarity is only 0.43. Deer and zebra both cluster at 0.83, while giraffe and elephant cluster together at 0.96—recapitulating the mammalian taxonomic hierarchy without any supervised labels. This demonstrates that ProtoRel's concept library P naturally forms semantically meaningful organization from unsupervised visual experience.

7 Discussion

7.1 Static Concept Library + Dynamic Application

Perhaps the most distinctive property of the minimally constrained variant is the clean division of computation into a static permanent concept library (P) and a dynamic application layer (A, S) recomputed for each input. The concept library $P \in \mathbb{R}^{K \times D}$ is learned through gradient descent across all training samples and frozen after training; all projection matrices similarly crystallize into stable knowledge representations. In contrast, both A and S are recomputed afresh for every input: $A = \text{softmax}(h \cdot P^T)$ selects which concepts apply to the current image, and $S = \text{softmax}(\text{proj}(P) \cdot \text{proj}(P)^T)$ weaves real-time associations among the activated concepts.

This separation mirrors an intuitive cognitive principle: stable concepts (e.g., "cat") persist across experiences, while each encounter activates a context-dependent subset of features. Transformers perform no such separation—Q, K, V projection weights are static, but "concepts" are never instantiated as persistent entities; every input must reconstruct its full semantic understanding from raw weight interactions, never selecting from a pre-existing library. The implication for continual learning follows naturally: when learning new categories, the concept library P can be extended with additional prototypes rather than overwritten—original prototypes retain their learned functions, and the relation matrix S accumulates richer associations without catastrophic interference.

7.2 Interpretability

The ProtoRel Network possesses intrinsic interpretability requiring zero additional tools. Five distinct forms of model introspection are available directly from trained parameters and

Table 10: Bidirectional concept relations from the trained S matrix. Positive values indicate co-occurring concepts; negative values indicate mutually exclusive ones.

Relation	Prototype Pair	S
Positive: concept clusters (co-occurrence)		
Zebra cluster	P3–P242	42.0%
Giraffe cluster	P386–P410	37.3%
Cross-cluster (both mammals)	P3–P386	23.9%
Negative: mutual exclusivity		
Zebra vs. non-animal	P3–P491	−12.5%
Giraffe vs. non-animal	P386–P491	−16.9%

activations, without probing classifiers, attention rollout, or GradCAM:

1. **S-matrix heatmap**: The $K \times K$ matrix, visualized directly, reveals clusters of mutually reinforcing prototypes via blocks of strong positive correlation along the diagonal, highlighting the model's internal conceptual groupings (see Figure 1).

2. **Prototype utilization**: The effective prototype count per layer (computed as $\exp(H(A))$ where H is the attention entropy) quantifies how many concepts each layer actively employs—127 in Layer 2 dropping to 122 in Layer 3, then expanding to 184 in Layer 4 (Table 6).

3. **Concept importance**: Prototype norms serve as direct indicators of a prototype's influence, progressing from 0.56 (Layer 1) to 1.06 (Layer 4), with increasing standard deviations indicating deeper-layer specialization.

4. **Concept-relation strength**: Individual entries $S_{k\ell}$ are directly readable as signed association strengths between prototype pairs, as demonstrated in Table 3.

5. **Class-concept linkage**: The classifier weight matrix $W \in \mathbb{R}^{C \times D}$ can be projected onto the prototype space to identify which concepts are most relevant for each class.

This zero-overhead interpretability is a fundamental differentiator from Transformers, where 32 attention heads per layer produce $N \times N$ matrices that require aggregation, rollout, or probing to interpret. In the ProtoRel Network, interpretability is an architectural property, not a post-hoc addition.

The S matrix from the self-supervised video experiment provides a concrete illustration (Table 10). Two distinct concept clusters emerge: one centered on zebra-dominant P3, the other on giraffe-dominant P386, with positive cross-cluster correlation confirming both are animal concepts. At the same time, P491 forms negative relations with both, encoding a "non-animal" pattern the model learned to exclude. This bidirectional interpretability—positive for semantic co-occurrence, negative for mutual exclusivity—is directly readable from the trained S matrix without any post-hoc tools.

7.3 The Third Route

The claim that the ProtoRel Network constitutes a "third AI architecture route" rests on its fundamental difference from both CNNs and Transformers. CNNs express relations implicitly through spatial locality; Transformers compute relations implicitly through per-sample

Table 11: Conceptual marginalization: per-image prototypes shift from animal concepts (P3/P386/P274) to new prototypes (P398/P368/P51) after city training, yet cross-animal relational geometry (bottom) remains stable.

