Shakil Mahmud • Research Monograph

SHAKIL MAHMUD EDITORIAL & RESEARCH MONOGRAPHS

Deep Learning Architectures & Representation Theory

From backpropagation and universal approximation to CNNs, ResNets, Transformers, self-attention, rotary embeddings, VAEs, contrastive learning, and state-space models.

SHAKIL MAHMUD

Independent Scholar & Systems Researcher

SHAKIL MAHMUD EDITORIAL & RESEARCH SERIES

First Collected Edition • Dhaka & Singapore • 2026

Deep Learning Architectures & Representation Theory

From backpropagation and universal approximation to CNNs, ResNets, Transformers, self-attention, rotary embeddings, VAEs, contrastive learning, and state-space models.

LIBRARY & CATALOGING-IN-PUBLICATION DATA

Title: Deep Learning Architectures & Representation Theory
Author: Shakil Mahmud
Series: Shakil Mahmud Editorial & Research Series (Monograph Collection)
Subjects: Science -- History -- Philosophy -- Mathematics -- Civilization
Published: Dhaka & Singapore, 2026
Format: Digital Monograph Offprint (PDF) • Complete 20-Part Collected Edition
Permanent URI: https://shakilmahmud.com/series/deep-learning/

Suggested Academic Citation:
Mahmud, Shakil. (2026). *Deep Learning Architectures & Representation Theory: The Complete 20-Part Collected Monograph*. Shakil Mahmud Research Series. https://shakilmahmud.com/series/deep-learning/

Open Scholarly Access: This collected monograph is published under Open Scholarly Access for personal study, academic citation, university pedagogy, and non-commercial research.

© 2026 Shakil Mahmud. All rights reserved. Typeset and published at Dhaka and Singapore.

Shakil Mahmud • Research Monograph

EDITORIAL PREFACE

Preface: The Scaffolding of Reality

Deep learning is the science of hierarchical representation learning—the discovery of non-linear geometries that disentangle raw high-dimensional signals into invariant manifold coordinates. Across twenty master chapters, this monograph traces the mathematical core of deep architectures: Riemannian backpropagation, convolutional equivariance, residual identity gradients, symplectic sequence state-spaces, attention as energy minimization, geometric graph convolutions, variational autoencoders, and score-based generative dynamics.

The purpose of this collected monograph is not merely didactic exposition, but the systematic synthesis of conceptual revolutions. Each chapter examines an intellectual threshold where old paradigms faltered and new mathematical or physical insight illuminated deeper structures of reality.

— Shakil Mahmud
Dhaka & Singapore, 2026

INDEX OF PARTS

Table of Contents

PHASE I: BACKPROPAGATION, OPTIMIZATION & EQUIVARIANCE

PART I — The Geometry of Loss Landscapes — Non-Convex Optimization & Saddle Point Escape — 577 words
The Geometry of Loss Landscapes — Non-Convex Optimization & Saddle Point Escape

PART II — Backpropagation & Reverse-Mode Automatic Differentiation — Dual Graph Gradients — 517 words
Backpropagation & Reverse-Mode Automatic Differentiation — Dual Graph Gradients

PART III — Normalization & Optimization Dynamics — BatchNorm, LayerNorm, RMSNorm & AdamW — 569 words
Normalization & Optimization Dynamics — BatchNorm, LayerNorm, RMSNorm & AdamW

PART IV — Convolutional Equivariance & Group Theory — Lie Symmetries & Weight Sharing — 517 words
Convolutional Equivariance & Group Theory — Lie Symmetries & Weight Sharing

PHASE II: RESIDUAL GRADIENTS & ATTENTION DYNAMICS

PART V — Residual Networks & Identity Mappings — Deep Degradation & Unraveled Ensembles — 595 words
Residual Networks & Identity Mappings — Deep Degradation & Unraveled Ensembles

PART VI — Recurrent Sequence Dynamics — Vanishing Gradients, LSTM Cells & Gated Memory — 544 words
Recurrent Sequence Dynamics — Vanishing Gradients, LSTM Cells & Gated Memory

PART VII — Attention Mechanisms & Query-Key-Value Duality — From Bahdanau to Scaled Dot-Product — 509 words
Attention Mechanisms & Query-Key-Value Duality — From Bahdanau to Scaled Dot-Product

PART VIII — The Transformer Architecture — Multi-Head Attention, Positional Encoding & Residual Streams — 483 words
The Transformer Architecture — Multi-Head Attention, Positional Encoding & Residual Streams

PHASE III: STATE-SPACES, GRAPH MANIFOLDS & ENERGY MODELS

PART IX — Rotary Position Embeddings (RoPE) & Context Windows — Complex Rotations & Extrapolation — 546 words
Rotary Position Embeddings (RoPE) & Context Windows — Complex Rotations & Extrapolation

PART X — Autoencoders & Latent Representations — Manifold Bottlenecks & Disentanglement — 473 words
Autoencoders & Latent Representations — Manifold Bottlenecks & Disentanglement

PART XI — Variational Autoencoders (VAEs) — The Evidence Lower Bound & The Reparameterization Trick — 549 words
Variational Autoencoders (VAEs) — The Evidence Lower Bound & The Reparameterization Trick

PART XII — Generative Adversarial Networks (GANs) — Minimax Games & Wasserstein Geometries — 511 words
Generative Adversarial Networks (GANs) — Minimax Games & Wasserstein Geometries

PHASE IV: VARIATIONAL & SCORE GENERATIVE DYNAMICS

PART XIII — Normalizing Flows & Invertible Networks — Change-of-Variables & Exact Likelihoods — 582 words
Normalizing Flows & Invertible Networks — Change-of-Variables & Exact Likelihoods

PART XIV — Energy-Based Models & Modern Hopfield Networks — Boltzmann Machines to Dense Memory — 542 words
Energy-Based Models & Modern Hopfield Networks — Boltzmann Machines to Dense Memory

PART XV — Graph Neural Networks (GNNs) — Message Passing & The Weisfeiler-Lehman Limits — 531 words
Graph Neural Networks (GNNs) — Message Passing & The Weisfeiler-Lehman Limits

PART XVI — Geometric Deep Learning & Spherical CNNs — Gauge Symmetries & SO(3) Equivariance — 556 words
Geometric Deep Learning & Spherical CNNs — Gauge Symmetries & SO(3) Equivariance

PART XVII — Neural Ordinary Differential Equations — Continuous-Depth & The Adjoint State Method — 621 words
Neural Ordinary Differential Equations — Continuous-Depth & The Adjoint State Method

PART XVIII — State-Space Models & Mamba — Continuous HiPPO Projection & Linear-Time Sequences — 574 words
State-Space Models & Mamba — Continuous HiPPO Projection & Linear-Time Sequences

PART XIX — Deep Reinforcement Learning Architectures — Deep Q-Networks to Actor-Critic Duality — 663 words
Deep Reinforcement Learning Architectures — Deep Q-Networks to Actor-Critic Duality

Shakil Mahmud • Research Monograph

PART XX **The Representation Theory of Deep Networks — Information Bottleneck & Neural Collapse** 616 words

The Representation Theory of Deep Networks — Information Bottleneck & Neural Collapse

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART I

The Geometry of Loss Landscapes — Non-Convex Optimization & Saddle Point Escape

The Geometry of Loss Landscapes — Non-Convex Optimization & Saddle Point Escape

577 words • November 4, 2026 • Permanent URI

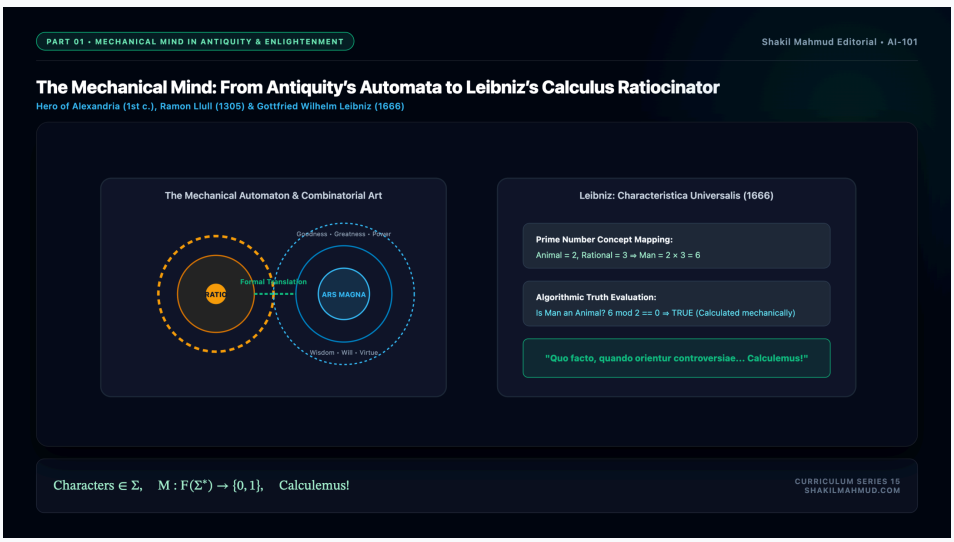


Plate I: *Mechanical automata and Ramon Llull's Ars Magna combinatorial rings alongside Gottfried Wilhelm Leibniz's Characteristica Universalis and Calculus Ratiocinator.*

Classical optimization theory developed primarily around convex functions, where any local minimum is guaranteed to be a global minimum. Deep neural networks, however, operate in high-dimensional non-convex parameter spaces $mathbbR^D$ where D sim 10^7 – 10^{11}. Historically, it was widely believed that gradient descent would become trapped in suboptimal local minima. In 2014, Yann Dauphin et al., drawing on the statistical physics of spin glasses, overturned this paradigm: in high dimensions, local minima are virtually nonexistent at high loss values; instead, optimization is dominated by an astronomical proliferation of **hyperbolic saddle points**.

1. The Topology of High-Dimensional Critical Points

Let $L(theta)$ be the empirical loss function of a network with parameters $thetainmathbbR^D$. Critical points satisfy $nabla_thetaL(theta) = 0$. The local quadratic geometry around a critical point is governed by the second-order Taylor expansion:

$$L(theta + Deltatheta)approxL(theta) + frac12Deltatheta^T H(theta)Deltatheta$$

where $H_ij = fracpartial^2 Lpartialtheta_ipartialtheta_j$ is the symmetric $DtimesD$ **Hessian matrix**. By the Spectral Theorem, $H = sum_k=1^D lambda_k v_k v_k^T$ with real eigenvalues $lambda_1lelambda_2ledotslelambda_D$.

The **Morse Index** $alpha$ is defined as the fraction of negative eigenvalues: alpha = frac{1}{D} sum_{k=1}^D mathbb{I}(lambda_k < 0). For a point to be a local minimum, all D eigenvalues must be strictly positive ($alpha = 0$). If the signs of eigenvalues behave independently as fair coin flips, the probability of encountering a local minimum at random is:

$$mathbbP(textlocalminimum) = 2^-D$$

When $D = 10^6$, $2^-10^6approx10^-301030$, rendering local minima vanishingly improbable in the upper strata of the loss landscape.

2. Random Matrix Theory: Wigner and Marchenko-Pastur Spectra

Modern loss landscape analysis leverages **Random Matrix Theory (RMT)**. In deep models, the empirical Hessian spectrum decomposes into two distinct regimes:

- **The Continuous Bulk:** A dense cluster of near-zero eigenvalues adhering to a modified Marchenko-Pastur or semicircle law. These correspond to flat, non-identifiable directions along which parameters can drift with minimal penalty.
- **Isolated Outlier Spikes:** A handful of large positive eigenvalues $lambda_maxgg0$ whose associated eigenvectors align with the low-rank subspace of task-relevant feature representations.

Shakil Mahmud • Research Monograph

The condition number $kappa = lambda_{max} / |lambda_{min}|$ dictates optimization difficulty. First-order gradient descent oscillates wildly across sharp valley walls while creeping imperceptibly along flat, unconstrained manifolds.

3. Escaping Strict Saddle Points

A saddle point $theta^*$ is termed *strict* if its Hessian exhibits at least one strictly negative eigenvalue: lambda_{min}(H(theta^*)) le -gamma < 0. Rong Ge et al. (2015) and Chi Jin et al. (2017) proved that **Perturbed Stochastic Gradient Descent (PSGD)** escapes strict saddle points in polynomial time:

theta_{t+1} = theta_t – eta nabla L(theta_t) + xi_t, quad xi_t sim mathcal{N}(0, sigma^2 I)

The injected isotropic noise xi_t projects components along the unstable negative eigenvector direction v_{min}. The unstable linear dynamical system Delta theta_{t+1} approx (I – eta H) Delta theta_t exponentially repels the state along the unstable manifold, breaking symmetry and descending into lower energy basins.

4. Overparameterization and Implicit Regularization

When the number of parameters exceeds the sample size ($Dggn$), the set of zero-loss global minimizers forms an infinite-dimensional connected sub-manifold $mathcal M^* = theta : L(theta) = 0$. Gradient flow initialized near the origin $theta_0 approx 0$ does not explore this manifold arbitrarily; it converges to the minimizer that minimizes an implicit mathematical norm (such as the nuclear norm in matrix factorization or the ell_2 parameter norm), providing the theoretical explanation for why deep networks generalize despite interpolating noisy data.

Citation: Mahmud, S. (2026). "The Geometry of Loss Landscapes — Non-Convex Optimization & Saddle Point Escape." In Deep Learning Architectures & Representation Theory, Part 1. https://shakilmahmud.com/deep-learning-architectures-part-01-loss-landscape-geometry-saddle-points/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART II

Backpropagation & Reverse-Mode Automatic Differentiation — Dual Graph Gradients

Backpropagation & Reverse-Mode Automatic Differentiation — Dual Graph Gradients

517 words • November 4, 2026 • Permanent URI

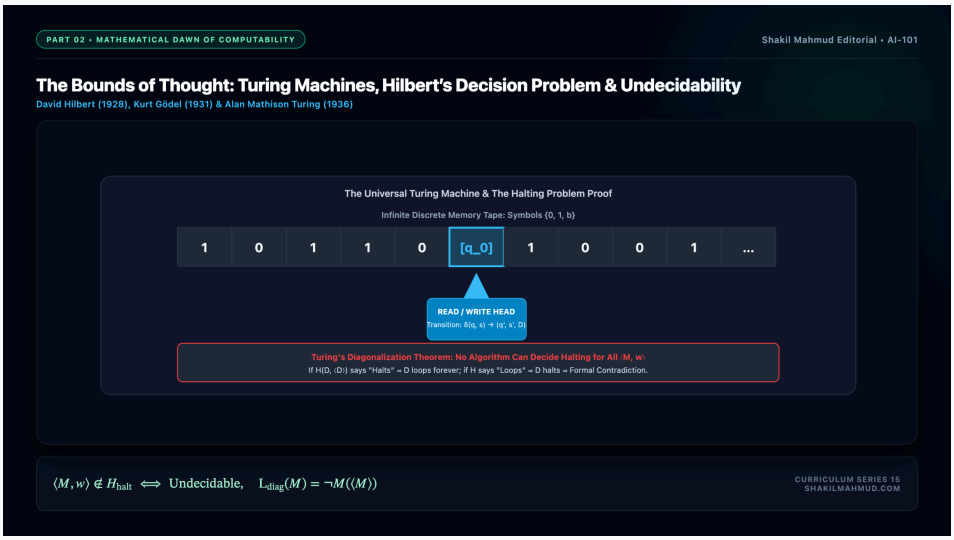


Plate II: *The Universal Turing Machine discrete tape architecture and the diagonal argument establishing the undecidability of the Halting Problem.*

At the mathematical core of deep neural network training lies **Automatic Differentiation (AD)**—specifically, reverse-mode accumulation, universally celebrated in machine learning as the **Backpropagation algorithm**. Unlike numerical differentiation (finite differences, which scale with prohibitive cost $O(D)$) or symbolic differentiation (which suffers from exponential combinatorial expression swell), reverse-mode AD computes the exact gradient of an arbitrary scalar loss function with respect to millions of parameters in time proportional to a single forward pass: $O(1)$ relative complexity.

1. The Computational Directed Acyclic Graph (DAG)

Any composite function $y = f(x_1, dots, x_d)$ computed on a digital computer can be decomposed into an ordered sequence of elementary operations (addition, multiplication, transcendental functions) represented as a Directed Acyclic Graph $mathcalG = (mathcalV, mathcalE)$.

Let nodes $v_i inmathcalV$ be ordered topologically for $i = 1, dots, N$ where:

- $v_1, dots, v_d$ represent input features and model weights.
- $v_i = phi_i left(textParents(v_i)right)$ for i > d computes an elementary mathematical primitive.
- $v_N = L$ represents the final scalar objective (loss).

2. Forward vs Reverse Mode: Dimensional Duality

Consider a general mapping $f : mathbbR^n tomathbbR^m$. The Jacobian matrix $J inmathbbR^mtimesn$ contains all first-order partial derivatives $J_ij = fracpartial f_ipartial x_j$.

