# Memory in Plain Sight

## A Survey of the Uncanny Resemblances between Diffusion Models and Associative Memories

**Benjamin Hoover**
IBM Research
Georgia Tech
benjamin.hoover@ibm.com

**Hendrik Strobelt**
MIT-IBM Watson AI Lab
IBM Research

**Dmitry Krotov**
MIT-IBM Watson AI Lab
IBM Research

**Judy Hoffman**
Georgia Tech

**Zsolt Kira**
Georgia Tech

**Duen Horng Chau**
Georgia Tech

## Abstract

Diffusion Models (DMs) have recently set state-of-the-art on many generation benchmarks. However, there are myriad ways to describe them mathematically, which makes it difficult to develop a simple understanding of how they work. In this submission, we provide a concise overview of DMs from the perspective of dynamical systems and Ordinary Differential Equations which exposes a mathematical connection to the highly related yet often overlooked class of energy-based models, called Associative Memories (AMs). Energy-based AMs are a theoretical framework that behave much like denoising DMs, but they enable us to *directly compute* a Lyapunov energy function on which we can perform gradient descent to denoise data. We finally identify the similarities and differences between AMs and DMs, discussing new research directions revealed by the extent of their similarities.

## 1 Introduction

Diffusion Models [1, 2, 3, 4] (DMs) have rapidly become the most perfomant class of generative models on images [5, 6, 7, 8, 9]. However, they operate differently than previous image generation models e.g.,GANs [10] and VAEs [11] that decode a latent input into a plausible image using a single pass through some network. Instead, a DM is trained to predict and remove a known amount of noise that was added to an image, a process that is applied recurrently at inference time to turn noisy images into clean ones; to generate an image a DM iteratively denoises a sample of pure noise. The goal of DMs can thus be stated as follows:

**Goal of Diffusion Models**: *Given a corrupted representation of some data, recreate the original uncorrupted data.*

Each denoising step aims to increase the probability (specifically, the *log-probability*) that the noisy data looks like a sample from the real data distribution. An example log-probability landscape with two peaks of "real-looking data" (located at the top right and bottom left of the landscape) is shown in the left half of Figure 1, where each denoising step pushes the initial noisy image (blue dot) towards regions of higher log-probability.

The *forward process* of DMs iteratively adds noise to some image (or other data), whereas the *reverse process* is trained to remove the noise introduced at each step of the forward process in an attempt to recreate the original image. The forward process on its own is un-parameterized and non-interesting; the computational power of Diffusion Models lies entirely in its reverse process.

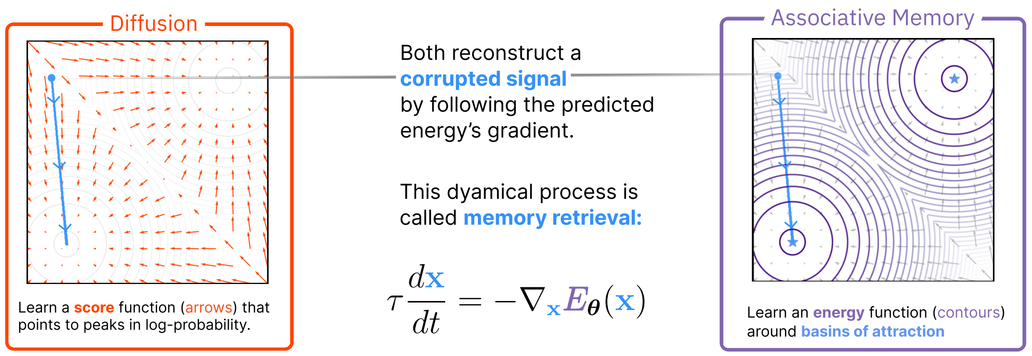
Diffusion
Learn a score function (arrows) that points to peaks in log-probability.
Both reconstruct a corrupted signal by following the predicted energy's gradient.
This dyamical process is called memory retrieval:
$\tau \frac{d\mathbf{x}}{dt} = -\nabla_{\mathbf{x}} E_{\boldsymbol{\theta}}(\mathbf{x})$
Associative Memory
Learn an energy function (contours) around basins of attraction

Figure 1: Comparing the emphases of Diffusion Models and Associative Memories tasked with learning the same energy (negative log-probability) landscape, represented with both contours and gradient arrows. Diffusion Models (left) train a score function (depicted as orange arrows) to model the gradient of the energy. The noisy starting signal (depicted as a blue circle) becomes less corrupted by following these gradients in the reverse denoising process. Associative Memories (right) instead learn a smooth energy function, depicted as contours. The "memory retrieval dynamics" is the process by which a fixed point is retrieved by following the energy gradient from the initial signal. *This process is mathematically equivalent to the objective of the reverse denoising process of Diffusion Models*. Memory retrieval dynamics always converge to fixed points (there are two in the figure, the top right and lower left) where the energy is at a local minimum. This guarantee does not exist for Diffusion Models.

### 1.1 Diffusion's Unseen Connection to Associative Memories

Associative Memories (AMs) are dynamical systems that are concerned with the storage and retrieval of data called *memories* [12]. These memories live at local minima of an energy landscape that also includes *all possible corruptions of those memories*. From the Boltzmann distribution (Eq. 1), we know that energy can be understood as a negative log-probability, where local peaks in the original probability distribution correspond exactly to local minima in the energy. The right side of Figure 1 shows an example energy landscape with two memories representing "real-looking data" (valleys located at the top right and bottom left of the landscape). *Memory retrieval is the process of extracting information from the model (thus reconstructing our initial signal) by descending the energy* according to the dynamical equation in the center of Figure 1.

The goal of Associative Memories as stated above is identical to that of DMs:

> **Goal of Associative Memories**: *Given a corrupted representation of some data, recreate the original uncorrupted data.*

Yet, though Diffusion Models (or score-based models in general) have been related to Markovian VAEs [13, 4, 1], Normalizing Flows [14], Neural ODEs [15], and Energy Based Models [2, 16], an explicit connection to Associative Memories has not been acknowledged. Such a connection would contribute to a growing body of literature that seeks to use modern AI techniques to unravel the mysteries of memory and cognition [17, 18, 19, 20, 21, 22, 23].

*See Appendix A for a description of all mathematical notation used in this paper.*

## 2 Denoising Diffusion is Memory Retrieval

This section is a gentle introduction to show how denoising diffusion is mathematically equivalent to energy-based memory retrieval under certain assumptions. Please see Table 1 and Appendix B for a detailed exploration of the nuances of this connection.

Diffusion Models use their parameters $\boldsymbol{\theta}$ to directly model the **score function** $\mathbf{F}_{\boldsymbol{\theta}}(\mathbf{x})$ (Eq. 2) of some data distribution $p_{\boldsymbol{\theta}}$ [15, 24, 25]. That is, starting from the Boltzmann Distribution

$$p_{\boldsymbol{\theta}}(\mathbf{x}) = \frac{e^{-E_{\boldsymbol{\theta}}(\mathbf{x})}}{Z_{\boldsymbol{\theta}}} \tag{1}$$

the score function is defined as the gradient of the log-likelihood, or equivalently, the negative gradient of the energy [2].

$$\mathbf{F}_{\boldsymbol{\theta}}(x) = \nabla_{\mathbf{x}} \log p_{\boldsymbol{\theta}}(\mathbf{x}) = -\nabla_{\mathbf{x}} E_{\boldsymbol{\theta}}(\mathbf{x}) - \nabla_{\mathbf{x}} \log Z_{\boldsymbol{\theta}} = -\nabla_{\mathbf{x}} E_{\boldsymbol{\theta}}(\mathbf{x}) \tag{2}$$

In both equations, $Z_{\boldsymbol{\theta}}$ is the partition function (a.k.a. normalizing constant) to enforce that the $\int p_{\boldsymbol{\theta}}(\mathbf{x}) d\mathbf{x} = 1$, and $E_{\boldsymbol{\theta}}(\mathbf{x})$ is the **energy function** (a.k.a. unnormalized probability function).