Prototype Change (Top-2 per image)	Relational Consistency
Zebra: P3(22%)P206(17%) → **P398(92%)**P222(2%)	Zebra–Deer: 0.985 → 0.928 (−5.8%)
Deer: P3(24%)P274(16%) → **P368(77%)**P47(12%)	Deer–Monkey: 0.994 → 0.994 (0.0%)
Giraffe: P386(39%)P274(12%) → **P51(34%)**P79(29%)	Zebra–Giraffe: 0.749 → 0.757 (+1.1%)
Elephant: P386(30%)P274(10%) → **P51(47%)**P79(13%)	Zebra R1–R2: 0.993 (post-city)
Monkey: P386(20%)P274(14%) → **P51(80%)**P341(5%)	Elephant–Giraffe: 0.978 (post-city)

dynamic QK^T; the ProtoRel Network makes relations explicit—through inspectable components P (which concepts exist) and S (how they relate). This explicitness enables the architecture's interpretability and cumulativity without sacrificing the global interaction capability of Transformers.

7.4 Conceptual Marginalization Under Domain Shift

The static concept library P implies a natural resistance to catastrophic forgetting: new knowledge can extend P rather than overwrite existing concepts. We tested this hypothesis by training a ProtoRel model from scratch on four diverse videos (nearly 3 hours), then exposing it to 12 minutes of city footage.

Table 11 shows the complete picture. Every animal prototype (P3, P386, P274, P206, P13) drops to near-zero after 12 minutes of city training, replaced by newly dominant concepts (P398 for zebra, P368 for deer, P51 for general mammal). Yet the relational geometry among animals—which pair is similar to which—survives the domain shift with minimal degradation. We term this **conceptual marginalization**: old concepts are preserved in P but no longer dominate the active attention space, pushed to the background by the newly learned city concepts. Full continual learning with explicit prototype locking and incremental expansion remains future work.

7.5 Adaptive Projection Dimension

A natural extension is adaptive control of the projection dimension d_{proj}, which governs $\text{rank}(S) \leq d_{\text{proj}}$ and serves as a built-in regularization knob unique to ProtoRel. Preliminary experiments on CIFAR-100 ($K = 512$, 50 epochs) support this direction: a learnable per-dimension gate $\text{sigmoid}(g) \in [0,1]^{d_{\text{proj}}}$ with $\max(d_{\text{proj}}) = 128$ achieved 35.4% accuracy through soft regularization (+0.6% over fixed $d_{\text{proj}} = 64$). Coupled with entropy-based pruning, the network autonomously reduced its projection dimension from 128 to 16 while retaining 34.4% accuracy (99% of the fixed 64-dim baseline), confirming that the eigendimension of the relation matrix is intrinsically low.

7.6 Limitations and Future Directions

This work represents an initial exploration; several directions remain open. **Real-time continual learning**: the static-dynamic separation and self-supervised training suggest that ProtoRel could learn continuously from camera streams—a step toward embodied AI. **Hierarchical concept libraries**: conceptual marginalization (Section 7.4) implies that a large K

(e.g., 4096) with natural prototype redundancy handles excess capacity via marginalization, requiring no explicit expansion mechanism. A "super-brain" ProtoRel trained on internet-scale data could share core concepts with domain-specific models. **Multimodal integration**: sharing the same P across vision, audio, and language would unify modalities under a single conceptual framework. **Scale**: all experiments here use $\leq$1.2M parameters; scaling to larger architectures and internet-scale data would test whether emergent concept formation generalizes.

8 Conclusion

We have presented the ProtoRel Network—a third deep learning architecture route that makes relations first-class citizens of computation through an explicit concept library P and a dynamic relation matrix S. The decomposition $R \approx A \cdot S \cdot A^T$ reduces complexity from $O(N^2)$ to $O(NK + K^2)$, while the static-dynamic separation (P permanent, A/S per-input) provides zero-overhead interpretability and natural resistance to catastrophic forgetting. Seven rounds of systematic failure showed that explicit design constraints consistently underperform the minimally constrained architecture, and self-supervised video watching demonstrated emergent concept formation—zebras, deer, giraffes, and elephants spontaneously differentiated without labels. Under domain shift, the architecture exhibits conceptual marginalization rather than forgetting (zebra–deer cosine 0.985$\rightarrow$0.928), while the S matrix provides bidirectional interpretability—positive relations reveal semantic co-occurrence and negative relations reveal mutual exclusivity—without any post-hoc tools. ProtoRel points toward a future where deep learning architectures do not merely fit data, but build, maintain, and expose structured knowledge through explicit concepts and relations.

Data Availability

CIFAR-100 is a publicly available benchmark dataset [7]. The video data used in self-supervised experiments are from publicly available footage. Code and trained models are available from the corresponding author upon reasonable request.

Conflicts of Interest

The authors declare no conflicts of interest.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In *NeurIPS*, 2012.

[2] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

[3] M. Tan and Q. V. Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019.

[4] A. Vaswani et al. Attention is all you need. In *NeurIPS*, 2017.

[5] A. Dosovitskiy et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021.

[6] J. Chen. Relational Gradient: Collective optimization beyond Adam. *Unpublished manuscript*, 2026.

[7] A. Krizhevsky and G. Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009.

[8] S. Wang et al. Linformer: Self-attention with linear complexity. *arXiv preprint*, 2020.

[9] K. Choromanski et al. Rethinking attention with performers. In *ICLR*, 2021.