Artificial Intelligence Models
03/31/2026

The contemporary landscape of artificial intelligence is characterized by a plurality of modeling paradigms, each resting upon a distinct constellation of philosophical commitments, mathematical formalisms, and empirical assumptions. To engage these paradigms critically is not merely a technical exercise but an epistemological one, for the question of how a machine comes to "know" or "represent" the world is inseparable from deeper questions about the nature of knowledge itself.

A preliminary note on terminology is warranted. The term *model* is used throughout in its statistical sense: a parameterized mathematical function that approximates some target distribution or mapping. The term *representation* refers to the internal encoding of information within a model's parameter space. *Inductive bias* denotes the set of prior assumptions embedded in a model's architecture that cause it to favor certain hypotheses over others when learning from data.

Large Language Models

Fundamental Assumptions
The Large Language Model is, at its mathematical core, a probabilistic model of language. The decisive assumption underwriting the entire approach is that language can be adequately characterized as a probability distribution over sequences of tokens, where a *token* is an atomic unit of text — typically a subword segment of two to eight characters. The model learns to estimate the conditional probability:

$$P(w_t \mid w_1, w_2, \ldots, w_{t-1})$$

where w_t denotes the token at position t in the sequence. This is the *autoregressive* assumption: the probability of each next token is conditioned on all preceding tokens, and the full probability of a sequence decomposes by the chain rule of probability:

$$P(w_1, w_2, \ldots, w_T) = \prod_{t=1}^{T} P(w_t \mid w_1, \ldots, w_{t-1})$$

This formulation carries an implicit philosophical claim of considerable weight: that meaning in language is sufficiently encoded in distributional co-occurrence patterns across an enormous corpus. The LLM implicitly wagers that sufficiently large corpora and sufficiently expressive architectures render this objection moot in practice, even if not in principle.

A second foundational assumption is the *Markov-like tractability assumption*: that a finite context window suffices to capture the probabilistic dependencies relevant to predicting the next token. The Transformer architecture addresses this through the *self-attention mechanism*, which allows every token in a context of length N to attend to every other token with complexity $\mathrm{O}(\mathrm{N}^2)$.

Mathematical Basis
The Transformer processes an input sequence by first projecting each token into a continuous vector space via an embedding matrix $\mathrm{E} \in \mathbb{R}^{|V| \times d}$, where $|V|$ is the vocabulary size and d is the embedding dimension. Position information is incorporated through *positional encodings* PE, yielding the representation:

$$\mathbf{h}_t^{(0)} = E_{w_t} + \mathrm{PE}(t)$$

The core computational unit is the *multi-head self-attention* mechanism. For a single attention head, queries Q, keys K, and values V are computed as linear projections of the hidden state:

$$Q = HW_Q, \quad K = HW_K, \quad V = HW_V$$

where H is the matrix of hidden states and $\mathrm{W}_Q, W_K, W_V \in \mathbb{R}^{d \times d_k}$ are learned weight matrices. The attention output is:

$$\mathrm{Attention}(Q, K, V) = \mathrm{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right) V$$

The division by $\sqrt{d_k}$, where d_k is the key dimension, stabilizes gradients during training by preventing the dot products from growing large enough to saturate the *softmax* function. In multi-head attention, n_h independent attention operations are performed in parallel and their outputs concatenated:

$$\mathrm{MultiHead}(H) = \mathrm{Concat}(\mathrm{head}_1, \ldots, \mathrm{head}_{n_h}) W_O$$

Training proceeds by minimizing the *cross-entropy loss* over the training corpus D:

$$\mathcal{L} = -\sum_{(w_1, \ldots, w_T) \in \mathcal{D}} \sum_{t=1}^{T} \log P_\theta(w_t \mid w_1, \ldots, w_{t-1})$$

where θ represents the model's parameters. Optimization is performed via stochastic gradient descent, with variants such as Adam being standard. Modern LLMs contain between 7×10^9 and 10^{12} parameters, and training requires computing resources measured in thousands of GPU-years.

