MEMORIZATION TO GENERALIZATION: EMERGENCE OF DIFFUSION MODELS FROM ASSOCIATIVE MEMORY

Bao Pham[*1]✉ **Gabriel Raya**[*2]✉ **Matteo Negri**[3] **Mohammed J. Zaki**[1]
Luca Ambrogioni[4 †] **Dmitry Krotov**[5 †]

[1]Rensselaer Polytechnic Institute, [2]Tilburg University, [3]University of Rome Sapienza
[4]Radboud University, [5]IBM Research

ABSTRACT

Hopfield networks are associative memory (AM) systems, designed for storing and retrieving patterns as local minima of an energy landscape. In the classical Hopfield model, an interesting phenomenon occurs when the amount of training data reaches its critical memory load — *spurious states*, or unintended stable points, emerge at the end of the retrieval dynamics. These particular states often appear as mixtures of the stored patterns, leading to incorrect recall. In this work, we examine diffusion models, commonly used in generative modeling, from the perspective of AMs. The training phase of diffusion model is conceptualized as memory encoding (training data is stored in the memory). The generation phase is viewed as an attempt of memory retrieval. In the small data regime the diffusion model exhibits a strong memorization phase, where the network creates distinct basins of attraction around each sample in the training set, akin to the Hopfield model below the critical memory load. In the large data regime, a different phase appears where an increase in the size of the training set fosters the creation of new attractor states that correspond to manifolds of the generated samples. Spurious states appear at the boundary of this transition and correspond to emergent attractor states, which are absent in the training set, but at the same time still have distinct basins of attraction around them. Our findings provide a novel perspective on the memorization-generalization phenomenon in diffusion models via the lens of AMs, which supports the view of diffusion models as AMs operating above the critical memory load.

1 INTRODUCTION

Hopfield networks are the simplest energy-based models that conceptualize memories as attractor states corresponding to local minima of the energy function, and the memory retrieval as dynamical convergence towards those attractor states (Hopfield, 1982; 1984; Amari, 1972). Recently, they have seen a resurgence of interest due to advances in their memorization capacity. Notably, Dense Associative Memories (DenseAMs), which are extensions of Hopfield networks with super-linear memory capacity (Krotov & Hopfield, 2016; 2018), have paved the way for more sophisticated associative memory (AM) systems (Demircigil et al., 2017; Agliari et al., 2020; Agliari & De Marzo, 2020; Krotov, 2021; Albanese et al., 2022; Millidge et al., 2022; Saha et al., 2023; Hoover et al., 2023a; Karakida et al., 2024; dos Santos et al., 2024; Albanese et al., 2024) — driven in part by their connection to the attention mechanism in transformers (Ramsauer et al., 2021; Krotov & Hopfield, 2021; Hoover et al., 2023a).

Simultaneously, generative diffusion models (Sohl-Dickstein et al., 2015) have gained considerable popularity, due to their flexibility and accuracy in modeling high-dimensional distributions for a variety of domains —such as image generation (Ho et al., 2020; Song et al., 2020; 2021), audio (Chen et al., 2020; Kong et al., 2020; Liu et al., 2023), video synthesis (Ho et al., 2022; Singer et al., 2022; Blattmann et al., 2023; Brooks et al., 2024), and other scientific applications. However,

✉ Correspondents: phamb@rpi.edu and g.raya@jads.nl
[*] denotes equal contribution as first authors.
† denotes equal contribution in advising of the work.

despite their effectiveness, diffusion models pose challenges related to privacy and security, as concerns grow about their tendency to replicate training data (Somepalli et al., 2023a;b; Carlini et al., 2023). Such matters consequently emphasize the need for further understanding of memorization and generalization behaviors in diffusion models.

Recent works (Hoover et al., 2023b; Ambrogioni, 2024; Raya & Ambrogioni, 2024) have begun establishing theoretical connections between DenseAMs and generative diffusion models, showing that the logarithm of the probability of the generated samples in diffusion models can be interpreted as the energy function commonly used in DenseAM model with the softmax activation. This provides the opportunity to study spurious patterns — a possible key to understanding the properties of the memorization-to-generalization transition.

Historically considered as detrimental in AM (Hopfield et al., 1983; Amit et al., 1985; Abu-Mostafa & St. Jacques, 1985), spurious patterns can be interpreted as interpolations of stored patterns, hinting at the network's ability to synthesize new patterns from existing training data. This blending of fundamental memories resembles generative modeling, where the learned representations are used to generate novel patterns. In this way, spurious patterns offer a fascinating framework for exploring the balance between memorization — where models store their training data — and generalization, where they use the underlying features to create genuinely new samples (Kalaj et al., 2024). Studying the conditions under which spurious patterns emerge and how they contribute to the network's behaviour can thus shed light on generalization in both AM and contemporary generative models.

Meanwhile, previous studies have investigated memorization in generative models through different approaches — measuring memorization (Meehan et al., 2020; den Burg & Williams, 2021), examining capacity limits (Somepalli et al., 2023a;b; Yoon et al., 2023), generalization gaps (Li et al., 2024), and learning dynamics (Kadkhodaie et al., 2023; Ventura et al., 2024; Achilli et al., 2024; Kamb & Ganguli, 2024; Ross et al., 2024) — the transition between memorization and generalization phases remains insufficiently understood.

Contributions. Most of the mentioned prior works on the memorization-to-generalization transition approach it from a generalization-centric perspective, while viewing memorization as a "small side effect" of the diffusion models. Our work adopts a complimentary approach by casting the diffusion modeling pipeline into the AM framework. In this approach the training phase of the diffusion model is conceptualized as *writing into the memory* operation and the generation phase as *an attempt of memory recall* — where successful recall leads to retrieving training samples (memorization), while unsuccessful recall results in generating unseen samples (generalization). This point of view allows us to apply theory developed in DenseAMs for analyzing the memorization-to-generalization transition. Notably, in DenseAMs, successful memory recall is known to dwindle as the number of stored data points increases, marked by the emergence of spurious states. Our theory predicts and we empirically verify that these spurious states must exist in conventional diffusion models trained and run using standard methods, appearing at the boundary between memorization and generalization phases.

In accord with previous literature, we find that **(1)** diffusion models undergo a phase transition from memorization to generalization as the number of training points increases; **(2)** at the onset of generalization we detect the emergence of spurious patterns, which have sizable basins of attraction, but are not memorized data points; and lastly, unlike previous works on diffusion models, **(3)** we provide theoretical descriptions distinguishing these spurious states from memorized and generalized patterns in terms of energy landscapes. All of these findings suggest a close parallel between diffusion models and DenseAMs above the critical storage capacity.