Figure 1 depicts the score function as vectors pointing to peaks in the log probability (equivalently, local minima in the energy function). In practice, we often think of the score function as predicting the noise we need to remove from noisy image $\mathbf{x}^t$. Thus, generating images from DMs can be construed as an iterative procedure that repeatedly removes predicted noise $\mathbf{F}_{\boldsymbol{\theta}}(\mathbf{x}^t)$ for some fixed number of steps $T$ (see Eq. 3). The final state $\mathbf{x}^T$ is declared to be a local peak (minimum) of the log-likelihood (energy) and is the model's "best guess" at a realistic sample drawn from the original distribution $p$. In practice, $\mathbf{F}_{\boldsymbol{\theta}}(\mathbf{x}^t; t)$ is additionally conditioned on time $t$ to help predict how much noise to remove. Formally,

$$\mathbf{x}^{t+1} = \mathbf{x}^t + \alpha(t) \mathbf{F}_{\boldsymbol{\theta}}(\mathbf{x}^t; t), \qquad t = 0, ..., T-1 \tag{3}$$

where $\alpha(t) \in \mathbb{R}$ is the "scheduled" amount of noise to remove at each step and is typically set during training. Note that Eq. 3 is described using the convention that *time progresses forward when reconstructing the data*. However, the literature around DMs describes *time in the reconstruction process as going backwards*, denoting $\mathbf{x}^T$ to refer to the sample drawn from pure noise and $\mathbf{x}^0$ to refer to the final reconstructed sample drawn from $p$ [1, 2, 4, 13, 15]. Eq. 4 rewrites Eq. 3 using the variable $s \triangleq T - t$ to represent the reverse-time convention used in most DM literature.

$$\mathbf{x}^{s-1} = \mathbf{x}^s + \alpha(s) \mathbf{F}_{\boldsymbol{\theta}}(\mathbf{x}^s; s), \qquad s = T, ..., 1 \tag{4}$$

[15, 26] use a *probability flow Ordinary Differential Equation* (PF-ODE) formulation for DMs to define a differential equation to noisify images (Eq. 5) using an infinitesimal amount of noise $\frac{d\mathbf{w}}{ds}$ scaled by some real-valued, time-dependent *diffusion coefficient* $\sigma(s)$.

$$\frac{d\mathbf{x}}{ds} = \sigma(s) \frac{d\mathbf{w}}{ds} \tag{5}$$

Reversing this corruption process is also a differential equation, depending only on the noise scale $\sigma$ and the score $\mathbf{F}_{\boldsymbol{\theta}}$ [15, 27].

$$\frac{d\mathbf{x}}{ds} = -\frac{1}{2} \sigma(s)^2 \mathbf{F}_{\boldsymbol{\theta}}(\mathbf{x}; s) \tag{6}$$

Eq. 7 rewrites Eq. 6 using forward time $t = T - s$, collecting the noise scale into a real-valued time-variable $\tau(s) \triangleq \frac{2}{\sigma(s)^2}$ to control the rate of change, and substituting $\mathbf{F}_{\boldsymbol{\theta}} = -\nabla_{\mathbf{x}} E_{\boldsymbol{\theta}}$.

$$\tau(s) \frac{d\mathbf{x}}{dt} = -\nabla_{\mathbf{x}} E_{\boldsymbol{\theta}}(\mathbf{x}; s), \qquad t \in [0, T] \tag{7}$$

*This is the definition of Associative Memory retrieval dynamics* when constraining time $t \in [0, T]$ [28, 29] and conditioning the dynamics on time. More details on AMs are included in Appendix C.

Table 1: Summarizing the similarities and differences between Diffusion Models and Associative Memory. Fields marked with a $^*$ indicate nuances that are clarified in § B.1 and § B.2.

| | **Diffusion** | **Associative Memory** |
|---|---|---|
| Parameterizes the... | Score function $\mathbf{F}_{\boldsymbol{\theta}}$ | Energy function $E_{\boldsymbol{\theta}}$ |
| Continuous Update$^*$ | $\tau(t)\frac{d\mathbf{x}}{dt} = \mathbf{F}_{\boldsymbol{\theta}}(\mathbf{x}; t)$ | $\tau\frac{d\mathbf{x}}{dt} = -\nabla_{\mathbf{x}} E_{\boldsymbol{\theta}}(\mathbf{x})$ |
| Discrete Update$^*$ | $\mathbf{x}^{t+1} = \mathbf{x}^t + \alpha(t)\mathbf{F}_{\boldsymbol{\theta}}(\mathbf{x}^t; t)$ | $\mathbf{x}^{t+1} = \mathbf{x}^t - \alpha\nabla_{\mathbf{x}} E_{\boldsymbol{\theta}}(\mathbf{x}^t)$ |
| Valid Time Domain | $t \in [0, T]$ | $t \geq 0$ |
| Fixed Point Attractor? | No$^*$ | Yes |
| Tractable Energy? | No$^*$ | Yes |
| Energy is Lyapunov? | No | Yes |
| Undoes Corruption of ... | Noise it was trained on$^*$ | Any kind |

## 3 Memory Retrieval is more than Denoising Diffusion

Special care needs to be prescribed to the design of a DM for it to be considered an AM. Namely, the score function $\mathbf{F}_{\boldsymbol{\theta}}$ must be the exact derivative of an energy function $E_{\boldsymbol{\theta}}$ that is Lyapunov (i.e., a function guaranteeing stable critical points). This implies that the dynamics of a true AM must be defined for all time $t > 0$, and the system (by nature of its architectural design) will always converge to fixed points $\mathbf{x}^\star$ (the *memories*) located at local minima of $E_{\boldsymbol{\theta}}$. That is,

$$\begin{aligned} &\frac{d\mathbf{x}^\star}{dt} = 0\ ; \quad \text{and} \\ &\nabla_{\mathbf{x}} E_{\boldsymbol{\theta}}(\mathbf{x}^\star + \delta) > 0 \qquad \text{for sufficiently small } \delta \neq 0. \end{aligned}$$

(where $\delta$ is a small perturbation of the memory). We tabulate the similarities and differences between DMs and AMs in Table 1, providing additional clarifications in Appendix B.

## 4 Conclusions

Diffusion Models and Associative Memories have remarkable similarities when presented using a unified mathematical notation: both aim to minimize some energy (maximize some log-probability) by following its gradient (score). The final solution of each approach represents some sort of *memory* that lies in a local minimum (maximum) of the energy (log-probability). However, these approaches are certainly not identical, as is evidenced by different validity constraints on architecture choices and time domains. The training philosophy behind each approach is also different: Diffusion Models assume that the energy function is intractable and fixate on training the gradient of the energy using known perturbations of training data as the objective, while AMs can instead focus on learning the fixed points of a tractable energy. See Appendix D for discussion on how this connection could affect future research into both fields.

Very few researchers will observe the rapid advances of AI today and notice a trend towards the dynamical processes of Associative Memories first established by John Hopfield in the 1980s. However, many of the theoretical guarantees of Associative Memories are captured in the design of increasingly popular Diffusion Models that have proven themselves fixtures for many applications of generative modeling. This paper represents a first step towards a more comprehensive understanding of the connections between Diffusion Models and Associative Memories. We hope that our work inspires further research into these exciting fields and that it helps to foster a new generation of AI systems that are capable of unlocking the secrets of memory and perception.

## References

[1] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep Unsupervised Learning using Nonequilibrium Thermodynamics, November 2015.

[2] Yang Song and Stefano Ermon. Generative Modeling by Estimating Gradients of the Data Distribution. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[3] Yang Song and Stefano Ermon. Improved Techniques for Training Score-Based Generative Models. In *Advances in Neural Information Processing Systems*, volume 33, pages 12438–12448. Curran Associates, Inc., 2020.

[4] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020.

[5] Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion Models Beat GANs on Image Synthesis. In *Advances in Neural Information Processing Systems*, November 2021.

[6] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical Text-Conditional Image Generation with CLIP Latents, April 2022.

[7] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Raphael Gontijo-Lopes, Burcu Karagol Ayan, Tim Salimans, Jonathan Ho, David J. Fleet, and Mohammad Norouzi. Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding. In *Advances in Neural Information Processing Systems*, May 2022.

[8] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10684–10695, 2022.

[9] Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models, March 2022.

[10] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[11] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. December 2013.

[12] Dmitry Krotov and John J. Hopfield. Large associative memory problem in neurobiology and machine learning. In *International Conference on Learning Representations*, 2021.

[13] David McAllester. On the Mathematics of Diffusion Models, February 2023.

[14] Will Grathwohl, Ricky T. Q. Chen, Jesse Bettencourt, Ilya Sutskever, and David Duvenaud. FFJORD: Free-Form Continuous Dynamics for Scalable Reversible Generative Models. In *International Conference on Learning Representations*, September 2018.

[15] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-Based Generative Modeling through Stochastic Differential Equations. In *International Conference on Learning Representations*, October 2020.