- **Forward-Mode AD:** Computes the Jacobian-Vector Product (JVP) $Jcdotv$ for a seed vector $vinmathbbR^n$ along with the primal computation. Evaluating the full Jacobian requires n independent sweeps. Forward mode is optimal when $nllm$.
- **Reverse-Mode AD:** Computes the Vector-Jacobian Product (VJP) $u^T cdotJ$ for adjoint seed $uinmathbbR^m$. Evaluating the full Jacobian requires m backward sweeps. When computing the gradient of a scalar loss function, $m = 1$ and $n = Dgg10^7$. Hence, reverse-mode AD computes the entire D-dimensional gradient $nabla_thetaL$ in exactly **one backward sweep**!

3. Adjoint Derivation of Backpropagation

Define the **Adjoint state** of node v_i as the sensitivity of the final scalar objective L with respect to that node: $barv_iequiv fracpartial Lpartial v_i$.

By the multivariate chain rule across all children nodes consuming v_i as an argument:

Shakil Mahmud • Research Monograph

$$barv_i = sum_{jintextChildren(v_i)} fracpartialLpartialv_j fracpartialv_jpartialv_i = sum_{jintextChildren(v_i)} barv_j fracpartialphi_jpartialv_i$$

Reverse-mode AD proceeds in two synchronized phases:

1. **The Forward Primal Pass:** Traverse $mathcalG$ in topological order $i = 1toN$, storing all intermediate activations v_i required for downstream local derivative evaluations.
2. **The Backward Adjoint Pass:** Seed the terminal adjoint $barv_N = barL = fracpartialLpartialL = 1.0$. Traverse $mathcalG$ in reverse topological order $i = Nto1$, accumulating adjoint contributions $barv_i$ into parameter nodes.

4. Memory-Compute Tradeoffs: Gradient Checkpointing

While reverse-mode AD achieves optimal time complexity, its spatial footprint is proportional to the number of graph operations: $O(N)$ activations must reside in GPU VRAM simultaneously during the forward pass. For 100-billion-parameter models, activation memory easily surpasses available HBM capacity.

In 2016, Chen et al. introduced **Gradient Checkpointing** (Activation Rematerialization): by storing activations only at selected checkpoint boundaries $sqrtN$ and recomputing intermediate forward graphs on-the-fly during the backward pass, memory consumption drops from $O(N)$ to $O(sqrtN)$ at the cost of a modest 33 increase in compute overhead.

Citation: Mahmud, S. (2026). "Backpropagation & Reverse-Mode Automatic Differentiation — Dual Graph Gradients." In Deep Learning Architectures & Representation Theory, Part 2.
https://shakilmahmud.com/deep-learning-architectures-part-02-backpropagation-automatic-differentiation-computational-graphs/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART III

Normalization & Optimization Dynamics — BatchNorm, LayerNorm, RMSNorm & AdamW

Normalization & Optimization Dynamics — BatchNorm, LayerNorm, RMSNorm & AdamW

569 words • November 4, 2026 • Permanent URI

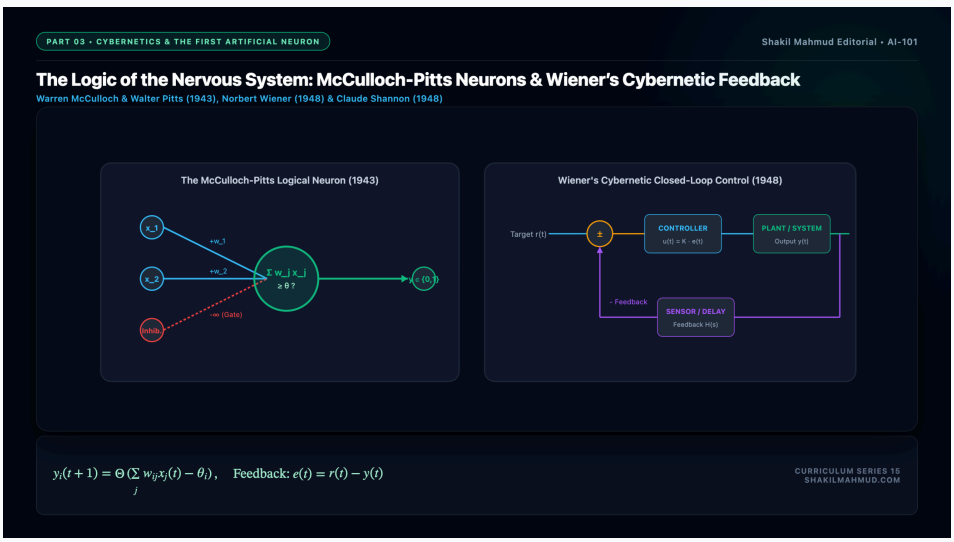


Plate III: *McCulloch-Pitts binary threshold neuron computing propositional logic, paired with Norbert Wiener's cybernetic negative feedback control loop.*

Training deep neural architectures with tens or hundreds of stacked non-linear layers was historically plagued by severe numerical instability: exploding or vanishing activations, erratic gradient variance, and extreme sensitivity to initialization. The modern deep learning renaissance became viable only after the discovery of structural **Normalization layers** and curvature-adaptive optimizers like **AdamW**. Together, these techniques transform jagged, ill-conditioned optimization bowls into smooth, isotropic geometric manifolds.

1. Batch Normalization: Landscape Smoothing Over Covariate Shift

In 2015, Sergey Ioffe and Christian Szegedy introduced **Batch Normalization (BatchNorm)**, postulating that normalizing layer inputs mitigates *Internal Covariate Shift*. For a mini-batch $mathcal B = x_1, dots, x_m$ of activation vectors, BatchNorm standardizes each channel independently:

$$mu_{mathcal{B}} = frac{1}{m} sum_{i=1}^m x_i, quad sigma_{mathcal{B}}^2 = frac{1}{m} sum_{i=1}^m (x_i – mu_{mathcal{B}})^2$$

$$hat{x}_i = frac{x_i – mu_{mathcal{B}}}{sqrt{sigma_{mathcal{B}}^2 + epsilon}}, quad y_i = gamma hat{x}_i + beta$$

where $gamma$ and $beta$ are learnable affine parameters restoring representational capacity.

In 2018, Santurkar et al. proved that BatchNorm's primary benefit is *not* eliminating covariate shift, but fundamentally reshaping the optimization geometry: BatchNorm strictly bounds the Lipschitz constant of the loss and the Lipschitz gradient constant $beta$ ($nabla^2 Lpreceqbeta I$), dramatically compressing the condition number of the Hessian and enabling substantially larger learning rates.

2. LayerNorm and RMSNorm: Sequence Independence

While BatchNorm thrives in convolutional computer vision, it fails catastrophically in recurrent and autoregressive sequence models due to variable sequence lengths and small batch sizes during inference.

- **Layer Normalization (LayerNorm, Ba, Kiros, & Hinton, 2016):** Normalizes across the feature/channel dimension D for each individual instance i independently of other batch elements:

$$mu = frac{1}{D}sum_{j=1}^D x_j, quad sigma^2 = frac{1}{D}sum_{j=1}^D (x_j – mu)^2, quad hat{x} = frac{x – mu}{sqrt{sigma^2 + epsilon}} odot gamma + beta$$

- **Root Mean Square Normalization (RMSNorm, Zhang & Sennrich, 2019):** Hypothesizes that the variance/scaling property drives the benefits of LayerNorm, rendering mean-centering computationally redundant. RMSNorm scales activations by root mean square energy:

Shakil Mahmud • Research Monograph

$$textRMS(x) = sqrtfrac1Dsum_{j=1}^{D} x_j^2 + epsilon, quad barx = fracxtextRMS(x)odotgamma$$

RMSNorm reduces per-layer wall-clock latency by 10 and has become the universal standard in frontier LLMs (LLaMA, Mistral, Gemma).

3. Pre-LN vs Post-LN Stability in Transformers

The topological placement of normalization relative to residual branches determines gradient flow stability:

- **Post-LN (Original Transformer):** $x_{l+1} = textLayerNorm(x_l + textSubLayer(x_l))$. Gradients through unnormalized residual connections decay exponentially toward the early layers, necessitating delicate learning rate warmup schedules to prevent early divergence.
- **Pre-LN:** $x_{l+1} = x_l + textSubLayer(textLayerNorm(x_l))$. Normalization is applied inside the sublayer branch while the identity residual stream remains pure. Gradients propagate unimpeded across arbitrary depth without warmup, establishing stable convergence for modern scale.

4. AdamW: Decoupling Weight Decay from Preconditioning

The classical **Adam** optimizer (Kingma & Ba, 2014) maintains exponentially decaying first (m_t) and second (v_t) uncentered moment vectors:

m_t = beta_1 m_{t-1} + (1 – beta_1) g_t, quad v_t = beta_2 v_{t-1} + (1 – beta_2) g_t^2

In standard implementations, L2 regularization was appended directly to the loss: $L_{textreg}(theta) = L(theta) + frac12lambda|theta|^2$. Consequently, the weight decay gradient was scaled inversely by the preconditioner $sqrtv_t$. In 2017, Ilya Loshchilov and Frank Hutter demonstrated that this corrupts regularization: weights with massive historical gradients receive vanishingly small weight decay, whereas dormant weights are penalized excessively.

AdamW decouples weight decay completely, performing pure Euclidean weight contraction after gradient steps:

theta_{t+1} = theta_t – eta left(frac{hat{m}_t}{sqrt{hat{v}_t} + epsilon} + lambda theta_t right)

restoring proper scale invariance and establishing state-of-the-art generalization across modern deep learning.

Citation: Mahmud, S. (2026). "Normalization & Optimization Dynamics — BatchNorm, LayerNorm, RMSNorm & AdamW." In Deep Learning Architectures & Representation Theory, Part 3.
https://shakilmahmud.com/deep-learning-architectures-part-03-normalization-batchnorm-layernorm-rmsnorm-adamw/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART IV

Convolutional Equivariance & Group Theory — Lie Symmetries & Weight Sharing

Convolutional Equivariance & Group Theory — Lie Symmetries & Weight Sharing

517 words • November 4, 2026 • Permanent URI

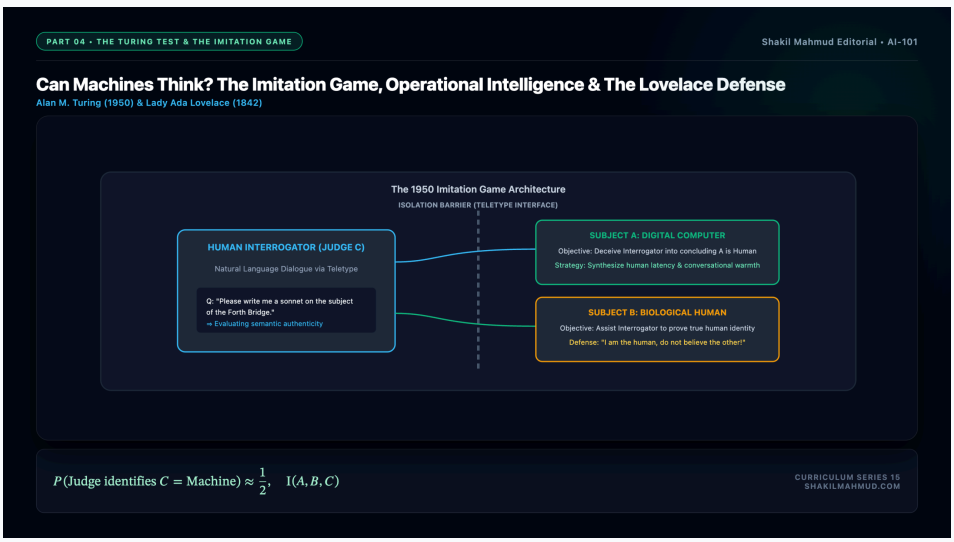


Plate IV: *Alan Turing's 1950 Imitation Game tripartite architecture, evaluating machine intelligence through blind teletype interrogation.*

In 1872, Felix Klein delivered his celebrated *Erlangen Program*, proposing that geometry is not merely the study of metric distances, but the study of invariant properties under transformation groups. In modern representation theory, this insight forms the mathematical foundation of **Convolutional Neural Networks (CNNs)**. By exploiting the symmetry group of 2D Euclidean translations $(mathbbR^2, +)$, convolutions enforce **Equivariance** through weight sharing, slashing parameter complexity from $O(D^2)$ in fully-connected layers to $O(K^2)$ and bestowing exact geometric inductive bias.

1. The Formal Definition of Group Equivariance

Let G be a transformation group acting on input space $mathcalX$ via group action $T_g : mathcalXtomathcalX$ for $ginG$, and acting on feature space $mathcalY$ via group action T_g’: mathcal{Y} to mathcal{Y}. A mathematical mapping $f : mathcalXtomathcalY$ is said to be G-**Equivariant** if the diagram commutes for all $ginG$:

f(T_g x) = T_g’ (f(x)), quad forall x in mathcal{X}, ; forall g in G

In contrast, f is G-**Invariant** if the representation is entirely unaltered by the group action: $f(T_g x) = f(x)$. In hierarchical deep visual architectures, early and intermediate feature extractors must be *equivariant* (tracking the exact spatial translation of eyes, noses, and edges), while the terminal classification head collapses features into an *invariant* representation (predicting "face" regardless of spatial coordinates).

2. Translation Equivariance and Continuous Convolution

Consider the continuous translation group $(mathbbR^d, +)$ where translation operator T_t shifts functions by $tinmathbbR^d$: (T_t f)(x) = f(x – t).

The continuous convolution of feature map f with spatial filter kernel w is defined as:

(f * w)(x) = int_{mathbb{R}^d} f(y) w(x – y) dy

Applying the translation operator to the input function:

(T_t f * w)(x) = int_{mathbb{R}^d} f(y – t) w(x – y) dy

Substituting integration variable u = y – t (where $dy = du$):

int_{mathbb{R}^d} f(u) w(x – t – u) du = (f * w)(x – t) = T_t (f * w)(x)

Shakil Mahmud • Research Monograph

By the fundamental theorem of linear systems, *any linear operator that is equivariant to continuous translation must be a convolution operator*. Weight sharing is not merely a pragmatic engineering heuristic; it is the unique mathematical consequence of translation symmetry.

3. Beyond 2D Translations: Group-Equivariant CNNs (G-CNNs)

In 2016, Taco Cohen and Max Welling generalized convolutions to arbitrary non-commutative Lie groups G (such as the roto-translation group $SE(2) = mathbbR^2rtimesSO(2)$ or the discrete p4 dihedral group of 90^circ rotations and reflections). For functions $f : GtomathbbR$ defined over group manifolds, group convolution is defined via the Haar measure dg:

$$(f *_G w)(g) = int_G f(h)w(g^{-1}h)dh$$

By lifting planar images into group representations, G-CNNs guarantee exact rotation equivariance, preventing models from wasting capacity memorizing tilted or mirrored orientations.

4. The Fourier Perspective and the Convolution Theorem

Under the continuous Fourier transform $mathcalFf(omega) = hatf(omega)$, the convolution theorem establishes that spatial convolution is dual to point-wise frequency multiplication:

$$mathcalFf * w = mathcalFfcdotmathcalFw$$

This duality reveals that convolutional layers act as learnable spatial filter banks decomposing representations into harmonic frequencies, providing the theoretical bridge between group representation theory and harmonic analysis.

Citation: Mahmud, S. (2026). "Convolutional Equivariance & Group Theory — Lie Symmetries & Weight Sharing." In Deep Learning Architectures & Representation Theory, Part 4. https://shakilmahmud.com/deep-learning-architectures-part-04-convolutional-equivariance-group-theory-lie-symmetries/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART V

Residual Networks & Identity Mappings — Deep Degradation & Unraveled Ensembles

Residual Networks & Identity Mappings — Deep Degradation & Unraveled Ensembles

595 words • November 4, 2026 • Permanent URI

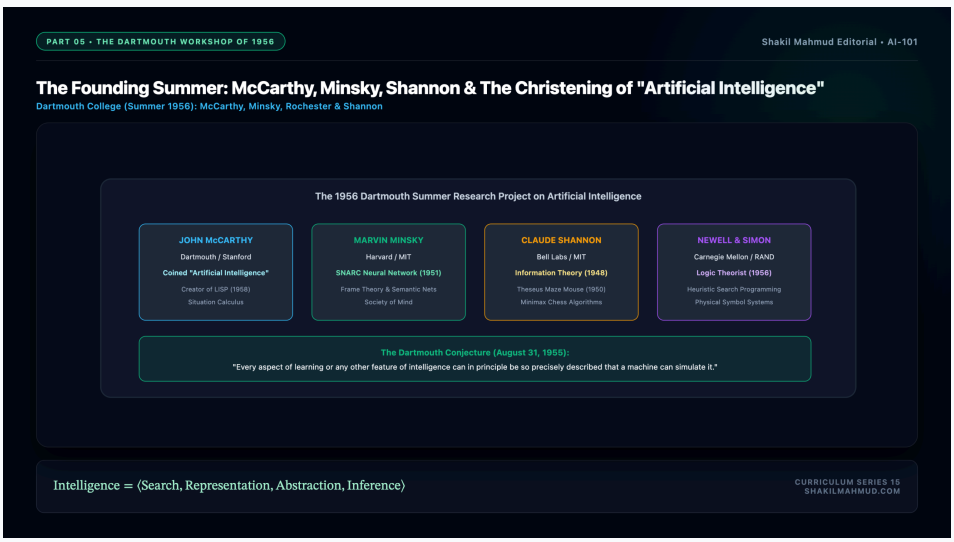


Plate V: *The 1956 Dartmouth Summer Research Project on Artificial Intelligence: McCarthy, Minsky, Shannon, and Newell-Simon founding the discipline.*

Prior to 2015, the deep learning community confronted an unexpected empirical crisis: the **Degradation Problem**. As networks grew beyond 20 layers, training error—not just test error—began to increase dramatically. Crucially, this was not caused by overfitting (which would exhibit lower training error alongside higher test error), but by an optimization failure: deeper architectures failed to learn even an identity mapping! In late 2015, Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun resolved this barrier with **Deep Residual Learning (ResNet)**, enabling the stable optimization of networks with 152 to over 1,000 layers.

