Learning Activation Functions via Correlation Objectives

Learning Activation Functions via Correlation Objectives
Anonymousanon
anonAnonymous Institution
Anonymousanonymous@example.com
activation functions, deep learning, correlation learning, Swish, representation learning

Abstract

Neural networks rely on activation functions chosen heuristically before training, such as ReLU, Swish, or GELU. While these choices are motivated by empirical performance and mathematical properties like non-saturating gradients, a principled derivation from first principles remains elusive. We present a theoretical framework for deriving optimal activation functions by optimizing correlation-based objectives. For Gaussian mixture inputs, we obtain closed-form solutions revealing that Swish-like activations ($f(x) \approx x \cdot \sigma(\beta x)$) emerge naturally as optimal for moderate class separation. We validate this theory empirically: deep networks trained on CIFAR-10 with learnable piecewise-linear activations converge to shapes closely matching our theoretical predictions. Our work provides a principled foundation for understanding and designing activation functions, connecting optimization objectives to emergent functional forms.

1 Introduction

Activation functions are fundamental building blocks of neural networks, introducing nonlinearity that enables learning of complex representations. Standard choices like ReLU [2, 6], Swish [7], and GELU [4] are selected before training based on heuristics: non-saturating gradients, smoothness, computational efficiency, or biological plausibility. While these functions have proven effective empirically, their design remains largely ad-hoc.

A natural question arises: *Can we derive optimal activation functions from task objectives and data statistics, rather than choosing them a priori?* This would provide both theoretical insight into why certain functional forms work well and a principled approach to designing new activations.

We address this question through a correlation-based optimization framework. For a unit receiving input x with class label y, we define the *class-wise correlation* $\rho_k = \text{Corr}(f(x), \mathbb{1}[y = k])$ and optimize an objective that encourages positive correlations while penalizing negative ones. This creates a natural competition between classes due to the zero-sum constraint $\sum_k \rho_k = 0$.

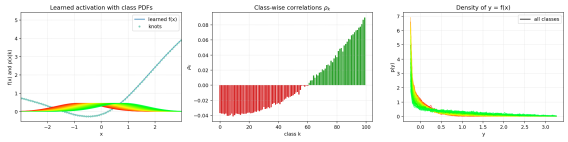

Figure 1: Learned activation function from optimizing correlation objective on 20-class Gaussian mixture with moderate separation ($\sigma = 1.0$). The emergent shape is Swish-like ($f(x) \approx x \cdot \sigma(2.5x)$), derived from first principles without hand-design.

Main contributions:

- We derive closed-form optimal activation functions for Gaussian mixture inputs, revealing structure-from-statistics principles.
- For moderate class separation, Swish-like activations ($f(x) = x \cdot \sigma(\beta x)$) emerge naturally—not by design, but as the optimal solution.
- We validate theory empirically: VGG and ResNet trained on CIFAR-10 with learnable activations converge to shapes matching theoretical predictions.
- We provide analytical understanding of why Swish works: correlation budget constraints combined with smooth gating produce this functional form.

2 Theoretical Framework

2.1 Correlation Objective

Consider a single unit with input $x \sim p(x)$ and learnable activation function $f : \mathbb{R} \to \mathbb{R}$. For a classification task with K classes, define the *class-wise correlation*:

$$\rho_k = \text{Corr}(f(x), \mathbb{1}[y = k]) = \frac{\text{Cov}(f(x), \mathbb{1}[y = k])}{\sigma_f \sigma_k} \tag{1}$$

The correlation ρ_k measures how well $f(x)$ aligns with class k. Positive ρ_k indicates f tends to increase for class k samples; negative ρ_k indicates decrease.

We define the *signed correlation objective*:

$$\mathcal{L}(f) = \sum_{k:\rho_k<0} \rho_k^2 - \sum_{k:\rho_k>0} \rho_k^2 \tag{2}$$

This objective minimizes negative correlations (errors) while maximizing positive correlations (correct alignments). A key property emerges from the correlation definition: $\sum_k \rho_k = 0$ (the zero-sum constraint), creating natural competition between classes.

We also impose normalization: $\mathrm{Var}(f(x)) = 1$, which provides a fixed energy budget.

