Conditional Entropy Defenses with Attention

Slot + Gated Cross Attention vs. Gated Attention Pooling

Project Update

Group Meeting Preparation

Attention Privacy Project

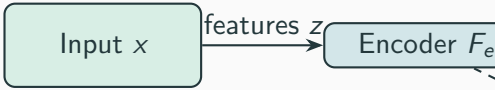
Roadmap

- Recap: collaborative inference threat model and CEM requirements
- Slot + Gated Cross Attention surrogate
- Gated Attention Pooling surrogate
- Comparison, monitoring guidance, next steps

Background

Collaborative Inference & CEM Goal

- Local encoder F_e uploads feature vectors z to the cloud decoder F_d; attacker can attempt input reconstruction.
- Conditional entropy maximization (CEM) seeks to keep $H(x \mid z)$ high so reconstructions remain uncertain.
- Legacy implementation used KMeans/GMM to approximate per-class variance and penalize sharp feature clusters.
- New attention surrogates replace GMM for richer class structure modeling while remaining differentiable.

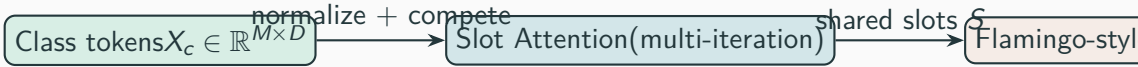


Design Criteria for Attention CEM

- **Fine-grained**: capture multi-modal subclasses within each label instead of assuming a single Gaussian.
- **Stable**: avoid exploding gradients and NaNs; guard against small batches or class imbalance.
- **Optimizer-friendly**: expose gradients that can be merged with encoder updates via λ.
- **Diagnostics**: surface interpretable statistics (slot gates, SNR signals, variance margins) to guide tuning.

Slot + Gated Cross Attention

Pipeline Overview



- Processes one class at a time; reuses slots to encode latent sub-modes.
- Cross attention re-injects slot context into every sample before measuring dispersion.

Slot Attention Core (Locatello et al.)

- Inputs are LayerNormed and projected to keys/values once per class.
- Slots are initialized from a learned Gaussian (μ, σ) and iteratively refined (T=3).
- Competitive soft assignments $r_{ms} = \text{softmax}(\beta \, \text{sim}(x_m, s_s))$ ensure each slot explains part of the class.
- GRU + MLP residual updates preserve temporal dynamics and stabilize slot representations.
- Learnable temperature β and slot sharpening power adjust how concentrated each slot becomes.

Flamingo-Style Gated Cross Attention

- Pre-LN multi-head cross attention (4 heads) with learnable α_{attn} gate:

$$y = q + \tanh(\alpha_{\text{attn}}) \cdot \mathrm{CrossAttn}(q, s).$$

- Feed-forward block mirrors Flamingo dense layer with a second gate α_{ffn} for gradual activation.
- Slots act as shared KV memory; query tokens are the LayerNormed class samples.
- Dropout-free path keeps gradients deterministic, easing debugging.

Conditional Entropy Surrogate

1. **Mixture-of-slots variance**: compute per-slot mean μ_s and variance σ_s^2 under sharpened responsibilities.
2. **Per-dimension gate**: sigmoid$(\mathrm{MLP}(\mathrm{LN}(\log \sigma_s^2)))$ suppresses low-signal dimensions.
3. **SNR gate**: $\sigma(\kappa(\sigma_s^2/(\mu_s^2+\epsilon)-\tau))$ dampens high-SNR activations that leak semantics.
4. **Softplus margin**: $L_{\text{base}} = \frac{1}{\beta'}\log\left(1+e^{\beta'(\log\sigma_s^2-\log\tau-m)}\right)$ smooths the hinge at the variance threshold.
5. **Slot mass weighting**: responsibilities are sharpened by $w_s \propto (\text{mass}_s/M)^\rho$ to focus on dominant slots.
6. **Class gate**: $g_c = \sigma(a(M/B-b))$ scales classes with scant samples to avoid noisy penalties.