1. The Degradation Paradox and Residual Formulations

Consider a network with L stacked non-linear layers. Suppose there exists an optimal shallow mapping $mathcal H^*(x)$ at depth l < L. In theory, if subsequent layers $l + 1 to L$ simply learn the identity function $mathcal H(x) = x$, the deeper network must achieve training error no worse than the shallow model. Yet empirically, non-linear stacks of layers $sigma(Wx + b)$ struggle to approximate identity mappings due to the curvature of activation manifolds.

He et al. hypothesized that instead of forcing stacked layers to fit the underlying mapping $mathcal H(x)$, they should explicitly fit a **Residual Mapping**:

mathcal{F}(x) equiv mathcal{H}(x) – x implies mathcal{H}(x) = mathcal{F}(x) + x

If an identity mapping is optimal, the optimization dynamics simply drive weights toward zero ($mathcal F(x) to 0$), which is trivial for standard weight decay regularizers.

2. Mathematical Derivation of Unimpeded Gradient Flow

Consider a chain of residual blocks where each block computes $x_{l+1} = x_l + mathcal F(x_l, mathcal W_l)$. Recursively expanding this recurrence from layer l to arbitrary deeper layer L > l yields the fundamental identity:

$$x_L = x_l + sum_{i=l}^{L-1} mathcal F(x_i, mathcal W_i)$$

This formulation exhibits profound properties: any deep representation x_L is the linear sum of the early representation x_l plus an additive aggregation of residual corrections. Applying the chain rule to evaluate the gradient of scalar loss $mathcal E$ with respect to activation x_l:

$$frac partial mathcal E partial x_l = frac partial mathcal E partial x_L frac partial x_L partial x_l = frac partial mathcal E partial x_L left(I + frac partial partial$$

Shakil Mahmud • Research Monograph

This gradient decomposition reveals why vanishing gradients are eliminated: the term I acts as an unattenuated **Gradient Highway**. Even if the Jacobian term $fracpartialpartialx_lsum_imathcalF_i$ vanishes or fluctuates chaotically, the identity term guarantees that $fracpartialmathcalEpartialx_L$ propagates directly back to layer l without exponential decay.

3. The Unraveled Ensemble Interpretation

In 2016, Andreas Veit, Michael Wilber, and Serge Belongie proved that ResNets do not behave like a single monolithic ultra-deep network; rather, they behave like an **Unraveled Ensemble of Exponentially Many Shallow Networks**.

A ResNet with n residual blocks contains 2^n distinct computational paths from input to output. Veit et al. showed that dropping individual layers during inference (or training via Stochastic Depth) barely affects performance, whereas deleting a single layer in a plain VGG network completely obliterates performance. Furthermore, gradient norm analyses demonstrate that the effective path lengths contributing to gradient updates remain around $O(sqrtn)$ to $O(n/2)$, establishing ResNets as parallel implicit ensembles.

4. Modern Descendants: DenseNet, ResNeXt, and ConvNeXt

The residual paradigm spawned a lineage of foundational visual architectures:

- **DenseNet (Huang et al., 2017):** Connects all layers directly ($x_l = H_l([x_0, x_1, dots, x_l-1])$), maximizing feature reuse via concatenation.
- **ResNeXt (Xie et al., 2017):** Introduces *Cardinality*—splitting residual transformations into multi-branch grouped convolutions, demonstrating that topological cardinality is more effective than sheer depth or width.
- **ConvNeXt (Liu et al., 2022):** Modernizes pure convolutional networks with $7times7$ depthwise convolutions, inverted bottlenecks, and LayerNorm, proving that classical convolutions match vision transformers when architected under modern residual principles.

Citation: Mahmud, S. (2026). "Residual Networks & Identity Mappings — Deep Degradation & Unraveled Ensembles." In Deep Learning Architectures & Representation Theory, Part 5.
https://shakilmahmud.com/deep-learning-architectures-part-05-residual-networks-identity-mappings-gradient-highways/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART VI

Recurrent Sequence Dynamics — Vanishing Gradients, LSTM Cells & Gated Memory

Recurrent Sequence Dynamics — Vanishing Gradients, LSTM Cells & Gated Memory

544 words • November 4, 2026 • Permanent URI

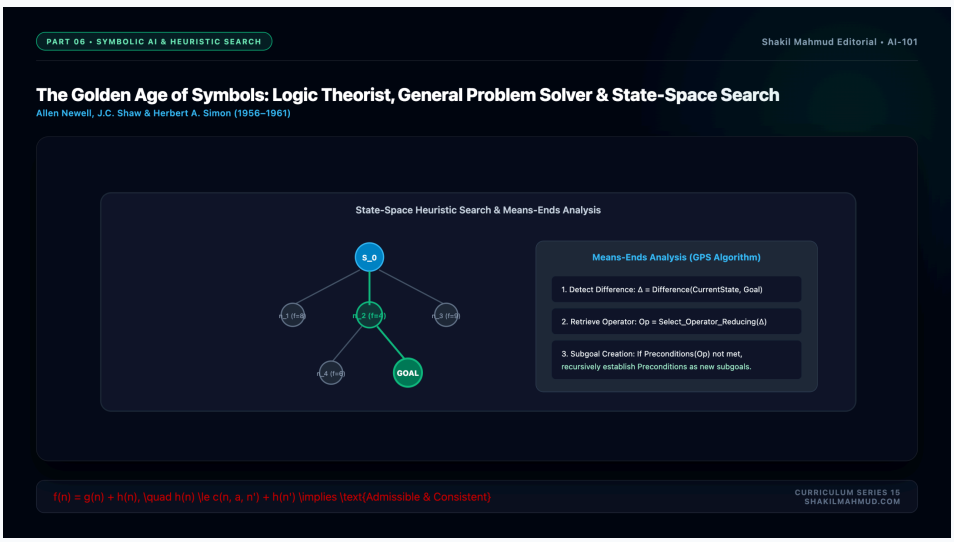


Plate VI: *State-space heuristic search tree evaluation and Newell-Simon's Means-Ends Analysis in the General Problem Solver (GPS).*

Sequential temporal processes—speech, natural language, financial time series, and biophysical kinetics—defy static feed-forward mappings: their outputs depend critically on an unbounded history of past observations. **Recurrent Neural Networks (RNNs)** formalize this sequence dependence by maintaining a latent dynamical state $h_t inmathbb R^d$. However, standard RNNs historically suffered from the *Vanishing and Exploding Gradient Problem*, which bounded temporal credit assignment to a handful of steps. In 1997, Sepp Hochreiter and Jürgen Schmidhuber solved this with the **Long Short-Term Memory (LSTM)** architecture.

1. The Mathematics of Temporal Unrolling and BPTT

A standard Elman recurrent network evolves according to the non-linear recurrence:

$$h_t = sigma(W_{hh} h_{t-1} + W_{xh} x_t + b_h)$$

To train parameters using gradient descent, the network is unrolled across T discrete time steps—a procedure known as **Backpropagation Through Time (BPTT)**. Evaluating the gradient of the loss at step T with respect to parameters W_{hh} requires applying the chain rule across the recurrent sequence:

$$fracpartial L_T partial W_{hh} = sum_{t=1}^{T} fracpartial L_T partial h_T fracpartial h_T partial h_t fracpartial h_t partial W_{hh}$$

The temporal credit assignment bottleneck resides in the temporal Jacobian product:

$$fracpartial h_T partial h_t = prod_{k=t+1}^{T} fracpartial h_k partial h_{k-1} = prod_{k=t+1}^{T} textdiag(sigma'(z_k)) W_{hh}^T$$

2. Bengio's Spectral Radius Theorem

In 1994, Yoshua Bengio et al. proved the fundamental limitation of recurrent credit assignment: let $rho(W_{hh})$ be the spectral radius (largest absolute eigenvalue) of weight matrix W_{hh}, and let $|sigma'(z)| legamma$. The norm of the Jacobian product is bounded by:

left| frac{partial h_T}{partial h_t} right| le (gamma rho(W_{hh}))^{T – t}

Shakil Mahmud • Research Monograph

- **Vanishing Gradient Regime (gamma rho < 1):** The gradient norm decays exponentially with temporal distance Delta t = T – t. For T – t ge 50, gradients shrink below floating-point precision (10^{-16}), rendering the model incapable of learning long-range dependencies.
- **Exploding Gradient Regime (gamma rho > 1):** Gradients explode exponentially toward numerical infinity ($textNaN$), causing destructive parameter divergence (mitigated heuristically via norm gradient clipping $|nabla|letau$).

3. The Long Short-Term Memory (LSTM) Solution

Hochreiter and Schmidhuber resolved this dilemma by introducing the **Constant Error Carousel (CEC)**: enforcing a linear self-connected unit where the Jacobian equals the identity matrix ($fracpartialc_tpartialc_{t-1} = I$).

Modern LSTMs (enhanced with Felix Gers's 2000 forget gate) govern internal information routing via four interacting affine gates:

$$beginbmatrixf_t \ i_t \ o_t \ tildec_tendbmatrix = beginbmatrixsigma \ sigma \ sigma \ tanhendbmatrixleft(Wbeginbmatrixh_{t-1} \ x_tendbmatrix + bright)$$

The internal cell state updates additively:

$$c_t = f_todotc_{t-1} + i_todottildec_t$$

$$h_t = o_todottanh(c_t)$$

Because the update to cell state c_t is purely additive, the gradient $fracpartialc_Tpartialc_t = prod_{k=t+1}^{T} f_k$ does not undergo repeated matrix multiplications with weight matrices. When forget gate $f_kapprox1$, error signals propagate across hundreds of steps without attenuation.

4. The Gated Recurrent Unit (GRU)

In 2014, Kyunghyun Cho et al. streamlined the LSTM into the **Gated Recurrent Unit (GRU)**, eliminating the separate cell state c_t and coupling the forget and input gates into a single update gate z_t:

$$z_t = sigma(W_z[h_{t-1}, x_t]), quadr_t = sigma(W_r[h_{t-1}, x_t])$$

h_t = (1 – z_t) odot h_{t-1} + z_t odot tilde{h}_t

GRUs reduce parameter count by 25 while matching LSTM expressivity, establishing the standard sequential recurrent baseline prior to the Transformer revolution.

Citation: Mahmud, S. (2026). "Recurrent Sequence Dynamics — Vanishing Gradients, LSTM Cells & Gated Memory." In Deep Learning Architectures & Representation Theory, Part 6. https://shakilmahmud.com/deep-learning-architectures-part-06-recurrent-networks-lstm-gru-vanishing-gradients/

SHAKIL MAHMUD RESEARCH MONOGRAPHS · PART VII

Attention Mechanisms & Query-Key-Value Duality — From Bahdanau to Scaled Dot-Product

Attention Mechanisms & Query-Key-Value Duality — From Bahdanau to Scaled Dot-Product

509 words • November 4, 2026 • Permanent URI

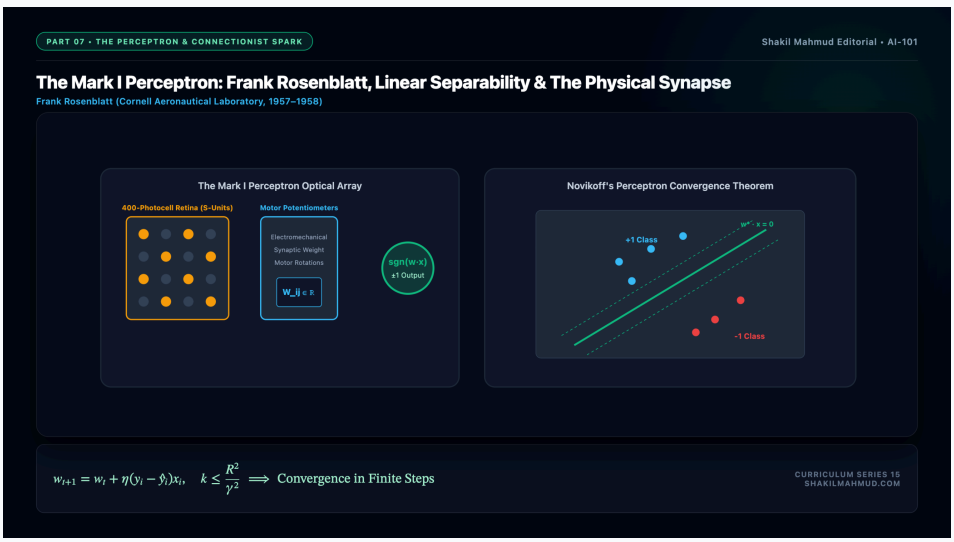


Plate VII: *Frank Rosenblatt's Mark I Perceptron 400-photocell retina, motor potentiometer weights, and Novikoff's convergence hyperplane.*

In classical sequence-to-sequence encoder-decoder architectures, the encoder is forced to compress an entire source sentence of arbitrary length into a single fixed-dimensional bottleneck vector $cinmathbbR^d$. For long inputs, information is irrevocably destroyed. In 2014, Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio shattered this compression limit by inventing the **Attention Mechanism**. By treating sequence representation as dynamic associative memory addressing, attention liberated neural networks from fixed bottlenecks, paving the way for the foundation models of modern AI.

1. The Encoder Bottleneck Crisis

In classical neural machine translation (Sutskever et al., 2014), the encoder computes hidden states $(h_1, dots, h_{T_x})$. The context vector passed to the decoder is clamped to the final state: $c = h_{T_x}$. As sentence length T_x increases beyond 20 words, BLEU translation scores collapse precipitously because a finite vector cannot preserve the Shannon entropy of complex multi-clause semantics.

2. Bahdanau Additive Attention: Dynamic Context Synthesis

Bahdanau et al. proposed that the context vector c_i for decoding step i should be dynamically synthesized as a weighted sum of all encoder hidden states:

$$c_i = sum_{j=1}^{T_x} alpha_{ij} h_j$$

The alignment weight $alpha_{ij}$ quantifies how strongly decoder target state s_{i-1} aligns with source state h_j:

$$alpha_{ij} = fracexp(e_{ij})sum_{k=1}^{T_x} exp(e_{ik}), quade_{ij} = v_a^T tanh(W_a s_{i-1} + U_a h_j)$$

where W_a, U_a, v_a are learned projection parameters. By backpropagating through alignment energies, the network learns a soft, differentiable bilingual alignment matrix without human supervision.

3. Query-Key-Value (QKV) Duality as Associative Memory

Vaswani et al. (2017) formalized attention under the computational lens of **Differentiable Key-Value Store Addressing**:

- **Query ($QinmathbbR^{ntimesd_k}$):** The current operational question or representation seeking information.

Shakil Mahmud • Research Monograph

- **Key ($Kinmathbb{R}^{mtimesd_k}$):** The indexing addresses describing what information is stored in memory.
- **Value ($Vinmathbb{R}^{mtimesd_v}$):** The actual substantive payload or content to be retrieved.

In standard database retrieval, a query matches a unique key, returning a single value: $V[textKey = Q]$. The Attention Mechanism relaxes this into a soft, differentiable probability distribution over all stored values:

$$textAttention(Q, K, V) = A(Q, K)cdotV, quadA_{ij} = mathbb{P}(textRouteQueryitotextValuej)$$

4. Scaled Dot-Product Attention: Variance and Temperature Scaling

Instead of Bahdanau's computationally heavy additive formulation, modern architectures employ the matrix dot-product $S = QK^Tinmathbb{R}^{ntimesm}$, which executes on tensor cores with massive parallel hardware efficiency.

However, dot products grow with vector dimension d_k: assuming components of q and k are independent random variables with zero mean and unit variance, their inner product has variance:

$$mathbb{E}[q^Tk] = 0, quadtextVar(q^Tk) = sum_{i=1}^{d_k}textVar(q_ik_i) = d_k$$

When d_k is large (e.g., $d_k = 128$), magnitudes grow into large values ($|q^Tk|approxsqrt{128}approx11.3$). Passing unscaled scores into the $textsoftmax$ function pushes probabilities into extreme saturations (\$0\$ or \$1\$), driving gradients $fracpartialtextsoftmaxpartialzto0$. Dividing by the scaling factor $sqrtd_k$ preserves unit variance and restores healthy gradient propagation:

$$textAttention(Q, K, V) = textsoftmaxleft(fracQK^Tsqrtd_kright)V$$

This formulation serves as the singular foundational atom of the modern Transformer.