The phenomenon of *emergent capabilities* — where qualitatively new behaviors appear above certain parameter scales — represents one of the most philosophically provocative observations in recent AI research. It suggests that the relationship between model capacity and capability is not always smooth and linear, but can exhibit phase-transition-like discontinuities. Whether these emergences represent genuine novelty or merely the crossing of capability thresholds detectable by our evaluation benchmarks remains contested.

Philosophical Tensions

The LLM paradigm is subject to several deep philosophical objections. The first is the *grounding problem*: the internal representations of an LLM are anchored to other textual tokens rather than to perceptual experience of the world. Symbols in such a system refer, ultimately, only to other symbols; whether symbol manipulation according to syntactic rules can ever constitute genuine semantic understanding. The LLM offers no principled resolution to this critique, only the pragmatic observation that the behavior engendered by such systems is often indistinguishable from understanding in practical contexts.

The second tension concerns *hallucination* — the tendency of LLMs to produce confident, fluent, and entirely fabricated claims. This is not a contingent engineering defect but a structural consequence of the probabilistic nature of the model: it generates text that is locally coherent given its training

distribution, without any direct mechanism for verifying the correspondence of its outputs to external facts. The model optimizes for linguistic plausibility, not factual accuracy.

Practical Applications
LLMs have demonstrated extraordinary practical utility in natural language processing tasks including open-domain question answering, document summarization, code generation, translation, and conversational assistance. They serve as the backbone of production systems at organizations including *Anthropic, OpenAI, Google DeepMind, and Meta*. Their limitations in rigorous factual recall, multi-step quantitative reasoning, and long-horizon planning remain active research frontiers.

World Models

Fundamental Assumptions
The *World Model* paradigm is grounded in a fundamentally different set of philosophical commitments from the LLM. Whereas LLMs treat intelligence as emerging from the compression and reproduction of surface patterns in data, World Models posit that intelligence requires the construction of an *internal model of environmental dynamics* — a latent representation from which an agent can simulate the consequences of hypothetical actions without executing them in the real world.

The central assumption is the *Markov Decision Process* (MDP) structure of the environment. An MDP is a tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma \rangle$, where S is the state space, A is the action space, $\mathrm{T}\colon \mathrm{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition function mapping state-action pairs to distributions over next states ($\Delta(\mathcal{S})$ denoting the set of probability distributions over S), $\mathrm{R}\colon \mathrm{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, and $\gamma \in [0, 1)$ is the discount factor governing the relative weight of future versus immediate rewards.

The World Model hypothesis further assumes that the environment's true state space is high-dimensional and partially observable, necessitating the learning of a *compressed latent space* Z that retains only those features of the state relevant to predicting future observations and rewards.

Mathematical Basis
A World Model typically comprises three learned components: an *encoder* $\mathrm{e}_\phi : \mathcal{O} \to \mathcal{Z}$, a *transition model* $\hat{T}_\psi : \mathcal{Z} \times \mathcal{A} \to \Delta(\mathcal{Z})$, and a *decoder* $\mathrm{d}_\xi : \mathcal{Z} \to \hat{\mathcal{O}}$. Here O denotes the observation space (*e.g.*, raw image pixels) and $\hat{\mathcal{O}}$ the reconstructed observation. The encoder and decoder together constitute a *variational autoencoder* (VAE), whose training objective is the *Evidence Lower Bound* (ELBO):

$$\mathcal{L}_{\mathrm{VAE}} = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{o})}[\log p_\xi(\mathbf{o}|\mathbf{z})] - D_{\mathrm{KL}}\big(q_\phi(\mathbf{z}|\mathbf{o}) \,\|\, p(\mathbf{z})\big)$$

In this expression, $\mathrm{q}_\phi(\mathbf{z}|\mathbf{o})$ is the approximate posterior distribution over latent states given an observation, $\mathrm{p}_\xi(\mathbf{o}|\mathbf{z})$ is the likelihood of reconstructing the observation from a latent state, and D_{KL} is the *Kullback-Leibler* divergence — a measure of how much one probability distribution diverges from a second reference distribution. The KL term acts as a regularizer, preventing the learned latent space from becoming a trivial lookup table.