2 Diffusion Models and Dense Associative Memory

Given a dataset of i.i.d. samples $\mathbf{y} \in \mathbb{R}^N$ drawn from a target data distribution $p(\mathbf{y})$, diffusion models are a class of generative models, which aims to approximate $p(\mathbf{y})$ by placing a reversible process that maps data to noise and back. The mapping to noise (or forward process) is described by the following Itô Stochastic Differential Equation (SDE) (Song et al., 2021):

$$\mathrm{d}\mathbf{x}_t = \mathbf{f}(\mathbf{x}_t, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}_t \tag{1}$$

which transforms the given data distribution ($\mathbf{x}_0 = \mathbf{y}$) into a simpler one, such as an isotropic Gaussian distribution. Here, $\mathbf{w}_t$ is the standaard Wiener process and $\mathbf{f}(\mathbf{x}_t, t)$ denotes the drift term

that guides the diffusion process, which we will assume to be zero for most part of this paper. Meanwhile, $g(t)$ represents the diffusion coefficient that controls the noise at each time step $t \to T$. In contrast, the reverse process, which removes the injected noise at each t step, is described as

$$\mathrm{d}\mathbf{x}_t = [\mathbf{f}(\mathbf{x}_t, t) - g(t)^2 \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)]\mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}}_t \tag{2}$$

where $\mathrm{d}\bar{\mathbf{w}}_t$ is the standard Wiener process. To effectively solve this equation, one must reliably estimate the score $\nabla_{\mathbf{x}} \log p_t(\mathbf{x})$ via parameterization of the score as a neural network $s_\theta(\mathbf{x}_t, t)$ using the following objective (Song et al., 2021):

$$\theta^* = \arg\min_\theta \mathop{\mathbb{E}}_{t,\mathbf{y},\mathbf{x}_t} \left[\lambda(t) \left\| s_\theta(\mathbf{x}_t, t) - \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t|\mathbf{y}) \right\|^2\right] \tag{3}$$

where $t \sim \mathcal{U}(1, T)$ is sampled from the uniform distribution $\mathcal{U}$ over the set $\{1, 2, \ldots, T\}$, $\mathbf{y} \sim p(\mathbf{y})$ and $\mathbf{x}_t \sim p(\mathbf{x}_t|\mathbf{y})$. Here $p(\mathbf{x}_t|\mathbf{y})$ is the forward process and $\lambda(t)$ is a positive weighting function.

Given the training data distribution, in the variance-exploding setting ($f(\mathbf{x}_t, t) = 0$, $g(t) = \sigma$) the marginal probability distribution of new samples can be computed exactly and is given by

$$p(\mathbf{x}_t, t) = \mathop{\mathbb{E}}_{\mathbf{y}\sim\text{data}} \left[\frac{1}{(2\pi\sigma^2 t)^{\frac{N}{2}}} \exp\Big(- \frac{\|\mathbf{x}_t - \mathbf{y}\|_2^2}{2\sigma^2 t} \Big) \right] \tag{4}$$

Assuming the empirical distribution of the data $p(\mathbf{y}) = \frac{1}{K} \sum_{\mu=1}^{K} \delta^{(N)}(\mathbf{y} - \boldsymbol{\xi}^\mu)$, where $\boldsymbol{\xi}^\mu$ represents an individual data point (with data size K), this marginal distribution can be written as

$$p(\mathbf{x}_t, t) = \frac{1}{K} \sum_{\mu=1}^{K} \frac{1}{(2\pi\sigma^2 t)^{\frac{N}{2}}} \exp\Big(- \frac{\|\mathbf{x}_t - \boldsymbol{\xi}^\mu\|_2^2}{2\sigma^2 t} \Big) \overset{\text{def}}{\equiv} \exp\Big(- \frac{E^{\text{DM}}(\mathbf{x}_t, t)}{2\sigma^2 t} \Big) \tag{5}$$

where we also defined the energy E^{DM} of diffusion model, which up to state-independent terms ($\mathbf{x}$ independent terms), is equal to

$$E^{\text{DM}}(\mathbf{x}_t, t) = -2\sigma^2 t \log \left[\sum_{\mu=1}^{K} \exp\Big(- \frac{\|\mathbf{x}_t - \boldsymbol{\xi}^\mu\|_2^2}{2\sigma^2 t} \Big) \right] \tag{6}$$

As already observed in Ambrogioni (2024), this energy function (6) is closely related to the modern extensions of Hopfield models or DenseAMs, which have large memory capacity (Krotov & Hopfield, 2016; Krotov, 2023). Of particular interest here is the energy function of the DenseAM model studied in Saha et al. (2023) (see also Millidge et al. (2022)):

$$E^{\text{AM}}(\mathbf{x}) = -\beta^{-1} \log \left[\sum_{\mu=1}^{K} \exp\Big(- \beta \|\mathbf{x} - \boldsymbol{\xi}^\mu\|_2^2 \Big) \right] \tag{7}$$

where β is the inverse "temperature", which controls the steepness of the energy landscape around memories $\boldsymbol{\xi}^\mu$. The energy formulas (6) and (7) highlight a close connection between these two models, where the data points in the diffusion model play the same role as the memories in AM. Moreover, the variance of the noise in the forward process acts as the effective temperature β^{-1} in AM, while the reverse process mirrors memory retrieval (Hoover et al., 2023b; Ambrogioni, 2024).

While DenseAMs and diffusion models share fundamental similarities, they operate under different conditions and serve distinct purposes. DenseAMs maintain a constant (typically high) β to accurately recall stored patterns, while diffusion models have time-dependent effective temperatures in non-equilibrium systems described by Eq. (6) and are designed for pattern syntheses. Despite these differences, both systems share a crucial property: their fixed points[1] align with the original data points — diffusion models through the reverse process (2), and AMs through high β retrieval dynamics. Our core message is that the data manifolds in diffusion models emerge when AM networks exceed their critical memory capacity, causing distinct memory attraction basins to merge. At this transition boundary, spurious states emerge, signaling the onset of generalization. Although DenseAMs theoretically possess exponential memory capacity for uncorrelated patterns, high correlations in real data significantly lower the critical memory load (Cortes et al., 1987; Gutfreund, 1988; Krogh & Hertz, 1988; Kanter & Sompolinsky, 1987; Van Hemmen, 1997). This theoretical framework unifies our understanding of how both systems transition from memorization to generalization in the correlated data setting despite their apparent differences.

[1] We remind the reader that the fixed points retrieved from the reverse process correspond to $t = 0$.