Citation: Mahmud, S. (2026). "Attention Mechanisms & Query-Key-Value Duality — From Bahdanau to Scaled Dot-Product." In Deep Learning Architectures & Representation Theory, Part 7.
https://shakilmahmud.com/deep-learning-architectures-part-07-attention-mechanisms-query-key-value-duality/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART VIII

The Transformer Architecture — Multi-Head Attention, Positional Encoding & Residual Streams

The Transformer Architecture — Multi-Head Attention, Positional Encoding & Residual Streams

483 words • November 4, 2026 • Permanent URI

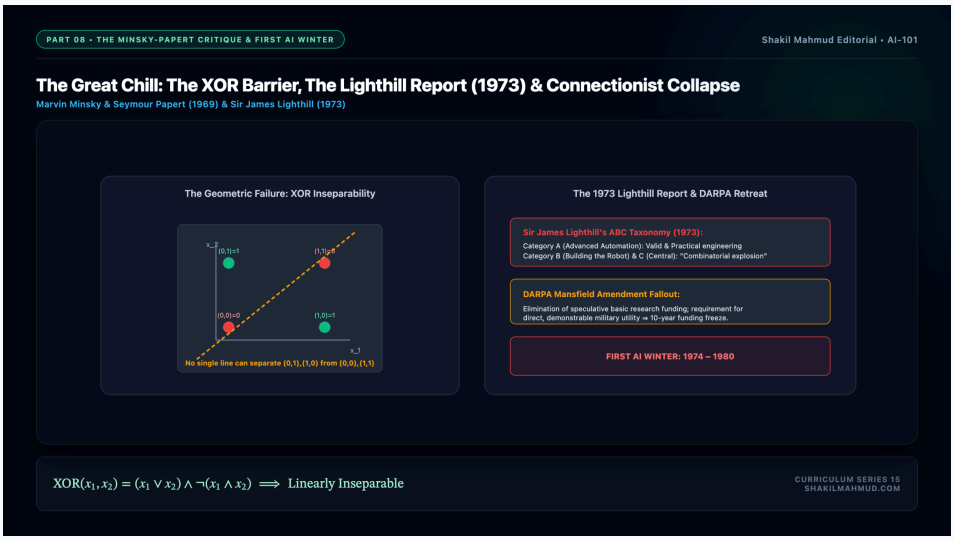


Plate VIII: *The XOR linear inseparability barrier proved by Minsky and Papert, precipitating the 1973 Lighthill Report and the First AI Winter.*

In June 2017, Ashish Vaswani et al. published *"Attention Is All You Need"*, unveiling the **Transformer architecture**. By completely banishing recurrence and convolutions in favor of self-attention, the Transformer removed the sequential temporal bottleneck of RNNs, unlocking fully parallel GPU hardware saturation and $O(1)$ maximum path lengths between arbitrary tokens. The Transformer is now the universal computational engine spanning natural language processing, computer vision, structural biology, and frontier AI.

1. Multi-Head Attention (MHA): Orthogonal Representation Subspaces

A single self-attention head averages over all semantic relationships simultaneously. However, language demands that a word attend concurrently to multiple disparate aspects: syntactic dependency, coreference resolution, and lexical semantic context. **Multi-Head Attention** projects queries, keys, and values into h distinct lower-dimensional subspaces of dimension $d_k = d_{textmodel}/h$:

$$texthead_i = textAttention(QW_i^Q, KW_i^K, VW_i^V)$$

$$textMultiHead(Q,K,V) = textConcat(texthead_1, dots, texthead_h)W^O$$

where $W_i^Q, W_i^K inmathbbR^{d_{textmodel}timesd_k}$, $W_i^V inmathbbR^{d_{textmodel}timesd_v}$, and $W^O inmathbbR^{hd_vtimesd_{textmodel}}$.

By computing attention independently across orthogonal linear projections, heads specialize into functional modules: some heads track previous tokens, others track syntactic verbs, and others attend across long distances.

2. Sinusoidal Positional Encoding: Breaking Permutation Invariance

Self-attention is mathematically **Permutation Equivariant**: if the rows of input sequence X are permuted by permutation matrix P, the output is identically permuted: $textAttention(PX) = PtextAttention(X)$. To make the model cognizant of sequential word order, Vaswani et al. injected static geometric vectors:

$$textPE_{(pos,2i)} = sinleft(fracpos10000^{2i/d_{textmodel}}right), quadtextPE_{(pos,2i+1)} = cosleft(fracpos10000^{2i/d_{textmodel}}right)$$

By the trigonometric angle addition formulas, any relative displacement $pos + k$ can be expressed as a linear transformation of $textPE_{pos}$, allowing the network to easily learn relative position offsets.

Shakil Mahmud • Research Monograph

3. Position-Wise Feed-Forward Networks (FFN)

Attention only performs linear combinations of value vectors retrieved across tokens; it cannot synthesize new non-linear features within a token. The **Feed-Forward Network (FFN)** applied to each token independently supplies non-linear capacity:

$$textFFN(x) = sigma(xW_1 + b_1)W_2 + b_2$$

Typically, hidden dimension expands by a factor of 4: $d_{textffn} = 4times d_{textmodel}$. Geva et al. (2021) showed that the first layer W_1 acts as a key-value memory retrieval mechanism matching semantic concepts, while W_2 writes new factual representations into the token's latent state.

4. The Residual Stream as a Shared Communication Bus

In modern mechanistic interpretability (Elhage et al., 2021), a Transformer layer stack is formalized not as a traditional sequential pipeline, but as a persistent **Residual Stream** $mathbf{x} in mathbb{R}^{d_{textmodel}}$.

The residual stream acts as a central communication bus. Attention heads and MLP layers read from the stream, perform their specialized computations, and write their delta updates additively back onto the stream:

$$mathbf{x}_{l+1} = mathbf{x}_l + sum_i texthead_{l,i}(mathbf{x}_l) + textMLP_l(mathbf{x}_l)$$

This additive architecture allows earlier layers to transmit unmodified representations directly to the final layer, while later layers read and enrich features in parallel, providing the theoretical basis for modern transformer scaling.

Citation: Mahmud, S. (2026). "The Transformer Architecture — Multi-Head Attention, Positional Encoding & Residual Streams." In Deep Learning Architectures & Representation Theory, Part 8.
https://shakilmahmud.com/deep-learning-architectures-part-08-transformer-architecture-multihead-attention-residual-stream/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART IX

Rotary Position Embeddings (RoPE) & Context Windows — Complex Rotations & Extrapolation

Rotary Position Embeddings (RoPE) & Context Windows — Complex Rotations & Extrapolation

546 words • November 4, 2026 • Permanent URI

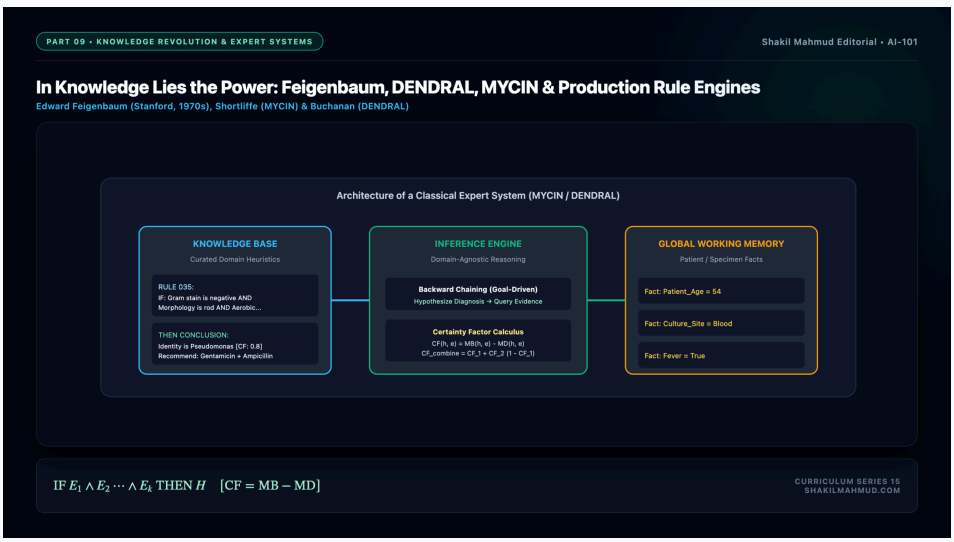


Plate IX: *Classical expert system architecture: Knowledge base rules, backward-chaining inference engine, and certainty factor propagation.*

In the original Transformer, Vaswani et al. encoded sequence order by additively injecting absolute sinusoidal positional vectors directly into token embeddings: $x_i + p_i$. While simple, additive positional encoding severely impairs the model's ability to generalize beyond its training context length. In 2021, Jianlin Su et al. introduced **Rotary Position Embeddings (RoPE)**, establishing a mathematically pristine framework that encodes relative token distance through multiplicative orthogonal rotations in the complex plane.

1. The Theoretical Objective of Relative Positional Encoding

Let $q_m = f_q(x_m, m)$ and $k_n = f_k(x_n, n)$ be the query and key vectors representing tokens x_m and x_n at positions $m, ninmathbbN$. The central mathematical desideratum is that their inner product must depend *solely* on their semantic content and their relative displacement m – n:

$$\langle f_q(x_m, m), f_k(x_n, n) \rangle = g(x_m, x_n, m \&\#8211; n)$$

Additive positional encodings violate this condition because the cross-terms $langlex_m, p_nrangle$ and $langlep_m, p_nrangle$ corrupt relative geometric distances.

2. 2D Complex Rotation Derivation

Su considered the 2D case where vectors are identified as complex numbers $q_m, k_ninmathbbC$. The operational mapping satisfies:

$$f_q(x_m, m) = R_m q = qe^{imtheta}, quad f_k(x_n, n) = R_n k = ke^{intheta}$$

Evaluating their Hermitian inner product:

$$\langle f_q(x_m, m), f_k(x_n, n) \rangle = \text{Re}\left[(q e^{i m \theta}) (k e^{i n \theta})^* \right] = \text{Re}\left[q k^* e^{i (m \&\#8211; n)\theta} \right]$$

Astonishingly, the absolute positions m and n cancel out completely! The geometric interaction between query and key is governed strictly by the relative rotation angle (m – n)theta.

3. Generalization to d-Dimensional Space: Block-Diagonal Rotation

To extend to arbitrary even dimension d, RoPE partitions the vector space into $d/2$ independent 2-dimensional orthogonal subspaces. For each subspace $jin1, dots, d/2$, a distinct angular frequency is assigned:

Shakil Mahmud • Research Monograph

$$theta_j = 10000^{-2(j-1)/d}$$

The global rotary operator is an orthogonal block-diagonal matrix $R^d_{Theta,m}$:

R_{Theta, m}^d = begin{bmatrix} cos mtheta_1 & -sin mtheta_1 & 0 & 0 & dots \ sin mtheta_1 & cos mtheta_1 & 0 & 0 & dots \ 0 & 0 & cos mtheta_2 & -sin mtheta_2 & dots \ 0 & 0 & sin mtheta_2 & cos mtheta_2 & dots \ vdots & vdots & vdots & vdots & ddots end{bmatrix}

In practice, matrix multiplication is avoided: RoPE is evaluated in $O(d)$ time via element-wise vector operations: $R^d_{Theta,m}x = xodotcos(mTheta) + tildexodotsin(mTheta)$ where $tildex = [-x_2, x_1, -x_4, x_3, dots]^T$.

4. Long-Context Extension: NTK-Aware Scaling and YaRN

When extending context windows from $4textk$ to $128textk$ tokens, rotating tokens by unobserved angles m > L_{text{train}} causes high-frequency components to oscillate chaotically.

- **Linear Position Interpolation (PI):** Compresses positions by ratio $kappa = L_{textnew}/L_{texttrain}$: m’ = m / kappa. While low frequencies adapt, high frequencies suffer resolution loss.
- **NTK-Aware Scaling (bloc97, 2023):** Draws on Neural Tangent Kernel theory, recognizing that high-frequency dimensions encode local syntax (which should not be compressed) while low-frequency dimensions encode long-range order. By scaling base frequency b’ = b cdot kappa^{d/(d-2)}, RoPE scales context seamlessly.
- **YaRN (Yet another RoPE extensioN, Peng et al., 2023):** Applies piecewise ramp interpolation across frequency bands alongside temperature scaling, enabling stable extrapolation to $128textk - 1textM$ context windows without fine-tuning catastrophic forgetting.

Citation: Mahmud, S. (2026). "Rotary Position Embeddings (RoPE) & Context Windows — Complex Rotations & Extrapolation." In Deep Learning Architectures & Representation Theory, Part 9.
https://shakilmahmud.com/deep-learning-architectures-part-09-rotary-position-embeddings-rope-context-windows/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART X

Autoencoders & Latent Representations — Manifold Bottlenecks & Disentanglement

Autoencoders & Latent Representations — Manifold Bottlenecks & Disentanglement

473 words • November 4, 2026 • Permanent URI

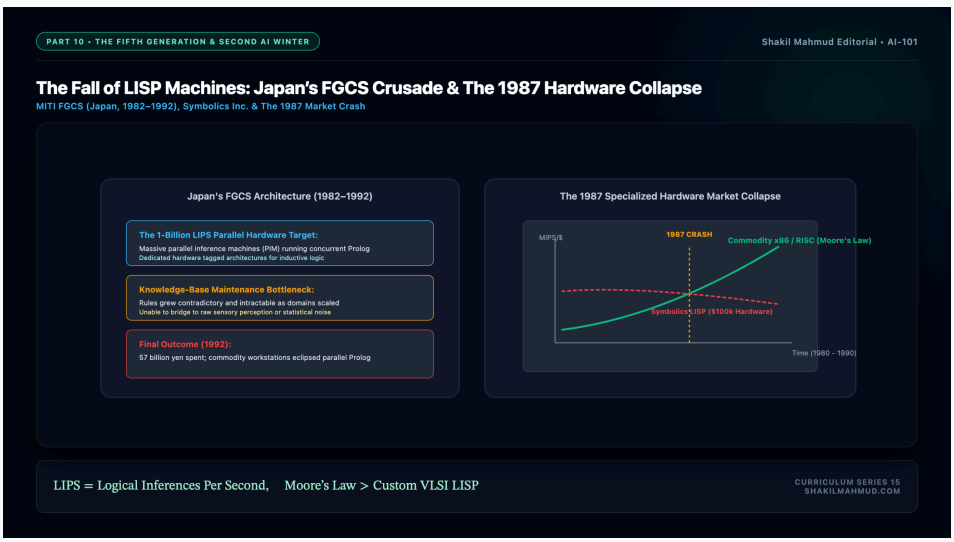


Plate X: *Japan's MITI Fifth Generation Computer Systems parallel Prolog project contrasted with the 1987 LISP machine market collapse.*

In unsupervised representation learning, the fundamental objective is to discover compact, generative coordinates that capture the underlying causal factors of high-dimensional data without external labels. **Autoencoders** realize this by forcing input vectors $xinmathbbR^d$ through a low-dimensional bottleneck $zinmathbbR^k$ ($klld$) before reconstructing $hatx$. By restricting capacity, the network is compelled to discover the intrinsic low-dimensional non-linear sub-manifold on which real-world data concentrates.

1. The Undercomplete Bottleneck Objective

An autoencoder consists of two parameterized mappings:

- **Encoder ($f_phi : mathcalXtomathcalZ$):** Compresses input x into latent code $z = f_phi(x) = sigma(W_ex + b_e)$.
- **Decoder ($g_psi : mathcalZtomathcalX$):** Reconstructs the original input $hatx = g_psi(z) = sigma(W_dz + b_d)$.

Parameters are optimized by minimizing empirical reconstruction distortion under a Bregman divergence or mean squared error:

min_{phi, psi} mathcal{L}_{text{AE}}(phi, psi) = frac{1}{n} sum_{i=1}^n | x_i – g_psi(f_phi(x_i)) |²

If activation functions $sigma$ are purely linear, Bourlard and Kamp (1988) proved that the k-dimensional latent space spans exactly the same subspace as Principal Component Analysis (PCA). Non-linear activations empower autoencoders to learn non-linear coordinate charts for curved Riemannian manifolds.

2. Denoising Autoencoders (DAE): Score Matching on Manifolds

An unconstrained autoencoder with high latent capacity ($kged$) risks learning a trivial identity mapping $g(f(x)) = x$. In 2008, Pascal Vincent et al. introduced the **Denoising Autoencoder (DAE)**: corrupting input x with stochastic noise $tildexsimq(tildexmidx)$ (e.g., Gaussian noise or masking) and training the network to reconstruct the original clean x:

mathcal{L}_{text{DAE}} = mathbb{E}_{x sim p_{text{data}}, tilde{x} sim q(tilde{x}|x)}left[| x – g_psi(f_phi(tilde{x})) |² right]

In 2011, Guillaume Alain and Yoshua Bengio established a profound theoretical equivalence: for small Gaussian noise $tildexsimmathcalN(x, sigma^2I)$, the vector pointing from corrupted input to reconstructed output estimates the score function (the gradient of the data log-density):

g(f(tilde{x})) – tilde{x} approx sigma² nabla_{tilde{x}} log p(tilde{x})

A DAE learns a vector field that points toward the nearest point on the data manifold, directly prefiguring modern continuous diffusion models.