[16] Phillip Lippe. Tutorial 7: Deep Energy-Based Generative Models, September 2021.

[17] Yu Takagi and Shinji Nishimoto. High-resolution image reconstruction with latent diffusion models from human brain activity. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14453–14463, 2023.

[18] James C. R. Whittington, Joseph Warren, and Tim E.J. Behrens. Relating transformers to models and neural representations of the hippocampal formation. In *International Conference on Learning Representations*, 2022.

[19] Jiakun Fu, Suhas Shrinivasan, Kayla Ponder, Taliah Muhammad, Zhuokun Ding, Eric Wang, Zhiwei Ding, Dat T. Tran, Paul G. Fahey, Stelios Papadopoulos, Saumil Patel, Jacob Reimer, Alexander S. Ecker, Xaq Pitkow, Ralf M. Haefner, Fabian H. Sinz, Katrin Franke, and Andreas S. Tolias. Pattern completion and disruption characterize contextual modulation in mouse visual cortex, March 2023.

[20] Furkan Ozcelik and Rufin VanRullen. Brain-Diffuser: Natural scene reconstruction from fMRI signals using generative latent diffusion, March 2023.

[21] Aria Y. Wang, Kendrick Kay, Thomas Naselaris, Michael J. Tarr, and Leila Wehbe. Incorporating natural language into vision models improves prediction and understanding of higher visual cortex, September 2022.

[22] Arjun Majumdar, Karmesh Yadav, Sergio Arnaud, Yecheng Jason Ma, Claire Chen, Sneha Silwal, Aryan Jain, Vincent-Pierre Berges, Pieter Abbeel, Dhruv Batra, Yixin Lin, Oleksandr Maksymets, Aravind Rajeswaran, and Franziska Meier. Where are we in the search for an Artificial Visual Cortex for Embodied Intelligence? In *Workshop on Reincarnating Reinforcement Learning at ICLR 2023*, March 2023.

[23] Leo Kozachkov, Ksenia V Kastanenka, and Dmitry Krotov. Building transformers from neurons and astrocytes. *Proceedings of the National Academy of Sciences*, 120(34):e2219150120, 2023.

[24] Roland S. Zimmermann, Lukas Schott, Yang Song, Benjamin Adric Dunn, and David A. Klindt. Score-Based Generative Classifiers. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, December 2021.

[25] Tim Salimans and Jonathan Ho. Should EBMs model the energy or the score? In *Energy Based Models Workshop - ICLR 2021*, April 2021.

[26] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the Design Space of Diffusion-Based Generative Models. In *Advances in Neural Information Processing Systems*, May 2022.

[27] Yang Song, Prafulla Dhariwal, Mark Chen, and Ilya Sutskever. Consistency Models. In *International Conference for Machine Learning*, January 2023.

[28] John Hopfield. Neurons With Graded Response Have Collective Computational Properties Like Those of Two-State Neurons. *Proceedings of the National Academy of Sciences of the United States of America*, 81:3088–92, June 1984.

[29] Dmitry Krotov and John J. Hopfield. Dense associative memory for pattern recognition. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[30] Nicolas Carlini, Jamie Hayes, Milad Nasr, Matthew Jagielski, Vikash Sehwag, Florian Tramer, Borja Balle, Daphne Ippolito, and Eric Wallace. Extracting training data from diffusion models. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 5253–5270, 2023.

[31] Vikram Voleti, Christopher Pal, and Adam Oberman. Score-based Denoising Diffusion with Non-Isotropic Gaussian Noise Models, November 2022.

[32] Eliya Nachmani, Robin San Roman, and Lior Wolf. Non Gaussian Denoising Diffusion Models, June 2021.

[33] Arpit Bansal, Eitan Borgnia, Hong-Min Chu, Jie S. Li, Hamid Kazemi, Furong Huang, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Cold Diffusion: Inverting Arbitrary Image Transforms Without Noise, August 2022.

[34] Fan Bao, Chongxuan Li, Yue Cao, and Jun Zhu. All are Worth Words: A ViT Backbone for Score-based Diffusion Models. In *NeurIPS 2022 Workshop on Score-Based Methods*, November 2022.

[35] Xiulong Yang, Sheng-Min Shih, Yinlin Fu, Xiaoting Zhao, and Shihao Ji. Your ViT is Secretly a Hybrid Discriminative-Generative Diffusion Model, August 2022.

[36] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *International Conference on Learning Representations*, October 2020.

[37] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Lecture Notes in Computer Science, pages 234–241, Cham, 2015. Springer International Publishing.

[38] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.

[39] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. CIFAR-10 (Canadian Institute for Advanced Research).

[40] Dmitry Krotov. Hierarchical Associative Memory, July 2021.

[41] Benjamin Hoover, Duen Horng Chau, Hendrik Strobelt, and Dmitry Krotov. A Universal Abstraction for Hierarchical Hopfield Networks. In *The Symbiosis of Deep Learning and Differential Equations II*, October 2022.

[42] Geoffrey E. Hinton. Training Products of Experts by Minimizing Contrastive Divergence. *Neural Computation*, 14(8):1771–1800, August 2002.

[43] J J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, April 1982.

[44] David J. C. MacKay and David J. C. Mac Kay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, September 2003.

[45] Y. Abu-Mostafa and J. St. Jacques. Information capacity of the Hopfield model. *IEEE Transactions on Information Theory*, 31(4):461–464, July 1985.

[46] Daniel J Amit, Hanoch Gutfreund, and Haim Sompolinsky. Storing infinite numbers of patterns in a spin-glass model of neural networks. *Physical Review Letters*, 55(14):1530, 1985.

[47] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[48] Mete Demircigil, Judith Heusel, Matthias Löwe, Sven Upgang, and Franck Vermet. On a Model of Associative Memory with Huge Storage Capacity. *J Stat Phys*, 168(2):288–299, July 2017.

[49] Hubert Ramsauer, Bernhard Schäfl, Johannes Lehner, Philipp Seidl, Michael Widrich, Lukas Gruber, Markus Holzleitner, Thomas Adler, David Kreil, Michael K. Kopp, Günter Klambauer, Johannes Brandstetter, and Sepp Hochreiter. Hopfield Networks is All You Need. In *International Conference on Learning Representations*, February 2022.

[50] Dmitry Krotov. A new frontier for hopfield networks. *Nature Reviews Physics*, pages 1–2, 2023.

[51] Honglak Lee, Roger Grosse, Rajesh Ranganath, and Andrew Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 609–616, Montreal Quebec Canada, June 2009. ACM.

[52] Louis A. Jaeckel. An alternative design for a sparse distributed memory. July 1989.

[53] Pentti Kanerva. *Sparse Distributed Memory*. MIT press, 1988.

[54] Jason Weston, Sumit Chopra, and Antoine Bordes. Memory Networks, November 2015.

[55] Danil Tyulmankov, Ching Fang, Annapurna Vadaparty, and Guangyu Robert Yang. Biological learning in key-value memory networks. *Advances in Neural Information Processing Systems*, 34:22247–22258, 2021.

[56] Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. Key-Value Memory Networks for Directly Reading Documents, October 2016.

[57] Geoffrey E Hinton, Terrence J Sejnowski, and David H Ackley. *Boltzmann Machines: Constraint Satisfaction Networks That Learn*. Carnegie-Mellon University, Department of Computer Science Pittsburgh, PA, 1984.

[58] David H Ackley, Geoffrey E Hinton, and Terrence J Sejnowski. A learning algorithm for Boltzmann machines. *Cognitive science*, 9(1):147–169, 1985.

[59] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[60] Beren Millidge, Tommaso Salvatori, Yuhang Song, Thomas Lukasiewicz, and Rafal Bogacz. Universal Hopfield Networks: A General Framework for Single-Shot Associative Memory Models. *Proc Mach Learn Res*, 162:15561–15583, July 2022.

[61] Thomas F Burns and Tomoki Fukai. Simplicial hopfield networks. In *The Eleventh International Conference on Learning Representations*, 2023.

[62] Yuchen Liang, Chaitanya K. Ryali, Benjamin Hoover, Leopold Grinberg, Saket Navlakha, Mohammed J. Zaki, and Dmitry Krotov. Can a Fruit Fly Learn Word Embeddings?, March 2021.

[63] Michael Widrich, Bernhard Schäfl, Milena Pavlović, Hubert Ramsauer, Lukas Gruber, Markus Holzleitner, Johannes Brandstetter, Geir Kjetil Sandve, Victor Greiff, Sepp Hochreiter, and Günter Klambauer. Modern Hopfield Networks and Attention for Immune Repertoire Classification. In *Advances in Neural Information Processing Systems*, volume 33, pages 18832–18845. Curran Associates, Inc., 2020.