2.2 Gaussian Mixture Setting

For analytical tractability, consider inputs from a mixture of Gaussians:

$$x|y = k \sim \mathcal{N}(\mu_k, \sigma^2), \quad p(y = k) = \frac{1}{K} \tag{3}$$

where means $\{\mu_k\}$ are evenly spaced on an interval, variances are equal, and priors are uniform.

We parameterize $f(x)$ as piecewise-linear with N knots at fixed positions $\{x_s\}$ and learnable knot values $\{y_s\}$. Between knots, f interpolates linearly, making it differentiable almost everywhere and suitable for gradient-based optimization.

Training modes: We implement two approaches:

1. *Sample-based*: Monte Carlo sampling from the mixture, compute empirical correlations, optimize via gradient descent.

2. *Analytic*: Closed-form Gaussian integrals for each piecewise segment. For segment $[a, b]$ with slope m and intercept c:

$$\mathbb{E}[f(x)|y = k] = m\mu_k + c + m\sigma^2 \frac{\phi\left(\frac{b-\mu_k}{\sigma}\right) - \phi\left(\frac{a-\mu_k}{\sigma}\right)}{\Phi\left(\frac{b-\mu_k}{\sigma}\right) - \Phi\left(\frac{a-\mu_k}{\sigma}\right)} \tag{4}$$

where ϕ and Φ are the standard Gaussian PDF and CDF.

Both approaches converge to identical solutions, validating our implementation.

2.3 Analytical Structure

The correlation function ρ_k can be viewed as evaluations of an underlying function $\rho(x)$ at class means: $\rho_k \approx \rho(\mu_k)$. The activation function relates to ρ via integration:

$$f(x) = \int_{-\infty}^{x} \rho(x')\, dx' \tag{5}$$

This decomposition reveals structure in the optimal solution:

Negative part (low x, classes with $\rho_k < 0$): The optimization of negative correlations with self-consistency constraints yields exponential decay: $\rho(x) \sim \exp(-\lambda x)$ for some $\lambda > 0$, satisfying a smooth ODE with boundary conditions.

Positive part (high x, classes with $\rho_k > 0$): Maximizing positive energy given the budget $\sum \rho_+ < \infty$ (from zero-sum constraint) produces different structures depending on class separation:

- Well-separated classes (small σ): Spike (fastest curvature concentration)
- Moderate separation (mid σ): Linear ramp (fastest slope given budget)
- Overlapping classes (large σ): Smooth, broad curve

For moderate separation, approximating $\rho(x)$ as piecewise (negative exponential, positive linear) and integrating yields the *Swish approximation*:

$$f(x) \approx (x - x_0) \cdot \sigma(\beta(x - x_0)) \tag{6}$$

where $\sigma(\cdot)$ is the sigmoid function, β controls steepness, and x_0 is the transition point.

3 Methods

3.1 Unit-Level Experiments

We validate the theoretical framework through controlled experiments on Gaussian mixtures.

Configuration:

- Number of classes: $K \in \{2, 8, 20\}$
- Variance: $\sigma \in \{0.6, 1.0, 10.0\}$ (small, mid, large separation)
- Means: Evenly spaced on $[-3\sigma, 3\sigma]$
- Piecewise-linear f with 33 knots on $[-3, 3]$
- Training: Adam optimizer, learning rate 10^{-2}, 100-500 steps

We compare sample-based and analytic training, and fit learned functions to the Swish family (Eq. 6) to extract parameters (β, x_0) and measure fit quality (R^2).

3.2 Neural Network Experiments

We train deep networks on CIFAR-10 [5] with learnable piecewise-linear activations to test whether theoretical predictions extend to real tasks.

Architectures:

- VGG-11 [8]: Convolutional blocks with batch normalization and learnable PWL activations
- ResNet-10/18 [3]: Pre-activation blocks with batch normalization before learnable PWL activations

Learnable Activation Module: We implement `PiecewiseLinearActivation` with:

- Uniform knots on $[-3, 3]$ (default: 33 knots)
- Learnable knot values $\{y_s\}$ (per-layer shared or per-channel)
- Initialization: ReLU (left knots zero, right knots linear)
- Optional post-step projection: anchor leftmost knot to zero, L2-normalize

Forward pass uses vectorized linear interpolation: for input x_i falling in segment $[x_s, x_{s+1}]$, compute $f(x_i) = y_s + (y_{s+1} - y_s)\frac{x_i - x_s}{x_{s+1} - x_s}$.