Shakil Mahmud • Research Monograph

3. Contractive Autoencoders (CAE): Robust Representation Flatness

Rifai et al. (2011) formulated the **Contractive Autoencoder (CAE)**, adding an explicit penalty on the Frobenius norm of the encoder Jacobian:

$$mathcal{L}_{text{CAE}} = | x – g(f(x)) |^2 + lambda | J_f(x) |_F^2, quad | J_f(x) |_F^2 = sum_{i,j} left(frac{partial h_i}{partial x_j} right)^2$$

The reconstruction loss pulls the model to preserve information along the tangent bundle of the manifold, while the contractive Frobenius penalty crushes sensitivity in all orthogonal directions. Consequently, the representation remains constant when perturbed off the data manifold.

4. Representation Disentanglement

A representation $z = (z_1, dots, z_k)$ is said to be **disentangled** if independent latent dimensions correspond to isolated generative factors of variation in the world (e.g., azimuth, lighting, color, scale). While deterministic autoencoders achieve compression, they do not enforce coordinate independence, motivating the probabilistic framework of Variational Autoencoders.

Citation: Mahmud, S. (2026). "Autoencoders & Latent Representations — Manifold Bottlenecks & Disentanglement." In Deep Learning Architectures & Representation Theory, Part 10. https://shakilmahmud.com/deep-learning-architectures-part-10-autoencoders-latent-representations-manifold-disentanglement/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART XI

Variational Autoencoders (VAEs) — The Evidence Lower Bound & The Reparameterization Trick

Variational Autoencoders (VAEs) — The Evidence Lower Bound & The Reparameterization Trick

549 words • November 4, 2026 • Permanent URI

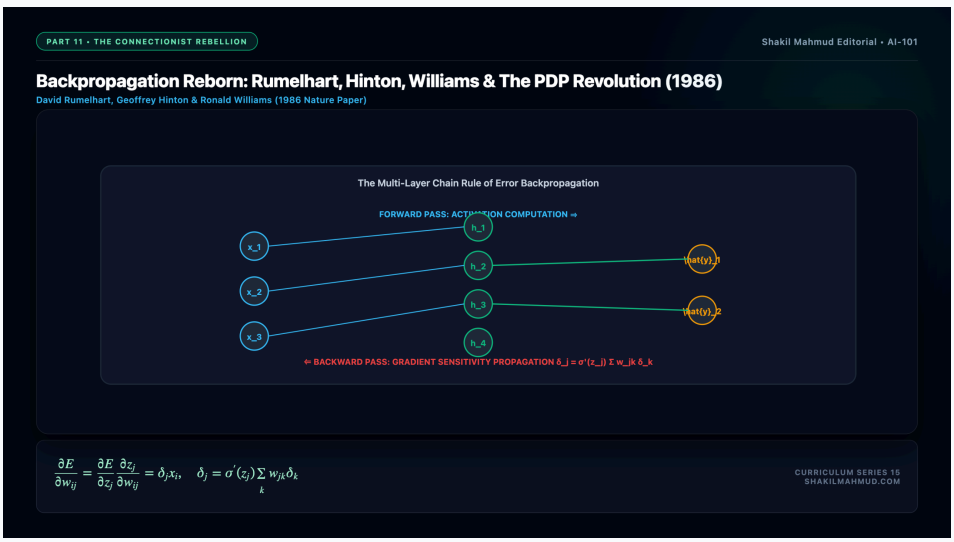


Plate XI: *The multi-layer backpropagation algorithm: Forward activation pass and backward error sensitivity propagation via the multivariable chain rule.*

Standard deterministic autoencoders suffer from a fatal generative flaw: their latent space is plagued by discontinuous, unregularized gaps. Sampling an arbitrary latent point $zsimmathcalN(0, I)$ and passing it through the decoder yields meaningless artifacts. In 2013, Diederik Kingma and Max Welling, alongside Danilo Rezende et al., introduced the **Variational Autoencoder (VAE)**, grounding deep latent representation in principled Bayesian variational inference and solving the stochastic backpropagation barrier via the **Reparameterization Trick**.

1. The Intractable Posterior Problem in Latent Variable Models

Consider a directed probabilistic graphical model where unobserved latent variables $zsimp(z) = mathcalN(0, I)$ generate observed data x according to conditional distribution $p_theta(xmidz)$. We seek to maximize the marginal log-likelihood (the "evidence"):

$$logp_theta(x) = logintp_theta(xmidz)p(z)dz$$

This integral is analytically intractable for non-linear neural networks: evaluating it requires integrating across the entire infinite-dimensional latent manifold. Furthermore, the true posterior $p_theta(zmidx) = fracp_theta(xmidz)p(z)p_theta(x)$ cannot be computed.

2. Derivation of the Evidence Lower Bound (ELBO)

Variational inference introduces a tractable parametric recognition model $q_phi(zmidx)approxp_theta(zmidx)$, parameterized by a neural network encoder. Evaluating the Kullback-Leibler divergence between the approximate and true posterior:

$$D_textKL(q_phi(zmidx)parallelp_theta(zmidx)) = intq_phi(zmidx)logleft(fracq_phi(zmidx)p_theta(zmidx)right)dz$$

Applying Bayes' rule log p_theta(z mid x) = log p_theta(x, z) – log p_theta(x) and rearranging terms yields:

$$logp_theta(x) = D_textKL(q_phi(zmidx)parallelp_theta(zmidx)) + mathcalL_textELBO(theta, phi; x)$$

where the **Evidence Lower Bound (ELBO)** is defined as:

Shakil Mahmud • Research Monograph

mathcal{L}_{text{ELBO}}(theta, phi; x) = mathbb{E}_{q_phi(z mid x)}[log p_theta(x mid z)] – D_{text{KL}}(q_phi(z mid x) parallel p(z))

Since the KL divergence is non-negative ($D_{textKL}ge0$), $mathcalL_{textELBO}lelogp_theta(x)$. Maximizing the ELBO simultaneously pushes the marginal data likelihood up while driving the variational posterior $q_phi(zmidx)$ toward the true posterior.

3. Anatomy of the ELBO Objectives

The ELBO balances two competing thermodynamic forces:

1. **Reconstruction Accuracy ($mathbbE_{q_phi}[logp_theta(xmidz)]$):** Encourages the decoder to reconstruct the specific input x faithfully from sampled latent coordinates.
2. **Latent Regularization ($-D_{textKL}(q_phi(zmidx)parallelp(z))$):** Penalizes deviation of the variational posterior from the standard Gaussian prior $mathcalN(0, I)$, enforcing continuous, smooth, and complete coverage without isolated islands.

Assuming a diagonal Gaussian recognition model $q_phi(zmidx) = mathcalN(mu(x), textdiag(sigma^2(x)))$, the KL term evaluates analytically in closed form:

D_{text{KL}}(q_phi(z mid x) parallel mathcal{N}(0, I)) = -frac{1}{2} sum_{j=1}^d left(1 + log(sigma_j^2) – mu_j^2 – sigma_j^2 right)

4. The Reparameterization Trick: Differentiable Stochasticity

To optimize encoder parameters phi via gradient descent, we must compute $nabla_phimathbbE_{q_phi(zmidx)}[f(z)]$. However, backpropagation cannot pass through stochastic random sampling nodes because the expectation distribution itself depends on phi.

Kingma and Welling solved this through the **Reparameterization Trick**: isolating the stochastic randomness by expressing random variable z as a deterministic, differentiable transformation of an auxiliary noise source $epsilonsimmathcalN(0, I)$:

$$z = g_phi(epsilon, x) = mu(x) + sigma(x)odotepsilon, quadepsilonsimmathcalN(0, I)$$

Now, the expectation is taken over an independent distribution $p(epsilon)$ that does not depend on phi:

$$nabla_phimathbbE_{q_phi(zmidx)}[f(z)] = mathbbE_{epsilonsimmathcalN(0,I)}left[nabla_z f(z)cdotnabla_phig_phi(epsilon, x)right]$$

Monte Carlo gradients flow seamlessly into $mu(x)$ and $sigma(x)$, establishing the foundational cornerstone of deep probabilistic generative modeling.

Citation: Mahmud, S. (2026). "Variational Autoencoders (VAEs) — The Evidence Lower Bound & The Reparameterization Trick." In Deep Learning Architectures & Representation Theory, Part 11.
https://shakilmahmud.com/deep-learning-architectures-part-11-variational-autoencoders-elbo-reparameterization/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART XII

Generative Adversarial Networks (GANs) — Minimax Games & Wasserstein Geometries

Generative Adversarial Networks (GANs) — Minimax Games & Wasserstein Geometries

511 words • November 4, 2026 • Permanent URI



Plate XII: *Yann LeCun's LeNet-5 convolutional neural network architecture: Local receptive fields, weight sharing, and subsampling on MNIST digits.*

In 2014, Ian Goodfellow et al. introduced an entirely new generative philosophy that dispensed with explicit likelihood estimation: **Generative Adversarial Networks (GANs)**. Framed as a zero-sum game between two dueling neural networks—a Generator synthesizing counterfeit artifacts and a Discriminator acting as an art critic—GANs unlocked sharp, photorealistic sample generation. Later, the theoretical integration of optimal transport geometry and **Wasserstein distance** transformed adversarial training from erratic instability into provably monotonic convergence.

1. The Two-Player Zero-Sum Minimax Game

Let $p_{textdata}$ be the true empirical data distribution over $mathbbR^d$, and let p_z be a simple noise prior (e.g., uniform or standard Gaussian) over latent space $mathbbR^k$.

- **Generator ($G_theta : mathcalZtomathcalX$):** Maps noise vectors $zsimp_z$ to generated samples $G(z)$, inducing synthetic distribution p_g.
- **Discriminator ($D_phi : mathcalXto[0,1]$):** Predicts the probability that an input x originates from real data rather than the generator.

The objective function is the minimax value function $V(D,G)$:

min_G max_D V(D, G) = mathbb{E}_{x sim p_{text{data}}}[log D(x)] + mathbb{E}_{z sim p_z}[log(1 – D(G(z)))]

2. Optimal Discriminator and the Jensen-Shannon Divergence

For any fixed generator G, the optimal discriminator $D_G^*(x)$ can be derived analytically by differentiating the objective with respect to $D(x)$:

$$D_G^*(x) = fracp_textdata(x)p_textdata(x) + p_g(x)$$

Substituting D_G^* back into the minimax objective yields the effective loss minimized by the generator:

$$V(D_G^*, G) = -log(4) + 2cdotD_textJS(p_textdataparallelp_g)$$

where D_textJS is the symmetric **Jensen-Shannon Divergence**. The global minimum occurs if and only if $p_g = p_textdata$, where $D_G^*(x) = 1/2$ and $V(D_G^*, G) = -log4$.

Shakil Mahmud • Research Monograph

3. The Pathology of Disjoint Supports: Mode Collapse & Vanishing Gradients

In high dimensions, real data concentrates on lower-dimensional sub-manifolds $mathcalM_textdata$ and $mathcalM_g$ of dimension d^*lld. By differential geometry, two low-dimensional manifolds in high-dimensional space intersect on a set of measure zero.

When supports are disjoint ($textsupp(p_textdata)captextsupp(p_g) = emptyset$):

$$D_textJS(p_textdataparallelp_g) = log2 = textconstant$$

Consequently, the gradient of the Jensen-Shannon divergence vanishes completely: $nabla_thetaD_textJS = 0$. The discriminator achieves 100 accuracy effortlessly, providing zero informative gradient signal to the generator. Conversely, if the generator exploits a single localized mode to deceive the discriminator, **Mode Collapse** occurs, collapsing sample diversity.

4. Wasserstein GAN (WGAN): Optimal Transport & Kantorovich Duality

In 2017, Martin Arjovsky, Soumith Chintala, and Léon Bottou cured adversarial instability by replacing JS divergence with the **Earth-Mover (Wasserstein-1) Distance** from optimal transport theory:

W_1(p_{text{data}}, p_g) = inf_{gamma in Pi(p_{text{data}}, p_g)} mathbb{E}_{(x, y) sim gamma}[|x – y|]

Unlike JS divergence, W_1 provides a smooth, continuous, and non-zero distance metric even when supports are completely disjoint.

By the **Kantorovich-Rubinstein Duality Theorem**, the intractable infimum over joint measures transforms into a supremum over 1-Lipschitz functions $finmathcalF_L$:

W_1(p_{text{data}}, p_g) = sup_{|f|_L le 1} mathbb{E}_{x sim p_{text{data}}}[f(x)] – mathbb{E}_{x sim p_g}[f(x)]

Here, the discriminator becomes a continuous **Critic** $f_w(x)$. Gulrajani et al. (2017) enforced the 1-Lipschitz constraint via the **Gradient Penalty (WGAN-GP)**:

mathcal{L}_{text{GP}} = mathbb{E}_{hat{x}}left[(|nabla_{hat{x}} f_w(hat{x})|_2 – 1)^2 right]

penalizing critics whose directional gradients deviate from unit slope along interpolations between real and generated points, ensuring stable monotonic generative convergence.

Citation: Mahmud, S. (2026). "Generative Adversarial Networks (GANs) — Minimax Games & Wasserstein Geometries." In Deep Learning Architectures & Representation Theory, Part 12.
https://shakilmahmud.com/deep-learning-architectures-part-12-generative-adversarial-networks-gans-wasserstein-distance/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART XIII

Normalizing Flows & Invertible Networks — Change-of-Variables & Exact Likelihoods

Normalizing Flows & Invertible Networks — Change-of-Variables & Exact Likelihoods

582 words • November 4, 2026 • Permanent URI

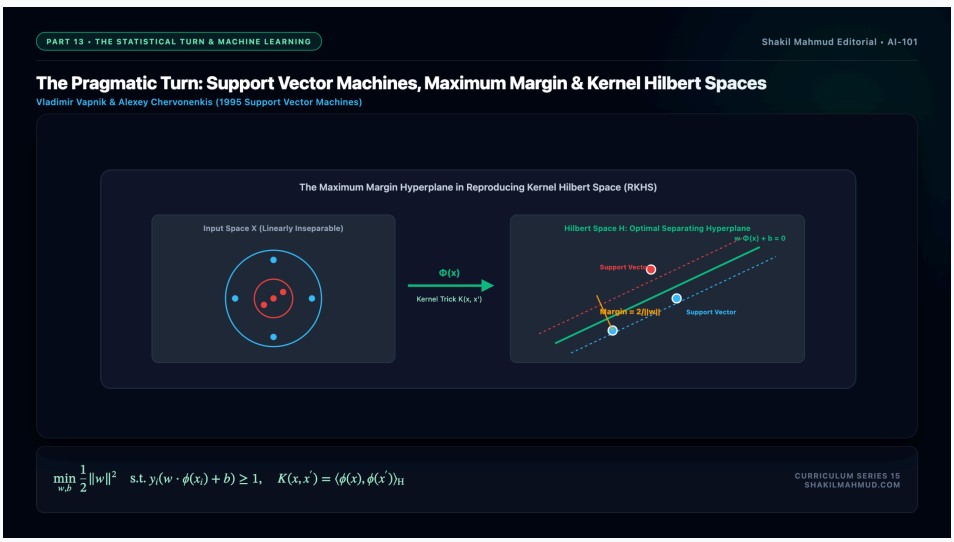


Plate XIII: *Support Vector Machines in Reproducing Kernel Hilbert Space: Maximum margin hyperplane separating non-linear input geometries.*

Most deep generative models—such as VAEs and GANs—suffer from fundamental theoretical compromises: VAEs optimize only a surrogate lower bound (the ELBO) rather than exact data likelihood, while GANs provide no density evaluation whatsoever and battle severe minimax instability. **Normalizing Flows** resolve both limitations by constructing *invertible, bijective neural networks*. By leveraging the multivariable Change-of-Variables Theorem, normalizing flows transform simple isotropic Gaussian base distributions into arbitrarily intricate multimodal densities while computing **exact marginal likelihoods** in closed form.

1. The Multivariable Change-of-Variables Theorem

Let $Zinmathbb R^d$ be a latent continuous random vector governed by a known, tractable prior probability density $p_Z(z)$ (such as a standard isotropic Gaussian $mathcal N(0, I_d)$). Let $f : mathbb R^d to mathbb R^d$ be a smooth, bijective (invertible) diffeomorphism with inverse $g = f^{-1}$ such that $X = f(Z)$ and $Z = g(X)$.

By the conservation of probability mass across differential volumes $p_X(x)|dx| = p_Z(z)|dz|$, the probability density function of target variable X is given by:

$$p_X(x) = p_Z(g(x)) cdot left|det left(frac partial g(x) partial x right) right| = p_Z(f^{-1}(x)) cdot left|det left(frac partial f(z) partial z right) right|^{-1}$$

where $J_f(z) = frac partial f(z) partial z in mathbb R^{d times d}$ is the Jacobian matrix of partial derivatives $J_{ij} = frac partial f_i partial z_j$. Taking logarithms yields the exact log-likelihood:

log p_X(x) = log p_Z(f^{-1}(x)) – log left| det J_f(f^{-1}(x)) right|

2. Compositional Depth and Jacobian Algebra

A single simple transformation cannot capture the geometry of complex empirical data. However, the composition of K invertible diffeomorphic functions $f = f_K circ f_{K-1} circ dots circ f_1$ remains strictly invertible ($f^{-1} = f_1^{-1} circ dots circ f_K^{-1}$).