[64] Dmitry Krotov and John Hopfield. Dense associative memory is robust to adversarial inputs. *Neural computation*, 30(12):3151–3167, 2018.

[65] Benjamin Hoover, Yuchen Liang, Bao Pham, Rameswar Panda, Hendrik Strobelt, Duen Horng Chau, Mohammed J. Zaki, and Dmitry Krotov. Energy Transformer, February 2023.

[66] Tim Brooks, Aleksander Holynski, and Alexei A Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18392–18402, 2023.

[67] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22500–22510, 2023.

[68] Rinon Gal, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H. Bermano, Gal Chechik, and Daniel Cohen-Or. An Image is Worth One Word: Personalizing Text-to-Image Generation using Textual Inversion, August 2022.

[69] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models, 2023.

[70] Gaurav Parmar, Krishna Kumar Singh, Richard Zhang, Yijun Li, Jingwan Lu, and Jun-Yan Zhu. Zero-shot image-to-image translation. In *ACM SIGGRAPH 2023 Conference Proceedings*, pages 1–11, 2023.

[71] Hila Chefer, Yuval Alaluf, Yael Vinker, Lior Wolf, and Daniel Cohen-Or. Attend-and-excite: Attention-based semantic guidance for text-to-image diffusion models. *ACM Transactions on Graphics (TOG)*, 42(4):1–10, 2023.

[72] Omer Bar-Tal, Lior Yariv, Yaron Lipman, and Tali Dekel. MultiDiffusion: Fusing Diffusion Paths for Controlled Image Generation, February 2023.

[73] Manuel Brack, Felix Friedrich, Dominik Hintersdorf, Lukas Struppek, Patrick Schramowski, and Kristian Kersting. SEGA: Instructing Diffusion using Semantic Dimensions, January 2023.

[74] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising Diffusion Implicit Models, October 2022.

[75] Tim Salimans and Jonathan Ho. Progressive Distillation for Fast Sampling of Diffusion Models. In *International Conference on Learning Representations*, October 2021.

[76] Hongkai Zheng, Weili Nie, Arash Vahdat, Kamyar Azizzadenesheli, and Anima Anandkumar. Fast Sampling of Diffusion Models via Operator Learning. In *NeurIPS 2022 Workshop on Score-Based Methods*, November 2022.

[77] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization, January 2017.

[78] Dong C. Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical Programming*, 45(1-3):503–528, August 1989.

[79] Yiping Lu, Aoxiao Zhong, Quanzheng Li, and Bin Dong. Beyond Finite Layer Neural Networks: Bridging Deep Architectures and Numerical Differential Equations, March 2020.

[80] Subhabrata Dutta, Tanya Gautam, Soumen Chakrabarti, and Tanmoy Chakraborty. Redesigning the transformer architecture with insights from multi-particle dynamical systems. *Advances in Neural Information Processing Systems*, 34:5531–5544, 2021.

[81] Bei Li, Quan Du, Tao Zhou, Yi Jing, Shuhan Zhou, Xin Zeng, Tong Xiao, Jingbo Zhu, Xuebo Liu, and Min Zhang. ODE transformer: An ordinary differential equation-inspired model for sequence generation. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8335–8351, 2022.

[82] Yaofeng Desmond Zhong, Tongtao Zhang, Amit Chakraborty, and Biswadip Dey. A Neural ODE Interpretation of Transformer Layers. In *The Symbiosis of Deep Learning and Differential Equations II*, November 2022.

[83] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling Laws for Neural Language Models, January 2020.

[84] Rishi Bommasani, Drew A. Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S. Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, Erik Brynjolfsson, Shyamal Buch, Dallas Card, Rodrigo Castellon, Niladri Chatterji, Annie Chen, Kathleen Creel, Jared Quincy Davis, Dora Demszky, Chris Donahue, Moussa Doumbouya, Esin Durmus, Stefano Ermon, John Etchemendy, Kawin Ethayarajh, Li Fei-Fei, Chelsea Finn, Trevor Gale, Lauren Gillespie, Karan Goel, Noah Goodman, Shelby Grossman, Neel Guha, Tatsunori Hashimoto, Peter Henderson, John Hewitt, Daniel E. Ho, Jenny Hong, Kyle Hsu, Jing Huang, Thomas Icard, Saahil Jain, Dan Jurafsky, Pratyusha Kalluri, Siddharth Karamcheti, Geoff Keeling, Fereshte Khani, Omar Khattab, Pang Wei Koh, Mark Krass, Ranjay Krishna, Rohith Kuditipudi, Ananya Kumar, Faisal Ladhak, Mina Lee, Tony Lee, Jure Leskovec, Isabelle Levent, Xiang Lisa Li, Xuechen Li, Tengyu Ma, Ali Malik, Christopher D. Manning, Suvir Mirchandani, Eric Mitchell, Zanele Munyikwa, Suraj Nair, Avanika Narayan, Deepak Narayanan, Ben Newman, Allen Nie, Juan Carlos Niebles, Hamed Nilforoshan, Julian Nyarko, Giray Ogut, Laurel Orr, Isabel Papadimitriou, Joon Sung Park, Chris Piech, Eva Portelance, Christopher Potts, Aditi Raghunathan, Rob Reich, Hongyu Ren, Frieda Rong, Yusuf Roohani, Camilo Ruiz, Jack Ryan, Christopher Ré, Dorsa Sadigh, Shiori Sagawa, Keshav Santhanam, Andy Shih, Krishnan Srinivasan, Alex Tamkin, Rohan Taori, Armin W. Thomas, Florian Tramèr, Rose E. Wang, William Wang, Bohan Wu, Jiajun Wu, Yuhuai Wu, Sang Michael Xie, Michihiro Yasunaga, Jiaxuan You, Matei Zaharia, Michael Zhang, Tianyi Zhang, Xikun Zhang, Yuhui Zhang, Lucia Zheng, Kaitlyn Zhou, and Percy Liang. On the Opportunities and Risks of Foundation Models, July 2022.

## A Mathematical Notations

In this survey we deviate from notations typically used for DMs and AMs. To minimize visual clutter, we prefer tensor notation over Einstein notation, representing scalars in non-bolded font (e.g., energies $E$ or time $t$), and tensors (e.g., vectors and matrices) in bold (e.g., states $\mathbf{x}$ or weights $\mathbf{W}$). These distinctions also apply to scalar-valued and tensor-valued functions (e.g., energy $E(\cdot)$ vs. activations $\mathbf{g}(\cdot)$). A collection of learnable parameters is expressed through the generic variable $\boldsymbol{\theta}$. Gradients are expressed using "nabla" notation of a scalar valued function, where $\nabla_{\mathbf{x}}(\cdot)$ will be a tensor of the same shape as vector $\mathbf{x}$. $\mathbf{g}^{\intercal}$ represents the transpose, whereas $\mathbf{g}^{T}$ represents the tensor $\mathbf{g}$ occurring at time $T$.

All mathematical notation used in this survey is documented in Table 2. When applicable, we describe the notation both by its data-science (e.g., "data point") and physics terminology (e.g., "particle").

Table 2: A summary of the mathematical notations and conventions used in this survey.

| Notation | Description |
|---|---|
| $t$ | **time** |
| $s$ | **reverse time** (as $s$ increases, time $t$ decreases. Used in diffusion literature) |
| $\boldsymbol{\theta}$ | **parameters** for neural network |
| $p_{\boldsymbol{\theta}}$ | **probability distribution** of some data, parameterized by $\boldsymbol{\theta}$ |
| $E_{\boldsymbol{\theta}}$ | **energy** of some data, parameterized by $\boldsymbol{\theta}$, proportional to the negative of log-probability |
| $Z_{\boldsymbol{\theta}}$ | **normalization constant** converting energy to probabilities a.k.a. the *partition function* |
| $\mathbf{F}_{\boldsymbol{\theta}}$ | **score function** or the *force* (equal to $-\nabla_{\mathbf{x}} E_{\boldsymbol{\theta}}$), parameterized by $\boldsymbol{\theta}$ |
| $\alpha$ | **discrete step size** when descending the energy |
| $\eta$ | **noise** added during discrete forward-process of diffusion |
| $d\mathrm{w}$ | **infinitesimal noise** added during continuous forward process of diffusion |
| $\sigma$ | **noise scale** of $d\mathrm{w}$ |
| $T$ | **max depth** of a Diffusion Model / time that fixed point of AM is reached |
| $\tau$ | **rate constant** of continuous time dynamics |
| $N_{\nu}$ | **number of neurons** in layer $\nu$ of AMs |
| $\mathbf{x}$ | **state** of a data point, evolves in time (*position* of particle) |
| $\mathbf{g}$ | **activations** of state $\mathbf{x}$ (*axonal output* of a neuron) |
| $\mathcal{L}$ | **Lagrangian** of a layer (dynamic variable) that defines that variable's energy and activations |
| $\mathbf{x}^{\star}$ | **fixed point** of the memory retrieval dynamics |
| $\mathbf{W}$ | **weights** of the Associative Memory network |
| $\mathbf{f}_{\theta}$ | **any generic function** parameterized by $\boldsymbol{\theta}$ |

## B Characterizing the Uncanny Resemblance

Here we clarify the summary of similarities and differences presented in Table 1.