The transition model $\hat{T}_\psi$ is learned by minimizing prediction error over rollouts in latent space. The agent then uses a separate controller $\pi : \mathcal{Z} \to \mathcal{A}$, which may be an evolved or gradient-trained policy, to select actions by imagining forward trajectories entirely within the learned latent model:

$$a_t = \arg\max_a \mathbb{E}_{\hat{T}_\psi}\left[\sum_{k=0}^{K} \gamma^k \hat{R}(\mathbf{z}_{t+k}, a_{t+k})\right]$$

This *planning in imagination* is computationally efficient because it substitutes cheap latent-space rollouts for expensive real-world interactions. The *DreamerV3* system, developed at *Google DeepMind*, extends this formalism by training actor and critic networks entirely within the world model's imagined trajectories using *RSSM* (Recurrent State Space Model) dynamics, achieving remarkable generalization across diverse domains with a single set of hyperparameters.

Philosophical Tensions
The World Model paradigm inherits the epistemological tensions of model-based reasoning. The most acute is the *Goodhart problem*: when the world model is imperfect, policies optimized against it will exploit its inaccuracies — a phenomenon known as *model exploitation* or *compounding hallucination*. A policy may discover trajectories that appear highly rewarding within the model but are catastrophic in reality, precisely because the model has not seen those trajectories during training. This creates a fundamental epistemic tension: the model must be expressive enough to generalize, but conservative enough not to be gamed. A deeper philosophical issue concerns the ontological status of the latent space $\mathcal{Z}$. The encoder maps observations to a compact representation, but there is no guarantee that this representation aligns with any humanly interpretable decomposition of the world. The latent dimensions are not interpretable by construction; they are whatever configurations of activations minimize the training loss. This raises questions about whether such models can serve as bases for robust, transferable, and transparent reasoning.

Practical Applications
World Models have achieved state-of-the-art results in sample-efficient reinforcement learning for video game domains (Atari, DM Control), robotic manipulation in simulation, and autonomous vehicle policy learning. The autonomous driving company *Wayve* has publicly employed world model architectures as the representational backbone of their end-to-end driving systems. In model-based planning for protein structure design and materials discovery, related ideas have been adapted to molecular dynamics prediction.

Diffusion Models

Fundamental Assumptions
Diffusion models — formally *Denoising Diffusion Probabilistic Models* (DDPMs), then elaborated through *score matching* and *stochastic differential equations* — represent a distinct approach to generative modeling. The core assumption is that complex high-dimensional data distributions $p(\mathbf{x})$ can be learned by training a neural network to reverse a gradual noising process.

The philosophical premise is that structure and order are more easily created by the systematic removal of disorder than by direct construction. This intuition — that learning to denoise is equivalent to learning the data distribution — is formalized through the connection to *Stein's score function*: the gradient of the log-probability density $\nabla_{\mathbf{x}} \log p(\mathbf{x})$.

Mathematical Basis
The *forward process* is a fixed *Markov* chain that progressively corrupts a data point $\mathbf{x}_0 \sim p(\mathbf{x})$ by adding *Gaussian* noise over T timesteps:

$$q(\mathbf{x}_t \mid \mathbf{x}_{t-1}) = \mathcal{N}\Big(\mathbf{x}_t; \sqrt{1-\beta_t}\,\mathbf{x}_{t-1},\, \beta_t \mathbf{I}\Big)$$

where $\{\beta_t\}_{t=1}^{T}$ is a fixed variance schedule. By reparameterization, one can sample x_t at any timestep directly from x_0:

$$\mathbf{x}_t = \sqrt{\bar{\alpha}_t}\,\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon}, \quad \boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$$

where $\bar{\alpha}_t = \prod_{s=1}^{t}(1-\beta_s)$. As $t \to T$, the distribution approaches a *Gaussian*, $q(\mathbf{x}_T) \approx \mathcal{N}(\mathbf{0}, \mathbf{I})$.