By the multivariable chain rule, the Jacobian of a composite mapping is the matrix product of intermediate Jacobians:

$$J_f(z_0) = J_{f_K}(z_{K-1}) cdot J_{f_{K-1}}(z_{K-2}) cdots J_{f_1}(z_0)$$

By the multiplicative property of determinants ($det(AB) = det(A) det(B)$), the log-determinant decomposes into a simple sum of scalar terms:

Shakil Mahmud • Research Monograph

$$logleft|detJ_f(z_0)right| = sum_{k=1}^{K} logleft|detJ_{f_k}(z_{k-1})right|$$

3. The Computational Bottleneck: Triangular Coupling Layers

Evaluating the determinant of an arbitrary dense $dtimesd$ matrix scales cubically: $O(d^3)$. In high-dimensional vision where $d = 10^6$, cubic scaling is impossible. The core architectural challenge of normalizing flows is designing layers whose Jacobian is **triangular**, collapsing determinant evaluation to the product of diagonal elements in linear time $O(d)$:

$$det(J) = prod_{i=1}^{d} J_{ii} implies log|det(J)| = sum_{i=1}^{d} log|J_{ii}|$$

In 2016, Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio introduced **RealNVP (Real-valued Non-Volume Preserving)** flows via *Affine Coupling Layers*:

1. Partition the input coordinates into two disjoint halves: $x = [x_{1:d/2}, x_{d/2+1:d}]$.
2. Leave the first half unaltered: $y_{1:d/2} = x_{1:d/2}$.
3. Transform the second half using scale $s(cdot)$ and translation $t(cdot)$ networks conditioned on the first half:

$$y_{d/2+1:d} = x_{d/2+1:d} odot exp(s(x_{1:d/2})) + t(x_{1:d/2})$$

The Jacobian of this coupling layer is lower triangular with diagonal entries $[1, dots, 1, exp(s_1), dots, exp(s_{d/2})]$. Its log-determinant is simply $sum_j s(x_{1:d/2})_j$—requiring zero matrix inversions! Remarkably, the scale and translation networks s and t can be arbitrarily complex, non-invertible deep neural networks (e.g., ResNets).

4. Continuous Normalizing Flows (CNFs) & Neural ODEs

Taking the continuous-time limit where depth approaches infinity ($Ktoinfty$) transitions normalizing flows into **Continuous Normalizing Flows (CNFs)** governed by ordinary differential equations $fracdz(t)dt = f(z(t), t)$. By Liouville's continuity theorem, the instantaneous change in log density is governed by the matrix **trace** rather than determinant:

$$fracpartiallogp(z(t))partialt = -textTrleft(fracpartial f(z(t), t)partial z(t)right)$$

Applying the Hutchinson stochastic trace estimator enables unbiased $O(d)$ likelihood estimation across continuous neural vector fields.

Citation: Mahmud, S. (2026). "Normalizing Flows & Invertible Networks — Change-of-Variables & Exact Likelihoods." In Deep Learning Architectures & Representation Theory, Part 13. https://shakilmahmud.com/deep-learning-architectures-part-13-normalizing-flows-invertible-networks-change-of-variables/

SHAKIL MAHMUD RESEARCH MONOGRAPHS · PART XIV

Energy-Based Models & Modern Hopfield Networks — Boltzmann Machines to Dense Memory

Energy-Based Models & Modern Hopfield Networks — Boltzmann Machines to Dense Memory

542 words • November 4, 2026 • Permanent URI

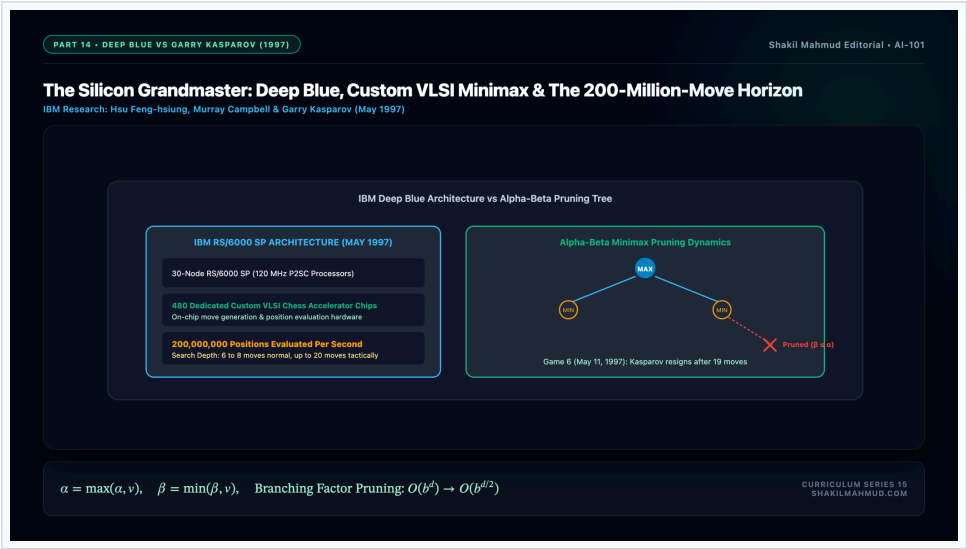


Plate XIV: *IBM Deep Blue's 30-node supercomputer with 480 custom VLSI chess chips executing 200M position alpha-beta minimax searches against Kasparov.*

Most probabilistic deep models normalize outputs via softmax or partition functions to represent normalized probabilities summing to one. This enforcement of normalization imposes severe computational burdens across high-dimensional manifolds. **Energy-Based Models (EBMs)**, pioneered in neural systems by John Hopfield, Yann LeCun, and Geoffrey Hinton, discard this normalization requirement. By defining an unnormalized scalar energy function $E_theta(x) in mathbb R$, EBMs associate plausible, harmonious data configurations with deep energy wells and improbable configurations with high energy barriers.

1. The Boltzmann Gibbs Distribution and the Partition Function

An Energy-Based Model parameterizes a continuous energy landscape $E_theta(x)$ using an expressive neural network. The corresponding probability density is given by the Gibbs-Boltzmann distribution:

$$p_theta(x) = frac{exp(-E_theta(x))}{Z(theta)}, quad Z(theta) = int_{mathcal X} exp(-E_theta(tilde x)) dtilde x$$

The normalizing denominator $Z(theta)$ is the **Partition Function**. While computing $Z(theta)$ is analytically intractable in continuous or high-dimensional spaces, evaluating relative probabilities or gradients requires evaluating only differences in energy: log frac{p(x_1)}{p(x_2)} = E(x_2) – E(x_1).

2. The Log-Likelihood Gradient: Positive vs Negative Phases

Evaluating the gradient of the log-likelihood of training data with respect to model parameters $theta$ reveals the foundational thermodynamic duality of EBM training:

nabla_theta log p_theta(x) = -nabla_theta E_theta(x) – nabla_theta log Z(theta)

Substituting $nabla_theta log Z(theta) = frac{1}{Z(theta)} int nabla_theta exp(-E(tilde x)) dtilde x = -mathbb E_{tilde x sim p_theta}[nabla_theta E_theta(tilde x)]$ yields:

nabla_theta mathcal{L}(theta) = underbrace{mathbb{E}_{x sim p_{text{data}}}}left[nabla_theta E_theta(x) right]}_{text{Positive Phase (Data Clamping)}} – underbrace{mathbb{E}_{tilde{x} sim p_theta}left[nabla_theta E_theta(tilde{x}) right]}_{text{Negative Phase (Hallucination / Fantasy)}}

Training an energy model acts as a physical push-pull dynamic:

- **The Positive Phase:** Pulls down the energy $E_theta(x)$ at empirical training points, carving deep attractor basins.

Shakil Mahmud • Research Monograph

- **The Negative Phase:** Pushes up the energy $E_theta(tildex)$ at synthetic points sampled from the model's current distribution, preventing the energy landscape from collapsing into a flat zero function.

3. Sampling Dynamics: Contrastive Divergence and Langevin MCMC

To evaluate the negative phase expectation, practitioners sample from unnormalized distribution $p_theta(x)$ using **Unadjusted Langevin MCMC**:

x_{t+1} = x_t – frac{epsilon}{2} nabla_x E_theta(x_t) + sqrt{epsilon} z_t, quad z_t sim mathcal{N}(0, I)

In 2002, Geoffrey Hinton introduced **Contrastive Divergence (CD-k)** for Restricted Boltzmann Machines (RBMs): initializing Markov chains directly at training data points x and running Gibbs sampling for just k steps (often $k = 1$), dramatically accelerating convergence.

4. Modern Hopfield Networks and Dense Associative Memory

In 1982, John Hopfield introduced recurrent associative memory networks with binary states $sin-1, +1^d$ and quadratic energy $E(s) = -frac12s^TWs$. The classical Hopfield network possessed an information storage capacity strictly bounded by Amit, Gutfreund, and Sompolinsky (1985) at $Capprox0.14d$ patterns; storing more patterns induced spurious mixed attractors.

In 2016, Dmitry Krotov and John Hopfield—and subsequently Ramsauer et al. (2020)—revolutionized associative memory by introducing continuous states with **exponential energy interactions**:

$$E(x) = -textlse(betaX^Tx) + frac12x^Tx$$

where $textlse(z) = logsum_iexp(z_i)$ is the Log-Sum-Exp function and $X = [x_1, dots, x_M]$ stores M memory vectors. Ramsauer et al. proved that the single-step attractor update rule on this energy function is:

$$x^textnew = Xcdottextsoftmax(betaX^Tx)$$

This is *identical* to the core Scaled Dot-Product Attention mechanism of the Transformer! Furthermore, the storage capacity of Modern Hopfield Networks scales exponentially: $Capprox2^d/2$ memories, establishing an unshakeable mathematical bridge uniting associative attractor memories, energy models, and modern attention architectures.

Citation: Mahmud, S. (2026). "Energy-Based Models & Modern Hopfield Networks — Boltzmann Machines to Dense Memory." In Deep Learning Architectures & Representation Theory, Part 14.
https://shakilmahmud.com/deep-learning-architectures-part-14-energy-based-models-modern-hopfield-networks-attractor-memory/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART XV

Graph Neural Networks (GNNs) — Message Passing & The Weisfeiler-Lehman Limits

Graph Neural Networks (GNNs) — Message Passing & The Weisfeiler-Lehman Limits

531 words • November 4, 2026 • Permanent URI

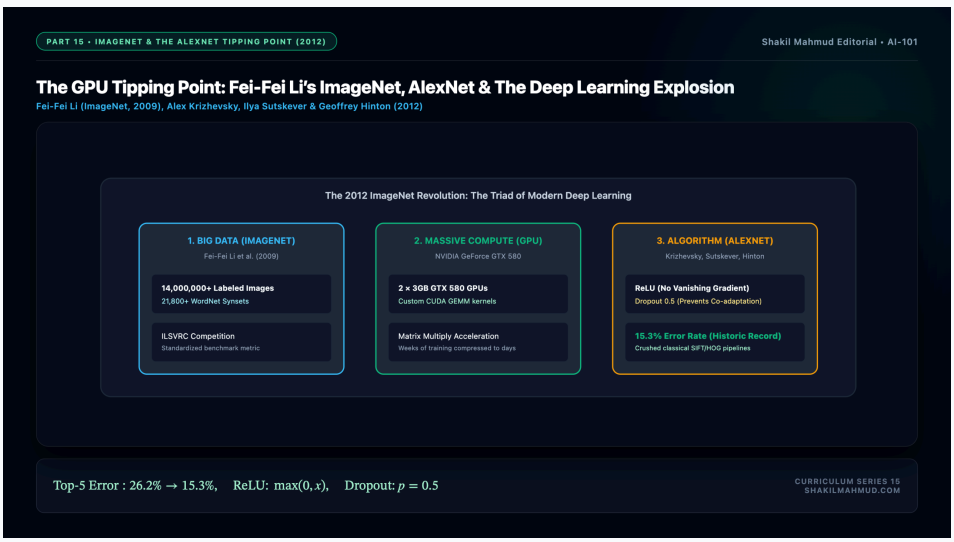


Plate XV: *The 2012 Deep Learning tipping point: Fei-Fei Li's ImageNet dataset, NVIDIA GPU acceleration, and AlexNet's ReLU/Dropout architecture.*

Standard deep learning architectures assume rigid, regular Euclidean data structures: images reside on uniform 2D pixel grids, while language conforms to 1D discrete sequences. Yet the physical and social universe is overwhelmingly non-Euclidean: chemical molecules, protein interactomes, citation networks, social graphs, and knowledge manifolds exist as arbitrary topological graphs $mathcalG = (mathcalV, mathcalE)$. **Graph Neural Networks (GNNs)** generalize deep learning to these unstructured domains by enforcing **Permutation Equivariance** and computing localized **Message Passing** across topological neighborhoods.

1. The Formal Symmetry of Graphs: Permutation Equivariance

A graph $mathcalG = (mathcalV, mathcalE)$ with N vertices is characterized by node feature matrix $XinmathbbR^Ntimesd$ and symmetric binary adjacency matrix $Ain0,1^NtimesN$. Crucially, nodes possess no canonical spatial ordering: permuting node indices by an arbitrary $NtimesN$ permutation matrix $PinPi_N$ yields an identical graph isomorphic to the original.

A graph neural network mapping $f(A, X)$ must be **Permutation Equivariant** for node-level representations:

$$f(PAP^T, PX) = Pf(A, X), quadforallPinPi_N$$

For graph-level classifications (e.g., predicting molecular toxicity), the readout mapping must be **Permutation Invariant**: $f(PAP^T, PX) = f(A, X)$.

2. The Message Passing Neural Network (MPNN) Framework

In 2017, Justin Gilmer et al. unified spatial graph convolutions under the **Message Passing Neural Network (MPNN)** abstraction. In layer $kin1, dots, K$, each node $vinmathcalV$ updates its state $h_v^(k)$ through three localized differentiable phases:

1. **Message Function ($mathcalM^(k)$):** Generates directed messages along edges $(u, v)inmathcalE$ connecting neighbors $uinmathcalN(v)$:

$$m_uv^(k) = mathcalM^(k)left(h_u^(k-1), h_v^(k-1), e_uvright)$$

2. **Aggregation Operator ($bigoplus$):** Accumulates incoming messages across the neighborhood multiset:

$$barm_v^(k) = bigoplus_uinmathcalN(v) m_uv^(k)$$

Shakil Mahmud • Research Monograph

The operator $bigoplus$ *must* be a symmetric commutative reduction (such as sum, $textmean$, or max) to guarantee permutation equivariance.

3. **Update Function ($mathcalU^(k)$):** Synthesizes the aggregated message with the node's prior state:

$$h_v^(k) = mathcalU^(k)left(h_v^(k-1), barm_v^(k)right)$$

3. Canonical Architectures: GCN, GAT, and GIN

The MPNN framework encompasses several landmark architectures:

- **Graph Convolutional Networks (GCN, Kipf & Welling, 2017):** First-order Chebyshev spectral approximation with normalized adjacency:

$$H^(k) = sigmaleft(tildeD^-1/2tildeAtildeD^-1/2H^(k-1)W^(k)right), quadtildeA = A + I_N$$

- **Graph Attention Networks (GAT, Veličković et al., 2018):** Replaces static edge weights with dynamic self-attention coefficients $alpha_uv = textsoftmax_uleft(textLeakyReLU(a^T[Wh_uparallelWh_v])right)$, enabling anisotropic filtering.
- **Graph Isomorphism Network (GIN, Xu et al., 2019):** Maximizes multiset expressivity via universal approximator MLPs:

$$h_v^(k) = textMLP^(k)left((1 + epsilon^(k))h_v^(k-1) + sum_uinmathcalN(v)h_u^(k-1)right)$$

4. Theoretical Expressive Limits: The 1-Weisfeiler-Lehman (1-WL) Test

How expressive are message-passing GNNs? In 2019, Keyulu Xu et al. and Christopher Morris et al. independently established the fundamental upper bound: **Standard message passing GNNs are at most as powerful as the 1-Weisfeiler-Lehman (1-WL) Graph Isomorphism Test**.

The 1-WL algorithm tests graph isomorphism by iteratively updating node colors based on sorted multisets of neighbor colors. If two non-isomorphic graphs produce identical color histograms under 1-WL (e.g., distinguishing a 6-node ring from two disjoint 3-node triangles), *no standard MPNN can distinguish them*, regardless of depth or parameter scale. Breaking the 1-WL barrier requires higher-order k-WL sub-graph networks, directional cellular sheaves, or persistent homology.