### B.1 Characterizing the Similarities

There are several reasons to be skeptical of significant overlap between generative models like DMs and AMs. As presented, DMs only compute gradients of energies and not energies themselves; thus they have no Lyapunov function guaranteeing stability and are not defined to operate for all $t > T$. However, it is clear that both DMs and AMs have very similar goals as shown in Eq. 7 and Eq. 10. We summarize the similarities between these two data modeling approaches as follows:

- **Both model the energy.** DMs learn a parameterized score function $\mathbf{F}_{\boldsymbol{\theta}}$ to approximate the gradient of some true energy function $E$ at every point $\mathbf{x}$ in the energy landscape such that $\mathbf{F}_{\boldsymbol{\theta}}(\mathbf{x}) \approx -\nabla_{\mathbf{x}} E(\mathbf{x})$. In AMs, this energy function is explicit and is defined by using architectures that directly model the energy $E_{\boldsymbol{\theta}}$ such that $E_{\boldsymbol{\theta}}(\mathbf{x}) \approx E(\mathbf{x})$.
- **Both generate samples by descending the predicted gradient of the energy.** A DM will directly output the estimated score $\mathbf{F}_{\boldsymbol{\theta}} \approx -\nabla_{\mathbf{x}} E(\mathbf{x})$, whereas an AM will directly output a smooth energy $E_{\boldsymbol{\theta}}(\mathbf{x})$ on which the gradient $-\nabla_{\mathbf{x}} E_{\boldsymbol{\theta}}(\mathbf{x})$ can be directly calculated and descended. The discrete (continuous) update rules of both are identical to a step size $\alpha$ (time variable $\tau$).
- **Both converge to a solution that lies in the neighborhood of a local energy minimum.** In DMs, this behavior is a consequence of the manner in which it is trained: the final output $\mathbf{x}^T$ exists in a region such that a small perturbation in any direction would increase its energy (remember that DMs are trained to remove infinitesimal noise that would increase the energy of the data point). In AMs, this statement is a requirement of the Lyapunov function; if the dynamics progress for a sufficiently long time, we are guaranteed to reach a true fixed point $\mathbf{x}^\star$ that lies at a local energy minimum.

## B.2 Reconciling the Differences

DMs and AMs are certainly not equivalent methods. However, we have discovered evidence that the theoretical differences between DMs and AMs are not so significant in practice. We present rebuttals to potential objections (which we call "misconceptions") that Diffusion Models do not simulate Associative Memories below.

**Misconception 1: Diffusion models are not fixed point attractors.** Though the dynamics of DMs have no theoretical guarantees of fixed point attractors, we notice that the design of DMs seems to intelligently engineer the behavior of fixed-point attractors like AMs without constraining the architecture to represent a Lyapunov function. We identify two fundamental tricks used by DMs that help approximate stable dynamics:

**Trick 1** DMs explicitly halt their reconstruction process at time $t = T$ (i.e., requiring $\mathbf{x}^\star = \mathbf{x}^T$, a form of boundary condition on the ODE defined by Eq. 6, see [27] for more details) and are thus only defined for time $t \in [0, T]$. $\mathbf{x}^T$ then represents a *contrived fixed point* because no further operations change it. We can say that $\mathbf{x}^{t \neq T}$ corresponds to a data point with some corruption and $\mathbf{x}^T$ corresponds to a *memory* in the language of AMs.

**Trick 2** We know that $\mathbf{x}^T$ approximates a local energy minimum because of the *noise annealing* trick introduced by [2] and used by all subsequent DMs. In the corruption process, points in the data distribution are perturbed with gradually increasing amounts of noise, implying that smaller noise is added at earlier points in time. This leads to a robust approximation of the true energy gradient localized around each training point, where the original data point lies at the minimum.

We additionally see evidence of DMs storing "memories" that are actual training points. [30] showed that one can retrieve training data almost exactly from publicly available DMs by descending an energy conditioned on prompts from the training dataset. It seems that this behavior is particularly evident for images considered outliers and for images that appear many times. Viewing DMs as AMs, this behavior is not surprising, as the whole function of AMs is to retrieve data (or close approximations to the data) that it has been seen before.

Tricks 1 & 2 also mean that a DM is inescapably bound to a knowledge of the current time $t$. By conditioning both the model $\mathbf{F}_{\boldsymbol{\theta}}$ and the step size $\alpha$ on the time $t$, we give information to the model and the optimizer about how much noise it should expect to remove from a given signal in a single step. Given a signal with an unknown quantity of noise, a user must either make a "best guess" for the time $t$ (expected noise level), or restart the dynamics at time $t = 0$. The model is always trained to expect large amounts of noise at time $t = 0$, which will cause the model to make large jumps around the energy landscape and likely land it in a different energy minimum. Though this is desirable for diverse generation, it makes DMs less reliable at memory retrieval. Currently, no works have tested AMs that are dependent on time, though conditioning an AM and its optimizer on the time $t$ (essentially, treating time as another input modality) maintains all Lyapunov constraints and is perfectly valid to try.

**Misconception 2: Diffusion models can only undo Gaussian noise.** In order for a DM to behave like an AM, they must be able to undo any kind of corruption (e.g., inpainting, blur, pixelation, etc.), not just the white- or Gaussian-noise associated with Brownian motion as originally formulated in [1, 2]. [31, 32] showed that the performance of DMs can actually improve when considering other types of noisy corruption in the forward process. However, it also seems that DMs can learn to reverse any kind of corrupting process. [33] empirically show that DMs can be trained to invert arbitrary image corruptions that generate samples just as well as those trained to invert Gaussian noise. This challenges the necessity of the mathematical formulation of infinitesimal gaussian noise $d\mathbf{w}$ in the forward process of Eq. 5. Because DMs can be trained to undo any kind of corruption, they can exhibit behavior identical to that of Associative Memory which focuses on the "fixed points" of the energy landscape rather than the kind of denoising/de-corrupting steps required to get there.

**Misconception 3: Unconstrained Diffusion Models work with any architecture.** DMs are a form of "unconstrained" architecture and can use any neural network to approximate the score function; in other words, the architecture is not required to be the true gradient of an energy. This is often considered advantageous over less flexible architectures such as those needed in AMs. The only requirement is that the neural network chosen to approximate the score must be *isomorphic* such that the function's output is the same shape as its input (i.e., $\mathbf{F}_{\boldsymbol{\theta}} : \mathbb{R}^d \mapsto \mathbb{R}^d$). However, not all isomorphic architectures are created equal and only select architectures are used in practice. Both standard feedforward networks [25] and vanilla Transformers have struggled to generate quality samples using diffusion [34, 35] (though recent work [34] has shown how to engineer vision Transformers [36] to achieve a similar reconstruction quality as U-Nets on images). Most applications of DMs use some modification of the U-Net architecture [37] originally used by [4] or shallow MLPs as used in the original DM paper [1]. Thus, the perceived advantage of flexibility comes at a cost – how can we know what architectures will likely work for a DM and what will not?

**Misconception 4: Diffusion Models with explicit energy don't work.** Though DMs characterize an energy landscape by modeling its gradient everywhere, they do not inherently have a concept of the energy value itself. However, [15] showed that one can actually compute an exact energy for DMs using the instantaneous change of variables formula from [38], with the caveat that this equation is expensive to compute; estimations of the energy computation are preferred over direct energy calculations in practice [14].