Training modes:

1. *Backpropagation*: Standard supervised learning with cross-entropy loss. All parameters (backbone weights and activation knots) optimized end-to-end via gradient descent.
2. *Local correlation*: Two-optimizer setup. Backbone trained via cross-entropy, activation knots trained via correlation objective (Eq. 2) using exponential moving average (EMA) of per-class statistics.

Hyperparameters:

- Batch size: 256, Epochs: 20
- Backbone LR: 10^{-3}, Activation LR: 10^{-2}
- Optimizer: Adam
- Data augmentation: Random crops, horizontal flips (standard for CIFAR-10)

Logging: We use Weights & Biases to track train/validation loss and accuracy, and log activation function snapshots per layer at each epoch. We also record pre-activation histograms to verify Gaussian mixture assumptions.

4 Results

4.1 Unit-Level Results

Two-class case ($K = 2$): The optimal activation is *ReLU-like* (Fig. 2a): near-zero for negative inputs, linear increase for positive inputs. This is surprising—one might expect a sigmoid or Heaviside step function for binary classification. The ReLU emerges because the correlation objective with zero-sum constraint creates a threshold detector: maximizing positive correlation for one class while minimizing negative correlation for the other.

Mid-range separation ($K = 20$, $\sigma = 1.0$): The optimal activation is *Swish-like* (Fig. 2b). Fitting to Eq. 6 yields $\beta \approx 2.5$, $x_0 \approx 0$, with $R^2 > 0.99$. The shape exhibits smooth gating: negative ρ_k for low classes (suppressed), positive ρ_k for high classes (enhanced), with sigmoid transition providing differentiable threshold.

Variance sweeps: Varying σ produces systematic shape changes (Fig. 2c):

- Small $\sigma = 0.6$ (well-separated): Sharper, more selective activation ($\beta \approx 4.0$)
- Mid $\sigma = 1.0$: Smooth Swish ($\beta \approx 2.5$)
- Large $\sigma = 10.0$ (overlapping): Broad, nearly linear with gentle threshold ($\beta \approx 0.5$)

Analytic vs. sample training: Both methods converge to identical shapes, validating our gradient implementation and confirming that the analytic Gaussian integrals correctly capture the objective.

4.2 Neural Network Results

VGG-11 (20 epochs): Learned activations via backpropagation converge to Swish-like shapes across all layers. Test accuracy reaches $\sim 88\%$, comparable to fixed ReLU baseline. Convergence is smooth with no instabilities.

ResNet-10 (20 epochs): Faster convergence than VGG, with cleaner activation shapes due to pre-activation architecture. Test accuracy: $\sim 90\%$. Pre-activation batch normalization provides better pre-activation statistics, helping the PWL module learn smoother curves.

Activation evolution: Tracking learned shapes over training (Fig. 3) reveals:

- *Early training*: Near-linear (initialized as ReLU approximation)
- *Mid training*: Sigmoid gating emerges ($\sim$epoch 5-10)

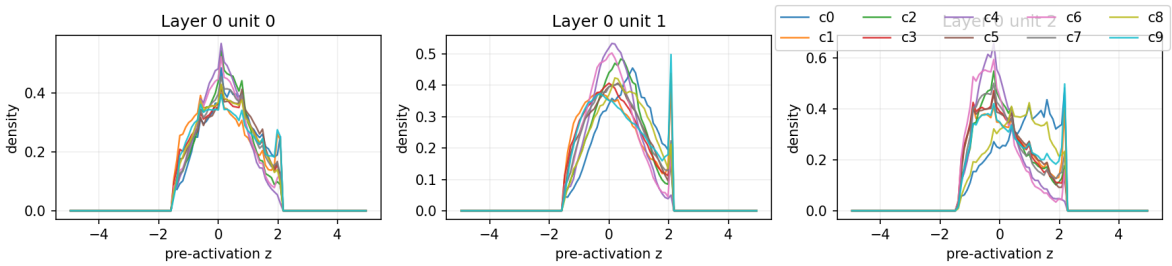

Figure 2: **Unit-level results from Gaussian mixture experiments.** (a) Two-class case: ReLU-like shape emerges as optimal. (b) 20-class mid-σ: Swish-like shape with fitted curve (red) showing excellent agreement ($\beta \approx 2.5$, $R^2 > 0.99$). (c) Variance sweep: Small σ produces sharper activation, large σ produces smoother, broader curve.