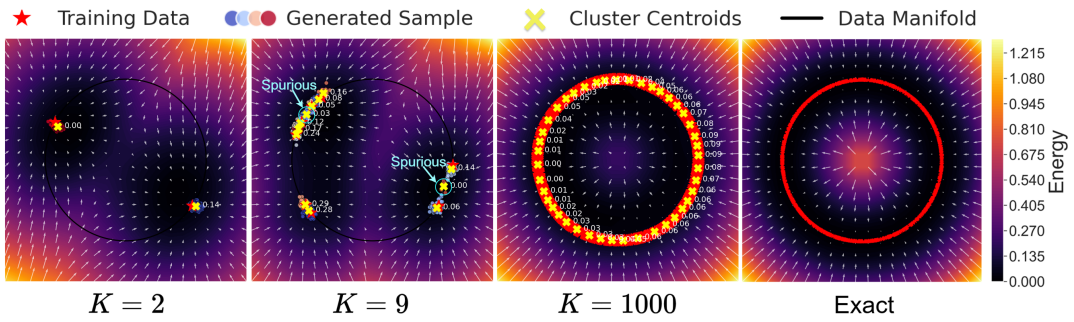

Figure 1: Energy landscapes of diffusion models trained on unit circle data with increasing sample sizes $K \in \{2, 9, 1000\}$. Generated samples are plotted alongside their energy values defined by Eq. (6), with vector fields showing the learned score $s_{\theta^*}(\mathbf{x}_t, t)$. Hierarchical clustering is applied to identify clusters within the generated data, with the energy of each cluster centroid computed. The rightmost panel shows the theoretical $K \to \infty$ solution from Eq. (13). The progression reveals three distinct phases: initial memorization with distinct attraction basins around training points ($K = 2$), emergence of spurious patterns marking the onset of generalization ($K = 9$), and finally a fully generalized regime ($K = 1000$) characterized by a nearly-flat continuous manifold of low-energy states.

3 Memorization-to-Generalization in 2D

To better illustrate the above phenomena, it is instructive to investigate a simple 2-dimensional toy model (see Fig. (1)), which exhibits many aspects of the memorization-generalization transition. Imagine that the training data lies on a unit circle. We are interested in exploring how the shape of the energy function (7) changes as the number of training data points increases.

$[\boldsymbol{K = 1}]$ For a single data point $\boldsymbol{\xi}^1$, Eq. (7) has a single local minimum at that stored pattern.

$$E^{\text{AM}}(\mathbf{x}) = -\beta^{-1} \log \left[\exp \left(-\beta \|\mathbf{x} - \boldsymbol{\xi}^1\|_2^2 \right) \right] = \|\mathbf{x} - \boldsymbol{\xi}^1\|_2^2 \tag{8}$$

The shape of this energy landscape is independent of the value of β.

$[\boldsymbol{K = 2}]$ For two patterns, the energy is given by

$$E^{\text{AM}}(\mathbf{x}) = -\beta^{-1} \log \left[\exp \left(-\beta \|\mathbf{x} - \boldsymbol{\xi}^1\|_2^2 \right) + \exp \left(-\beta \|\mathbf{x} - \boldsymbol{\xi}^2\|_2^2 \right) \right] \tag{9}$$

While for very large $\beta \to \infty$ this energy has two local minima corresponding to the stored patterns. For finite small values of β, there exist configurations of patterns such that the energy has only one local minimum:

$$\boldsymbol{\eta} = \arg\min_{\mathbf{x}} E^{\text{AM}}(\mathbf{x}) \tag{10}$$

such that this local minimum is different from either of the two stored patterns, i.e., $\boldsymbol{\eta} \neq \boldsymbol{\xi}^1$ and $\boldsymbol{\eta} \neq \boldsymbol{\xi}^2$. This "emergent" local minimum is the *spurious state*.

$[\boldsymbol{K \to \infty}]$ Finally, consider the case when the number of training data points is infinite. In this case they can be described by a continuous density of states:

$$p(\mathbf{y}) = \frac{1}{\pi} \delta\left(y_1^2 + y_2^2 - 1\right) \tag{11}$$

The probability of the generated data is proportional (up to terms independent of the state $\mathbf{x}$) to

$$p(\mathbf{x}) \sim \int_{-\infty}^{+\infty} dy_1 dy_2 \; p(\mathbf{y}) \; e^{-\beta \|\mathbf{x} - \mathbf{y}\|_2^2} = e^{-\beta(R^2+1)} I_0(2\beta R) \tag{12}$$

where $I_0(\cdot)$ is a modified Bessel function of the first kind. Refer to Appendix A for more details on deriving Eq. (12). Thus, the energy of the model is given by

$$E^{\text{AM}}(R, \phi) = R^2 + 1 - \frac{1}{\beta} \log \left[I_0(2\beta R) \right] \underset{\beta \to \infty}{\approx} (R - 1)^2 \tag{13}$$

Notice, the dependence on the angle ϕ completely disappeared from the final result. The local minima of Eq. (13) form a *continuous manifold*, corresponding to $R = 1$. Data samples from the

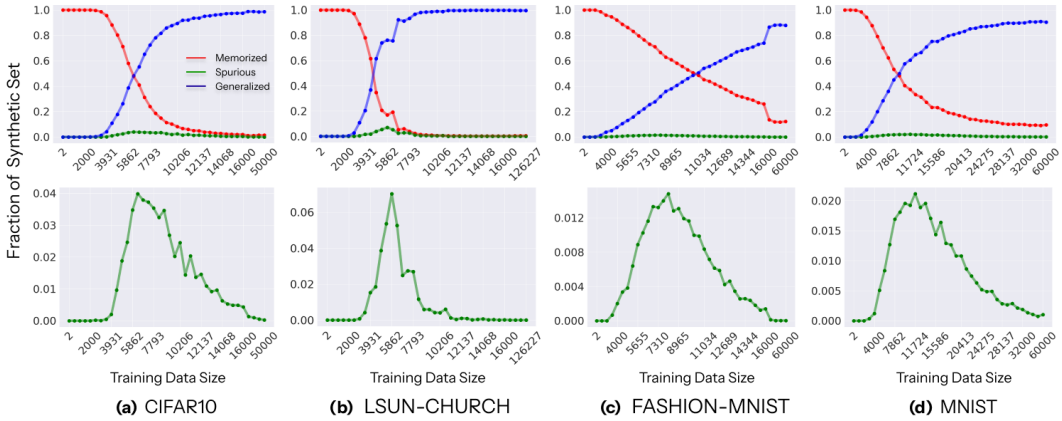

Figure 2: The fractions of memorized, spurious, and generalized samples in synthetic sets across training sizes and different datasets. As the training data size K increases, memorization decreases while the emergence of spurious patterns follows. The fraction of spurious patterns rises and decreases at the boundary between the memorization and generalization phases.

model occupy the vicinity of that manifold, illustrating the fully generalized phase. In the limit of large β the energy landscape is described by a parabola centered around $R = 1$.