Another approach for enforcing an energy on DMs is to choose an architecture that parameterizes an actual energy function, whose gradient is the score function. [25] researched exactly this, exploring whether a generic learnable function $\mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}; t)$ that is constrained to be the true gradient of a parameterized energy function as in Eq. 8 is able to attain sample quality results similar to those of unconstrained networks.

$$E_{\boldsymbol{\theta}}(\mathbf{x}; t) = \frac{1}{2\sigma(t)} ||\mathbf{x} - \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}; t)||^2 \tag{8}$$

The score $\mathbf{F}_{\boldsymbol{\theta}}$ of this energy can be written by computing the analytical gradient

$$\mathbf{F}_{\boldsymbol{\theta}}(\mathbf{x}; t) = \frac{1}{\sigma(t)} (\mathbf{x} - \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}; t)) \nabla_{\mathbf{x}} \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}; t) - \frac{1}{\sigma(t)} (\mathbf{x} - \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}; t)) \,. \tag{9}$$

Ref. [25] notes that the second term of the equation is the standard DM, while the first term involving $\nabla_{\mathbf{x}} \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}; t)$ is new and helps guarantee that $\mathbf{F}_{\boldsymbol{\theta}}(\mathbf{x}; t)$ is a conservative vector field. They also showed that constraining the score function to be the gradient of the energy in Eq. 8 does not hurt the generation performance on the CIFAR dataset [39] and provides hope that AMs with constrained energy can match the performance of unconstrained DMs.

## C More on Associative Memories

An Associative Memory (AM) is a dynamical system concerned with the storage and retrieval of signals (data). In AMs all signals exist on an *energy landscape*; in general, the more corrupted a signal the higher its energy value. Uncorrupted signals or *memories* live at local minima of the system's energy or *Lyapunov* function. The process of *memory retrieval* is the dynamic process of descending the energy to a fixed point or *memory* as shown in Figure 1, a phenomenon that is guaranteed to occur by constraints placed on the architecture (see § C.1).

The following subsections describing AMs can get quite technical and rely on modeling techniques used regularly in physics (e.g., Lagrangian functions, Lyapunov functions, Legendre transforms [12, 40]) but that are foreign to experts in modern AI. We thus deem it important to summarize the core points of AMs before diving into them and their history.

AMs are a class of neural architectures that defines an energy function $E_{\boldsymbol{\theta}}(\mathbf{x}) \in \mathbb{R}$, where $\boldsymbol{\theta}$ is a set of learnable parameters and $\mathbf{x}$ represents a data point. In other words, an AM is designed to compute a scalar energy on every possible input signal $\mathbf{x}$. Given some initial input $\mathbf{x}^0$ at time $t = 0$ (this could be any kind of signal, including pure noise), we want to minimize the energy (by descending the gradient) to a fixed point that represents a memory learned from the original data distribution.

$$\tau \frac{d\mathbf{x}}{dt} = -\nabla_{\mathbf{x}} E_{\boldsymbol{\theta}}(\mathbf{x}), \qquad t > 0 \tag{10}$$

Here, $\tau$ is a time constant that governs how quickly the dynamics evolve. This equation can of course be discretized as

$$\mathbf{x}^{t+1} = \mathbf{x}^t - \frac{dt}{\tau} \nabla_{\mathbf{x}} E_{\boldsymbol{\theta}}(\mathbf{x}^t) \tag{11}$$

and treated as a neural network that is recurrent through time. The energy of AMs is constructed in such a way that the dynamics will eventually converge to a fixed point $\mathbf{x}^\star$ at some $t = T$. That is,

$$\frac{d\mathbf{x}^\star}{dt} = -\nabla_{\mathbf{x}} E_{\boldsymbol{\theta}}(\mathbf{x}^\star) = 0$$

In practice there are some valid AM architectures where the variable $\mathbf{x}^t$ itself may not converge, which depends on our choice of *activation function* $\mathbf{g}(\mathbf{x}^t)$. E.g., the $\mathbf{x}^t$ operating under activations like the $\mathrm{sigmoid}$ and $\mathrm{relu}$ are guaranteed to converge, whereas the $\mathbf{x}^t$ under activation functions with some normalizing constant like the $\mathrm{softmax}$ or the $\mathrm{layernorm}$ may never converge. In these cases the AM dynamics are well behaved on the activations $\mathbf{g}^t$, which are still guaranteed to converge.

## C.1 The *TinyBrain* Sandbox for Building Associative Memories

AMs are constrained to use only neural architectures that are themselves a Lyapunov function: that is, these architectures must compute a scalar energy and be fixed point attractors. In this section we discuss the architectural constraints needed to ensure a Lyapunov function on AMs, following the abstractions of [41].

AMs were originally inspired by neurons and synapses within the brain [43]. As such, it is useful to consider a simple model we call the *TinyBrain* model[1] (depicted in Figure 2). *TinyBrain* captures the simplest Associative Memory configuration consisting of two dynamic variables called *neuron layers* connected to each other by a single synaptic weight matrix $\mathbf{W}$ that defines the relationship between variables. This setup is distinct from general AMs which can be made hierarchical (see § C.4). Each neuron layer (dynamic variable) can be described by an *internal state* $\mathbf{x} \in \mathbb{R}^N$ (which one can think of as the *membrane voltage* for each of the $N$ neurons in that layer), and their *axonal state* $\mathbf{g} \in \mathbb{R}^N$ (which is analogous to the *firing rate* of each neuron). We call the axonal state the *activations* and they are uniquely constrained to be the gradient of a scalar, convex Lagrangian function $\mathcal{L} : \mathbb{R}^N \mapsto \mathbb{R}$ we choose for that layer; that is, $\mathbf{g} = \nabla_{\mathbf{x}} \mathcal{L}$.

The *Legendre Transform* of the Lagrangian defines the energy $E_\nu^{\text{layer}}(\mathbf{g}_\nu, \mathbf{x}_\nu) \in \mathbb{R}$ of a neuron layer $\nu$, shown in Eq. 12 [12, 40, 41]. If all neuron-layers of a system have a Lagrangian, a synaptic energy can be easily found such that the entire system has a defined energy.

[1]Our use of the term "brain" does not claim resemblance to real brains or brain-like behavior; instead, the term encapsulates a useful thought experiment around the usage of the terms "neurons" and "synapses".

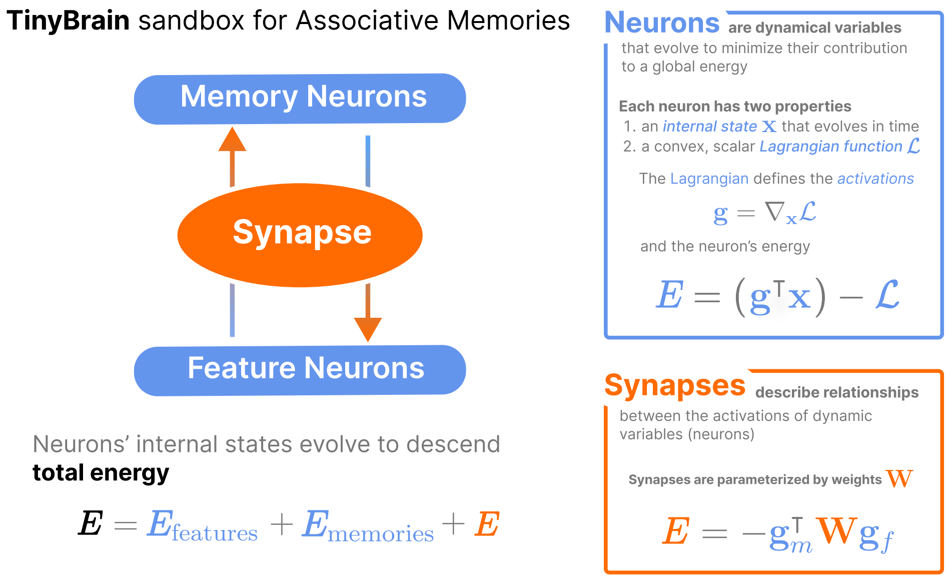
**TinyBrain** sandbox for Associative Memories
Memory Neurons
Synapse
Feature Neurons
Neurons' internal states evolve to descend **total energy**
$E = E_{\text{features}} + E_{\text{memories}} + E$
**Neurons** are dynamical variables that evolve to minimize their contribution to a global energy
Each neuron has two properties
1. an *internal state* $\mathbf{x}$ that evolves in time
2. a convex, scalar *Lagrangian function* $\mathcal{L}$
The Lagrangian defines the *activations*
$\mathbf{g} = \nabla_{\mathbf{x}} \mathcal{L}$
and the neuron's energy
$E = (\mathbf{g}^\intercal \mathbf{x}) - \mathcal{L}$
**Synapses** describe relationships between the activations of dynamic variables (neurons)
Synapses are parameterized by weights $\mathbf{W}$
$E = -\mathbf{g}_m^\intercal \mathbf{W} \mathbf{g}_f$