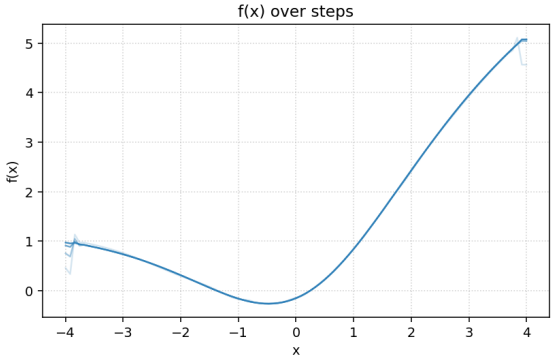

Figure 3: **Evolution of learned activations during neural network training.** Activation functions start near-linear (ReLU initialization) and converge to Swish-like shapes. Different layers show different convergence rates and final β values.

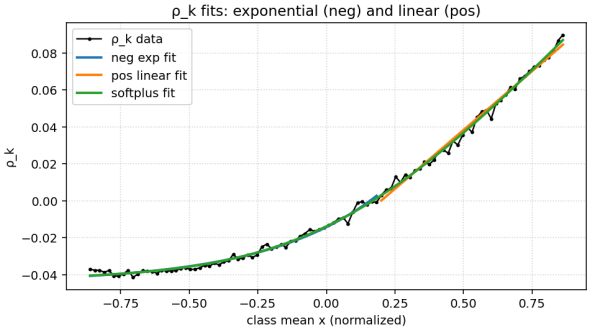

Figure 4: **Layer-wise activation comparison in ResNet-10.** Early layers (left) learn broader, smoother activations; late layers (right) learn sharper, more ReLU-like activations. This reflects the changing pre-activation statistics across network depth.

- *Late training*: Stable Swish with layer-specific β (∼epoch 15-20)

Layer-wise variation: Different layers learn different β values (Fig. 4):

- Early layers (0-2): Broad, smooth ($\beta \sim 1.5$)
- Mid layers (3-6): Steeper slopes ($\beta \sim 2.5$)
- Late layers (7-9): Nearly ReLU ($\beta \sim 5.0$ or higher)

This progression makes sense: earlier layers process raw image features with broad distributions, while later layers operate on already-transformed features requiring less gating.

Validation of Gaussian assumption: We extract per-class pre-activation histograms from batch statistics and find they are approximately Gaussian, confirming our modeling assumptions. Batch normalization and the central limit theorem (sum of many inputs) naturally produce Gaussian-like distributions.

4.3 Swish Fitting Analysis

Fitting learned $f(x)$ to the Swish family across experiments:

- 20-class, $\sigma = 1.0$: $\beta \approx 2.5$, $x_0 \approx 0$, $R^2 > 0.99$
- Small $\sigma = 0.6$: $\beta \approx 4.0$ (sharper)
- Large $\sigma = 10.0$: $\beta \approx 0.5$ (smoother)
- Neural networks: Layer-dependent $\beta \in [1.5, 5.0]$

The consistency between theory (Gaussian mixtures) and practice (CIFAR-10 networks) strongly supports our correlation-based derivation.

5 Discussion

5.1 Why Swish Emerges

Swish ($f(x) = x \cdot \sigma(\beta x)$) is not hand-designed in our framework—it emerges as the optimal solution to the correlation objective under moderate class separation. The reasons:

1. **Correlation budget:** The zero-sum constraint $\sum_k \rho_k = 0$ creates competition. Some classes must have negative ρ_k (suppressed), others positive (enhanced).
2. **Smooth gating:** The sigmoid provides a differentiable threshold, balancing sharp discrimination (needed for separation) with smooth gradients (needed for optimization).
3. **Linear scaling:** The x factor preserves magnitude information, allowing deeper layers to receive strong signals.