Two key questions remain regarding this toy model: **(1)** How many training samples are needed for the model to transition from memorization to generalization under realistic diffusion conditions? **(2)** What role does the neural network modeling the score function $s_\theta(\mathbf{x}_t, t)$ play in this transition? To investigate, we trained a family of diffusion models in this controlled setting (see Appendix B for details). Each model was trained on a dataset of size K, and the resulting energy landscapes obtained from Eq. (6) are shown in Fig. (1). For small training data size (e.g., $K = 2$), the model exhibits memorization, with energy minima tightly aligned with training samples. At $K = 9$, we observe the emergence of spurious states, marking the onset of generalization. Here, the model begins forming new, non-training local minima in the energy landscape. As the training data size further increases, full generalization occurs, illustrated at $K = 1000$, where generated samples closely align with the true data manifold. The right panel of Fig. (1) compares the numerical energy landscape to the analytical expression (13), revealing close resemblance between the two at $K = 1000$.

To better illustrate this transition (detailed in Fig. (2)), we developed detection metrics in Appendix D and computed the fractions of memorized, spurious, and generalized samples for increasing training data sizes K of different datasets, including MNIST (Deng, 2012), FASHION-MNIST (Xiao et al., 2017), CIFAR10 (Krizhevsky et al., 2014), and LSUN-CHURCH (Yu et al., 2015). The results in Fig. (2) clearly demonstrate the memorization-to-generalization transition as K increases, similarly to the toy model in Fig. (1). The collected generated samples of the three sample types, which come from diffusion models trained on different data sizes K, show distinct features of each phase (see Fig. (4) in Appendix E). Specifically, memorized samples are copies of the training data points, spurious ones resemble mixtures of such points while generalized patterns are genuinely novel, resembling very little to any of the other generated patterns (see figures in Appendix G). Please refer to sections (D, E, and F) for full details on the experimentation of the mentioned real datasets and further discussion.

4 Conclusion

In this work we established a novel connection between diffusion models and Dense Associative Memories. We identified emergence of creativity and generalization in diffusion models with a failure of a successful memory recall. We developed an energy-based theoretical description of diffusion model's generative capabilities. Using this theory, we identified a novel phase – *spurious states* – previously overlooked in the memorization-generalization literature on diffusion models. We designed a simple (but non-trivial) analytically solvable model (2D toy model) where all the aspects of this transition can be investigated analytically and tested numerically. Finally, we established that diffusion models trained on natural datasets follow the same transition (memorization to spurious phase to generalization) as predicted by our theory, and as observed in the 2D synthetic model. Thus, spurious states are real, and the peak of their frequency among the generated samples marks the onset of generalization as the training set size is increased.

REFERENCES

Y. Abu-Mostafa and J. St. Jacques. Information capacity of the hopfield model. *IEEE Transactions on Information Theory*, 31(4):461–464, 1985. doi: 10.1109/TIT.1985.1057069.

Beatrice Achilli, Enrico Ventura, Gianluigi Silvestri, Bao Pham, Gabriel Raya, Dmitry Krotov, Carlo Lucibello, and Luca Ambrogioni. Losing dimensions: Geometric memorization in generative diffusion. *arXiv preprint arXiv:2410.08727*, 2024.

Elena Agliari and Giordano De Marzo. Tolerance versus synaptic noise in dense associative memories. *The European Physical Journal Plus*, 135(11):1–22, 2020.

Elena Agliari, Francesco Alemanno, Adriano Barra, Martino Centonze, and Alberto Fachechi. Neural networks with a redundant representation: Detecting the undetectable. *Physical review letters*, 124(2):028301, 2020.

Linda Albanese, Francesco Alemanno, Andrea Alessandrelli, and Adriano Barra. Replica symmetry breaking in dense hebbian neural networks. *Journal of Statistical Physics*, 189(2):24, 2022.

Linda Albanese, Andrea Alessandrelli, Alessia Annibale, and Adriano Barra. About the de almeida–thouless line in neural networks. *Physica A: Statistical Mechanics and its Applications*, 633: 129372, 2024.

S-I Amari. Learning patterns and pattern sequences by self-organizing nets of threshold elements. *IEEE Transactions on computers*, 100(11):1197–1206, 1972.

Luca Ambrogioni. In search of dispersed memories: Generative diffusion models are associative memory networks. *Entropy*, 26(5):381, 2024.

Daniel J. Amit, Hanoch Gutfreund, and H. Sompolinsky. Storing infinite numbers of patterns in a spin-glass model of neural networks. *Phys. Rev. Lett.*, 55:1530–1533, Sep 1985. doi: 10.1103/PhysRevLett.55.1530. URL https://link.aps.org/doi/10.1103/PhysRevLett.55.1530.

Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

Tim Brooks, Bill Peebles, Connor Holmes, Will DePue, Yufei Guo, Li Jing, David Schnurr, Joe Taylor, Troy Luhman, Eric Luhman, Clarence Ng, Ricky Wang, and Aditya Ramesh. Video generation models as world simulators, 2024.

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)*, pp. 5253–5270, 2023.

Nanxin Chen, Yu Zhang, Heiga Zen, Ron J Weiss, Mohammad Norouzi, and William Chan. Wavegrad: Estimating gradients for waveform generation. *arXiv preprint arXiv:2009.00713*, 2020.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, pp. 6572–6583, Red Hook, NY, USA, 2018. Curran Associates Inc.

C Cortes, A Krogh, and JA Hertz. Hierarchical associative networks. *Journal of Physics A: Mathematical and General*, 20(13):4449, 1987.

Mete Demircigil, Judith Heusel, Matthias Löwe, Sven Upgang, and Franck Vermet. On a model of associative memory with huge storage capacity. *Journal of Statistical Physics*, 168(2):288–299, May 2017. ISSN 1572-9613. doi: 10.1007/s10955-017-1806-y. URL http://dx.doi.org/10.1007/s10955-017-1806-y.

Gerrit J.J. Van den Burg and Chris Williams. On memorization in probabilistic deep generative models. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=PlGSgjFK2oJ.

Li Deng. The mnist database of handwritten digit images for machine learning research [best of the web]. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012. doi: 10.1109/MSP.2012.2211477.

J.R. Dormand and P.J. Prince. A family of embedded runge-kutta formulae. *Journal of Computational and Applied Mathematics*, 6(1):19–26, 1980. ISSN 0377-0427. doi: https://doi.org/10.1016/0771-050X(80)90013-3. URL https://www.sciencedirect.com/science/article/pii/0771050X80900133.