Figure 2: The *TinyBrain* sandbox for understanding Associative Memories is a fully connected bipartite graph (structurally similar to the Restricted Boltzmann Machine [42]). Both feature neurons and memory neurons have states $\mathbf{x}_f$ and $\mathbf{x}_m$ respectively that evolve in time; these states have corresponding activations $\mathbf{g}_f$ and $\mathbf{g}_m$. The energy of the synapse is minimized when the memory activations $\mathbf{g}_m$ perfectly align with the feature activations $\mathbf{g}_f$ according to the learned parameters $\mathbf{W}$. The energy for the entire system is defined by this the energies of each individual component, and the internal states of each neuron layer evolve by descending the global energy's gradient.

$$E_\nu^{\text{layer}} = \mathbf{g}_\nu^\intercal \mathbf{x}_\nu - \mathcal{L}_\nu(\mathbf{x}_\nu) \tag{12}$$

For historical reasons, we call one neuron layer in our *TinyBrain* the "feature neurons" (fully described by the internal state $\mathbf{x}_f$ and Lagrangian $\mathcal{L}_f$ of the features) and the other the "memory neurons" (fully described by the internal state $\mathbf{x}_m$ and Lagrangian $\mathcal{L}_m$ of the memories). These layers are connected to each other via a synaptic weight matrix $\mathbf{W} \in \mathbb{R}^{N_m \times N_f}$, creating a synaptic energy $E^{\text{synapse}}(\mathbf{g}_f, \mathbf{g}_m; \mathbf{W}) \in \mathbb{R}$ defined as

$$E^{\text{synapse}} = -\mathbf{g}_m^\intercal \mathbf{W} \mathbf{g}_f \,. \tag{13}$$

We now have defined all the *component energies* necessary to write the total energy $E^{\text{system}}$ for *TinyBrain* as in Eq. 14. Note that Eq. 14 is a simple summation of the energies of each component of the *TinyBrain*: two layer energies and one synaptic energy. This perspective of modular energies for understanding AMs was originally proposed by [41].

$$E^{\text{system}} = E_f^{\text{layer}} + E_m^{\text{layer}} + E^{\text{synapse}} \tag{14}$$

The hidden states of our neurons $\mathbf{x}_f$ and $\mathbf{x}_m$ evolve in time according to the general update rule:

$$\begin{cases} \tau_f \dfrac{d\mathbf{x}_f}{dt} & = -\nabla_{\mathbf{g}_f} E^{\text{system}} \\ \tau_m \dfrac{d\mathbf{x}_m}{dt} & = -\nabla_{\mathbf{g}_m} E^{\text{system}} \,. \end{cases} \tag{15}$$

Eq. 15 reduces to the manually derived update equations for the feature and memory layers presented in Eq. 16 [12].

$$\begin{cases} \tau_f \dfrac{d\mathbf{x}_f}{dt} &= \mathbf{W}^\intercal \mathbf{g}_m - \mathbf{x}_f \\ \tau_m \dfrac{d\mathbf{x}_m}{dt} &= \mathbf{W}\mathbf{g}_f - \mathbf{x}_m \end{cases} \tag{16}$$

where $\mathbf{W}^\intercal \mathbf{g}_m$ and $\mathbf{W}\mathbf{g}_f$ represent the *input currents* to feature neurons and memory neurons respectively, and $-\mathbf{x}_f$ and $-\mathbf{x}_m$ represent exponential decay (this implies that the internal state $\mathbf{x}$ will exponentially decay to $\mathbf{0}$ in the absence of input current). Note that the synaptic matrix $\mathbf{W}$ plays a symmetric role: the same weights that modulate the current from feature neurons $\to$ memory neurons are used to modulate the current from memory neurons $\to$ feature neurons. This "symmetric weight constraint" present in AMs does not exist in real neurons and synapses [12, 44].

Krotov and Hopfield identified [12] that this toy model is mathematically equivalent to the original Hopfield Network [43, 28] under certain choices of the activation function $\mathbf{g}$; hence, it is through the lens of the abstraction presented in this section that we explore the long history of AMs, beginning with the famous Hopfield Network.

### C.2 Hopfield Networks are the First Energy-Based Associative Memory

John Hopfield formalized the dynamic retrieval process of Associative Memory in the 80s [43, 28] in an energy-based model that became famously known as the *Hopfield Network* (HN). Unlike previous models for associative memory (see § C.5), HNs performed the task of memory retrieval through gradient descent of an energy function. A HN resembles in form a single-layer McCulloch & Pitts perceptron [29], with input feature neurons $\mathbf{x}_f$ and a single weight matrix; however, unlike perceptron-based models, the HN operated continuously in time, repeatedly updating the inputs $\mathbf{x}_f^t$ over time to minimize some energy.

Hopfield described the AM energy dynamics for both binary [43] and graded (continuous) [28] neurons. Like the *TinyBrain* model, a continuous HN consists of $N_f$ feature neurons connected via a synaptic weight matrix $\mathbf{W} \in \mathbb{R}^{N_m \times N_f}$ to $N_m$ memory neurons. Feature neurons $\mathbf{x}_f$ have corresponding activations $\mathbf{g}_f = \mathrm{sigmoid}(\mathbf{x}_f)$, whereas memory neurons $\mathbf{x}_m$ have a linear activation function $\mathbf{g}_m = \mathbf{x}_m$. We call this configuration of *TinyBrain* the *Classical Hopfield Network* (CHN).

Hopfield never considered the time evolution of the memory neurons in his model, instead assuming that $\mathbf{x}_m^t = W\mathbf{g}_f^t$ at any point in time (i.e., that the state of the memory neurons was instantaneously defined by the activations of the feature neurons). This simplification of the two-layer *TinyBrain* did not change the fixed points of the AM dynamics and allowed him to consider the time-evolution for only the feature neuron layer. The simplified energy function and update rules for the CHN are shown in Eq. 17 and Eq. 18 respectively, where $\mathcal{L}_f(\mathbf{x}_f) \triangleq \sum_{N_f} \log\left(1 + \exp(\mathbf{x}_f)\right)$ is the Lagrangian of the $\mathrm{sigmoid}$ activation function and $(\cdot)^2$ operates elementwise.

$$E^{\text{system}}(\mathbf{g}_f, \mathbf{x}_f) = -\frac{1}{2} \sum_{N_m} \left(\mathbf{W}\mathbf{g}_f\right)^2 + \mathbf{g}_f^\intercal \mathbf{x}_f - \mathcal{L}_f(\mathbf{x}_f) \tag{17}$$

$$\tau \frac{d\mathbf{x}_f}{dt} = \mathbf{W}^\intercal(\mathbf{W}\mathbf{g}_f) - \mathbf{x}_f \,. \tag{18}$$

A full derivation is included in [12].

### C.3 Solving the Small Memory Capacity of Hopfield Networks

The CHN unfortunately suffered from a tiny memory capacity that scaled linearly according the number of input features $N_f$; specifically, the maximum storage capacity of the classical Hopfield Network was discovered to be $\sim 0.14 N_f$ by [45, 43, 46]. Consider the problem of building an associative memory on the 60k images in MNIST [47], where each image can be represented as a binary vector with $N_f = 784$ features. If the patterns were random, one could only reliably store a maximum of $0.14(784) \approx 110$ images using the classical Hopfield paradigm, **no matter how many memories $N_m$ you add to the synaptic matrix $\mathbf{W}$**. Given that MNIST images have strong correlations between their pixel intensities, this bound is even lower.

A breakthrough in the capacity of Hopfield Networks was proposed by Krotov & Hopfield over 30 years later [29]. Their new network, called the Dense Associative Memory (DAM), enabled the energy dynamics of the CHN to store a super-linear number of memories. The core idea was to use a rapidly growing non-linear activation functions $\mathbf{g}_m(\cdot)$ on the memory neurons. For instance, choosing $\mathbf{g}_m(\cdot)$ to be higher orders of (optionally rectified) polynomials allowed much greater memory storage capacity than the CHN. Extending this idea to exponential functions $\mathbf{g}_m = \exp(\mathbf{x}_m)$ can even lead to exponential storage capacity [48]. The intuition is that the "spikier" the activation function $\mathbf{g}_m$ (i.e., the faster it grows in the region around $\mathbf{x}_m$), the more memories the network can store and retrieve.