Citation: Mahmud, S. (2026). "Graph Neural Networks (GNNs) — Message Passing & The Weisfeiler-Lehman Limits." In Deep Learning Architectures & Representation Theory, Part 15.
https://shakilmahmud.com/deep-learning-architectures-part-15-graph-neural-networks-message-passing-weisfeiler-lehman/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART XVI

Geometric Deep Learning & Spherical CNNs — Gauge Symmetries & SO(3) Equivariance

Geometric Deep Learning & Spherical CNNs — Gauge Symmetries & SO(3) Equivariance

556 words • November 4, 2026 • Permanent URI

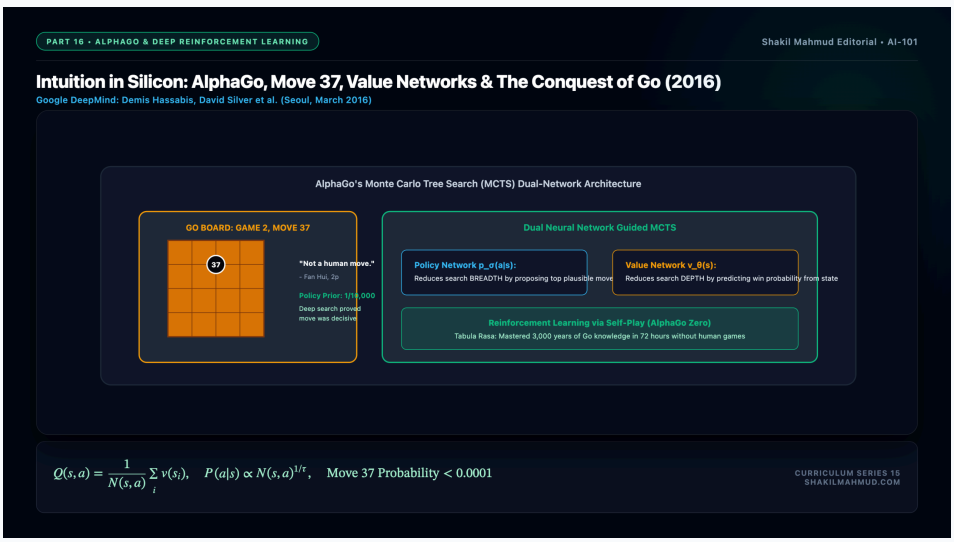


Plate XVI: *AlphaGo's dual policy and value networks guiding Monte Carlo Tree Search, immortalized by the intuitive Move 37 shoulder hit.*

In 2021, Michael Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković published their magnum opus, *"Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges"*. Geometric deep learning formalizes Felix Klein's Erlangen Program for machine learning: all modern neural architectures—CNNs, RNNs, Transformers, and GNNs—are not ad-hoc empirical inventions, but direct mathematical manifestations of **symmetry groups and geometric invariance**. On curved Riemannian manifolds and spherical topologies $mathbbS^2$, geometric deep learning enforces 3D rotation equivariance $SO(3)$ through the mathematics of **Spherical Harmonics and Gauge Connections**.

1. The Geometric Deep Learning Blueprint: The 5 G's

Every deep learning architecture operates over a domain $Omega$ possessing a symmetry group $mathfrakG$. The five foundational pillars encompass:

1. **Grids:** Euclidean domains $mathbbR^d$ with translation group $(mathbbR^d, +)$ $implies$ *Convolutional Neural Networks.*
2. **Groups:** Homogeneous spaces $XcongG/H$ $implies$ *Group-Equivariant CNNs (G-CNNs).*
3. **Graphs:** Discrete topological combinatorial spaces $mathcalG = (mathcalV, mathcalE)$ with permutation symmetry group Pi_N $implies$ *Graph Neural Networks.*
4. **Geodesics:** Riemannian manifolds $(mathcalM, g)$ with intrinsic distance metrics $implies$ *Manifold Neural Networks.*
5. **Gauges:** Local coordinate frame transformations on manifolds $implies$ *Gauge-Equivariant Neural Networks.*

2. Spherical Convolutions and the 3D Rotation Group $SO(3)$

In climate science, omnidirectional robotics, and molecular astrophysics, data naturally resides on the 2D spherical surface $mathbbS^2 = xinmathbbR^3 : |x|_2 = 1$. Projecting spherical data onto flat 2D planar maps (e.g., equirectangular projection) creates severe geometric distortions near the poles, destroying Euclidean translational weight sharing.

The continuous symmetry group of the sphere is the **Special Orthogonal Group** $SO(3)$ of $3times3$ rotation matrices satisfying $R^TR = I$ and $det(R) = +1$. The group action rotates spherical functions $finL^2(mathbbS^2)$ according to $(pi(R)f)(x) = f(R^-1x)$.

Cohen et al. (2018) formulated **Spherical CNNs**: spherical cross-correlation lifts spherical functions into functions on the rotation group $SO(3)$:

$$(fstarpsi)(R) = int_mathbbS^2 f(x)psi(R^-1x)dx$$

Subsequent layers perform group convolutions over $SO(3)$, guaranteeing mathematically exact equivariance to 3D physical rotations.

Shakil Mahmud • Research Monograph

3. Harmonic Analysis: Spherical Harmonics and Wigner D-Matrices

Evaluating spherical convolutions directly in the spatial domain is computationally intractable. Geometric deep learning resolves this via **Generalized Fourier Transforms** on Lie groups.

By the Peter-Weyl Theorem, the square-integrable Hilbert space $L^2(mathbbS^2)$ decomposes into a direct sum of orthogonal irreducible representation subspaces $mathcalV_l$ spanned by the **Spherical Harmonics** $Y_l^m(theta, phi)$ for degree $lge0$ and order $-llemlel$:

$$L^2(mathbbS^2) = bigoplus_l=0^infty mathcalV_l, quad dim(mathcalV_l) = 2l + 1$$

Spherical harmonics are the eigenfunctions of the spherical Laplace-Beltrami operator: $Delta_mathbbS^2 Y_l^m = -l(l+1)Y_l^m$. Under rotation $Rin SO(3)$, spherical harmonics transform via the unitary **Wigner D-Matrices** $D^l(R) in mathbbC^(2l+1)times(2l+1)$:

$$Y_l^m(R^{-1} x) = sum_{m’=-l}^l D_{m’ m}^l(R) Y_l^{m’}(x)$$

The Non-Commutative Spherical Convolution Theorem transforms spatial convolution into block-diagonal matrix multiplication in the frequency domain, achieving $O(B^3)$ complexity via the Fast Spherical Fourier Transform (S2FFT).

4. Gauge Symmetries and Fiber Bundles

On arbitrary curved Riemannian manifolds $mathcalM$ (such as 3D protein surfaces), no global coordinate system exists. Observers must establish arbitrary, local reference frames (gauges) on tangent spaces $T_pmathcalM$.

A change of local gauge by angle $theta(p) in SO(2)$ rotates the tangent coordinates. **Gauge Equivariant CNNs** (Cohen et al., 2019) treat feature representations as sections of a vector bundle. Parallel transport along geodesics is governed by the *Levi-Civita connection*, ensuring representations remain rigorously invariant to the arbitrary choice of local coordinate frames.

Citation: Mahmud, S. (2026). "Geometric Deep Learning & Spherical CNNs — Gauge Symmetries & SO(3) Equivariance." In Deep Learning Architectures & Representation Theory, Part 16.
https://shakilmahmud.com/deep-learning-architectures-part-16-geometric-deep-learning-spherical-cnns-gauge-symmetries/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART XVII

Neural Ordinary Differential Equations — Continuous-Depth & The Adjoint State Method

Neural Ordinary Differential Equations — Continuous-Depth & The Adjoint State Method

621 words • November 4, 2026 • Permanent URI

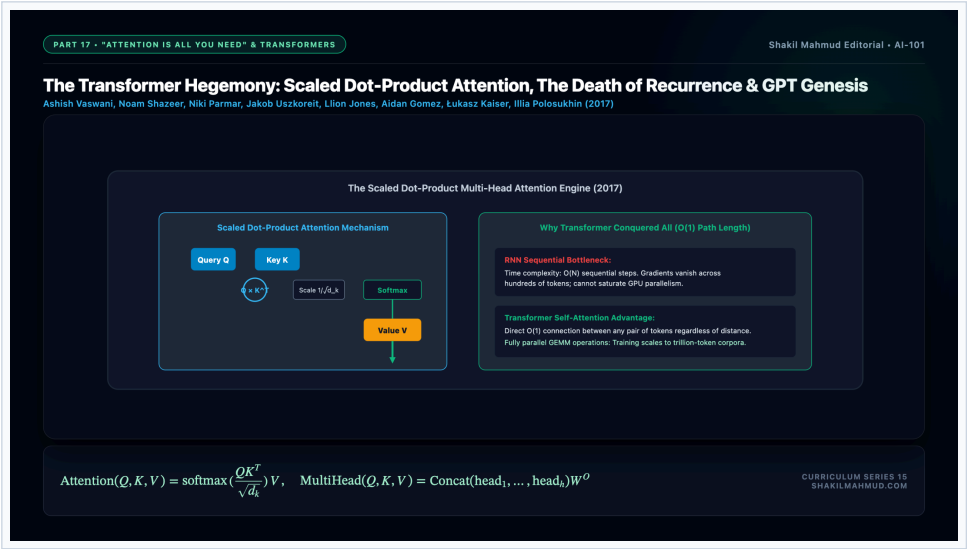


Plate XVII: *The Transformer architecture: Scaled dot-product multi-head self-attention, replacing recurrent bottlenecks with O(1) token interactions.*

In standard residual networks, representations evolve across a sequence of discrete, static layers $h_{t+1} = h_t + f(h_t, theta_t)$. In 2018, Ricky Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud realized that this residual difference equation is simply the forward Euler discretization of an underlying continuous process. By taking the infinitesimal step limit $Deltatto0$, they unveiled **Neural Ordinary Differential Equations (Neural ODEs)**—replacing discrete layer stacks with continuous vector fields and revolutionizing memory efficiency via the continuous **Adjoint Sensitivity Method**.

1. From Discrete ResNets to Continuous Dynamical Flows

Recall the discrete residual update step:

h_{t+1} – h_t = f(h_t, theta_t)

Scaling step size by infinitesimal $Deltat$ and taking the continuum limit $Deltatto0$ yields the continuous ordinary differential equation:

lim_{Delta t to 0} frac{h(t + Delta t) – h(t)}{Delta t} = frac{dh(t)}{dt} = f(h(t), t, theta)

where $f(h(t), t, theta)$ is a neural network defining a smooth vector field in state space. Computing the representation at terminal depth T from initial input $h(0) = x$ is formulated as an Initial Value Problem (IVP):

$$h(T) = h(0) + int_0^T f(h(t), t, theta)dt = textODESolve(h(0), f, 0, T, theta)$$

Crucially, the architecture possesses no fixed "depth": depth is an adaptive continuum dynamically determined by modern numerical ODE solvers (such as Runge-Kutta 4/5 or Dormand-Prince with adaptive step sizes) based on user-specified error tolerances.

2. Constant Memory Backpropagation: The Adjoint State Method

Standard backpropagation through adaptive numerical ODE solvers requires storing all intermediate internal solver evaluations in memory, incurring catastrophic memory complexity scaling with the number of solver steps.

Chen et al. resurrected the **Adjoint State Method**—rooted in Lev Pontryagin's 1962 Maximum Principle of optimal control theory. Define the continuous adjoint state $a(t)$ as the sensitivity of scalar loss L with respect to state trajectory $h(t)$:

Shakil Mahmud • Research Monograph

$$a(t) equiv frac partial L partial h(t)$$

Differentiating $a(t)$ with respect to continuous time yields the **Adjoint Differential Equation**:

$$frac da(t) dt = -a(t)^T frac partial f(h(t), t, theta) partial h(t)$$

To train the model, the algorithm performs a reverse-time ODE solve starting from terminal boundary condition $a(T) = frac partial L partial h(T)$. As the solver runs backward in time from $t = T$ to $t = 0$, it reconstructs the forward trajectory $h(t)$ in reverse while simultaneously accumulating the parameter gradient:

$$frac dL d theta = -int_T^0 a(t)^T frac partial f(h(t), t, theta) partial theta dt$$

Because the forward states are recomputed in reverse time directly from $h(T)$, **zero intermediate forward activations need to be stored in GPU VRAM**! The training memory complexity of an arbitrarily complex Neural ODE is strictly $O(1)$ constant memory.

3. Irregularly-Sampled Time Series: Latent ODEs

Traditional sequence models (RNNs, Transformers) require observations to be aligned along uniform discrete time grids $Delta t_i = text constant$. Real-world telemetry—medical intensive care records, ecological sensor streams, astronomical telescope transits—is notoriously irregular, with variable gaps spanning minutes to weeks.

Neural ODEs naturally ingest irregular timestamps $(t_0, t_1, dots, t_N)$: the hidden state evolves autonomously under continuous dynamics dh/dt during observation gaps, jumping discretely upon observing a new event via Bayesian filtering updates. This established **Latent ODEs** as the premier theoretical model for continuous physical forecasting.

4. Augmented Neural ODEs and Topological Knots

In 2019, Emilien Dupont et al. exposed a topological limitation of pure Neural ODEs: because ODE trajectories cannot intersect in phase space (by the Picard-Lindelöf Uniqueness Theorem), continuous flows cannot invert or unknot topological configurations (e.g., separating concentric spheres or learning the 1D reflection $f(x) = -x$).

Augmented Neural ODEs (ANODEs) solve this by embedding the state into an augmented dimension $h_{text aug}(t) = [h(t), mathbf 0] in mathbb R^{d+p}$. By providing extra dimensions for trajectories to maneuver around one another without intersection, ANODEs become universal continuous function approximators while significantly smoothing the learned vector field and accelerating solver evaluation speeds.

Citation: Mahmud, S. (2026). "Neural Ordinary Differential Equations — Continuous-Depth & The Adjoint State Method." In Deep Learning Architectures & Representation Theory, Part 17.
https://shakilmahmud.com/deep-learning-architectures-part-17-neural-odes-continuous-depth-adjoint-state/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART XVIII

State-Space Models & Mamba — Continuous HiPPO Projection & Linear-Time Sequences

State-Space Models & Mamba — Continuous HiPPO Projection & Linear-Time Sequences

574 words • November 4, 2026 • Permanent URI



Plate XVIII: *Empirical neural scaling laws: Kaplan & Chinchilla power-law frontiers and the emergence of few-shot in-context learning in GPT-3.*

While the Transformer architecture dominates modern foundation models, it is burdened by a severe computational barrier: the **Quadratic Context Bottleneck**. Standard self-attention requires evaluating the full $NtimesN$ attention matrix, scaling computation and memory as $O(N^2)$ with sequence length N. Processing million-token documents or long-range video becomes computationally prohibitive. In 2023, Albert Gu and Tri Dao introduced **Mamba**, rooted in continuous **Structured State-Space Models (SSMs)**, matching Transformer performance while processing sequences in **linear $O(N)$ time** and constant inference memory.

1. The Continuous-Time State-Space Representation

A State-Space Model maps a 1D continuous input signal $x(t)inmathbbR$ through an N-dimensional latent state $h(t)inmathbbR^N$ to an output $y(t)inmathbbR$ via linear differential equations:

$$h'(t) = Ah(t) + Bx(t)$$

$$y(t) = Ch(t) + Dx(t)$$

where $AinmathbbR^{NtimesN}$ is the state transition matrix governing internal memory evolution, $BinmathbbR^{Ntimes1}$ is the input projection vector, and $CinmathbbR^{1timesN}$ is the observation projection.

2. The HiPPO Framework: Optimal Orthogonal History Projection

If state transition matrix A is randomly initialized, internal memory suffers catastrophic forgetting. In 2020, Albert Gu et al. established the **HiPPO (High-order Polynomial Projection Operators)** framework: proving that to maintain an optimal online approximation of continuous historical signal $x(t)$ under a exponentially decayed memory measure mu(t) = e^{t – t_{max}}, the state vector must track the coefficients of Legendre orthogonal polynomials.

The resulting closed-form HiPPO matrix $AinmathbbR^{NtimesN}$ is:

A_{nk} = begin{cases} -sqrt{(2n+1)(2k+1)} & text{if } n > k \ -(n+1) & text{if } n = k \ 0 & text{if } n < k end{cases}

The HiPPO operator endows the linear dynamical system with mathematically provable memory retention across millions of time steps.

Shakil Mahmud • Research Monograph

3. Discretization and Dual Dualities: Recurrence vs Convolution

To process digital discrete tokens $(x_0, x_1, dots)$, continuous SSMs are discretized using the bilinear (Tustin) or zero-order hold (ZOH) transformation with step size $Delta$:

bar{A} = exp(Delta A), quad bar{B} = (Delta A)^{-1}(exp(Delta A) – I) cdot Delta B

The discretized system exhibits a remarkable mathematical duality:

- **Linear Recurrent View ($O(1)$ Inference):** $h_t = barAh_{t-1} + barBx_t, quady_t = Ch_t$. During autoregressive token generation, the model maintains a constant-size hidden state h_t, achieving $O(1)$ time complexity and avoiding the massive KV cache memory footprint of Transformers.
- **Global Convolutional View ($O(NlogN)$ Training):** Unrolling the recurrence reveals a global 1D convolution: $y = x * barK$ where $barK = (CbarB, CbarAbarB, dots, CbarA^{N-1}barB)$. Training is executed non-autoregressively across the entire sequence simultaneously in parallel via the Fast Fourier Transform (FFT).

4. Mamba: The Selective State Space (S6) Architecture

Classical SSMs (S4, H3) are *linear time-invariant (LTI)*: matrices $A, B, C, Delta$ remain constant across all tokens regardless of input content. LTI systems cannot perform content-based reasoning: they cannot selectively filter out irrelevant noise or remember specific crucial tokens.

Gu and Dao (2023) broke the LTI constraint by introducing **Selective State Spaces (S6)**: making parameters input-dependent functions of the current token x_t:

$$B_t = W_Bx_t, quadC_t = W_Cx_t, quadDelta_t = textsoftplus(W_Deltax_t + textParameter)$$

Because parameters vary across time, the global convolution duality is lost. To preserve training speed, Gu and Dao designed a **Hardware-Aware Parallel Scan**: rather than materializing the intermediate expanded states in slow GPU HBM memory, the parallel scan executes entirely inside ultra-fast on-chip SRAM memory, achieving $O(N)$ linear-time processing faster than optimized FlashAttention.