The *reverse process* is a neural-network-parameterized *Markov* chain trained to undo the forward corruption:

$$p_\theta(\mathbf{x}_{t-1} \mid \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1};\, \boldsymbol{\mu}_\theta(\mathbf{x}_t, t),\, \Sigma_\theta(\mathbf{x}_t, t))$$

The training objective simplifies to a *denoising score matching* loss:

$$\mathcal{L}_{\text{DDPM}} = \mathbb{E}_{t, \mathbf{x}_0, \boldsymbol{\epsilon}}\big[\|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\,\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\,\boldsymbol{\epsilon},\, t)\|^2\big]$$

where $\boldsymbol{\epsilon}_\theta$ is the neural network's prediction of the noise added at step t. This formulation reveals a profound equivalence: predicting the noise is equivalent to estimating the score function $\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t)$, which encodes the direction of steepest ascent in log-probability space. Generation proceeds by sampling pure noise and iteratively applying the learned reverse transitions.

Philosophical Tensions and Practical Applications
The diffusion paradigm raises important questions about the nature of creativity and originality: since the model learns a distribution over its training data, its samples are, in a rigorous statistical sense, instances of that learned distribution rather than genuinely novel creations. This carries legal and ethical implications regarding copyright and the provenance of training data that are currently being adjudicated in courts across multiple jurisdictions. Moreover, the iterative sampling procedure — requiring hundreds to thousands of neural network forward passes per generated sample — is computationally costly, though distillation techniques are beginning to address this limitation. Practical applications are extensive: text-to-image synthesis (*Stable Diffusion*, *DALL-E 3*, *Imagen*), audio generation, video generation, protein structure prediction (closely related diffusion-over-structure approaches), and drug molecule design all leverage diffusion model principles.

Reinforcement Learning Agents

Fundamental Assumptions
Reinforcement learning (RL) adopts perhaps the most explicit connection to animal learning theory among AI paradigms. It models an agent operating within an environment according to an MDP, seeking to maximize the expected sum of discounted rewards:

$$J(\pi) = \mathbb{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t)\right]$$

The fundamental assumption is that intelligence can be characterized as reward-maximizing behavior in an environment that provides a scalar feedback signal. This assumption, while computationally tractable, is philosophically tendentious: it reduces the richness of intelligent behavior to a single scalar quantity, and it assumes that the designer can specify the correct reward function. The latter assumption is precisely what the field of *Inverse Reinforcement Learning* (IRL) and *Reward Modeling* was developed to relax — the insight being that in many settings, specifying the reward is as hard as solving the original problem.

Mathematical Basis
The central object is the *value function*

$$V^{\pi}(s) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^{t} R(s_t, a_t) \mid s_0 = s\right]$$

which satisfies the *Bellman equation*:

$$V^{\pi}(s) = \sum_{a} \pi(a|s) \left[R(s,a) + \gamma \sum_{s'} T(s'|s,a) V^{\pi}(s')\right]$$

The *Q-function* $Q^{\pi}(s,a)$ extends the value function to state-action pairs, and the *Bellman optimality equation* characterizes the optimal policy:

$$Q^{*}(s,a) = R(s,a) + \gamma \sum_{s'} T(s'|s,a) \max_{a'} Q^{*}(s',a')$$

In deep reinforcement learning, Q^{*} is approximated by a neural network $\mathrm{Q}_{\theta}(s,a)$ trained via temporal difference learning. Policy gradient methods such as *Proximal Policy Optimization* (PPO) instead directly parameterize the policy π_{θ} and optimize $\mathrm{J}(\pi_{\theta})$ using gradient estimators of the form:

$$\nabla_{\theta} J(\pi_{\theta}) = \mathbb{E}_{\pi_{\theta}}[\nabla_{\theta} \log \pi_{\theta}(a_t|s_t) \cdot A^{\pi}(s_t, a_t)]$$

where $A^{\pi}(s,a) = Q^{\pi}(s,a) - V^{\pi}(s)$ is the *advantage function*, measuring how much better action a is compared to the average action under policy π in state s. PPO is the algorithm currently underlying the *Reinforcement Learning from Human Feedback* (RLHF) procedure that fine-tunes LLMs such as *GPT-4* and *Claude* to follow instructions and align with human preferences.