Recall that a CHN could reliably store a maximum of 110 MNIST images. Using the exponential DAM, one can increase the number of stored memories up to $N_m \sim \exp(784/2)$, assuming no correlations, *and still reliably retrieve each individual memory*. This marked difference has led to DAMs being branded as the "Modern Hopfield Network" (MHN) [49] and opened the door for a new frontier in AMs [50].

## C.4 Adding Hierarchy to Shallow Associative Memories

The CHN and DAM have another fundamental limitation: like the *TinyBrain*, both only have a single weight matrix $\mathbf{W}$ and thus cannot learn hierarchical abstractions that simplify the task of memorizing complex signals. This is not a limitation of Deep Learning networks today, which can be seen as a stack of distinct learnable functions ("layers") each processing the output of a previous layer. These architectures are inherently *hierarchical*; that is, deeper layers operate on higher levels of abstraction output by previous layers. A classic example of this occurs in deep Convolutional Neural Networks (CNNs), where neurons in earlier layers (i.e., layers closer to the image) detect simple shapes, and those in deeper layers respond to more complex objects (see Figure 3).

However, Eq. 12 makes it easy to constrain the energy of an AM. Why can't these systems be extended beyond the *TinyBrain* in § C.1 to include more interacting layers? This is the realization of [40, 41] who generalize the theoretical abstraction of AMs in § C.1 to connect arbitrary numbers of neuron layers via arbitrary synaptic relationships that can resemble the convolutional, pooling, or even attention operations in modern architectures. This version of AMs is known as a Hierarchical Associative Memory (HAM).

The HAM has given AMs a theoretical maturity and flexibility to rival the representational capacity of any existing neural architecture while guaranteeing stable attractor points. However, the HAM is still a very young architecture, having been proposed in 2021, and has yet to establish itself as a viable alternative to traditional Deep Learning architectures in practice.

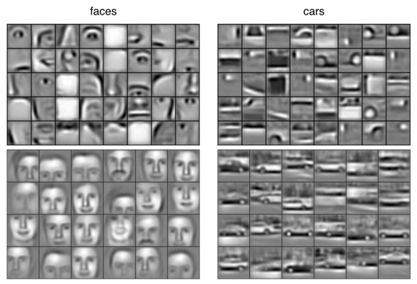

Figure 3: An explanation of hierarchical representations as learned in images, adapted from [51]. The top row shows the filters learned by the second layer in a convolutional deep belief network, whereas the bottom row the filters of the third layer. The deeper filters are able to recognize more complex objects and are thus considered more "abstract".

## C.5 Other models for Associative Memory

The term "associative memory" has become a catch-all term for many different types of Content Addressable Memories (CAMs), including models like Sparse Distributed Memories [52, 53],Memory Networks [54], Key-Value Memories [55, 56], Hopfield Networks [43, 28], and Boltzmann Machines [57, 58, 42]. Even the popular attention mechanism in Transformers [59] is itself a differentiable form of associative memory [49] where tokens act as queries to retrieve values stored by the other tokens. In this paper we have considered a particular class of AMs that refers to systems with a defined Lyapunov function – that is, a CAM must have both a tractable energy function and guaranteed stable states to be considered in this survey. The paradigmatic examples of this class of AMs is of course the Hopfield Network (HN) [43, 28] and its modern counterparts [29, 40] which have been discussed in earlier sections.

## D Open Challenges

### D.1 Directions for Associative Memories

Associative Memories have not gained nearly the traction of Diffusion Models in AI applications. Many researchers in the field focus on the *TinyBrain* architecture, trying to improve its theoretical memory capacity [60, 61] or apply related models to modern problems [62]. Other researchers are integrating the memory-retrieval capabilities of AMs into existing feed-forward architectures [49, 63, 64]; in doing so they discard the idea of global optimization on the energy. In part, these other research directions exist because no pure AM has shown impressive performance on large data until [65] introduced the Energy Transformer, though even this AM does not yet show significant performance gain over traditional methods.

The empirical success of Diffusion Models across many domains should provide hope that modern AM architectures [40, 41] can achieve performance parity on similar tasks. Constrained Diffusion Models show no worse performance than unconstrained models [25], and the introduction of HAMs allow AMs to be built that resemble U-Nets [41]. Even the training pipeline of a Diffusion Model can be mapped directly to an AM paradigm if we condition the energy function and optimization procedure on the time: from a starting image $\mathbf{x}^\star$ in the training set, repeatedly add varying amounts of random noise and train the score function $-\nabla_{\mathbf{x}} E_\theta(\mathbf{x}^t; t)$ to predict the added noise. Once trained, Diffusion Models have even shown strong steerability – the ability for a user to modify a trained Diffusion Model to complete tasks it was not trained to perform [1, 66, 67, 68, 69, 70, 71, 72, 73]. We can expect similar levels of controllability from a fully trained AM.

### D.2 Directions for Diffusion Models

Diffusion Models should find the theoretical framework of the Lyapunov function from AMs compelling, as it defines systems around fixed-point attractors that seem to already exist in Diffusion Models. Perhaps the score function $\mathbf{F}_\theta$ learned by Diffusion Models has already learned to behave similarly? If the score function is shown to be a conservative vector field as in [25], perhaps the constraint of finite time in Diffusion Models is unnecessary and Diffusion Models can behave well in continuous time $t > T$. Viewing Diffusion Models as fundamentally fixed-point attractors like AMs, it also becomes possible to theoretically characterize its memory capacity. Finally, recent research has focused on optimizing the sampling speed of Diffusion Models by improving the scheduling step in the update rule [27, 74, 75, 76]. By viewing this sampling procedure as ordinary gradient descent (as is the case in AMs), smart gradient optimization techniques already used in Deep Learning like ADAM [77] and L-BFGS [78] become available.

### D.3 Beyond Diffusion: Transformers Resemble Associative Memories

Diffusion Models are not the only method in modern Deep Learning to show similarities to AMs. In fact, recent literature shows increasing connections between the operations of AMs and deep architectures, e.g. feed-forward networks [64], convolutional neural networks [40], Transformer architecture [49], and optimization processes of Ordinary Differential Equations (ODEs) [79, 80, 81, 82]. In 2020 [49] discovered that the attention mechanism of Transformers strongly resembled a single-step update of a Dense Associative Memory [29] under the $\mathrm{softmax}(\cdot)$ activation function, which exhibits similar properties to the power and $\exp(\cdot)$ functions studied in [29, 48]. However, it is incorrect to call their contrived energy function an AM as it is integrated as a conventional feed-forward layer in the standard Transformer block and applied only using a single gradient descent step. Of particular interest to the scope of this paper is the following question: if Transformers have strong resemblances to ODEs and the attention operation is similar to a single step update of a DAM, what are the differences between an AM with desirable attractor dynamics and a Transformer?

A recent work by [65] explores this question, deriving a new "Energy Transformer" block that strongly resembles the conventional Transformer block, but whose fundamental operation outputs an energy. This energy satisfies all the desired properties of an AM, which allows us to interpret the forward pass through a stack of Transformer blocks as gradient descent down the block's energy.

## D.4 Scaling Laws from the Perspective of Associative Memory

The performance of Transformers on language is famously characterized by the "scaling laws", which claim that a model's performance will improve as a power-law with model size, dataset size, and the amount of compute used for training [83]. We expect similar behaviors to hold for Diffusion Models, though a similar study has not yet been conducted. However, the "scaling laws" are empirical only, and there is little theory to justify why a model's performance would continue to grow with the size. AMs offer one possible theory by characterizing large-model performance as one of *memory capacity* (see § C.3). In the world of AMs, this scaling behavior makes intuitive sense: more parameters means more possible memories that can be stored; more data means more meaningful local minima in the energy; and more compute means the model can descend further down the energy, making it easier to distinguish between nearby fixed points (alternatively, more compute can imply that longer training allows the model to distribute the fixed points in the energy more efficiently, allowing for greater memory capacity). These hypotheses for understanding the scaling law come from intuitively understanding large models as AMs, but this is still an open research question.

Both Transformers and Diffusion Models are ubiquitous choices for foundation models [84] in Deep Learning today, and both strongly resemble Associative Memories. We believe that the trajectory of AI research would benefit by interpreting the problem of unsupervised learning on large, unstructured data from the perspective of Associative Memory.