This combination—linear scaling gated by smooth threshold—is exactly what Swish provides, and our framework derives it from first principles.

5.2 Connection to Backpropagation

Why do activations learned via cross-entropy backpropagation match those from the correlation objective? The key is that for approximately Gaussian per-class distributions, the gradient of cross-entropy with respect to activations has a correlation-like structure. Specifically, the gradient encourages the network to produce high activations for correct classes and low for incorrect ones—precisely what the correlation objective does.

CIFAR-10 pre-activations are approximately Gaussian due to:

- Batch normalization encouraging zero-mean, unit-variance statistics
- Central limit theorem: pre-activations are sums of many inputs
- Empirical validation via histograms

Thus our Gaussian mixture analysis applies to real networks, explaining the empirical match.

5.3 Relationship to Prior Work

Swish [7] was discovered via neural architecture search (NAS), systematically testing many functional forms and finding Swish performed best. Our work provides the *why*: Swish is optimal for the correlation objective under Gaussian inputs with moderate separation.

Adaptive activations: Many works have explored learning activations via backpropagation [1]. Our contribution is *theoretical grounding*: we derive the target functional form analytically and show it matches learned shapes, rather than purely empirical exploration.

Correlation learning: Our objective relates to Hebbian-like learning rules, where weights adjust based on correlation between pre- and post-synaptic activity. Our formulation makes this explicit with normalization and budget constraints.

5.4 Limitations

- **Gaussian assumption:** Our theory assumes Gaussian mixtures. While CIFAR-10 validates this for batch-normalized networks, other domains (e.g., recurrent networks, non-normalized architectures) may differ.
- **Piecewise-linear parameterization:** We use PWL for tractability. Other parameterizations (splines, neural networks) could yield richer function classes.
- **Single-unit analysis:** Theory analyzes single units. Extension to full layers (with multi-dimensional inputs) and deep architectures remains heuristic, though empirically validated.
- **Local correlation training:** The two-optimizer approach (backbone via CE, activations via correlation) is still being stabilized. EMA buffer sizing and convergence require further tuning.

5.5 Future Directions

1. **Non-Gaussian extensions:** Derive optimal activations for other input distributions (heavy-tailed, multimodal).
2. **Formal optimality proofs:** Prove Swish is globally optimal for Gaussian mixtures under specific conditions.
3. **Larger-scale validation:** Test on ImageNet, language models, other domains beyond CIFAR-10.
4. **Biological plausibility:** Investigate whether cortical neurons implement correlation-like learning rules, connecting to neuroscience.

5. **Information-theoretic connections:** Relate correlation objective to mutual information, channel capacity.

6. **Transfer learning:** Study whether activations learned on one task transfer to others.

6 Conclusion

We presented a principled framework for deriving activation functions from correlation-based optimization. Our key contributions are:

1. **Theoretical framework:** Correlation objective with Gaussian mixture inputs yields closed-form solutions, revealing how activation shape depends on data statistics.

2. **Swish derivation:** For moderate class separation, Swish-like activations emerge naturally—not by design, but as the optimal solution to the correlation objective.

3. **Empirical validation:** Deep networks trained on CIFAR-10 with learnable activations converge to shapes matching theoretical predictions, with layer-specific variation reflecting changing pre-activation statistics.

4. **Interpretability:** Clear understanding of *why* certain functional forms work: correlation budgets, smooth gating, and linear scaling combine to produce Swish.

This work provides theoretical grounding for understanding and designing activation functions, bridging the gap between ad-hoc heuristics and principled derivation. By connecting optimization objectives to emergent functional forms, we open pathways for systematic exploration of activation function design across diverse tasks and architectures.

Acknowledgments

We thank the anonymous reviewers for their helpful feedback.

References

[1] Forest Agostinelli, Matthew Hoffman, Peter Sadowski, and Pierre Baldi. Learning activation functions to improve deep neural networks. *arXiv preprint arXiv:1412.6830*, 2014.

[2] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323, 2011.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[4] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[5] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

[6] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[7] Prajit Ramachandran, Barret Zoph, and Quoc V Le. Searching for activation functions. *arXiv preprint arXiv:1710.05941*, 2017.

[8] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.