Saul José Rodrigues dos Santos, Vlad Niculae, Daniel C McNamee, and Andre Martins. Sparse and structured hopfield networks. In *Forty-first International Conference on Machine Learning*, 2024.

Izrail Solomonovich Gradshteyn and Iosif Moiseevich Ryzhik. *Table of integrals, series, and products*. Academic press, 2014.

Hanoch Gutfreund. Neural networks with hierarchically correlated patterns. *Physical Review A*, 37 (2):570, 1988.

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

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 2020.

Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *arXiv preprint arXiv:2204.03458*, 2022.

Benjamin Hoover, Yuchen Liang, Bao Pham, Rameswar Panda, Hendrik Strobelt, Duen Horng Chau, Mohammed Zaki, and Dmitry Krotov. Energy transformer. In A. Oh, T. Neumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 27532–27559. Curran Associates, Inc., 2023a.

Benjamin Hoover, Hendrik Strobelt, Dmitry Krotov, Judy Hoffman, Zsolt Kira, and Duen Horng Chau. Memory in plain sight: A survey of the uncanny resemblances between diffusion models and associative memories. *arXiv preprint arXiv:2309.16750*, 2023b.

J J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, 1982. doi: 10.1073/pnas.79.8.2554. URL https://www.pnas.org/doi/abs/10.1073/pnas.79.8.2554.

John J Hopfield. Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the national academy of sciences*, 81(10):3088–3092, 1984.

John J. Hopfield, David I. Feinstein, and Richard G. Palmer. 'unlearning' has a stabilizing effect in collective memories. *Nature*, 304:158–159, 1983. URL https://api.semanticscholar.org/CorpusID:4269710.

Michael F Hutchinson. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics-Simulation and Computation*, 18(3):1059–1076, 1989.

Zahra Kadkhodaie, Florentin Guth, Eero P Simoncelli, and Stéphane Mallat. Generalization in diffusion models arises from geometry-adaptive harmonic representation. *arXiv preprint arXiv:2310.02557*, 2023.

Silvio Kalaj, Clarissa Lauditi, Gabriele Perugini, Carlo Lucibello, Enrico M Malatesta, and Matteo Negri. Random features hopfield networks generalize retrieval to previously unseen examples. *arXiv preprint arXiv:2407.05658*, 2024.

Mason Kamb and Surya Ganguli. An analytic theory of creativity in convolutional diffusion models. *arXiv preprint arXiv:2412.20292*, 2024.

I Kanter and Haim Sompolinsky. Associative recall of memory without errors. *Physical Review A*, 35(1):380, 1987.

Ryo Karakida, Toshihiro Ota, and Masato Taki. Hierarchical associative memory, parallelized mlp-mixer, and symmetry breaking. *arXiv preprint arXiv:2406.12220*, 2024.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL https://arxiv.org/abs/1412.6980.

Peter E Kloeden, Eckhard Platen, Peter E Kloeden, and Eckhard Platen. *Stochastic differential equations*. Springer, 1992.

Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. *arXiv preprint arXiv:2009.09761*, 2020.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. URL https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. The cifar-10 dataset. *online: http://www. cs. toronto. edu/kriz/cifar. html*, 55(5):2, 2014.

A Krogh and JA Hertz. Mean-field analysis of hierarchical associative networks with'magnetisation'. *Journal of Physics A: Mathematical and General*, 21(9):2211, 1988.

Dmitry Krotov. Hierarchical associative memory. *arXiv preprint 2107.06446*, 2021. URL https://arxiv.org/abs/2107.06446.

Dmitry Krotov. A new frontier for hopfield networks. *Nature Reviews Physics*, 5(7):366–367, 2023.

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

Dmitry Krotov and John J. Hopfield. Dense associative memory for pattern recognition. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016. URL https://proceedings.neurips.cc/paper_files/paper/2016/file/eaae339c4d89fc102edd9dbdb6a28915-Paper.pdf.

Dmitry Krotov and John J Hopfield. Large associative memory problem in neurobiology and machine learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=X4y_10OX-hX.

Puheng Li, Zhong Li, Huishuai Zhang, and Jiang Bian. On the generalization properties of diffusion models. *Advances in Neural Information Processing Systems*, 36, 2024.

Haohe Liu, Zehua Chen, Yi Yuan, Xinhao Mei, Xubo Liu, Danilo Mandic, Wenwu Wang, and Mark D Plumbley. AudioLDM: Text-to-audio generation with latent diffusion models. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pp. 21450–21474, 2023.

Casey Meehan, Kamalika Chaudhuri, and Sanjoy Dasgupta. A non-parametric test to detect data-copying in generative models. In *International Conference on Artificial Intelligence and Statistics*, 2020.

Beren Millidge, Tommaso Salvatori, Yuhang Song, Thomas Lukasiewicz, and Rafal Bogacz. Universal hopfield networks: A general framework for single-shot associative memory models. In *International Conference on Machine Learning*, pp. 15561–15583. PMLR, 2022.

Bernt Oksendal. *Stochastic differential equations (3rd ed.): an introduction with applications*. Springer-Verlag, Berlin, Heidelberg, 1992. ISBN 3387533354.

Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Searching for activation functions. *CoRR*, abs/1710.05941, 2017. URL http://arxiv.org/abs/1710.05941.

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*, 2021. URL https://openreview.net/forum?id=tL89RnzIiCd.

Gabriel Raya and Luca Ambrogioni. Spontaneous symmetry breaking in generative diffusion models. *Advances in Neural Information Processing Systems*, 36, 2024.

Brendan Leigh Ross, Hamidreza Kamkari, Tongzi Wu, Rasa Hosseinzadeh, Zhaoyan Liu, George Stein, Jesse C Cresswell, and Gabriel Loaiza-Ganem. A geometric framework for understanding memorization in generative models. *arXiv preprint arXiv:2411.00113*, 2024.

Bishwajit Saha, Dmitry Krotov, Mohammed J Zaki, and Parikshit Ram. End-to-end differentiable clustering with associative memories. In *International Conference on Machine Learning*, pp. 29649–29670. PMLR, 2023.

Tim Salimans, Andrej Karpathy, Xi Chen, and Diederik P Kingma. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. *arXiv preprint arXiv:1701.05517*, 2017.

Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, et al. Make-a-video: Text-to-video generation without text-video data. *arXiv preprint arXiv:2209.14792*, 2022.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, 2015.

Gowthami Somepalli, Vasu Singla, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Diffusion art or digital forgery? investigating data replication in diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6048–6058, 2023a.