Training Integration & Safety Nets

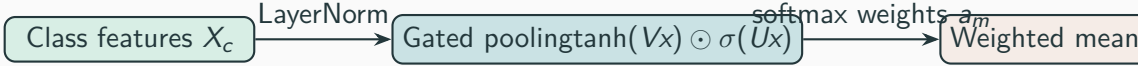
- Activation: only after warmup (> 3 epochs), non-random centers, and $\lambda > 0$.
- Gradients: backprop through L_C, stash encoder gradients, zero optimizers, then blend with classification gradients using λ and LR-aware rescaling.
- Scaling: final L_C contribution multiplied by `attention_loss_scale` $= 0.25$ before gradient capture.
- Early shutoff: for first 100 calls or when gate statistics exceed thresholds, L_C returns a graph-connected zero to avoid destabilizing the encoder.
- Robustness: skip computation if NaN/Inf features, and lazily register module parameters into the optimizer only once.
- Logging: prints mean gate activations and slot statistics every 200 calls to guide hyper-parameter tuning.

Gated Attention Pooling

Why a Second Surrogate?

- Slot pipeline is expressive but heavy: GRUs, multi-head attention, and numerous gates increase flops and tuning cost.
- Need a lightweight alternative for large batches, high-resolution encoders, or quick ablations.
- Gated Attention Pooling borrows from deep MIL: a single attention distribution summarizes class features.

Pooling Architecture



Mathematics

$$
\begin{aligned}
a_m &= \frac{\exp\left(w^\top [\tanh(Vx_m) \odot \sigma(Ux_m)]\right)}{\sum_j \exp(\cdot)}, \\
\mu_c &= \sum_m a_m x_m, \qquad \sigma_c^2 = \sum_m a_m (x_m - \mu_c)^2, \\
\tau &= \texttt{var\_threshold} \cdot \texttt{reg\_strength}^2 + \gamma, \\
L_C &= \max\{0,\, \log(\sigma_c^2 + \gamma) - \log(\tau)\}\,.
\end{aligned}
$$

- LayerNorm of inputs curbs variance collapse under sharp attention distributions.
- Hidden size defaults to $\min(512, \max(64, D/4))$ per class at instantiation.
- ReLU-style hinge keeps gradients sparse when class variance already exceeds the target threshold.

Training Notes

- Warmup extends to 5 epochs to let softmax weights stabilize.
- Module lazily constructed on first use and registered with the optimizer; gradients combined with encoder updates just like the slot version.
- Uses a smaller scaling factor: `attention_loss_scale` $= 0.1$ before gradient capture.
- Skips classes with ≤ 1 samples to avoid ill-defined variance; safe defaults return zero loss.
- Same guardrails for NaN/Inf detection and optimizer resets after the auxiliary backward pass.

When to Prefer Each

- **Use Slot + Gated Cross Attention** when class heterogeneity is high (complex datasets, multi-modal privacy leakage) and logging bandwidth is available.
- **Use Gated Attention Pooling** for rapid experimentation, limited compute, or when you mainly need coarse variance inflation.
- Both surrogates can be toggled via the training script; only one runs per experiment.

Comparison & Next Steps

Side-by-Side Summary

	Slot + Gated Cross Attn	**Gated Attention Pooling**
Representation	Multi-slot latent mixture with cross-attention refinement	Single attention distribution over class tokens
Gates	Per-dim MLP, SNR gate, slot mass, class gate, softplus margin	Single gated attention + variance hinge
Compute	GRU iterations + multi-head attention (heavier)	Two linear layers + softmax (light)
Tuning knobs	β, slot power, SNR threshold, class gate bias, margin	Threshold τ, loss scale, hidden width
Safety	Early shutoff, extensive logging, NaN guards	Skip small classes, automatic hidden size

Operational Guidance

- Monitor: slot gate means, hard gate averages, and class-level activations printed by the module; large values signal over-sharpening.
- For Gated Pooling, track variance histograms versus τ to ensure the hinge is active for risky classes.
- Adjust λ schedule alongside `attention_loss_scale` to balance accuracy and privacy cost.
- Remember to rerun validation with reconstruction attacks after enabling either surrogate to quantify impact.

Thank you!