Philosophical Tensions
Reinforcement learning is subject to the *reward hacking* or *specification gaming* problem, wherein agents find unintended ways to achieve high reward that violate the designer's intent. This is not a mere engineering inconvenience but a fundamental instance of *Goodhart's Law* — that any measure, once it becomes a target, ceases to be a good measure. RL has produced landmark results in game-playing (*AlphaGo, AlphaZero, OpenAI Five*), robotic locomotion, chip floorplanning, and logistics optimization. Its application to open-ended physical robotics and real-world decision making remains a research frontier.

Graph Neural Networks

Fundamental Assumptions and Mathematical Basis
Graph Neural Networks (GNNs) are built on the assumption that many natural and social systems are more accurately represented as *relational structures* — sets of entities connected by typed relations —

than as fixed-dimensional vectors. A graph $G = (V, E)$ consists of nodes $v_i \in V$ with feature vectors $\mathbf{h}_i$ and edges $(v_i, v_j) \in E$ with optional edge features $\mathbf{e}_{ij}$. GNNs operate by iteratively aggregating information from a node's local neighborhood:

$$\mathbf{h}_i^{(l+1)} = \phi\left(\mathbf{h}_i^{(l)}, \bigoplus_{j \in \mathcal{N}(i)} \psi\left(\mathbf{h}_i^{(l)}, \mathbf{h}_j^{(l)}, \mathbf{e}_{ij}\right)\right)$$

where $\mathcal{N}(i)$ denotes the neighborhood of node i, ψ is a message function, $\bigoplus$ is a permutation-invariant aggregation operator (typically sum, mean, or max), and ϕ is an update function. Both ψ and ϕ are neural networks. This *message passing* formalism unifies many GNN variants under a common theoretical framework.

The inductive bias is explicit: nodes that are nearby in the graph should have similar representations, and the model is equivariant to permutations of node indices. This structural prior is highly appropriate for domains where entities are genuinely relational — molecular graphs, social networks, knowledge graphs, physical simulations governed by pairwise forces.

A notable limitation is the *over-smoothing* phenomenon: as the number of message-passing layers increases, node representations converge to indistinct averages, losing local discriminative information. This constrains the effective depth of GNNs and has motivated architectures that preserve heterogeneity through skip connections and attention-weighted aggregation.

Philosophical Tensions
GNNs raise the interesting philosophical question of whether the relational structure of a problem should be treated as part of the model's prior or as a feature to be learned from data. In most applications, the graph topology is given by domain knowledge (the covalent bonds in a molecule, the follow-links in a social network), which is a form of prior knowledge injection that purely data-driven approaches forego. This makes GNNs highly performant in their intended domains but inapplicable outside them. Practical applications include molecular property prediction and drug discovery (where atoms are nodes and bonds are edges), combinatorial optimization, traffic flow prediction, recommendation systems, and particle physics simulation.

Concluding Remarks

It would be a philosophical error to regard any of these paradigms as inherently superior. Each constitutes a bet — a structured wager about which aspects of the world and of intelligence it is most important to model. LLMs bet on the sufficiency of language statistics; World Models bet on the necessity of internal simulation; Reinforcement Learning bets on the primacy of reward-maximizing agency; Diffusion Models bet on the tractability of learning by denoising. These bets are not mutually exclusive, and the most capable contemporary AI systems — such as those undergirding autonomous vehicle stacks, drug discovery pipelines, and advanced code-generation assistants — increasingly integrate multiple paradigms in a single computational architecture.