Gowthami Somepalli, Vasu Singla, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Understanding and mitigating copying in diffusion models. *Advances in Neural Information Processing Systems*, 36:47783–47803, 2023b.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*, 2020.

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*, 2021.

Aaron Van den Oord, Nal Kalchbrenner, Lasse Espeholt, Oriol Vinyals, and Alex Graves. Conditional image generation with pixelcnn decoders. *Advances in neural information processing systems*, 29, 2016.

JL Van Hemmen. Hebbian learning, its correlation catastrophe, and unlearning. *Network: Computation in Neural Systems*, 8(3):V1, 1997.

Enrico Ventura, Beatrice Achilli, Gianluigi Silvestri, Carlo Lucibello, and Luca Ambrogioni. Manifolds, random matrices and spectral gaps: The geometric phases of generative diffusion. *arXiv preprint arXiv:2410.05898*, 2024.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint 1708.07747*, 2017.

TaeHo Yoon, Joo Young Choi, Sehyun Kwon, and Ernest K Ryu. Diffusion probabilistic models generalize when they fail to memorize. In *ICML 2023 Workshop on Structured Probabilistic Inference Generative Modeling*, 2023.

Fisher Yu, Yinda Zhang, Shuran Song, Ari Seff, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018.

APPENDIX

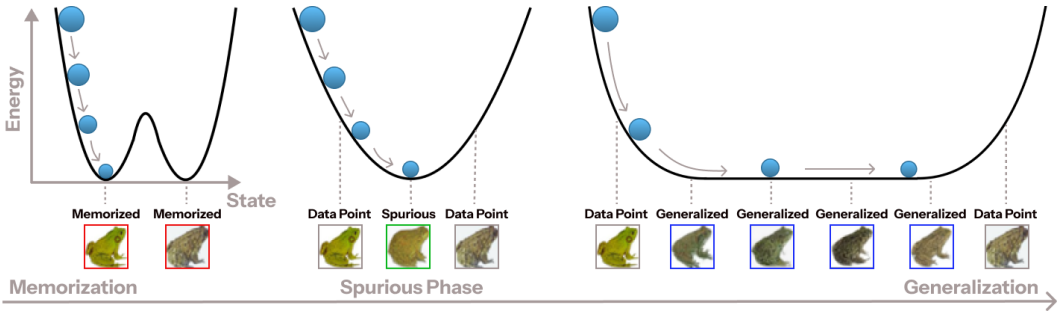

Figure 3: A simple illustration depicting the change in the energy landscape as the size of the training dataset is increased. In the small data regime, the model memorizes the training data points as local minima of an energy function. When the amount of training data exceeds the memory capacity of the model, spurious patterns are formed and training data points are no longer the minima. Subsequent increase of the training set size leads to the generalization phase, which is defined by the formation of continuous manifold of the low energy states. Note, although the energy landscape in the generalization phase is shown as flat, small bumps are permitted as long as they are small compared to the thermal noise in the generation phase.

A FURTHER DETAILS OF THE 2D TOY EXAMPLE

In order to obtain the results in Section (3), it is straightforward to introduce polar coordinates for both the state vector and the training data:

$$\begin{cases} x_1 = R\cos(\phi) \\ x_2 = R\sin(\phi) \end{cases} \qquad \begin{cases} y_1 = r\cos(\varphi) \\ y_2 = r\sin(\varphi) \end{cases}$$

The integral in equation (12) can then be written as

$$p(\mathbf{x}) \sim \int_0^{2\pi} d\varphi \int_0^{\infty} r dr \frac{1}{\pi}\delta(r^2-1)e^{-\beta[R^2+r^2-2Rr\cos(\varphi-\phi)]} = e^{-\beta(R^2+1)}I_0(2\beta R)$$

and explicitly computed using the definition of the modified Bessel functions (Gradshteyn & Ryzhik, 2014).

B DETAILS OF THE 2D TOY EXAMPLE

This section describes the two-dimensional toy example used in Section (3), including details about the neural network architecture, training, generation, and computation of the model's energy. All of the corresponding code will be made publicly available.

Dataset. To construct the dataset, we uniformly sample points on the unit circle. Concretely, we take $\mathbf{y} \sim p(\mathbf{y}) = \frac{1}{2\pi}\delta(r-1)$, where $r = \sqrt{y_1^2+y_2^2}$ is the radius and $\delta(r-1)$ is the delta function ensures that all probability mass lies on radius $r = 1$. The factor $\frac{1}{2\pi}$ guarantees that the angles are sampled uniformly, relying on the identity $\delta(\sqrt{y_1^2+y_2^2}-1) = 2\delta(y_1^2+y_2^2-1)$ in $p(\mathbf{y}) = \frac{1}{\pi}\delta(y_1^2+y_2^2-1)$. In practice, we sample polar coordinates (r, φ) with $r = 1$ and uniformly sample the angular coordinate φ from $[0, 2\pi]$. We then convert (r, φ) to Cartesian coordinates (y_1, y_2) using $y_1 = r\cos\varphi$ and $y_2 = r\sin\varphi$.

We first created a data set of $K = 60000$ points using a fixed random seed, then built smaller subsets (e.g., $K \in \{2, 4, 9, 1000, 10000\}$) by progressively adding distinct samples without replacement, ensuring that each subset is a strict subset of the next. This approach allows us to systematically examine how the model behaves, as we vary the size of the training set.

Exact Score. Following Eq. (13) specified in Section (3), energy can be written in closed form, whose local minima form the continuous data manifold at $R = 1$ (throughout this paper we use r for the polar radius of the training data and R for the polar radius of the generated samples). Furthermore, in the case of many data points or $K \to \infty$, we have $p(\mathbf{y}) = \frac{1}{\pi}\delta(y_1^2 + y_2^2 - 1)$. Since $p(R) \propto \exp^{-E^{\text{AM}}(R)}$, the score can be described as

$$\begin{aligned} -\nabla_R \log p(R) &= \nabla_R E^{\text{AM}}(R) \\ &= 2R - \frac{2\beta I_1(2\beta R)}{\beta I_0(2\beta R)} \\ &= 2R - 2\frac{I_1(2\beta R)}{I_0(2\beta R)} \end{aligned} \tag{14}$$

where $E^{\text{AM}}(R)$ is Eq. (13), and I_1 and I_0 are the first-order and zero-order modified Bessel functions of the first kind, respectively. With further expansion, the score becomes

$$\nabla_R \log p(R) = 2\left(\frac{I_1(2\beta R)}{I_0(2\beta R)} - R\right) \qquad \Rightarrow \qquad \nabla_x \log p(x) = 2\left(\frac{I_1(2\beta R)}{I_0(2\beta R)} - R\right)\frac{x}{R} \tag{15}$$

We implemented Eq. (13) on a 20×20 grid to obtain the exact energy and score of the model, overlaying 1000 samples from the training set. For visualization purposes, we set $\beta = 20$, ensuring a well-defined energy landscape. Additionally, we normalized the energy by subtracting its minimum value to ensure that the lowest energy point remains at zero.