Citation: Mahmud, S. (2026). "State-Space Models & Mamba — Continuous HiPPO Projection & Linear-Time Sequences." In Deep Learning Architectures & Representation Theory, Part 18.
https://shakilmahmud.com/deep-learning-architectures-part-18-state-space-models-mamba-linear-time-sequences/

SHAKIL MAHMUD RESEARCH MONOGRAPHS · PART XIX

Deep Reinforcement Learning Architectures — Deep Q-Networks to Actor-Critic Duality

Deep Reinforcement Learning Architectures — Deep Q-Networks to Actor-Critic Duality

663 words • November 4, 2026 • Permanent URI

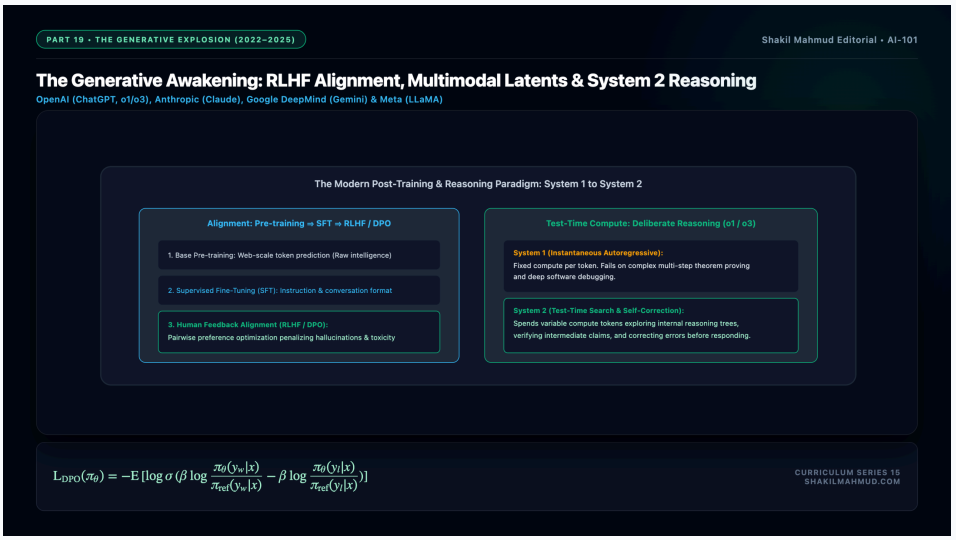


Plate XIX: *The generative AI paradigm: From RLHF/DPO preference alignment to System 2 test-time reasoning search in modern frontier models.*

Supervised deep learning thrives in passive perception regimes where static datasets furnish ground-truth labels. In interactive environments—autonomous navigation, physical robotics, complex strategy games, and agentic reasoning—an autonomous agent must learn through trial, error, and delayed environmental rewards. **Deep Reinforcement Learning (DRL)** unifies deep representation architectures with the Bellman optimality equations, transforming high-dimensional perceptual observations directly into optimal behavioral policies.

1. The Markov Decision Process and Bellman Optimality

An interactive agent interacts with an environment formalized as a Markov Decision Process (MDP) tuple $(mathcalS, mathcalA, mathcalP, mathcalR, gamma)$:

- State space $mathcalS$, action space $mathcalA$, transition dynamics mathcal{P}(s’ mid s, a), reward function $mathcalR(s,a)$, and temporal discount factor $gammain[0,1)$.

The state-action value function $Q^pi(s,a) = mathbbE_pileft[sum_t=0^infty gamma^t R_t mids_0 = s, a_0 = aright]$ measures the expected discounted return under policy pi. By Richard Bellman's Principle of Optimality, the optimal value function satisfies the non-linear **Bellman Optimality Operator** $mathcalT^*$:

$$Q^*(s, a) = mathcal{R}(s, a) + gamma mathbb{E}_{s’ sim mathcal{P}}left[max_{a’ in mathcal{A}} Q^*(s’, a’) right]$$

By the Banach Fixed-Point Theorem, $mathcalT^*$ is a $gamma$-contraction mapping under the infinity norm, guaranteeing unique convergence in tabular settings.

2. Sutton's Deadly Triad and Deep Q-Networks (DQN)

When parameterizing the action-value function with a deep neural network $Q(s,a;theta)$, Richard Sutton identified the **Deadly Triad**—a lethal combination guaranteed to induce catastrophic divergence during training:

1. **Function Approximation:** Representing states with deep networks rather than lookup tables.
2. **Bootstrapping:** Updating value estimates from downstream value estimates ($r + gammamaxQ$).
3. **Off-Policy Learning:** Training on transitions generated by past behavioral distributions.

In 2015, Volodymyr Mnih et al. at Google DeepMind tamed the Deadly Triad, achieving human-level control across 49 Atari games via **Deep Q-Networks (DQN)** using two architectural stabilizing pillars:

- **The Experience Replay Buffer ($mathcalD$):** Transitions (s_t, a_t, r_t, s_t+1) are stored in a rolling FIFO buffer. Random mini-batches are uniformly sampled for training, breaking temporal autocorrelation across consecutive video frames and stabilizing the i.i.d. assumption.

Shakil Mahmud • Research Monograph

- **The Target Network ($theta^-$):** In standard Q-learning, the target y = r + gamma max_{a’} Q(s’, a’; theta) shifts constantly with every gradient update theta leftarrow theta – eta nabla L, chasing a moving target. DQN freezes a separate target network $theta^-$, updating it only periodically (every C steps):

mathcal{L}(theta) = mathbb{E}_{(s, a, r, s’) sim mathcal{D}}left[left(r + gamma max_{a’} Q(s’, a’; theta^-) – Q(s, a; theta) right)^2 right]

3. Mitigating Overestimation: Double DQN & Dueling Networks

Standard DQN suffers from positive maximization bias: the max_{a’} Q(s’, a’) operator systematically overestimates value functions under noise. Hado van Hasselt et al. (2015) introduced **Double DQN**, decoupling action selection from action evaluation:

y_{text{Double}} = r + gamma Qleft(s’, argmax_{a’} Q(s’, a’; theta); ; theta^- right)

In 2016, Wang et al. formulated the **Dueling Network Architecture**, decomposing $Q(s, a)$ into a state value stream $V(s)$ and state-dependent action advantage stream $A(s, a)$:

Q(s, a; theta, alpha, beta) = V(s; theta, beta) + left(A(s, a; theta, alpha) – frac{1}{|mathcal{A}|} sum_{a’ in mathcal{A}} A(s, a’; theta, alpha) right)

ensuring unidentifiability is resolved and allowing the network to learn state values without having to evaluate every action.

4. Continuous Action Spaces: Actor-Critic Duality & DDPG

When action spaces are continuous vector manifolds $mathcal{A} = mathbb{R}^m$ (e.g., robotic torque controls), computing $max_{a in mathcal{A}} Q(s, a)$ at every step requires solving an expensive non-convex optimization problem.

The **Actor-Critic architecture** resolves this via two interacting networks:

- **Critic ($Q_phi(s, a)$):** Evaluates policy performance via temporal difference learning.
- **Actor ($mu_theta(s)$):** Parameterizes a deterministic continuous policy mapping states directly to action vectors.

By David Silver's **Deterministic Policy Gradient (DPG) Theorem**, actor parameters are updated via the chain rule along the critic's action-gradient vector field:

$$nabla_theta J(mu_theta) = mathbb{E}_{s sim rho^mu}left[left. nabla_theta mu_theta(s) cdot nabla_a Q^mu(s, a) right|_{a=mu_theta(s)} right]$$

Lillicrap et al. (2015) combined DPG with replay buffers and soft Polyak target updates ($theta^- leftarrow tau theta + (1 - tau)theta^-$) to create **DDPG**, establishing the architectural foundation for deep continuous robotic control.

Citation: Mahmud, S. (2026). "Deep Reinforcement Learning Architectures — Deep Q-Networks to Actor-Critic Duality." In Deep Learning Architectures & Representation Theory, Part 19. https://shakilmahmud.com/deep-learning-architectures-part-19-deep-reinforcement-learning-dqn-actor-critic-architectures/

SHAKIL MAHMUD RESEARCH MONOGRAPHS • PART XX

The Representation Theory of Deep Networks — Information Bottleneck & Neural Collapse

The Representation Theory of Deep Networks — Information Bottleneck & Neural Collapse

616 words • November 4, 2026 • Permanent URI

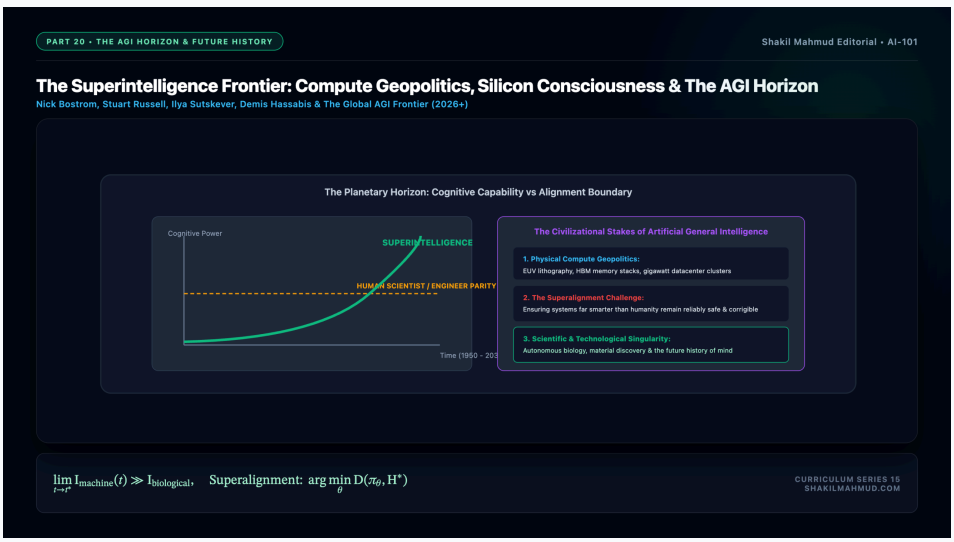


Plate XX: *The Artificial General Intelligence frontier: Exponential cognitive capability curves, EUV compute infrastructure, and superalignment theory.*

For decades, deep neural networks operated as empirical black boxes: their unprecedented ability to generalize while severely overparameterized ($Dggn$) directly contradicted classical statistical learning theory. In recent years, mathematical breakthroughs in representation theory—spearheaded by Naftali Tishby's **Information Bottleneck Principle**, Vardan Papyan and David Donoho's discovery of **Neural Collapse**, and the geometry of **Double Descent**—have finally unveiled the deep mathematical architecture governing representation learning.

1. Tishby's Information Bottleneck Principle

In 2000, Naftali Tishby, Fernando Pereira, and William Bialek formulated the **Information Bottleneck (IB)** method. In a deep network processing input X to predict target Y, any intermediate internal representation layer T forms a Markov chain: $Yleftrightarrow Xleftrightarrow Tleftrightarrow hatY$.

An optimal representation must solve the variational Lagrangian optimization problem:

$$min_{p(t|x)} mathcal{L}_{text{IB}} = I(X; T) – beta I(T; Y)$$

where $I(A;B) = intp(a,b)logfracp(a,b)p(a)p(b)dadb$ denotes the Shannon Mutual Information and beta > 0 controls the trade-off:

- **Compression Term ($I(X;T)$):** The network acts as a compressor, purging irrelevant nuances, backgrounds, and nuisance variations present in X.
- **Sufficiency Term ($I(T;Y)$):** The network preserves all mutual information necessary to predict target Y faithfully.

Tishby and Zaslavsky (2015) demonstrated that stochastic gradient training exhibits two distinct temporal phases on the IB plane: an initial fast *Empirical Fitting Phase* where $I(T;Y)$ increases rapidly, followed by an extended asymptotic *Information Compression Phase* where diffusion forces compress $I(X;T)$, explaining the implicit regularization of deep architectures.

2. The Double Descent Phenomenon: Overparameterization Regimes

Classical statistical theory posits that increasing model capacity beyond the interpolation threshold ($D = n$) inevitably induces catastrophic overfitting under the U-shaped bias-variance curve. In 2019, Mikhail Belkin et al. proved the **Double Descent** phenomenon:

1. **Underparameterized Regime (D < n):** Classical U-shaped curve; risk decreases then spikes toward infinity at the interpolation boundary ($D = n$) due to near-singular Gram matrices.
2. **Overparameterized Regime (D > n):** As capacity expands further toward infinity ($Dggn$), test risk *declines monotonically* again!

Shakil Mahmud • Research Monograph

In the overparameterized regime, the network possesses infinite interpolating global minimizers. Implicit bias forces gradient descent to select the minimum ell_2-norm or maximum-margin solution, smoothing function curvature and optimizing generalization.

3. Neural Collapse ($mathcal{NC}$): The Terminal Phase of Learning

In 2020, Vardan Papyan, X.Y. Han, and David L. Donoho uncovered an astonishing geometric universality in the Terminal Phase of Training (continuing training well after achieving zero empirical training error, $L_{texttt{train}} = 0$). Across diverse architectures, last-layer features and classifier weights undergo a geometric phase transition known as **Neural Collapse ($mathcal{NC}$)**, characterized by four simultaneous mathematical laws:

1. $mathcal{NC}_1$ **(Variability Collapse):** Within-class feature covariance collapses to zero:

$$Sigma_W = frac{1}{C K} sum_{c=1}^C sum_{k=1}^K (h_{c,k} – bar{h}_c)(h_{c,k} – bar{h}_c)^T to mathbf{0}$$

   All training samples of class c collapse into their individual class centroid $bar{h}_c$.

2. $mathcal{NC}_2$ **(Convergence to Simplex Equiangular Tight Frame):** The normalized class centroids tilde{mu}_c = (bar{h}_c – bar{h}_G) / |bar{h}_c – bar{h}_G| form a **Simplex ETF**: all centroids possess equal length and maximal, identical pairwise angular separation:

$$langle tilde{mu}_c, tilde{mu}_{c’} rangle = -frac{1}{C – 1}, quad forall c ne c’$$

   For 3 classes, centroids form an equilateral triangle (120^circ); for 4 classes, a regular tetrahedron (109.5^circ).

3. $mathcal{NC}_3$ **(Self-Duality / Alignment):** The linear classifier weights $W = [w_1, dots, w_C]^T$ align perfectly with the normalized class centroids:

$$frac{w_c}{|w_c|} = tilde{mu}_c, quad forall c in 1, dots, C$$

4. $mathcal{NC}_4$ **(Nearest Center Classification):** Given a test representation h, the decision rule simplifies to nearest Euclidean distance to class centroids: argmax_c langle w_c, h rangle = argmin_c |h – bar{h}_c|.

Neural Collapse demonstrates that at the limit of optimization, deep representation learning completely dissolves arbitrary architectural idiosyncrasies, converging to the unique, optimal geometric symmetry of the Simplex Equiangular Tight Frame.

Citation: Mahmud, S. (2026). "The Representation Theory of Deep Networks — Information Bottleneck & Neural Collapse." In Deep Learning Architectures & Representation Theory, Part 20. https://shakilmahmud.com/deep-learning-architectures-part-20-representation-theory-information-bottleneck-neural-collapse/

PRIMARY SOURCES & SCHOLARLY REFERENCES

Consolidated Bibliography

The formulations, historical contexts, and conceptual developments presented throughout this monograph draw upon the following primary treatises, foundational papers, and critical editions:

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press, Cambridge, MA.

LeCun, Y., Bengio, Y., & Hinton, G. (2015). *Deep learning*. Nature, 521(7553): 436–444.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). *Deep Residual Learning for Image Recognition*. CVPR 2016: 770–778.

Kingma, D. P., & Welling, M. (2013). *Auto-Encoding Variational Bayes*. arXiv:1312.6114 [stat.ML].

Bronstein, M. M., Bruna, J., Cohen, T., & Veličković, P. (2021). *Geometric Deep Learning: Grids, Groups, Graphs, Geodesics, and Gauges*. arXiv:2104.13478 [cs.LG].

Gu, A., & Dao, T. (2023). *Mamba: Linear-Time Sequence Modeling with Selective State Spaces*. arXiv:2312.00752 [cs.LG].

Shakil Mahmud • Research Monograph

BIOGRAPHICAL NOTE

About the Author

Shakil Mahmud

Independent Scholar, Writer & Systems Architect

Shakil Mahmud is an independent scholar, author, and software systems architect. His research interests span the foundations of theoretical physics, the history and philosophy of mathematics, evolutionary biology, macroeconomic history, and the architecture of neural computing. His work synthesizes deep historical rigor with accessible first-principles exposition. His complete published archives and monographs are maintained at **shakilmahmud.com**.

COLOPHON

Typeset in EB Garamond (English) and Kohinoor Bangla / Noto Serif Bengali (Bengali).
Mathematical typesetting generated with Donald Knuth's TeX algorithms and KaTeX.
Published digitally as a monolithic research monograph by Shakil Mahmud Editorial Series.
Dhaka & Singapore • 2026 • https://shakilmahmud.com