VE-SDE. To align with the theoretical framework, we employ a variance exploding (VE) SDE of the form

$$\mathrm{d}\mathbf{x}_t = \sigma\, \mathrm{d}\mathbf{w}_t \tag{16}$$

where the diffusion coefficient $g(t) = \sigma$. Assuming $t \in (0, 1]$, the variance of the diffusion kernel is given by $\int_0^t g^2(s)\, \mathrm{d}s = \sigma^2 t$, which allows us to construct the corresponding Gaussian kernel

$$p(\mathbf{x}_t|\mathbf{y}) = \mathcal{N}(\mathbf{x}_t; \mathbf{y}, \sigma^2 t\mathbf{I})$$

However, in practice we used $t \in [\epsilon, 1]$ with $\epsilon = 10^{-5}$ for numerical stability. The generative dynamics are given by

$$\mathrm{d}\mathbf{x}_t = \big[-\sigma^2 \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)\big]\mathrm{d}t + \sigma^2\, \mathrm{d}\mathbf{w}_t. \tag{17}$$

where it runs backwards in time to match the description of Song et al. (2021). The deterministic Probability Flow ODE is

$$\frac{\mathrm{d}\mathbf{x}_t}{\mathrm{d}t} = -\frac{1}{2}\sigma^2 \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t) \tag{18}$$

which shares the same marginal distributions as the SDE in Eq. (17) and is useful for likelihood estimation.

Model and Training Details. For each training size K in Fig. (1), we use a Unet inspired MLP-based network to estimate the score $s_\theta(\mathbf{x}_t, t)$, without using any down- or up- sampling blocks. The general architecture includes **(1)** a Fourier random feature (FRF) timestep embedding layer and a linear layer which projects the concatenation of FRF timestep embedding and input into a latent dimension of 256; **(2)** an encoder which consists of four non-convolutional residual blocks (using the same latent dimension) with Swish (Ramachandran et al., 2017) activation in between; **(3)** a decoder which consists of the same number of residual blocks; and **(4)** a linear block which projects the latent variable back to the 2D space.

The network was trained in continuous time (with 1000 discretized steps) with $\sigma = 1$ using the objective function from (Ho et al., 2020; Song et al., 2021). Optimization is performed using the Adam (Kingma & Ba, 2017) optimizer with a learning rate $lr = 10^{-4}$. The batch size is set to $\textbf{min}(K, 500)$. All models are trained for $800,000$ iterations with the maximum batch size of 500. Meanwhile, we use Euler-Maruyama discretization method (Kloeden et al., 1992) to solve Eq. (17) of the reverse SDE for the generation of samples.

Empirical distribution. We represent the data as an empirical distribution

$$p(\mathbf{y}) = \frac{1}{K}\sum_{\mu=1}^{K} \delta^{(N)}(\mathbf{y} - \boldsymbol{\xi}^\mu) \tag{19}$$

where N is the dimensionality of the data point $\mathbf{y}$. Since $\mathbf{x}_t$ is drawn from the forward process distribution $p(\mathbf{x}_t|\mathbf{y})$ that is conditioned on the data point $\mathbf{y}$, which can be expressed as a Gaussian kernel $\mathcal{N}(\mathbf{x}_t; \mathbf{y}, \sigma^2 t\mathbf{I})$. We can obtain the distribution $p(\mathbf{x}_t, t)$ as

$$\begin{aligned} p(\mathbf{x}_t, t) = \int p(\mathbf{x}_t \mid \mathbf{y})\, p(\mathbf{y})\, \mathrm{d}\mathbf{y} &= \frac{1}{K} \sum_{\mu=1}^{K} \int \mathcal{N}\big(\mathbf{x}_t; \mathbf{y}, \sigma^2 t\mathbf{I}\big)\, \delta^{(N)}\big(\mathbf{y} - \boldsymbol{\xi}^{\mu}\big)\, \mathrm{d}\mathbf{y} \\ &= \frac{1}{K} \sum_{\mu=1}^{K} \mathcal{N}\big(\mathbf{x}_t;\, \boldsymbol{\xi}^{\mu},\, \sigma^2 t\, \mathbf{I}\big) \\ &= \frac{1}{K} \sum_{\mu=1}^{K} \frac{1}{(2\pi\, \sigma^2 t)^{\frac{N}{2}}} \exp\Big(-\frac{\|\mathbf{x}_t - \boldsymbol{\xi}^{\mu}\|_2^2}{2\, \sigma^2 t}\Big) \end{aligned} \tag{20}$$

which is Eq. (5), when the variance of the Gaussian kernel is $\sigma^2 t$.

Likelihood Computation. Following Song et al. (2021) we can compute the log-likelihood $\log p_\theta(\mathbf{x}_0)$ given by a diffusion model, with the instantaneous change of variable formula (Chen et al., 2018), where $\mathbf{x}_0$ are the generated samples from the model. Replacing Eq. (17) into the log-likelihood equation,

$$\log p_0(\mathbf{x}_0; \theta) = \log p_T(\mathbf{x}_T) + \int_0^T \nabla \cdot \tilde{\mathbf{f}}(\mathbf{x}_t, t) \mathrm{d}t \tag{21}$$

where $\tilde{\mathbf{f}}(\mathbf{x}_t, t) = -\frac{1}{2}\sigma^2 \nabla_{\mathbf{x}_t} \log p_t(\mathbf{x}_t)$. The function $\tilde{\mathbf{f}}(\mathbf{x}_t, t)$ came from the above Probability Flow ODE (Eq. (18)), and $\nabla \cdot \tilde{\mathbf{f}}(\mathbf{x}_t, t)$ denotes the divergence of the function (or the trace of its Jacobian).

To estimate the likelihood of the model, we integrate $\nabla \cdot \tilde{\mathbf{f}}(\mathbf{x}_t, t)$ from a small time $\epsilon = 10^{-5}$ to $T = 1$ using a numerical integrator and add the prior logarithmic likelihood to it following Eq. (21). The divergence term $\nabla \cdot \tilde{\mathbf{f}}(\mathbf{x}_t, t)$ is computed using the Laplacian, which is computationally feasible for this toy example, instead of using the Hutchinson trace estimator (Hutchinson, 1989) done in Song et al. (2021). We use the RK45 method (Dormand & Prince, 1980) implemented in the ODE solver *scipy.integrate.solve_ivp* from Scipy (Virtanen et al., 2020) to solve the above integral.

Energy Computation. To compute the energy, we use Eq. (6), which can be also computed using Eq. (21). From the Boltzmann distribution $p(\mathbf{x}) = \frac{1}{Z}\exp\big[-\beta E(\mathbf{x})\big]$, we obtained $\log p(\mathbf{x}) \propto -\beta E(\mathbf{x})$. In our case the inverse temperature is $\beta = \frac{1}{2\sigma^2 t}$. The equation for our VE-SDE thus becomes

$$E^{\mathrm{DM}}(\mathbf{x}_t, t) = -2\sigma^2 t \log p(\mathbf{x}_t, t) \tag{22}$$

To visualize the energy landscape, we observe the energy at $t = 0.15$ which corresponds to $\beta = 3.3$.

Sample Clustering. To cluster the generated samples, we use the `AgglomerativeClustering` algorithm from scikit-learn, which is a part of SciPy (Virtanen et al., 2020).

C Additional Details on the Likelihood of Diffusion Models

For concreteness, we further explain the likelihood computation in Eq. (21) within this section. Specifically, we want to highlight the formulation done by Song et al. (2021) which resulted in Eq. (21), and Chen et al. (2018) for the term $\int_0^T \nabla \cdot \tilde{\mathbf{f}}(\mathbf{x}, t)\mathrm{d}t$ in the same equation.

Probability Flow ODE. Suppose we have the following forward process

$$\mathrm{d}\mathbf{x}_t = \mathbf{f}(\mathbf{x}_t, t)\mathrm{d}t + \mathbf{g}(\mathbf{x}_t, t)\mathrm{d}\mathbf{w}_t \tag{23}$$

where $\mathbf{f}(\cdot, t) : \mathbb{R}^N \to \mathbb{R}^N$, $\mathbf{g}(\cdot, t) : \mathbb{R}^N \to \mathbb{R}^{N\times N}$ and N denotes the dimensionality of $\mathbf{x}_t$. Using the derivations done by Song et al. (2021) in their Appendix D, the evolution of the marginal

probability density $p_t(\mathbf{x}_t)$ is

$$\frac{\partial p_t(\mathbf{x}_t)}{\partial t} = -\sum_{i=1}^{N} \frac{\partial}{\partial x_i}\big[f_i(\mathbf{x}_t, t)p_t(\mathbf{x}_t)\big] + \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \frac{\partial^2}{\partial x_i \partial x_j}\left[\sum_{k=1}^{N} g_{ij}(\mathbf{x}_t, t)g_{jk}(\mathbf{x}_t, t)p_t(\mathbf{x}_t)\right] \tag{24}$$

which corresponds to Fokker-Planck equation (Oksendal, 1992).

$$\begin{aligned}
\frac{\partial p_t(\mathbf{x}_t)}{\partial t} &= -\sum_{i=1}^{N} \frac{\partial}{\partial x_i}\big[f_i(\mathbf{x}_t, t)p_t(\mathbf{x}_t)\big] + \frac{1}{2}\sum_{i=1}^{N} \frac{\partial}{\partial x_i}\left[\sum_{j=1}^{N} \frac{\partial}{\partial x_j}\left[\sum_{k=1}^{N} g_{ij}(\mathbf{x}_t, t)g_{jk}(\mathbf{x}_t, t)p_t(\mathbf{x}_t)\right]\right] \\
&= -\sum_{i=1}^{N} \frac{\partial}{\partial x_i}\big[f_i(\mathbf{x}_t, t)p_t(\mathbf{x}_t)\big] \\
&\quad + \frac{1}{2}\sum_{i=1}^{N} \frac{\partial}{\partial x_i}\Big[p_t(\mathbf{x}_t)\,\nabla \cdot \big[\mathbf{g}(\mathbf{x}_t, t)\mathbf{g}(\mathbf{x}_t, t)^\top\big] + p_t(\mathbf{x}_t)\big[\mathbf{g}(\mathbf{x}_t, t)\mathbf{g}(\mathbf{x}_t, t)^\top\big]\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t)\Big] \\
&= -\sum_{i=1}^{N} \frac{\partial}{\partial x_i}\Big\{f_i(\mathbf{x}_t, t)p_t(\mathbf{x}_t) \\
&\quad - \frac{1}{2}\Big[\nabla \cdot \big[\mathbf{g}(\mathbf{x}_t, t)\mathbf{g}(\mathbf{x}_t, t)^\top\big] + \big[\mathbf{g}(\mathbf{x}_t, t)\mathbf{g}(\mathbf{x}_t, t)^\top\big]\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t)\Big]p_t(\mathbf{x}_t)\Big\} \\
&= -\sum_{i=1}^{N} \frac{\partial}{\partial x_i}\big[\tilde{f}_i(\mathbf{x}_t, t)\big]p_t(\mathbf{x}_t)
\end{aligned} \tag{25}$$

where $\tilde{\mathbf{f}}(\mathbf{x}_t, t) = \mathbf{f}(\mathbf{x}_t, t) - \frac{1}{2}\nabla \cdot \big[\mathbf{g}(\mathbf{x}_t, t)\mathbf{g}(\mathbf{x}_t, t)^\top\big] - \frac{1}{2}\big[\mathbf{g}(\mathbf{x}_t, t)\mathbf{g}(\mathbf{x}_t, t)^\top\big]\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t)$. With careful inspection of Eq. (25), it is equal to the Liouville equation if the diffusion term $\tilde{\mathbf{g}}(\mathbf{x}, t) = 0$ and essentially, it is the probability flow ODE:

$$\begin{aligned}
\mathrm{d}\mathbf{x}_t &= \tilde{\mathbf{f}}(\mathbf{x}_t, t)\mathrm{d}t + \tilde{\mathbf{g}}(\mathbf{x}_t, t)\mathrm{d}\mathbf{w}_t \\
&= \left\{\mathbf{f}(\mathbf{x}_t, t) - \frac{1}{2}\nabla \cdot \big[\mathbf{g}(\mathbf{x}_t, t)\mathbf{g}(\mathbf{x}_t, t)^\top\big] - \frac{1}{2}\big[\mathbf{g}(\mathbf{x}_t, t)\mathbf{g}(\mathbf{x}_t, t)^\top\big]\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t)\right\}\mathrm{d}t
\end{aligned} \tag{26}$$

Using Eq. (26), we can derive an appropriate probability flow ODE from the forward process (1) introduced in the main text. For example, we have the following equation

$$\mathrm{d}\mathbf{x}_t = \underbrace{\left\{\mathbf{f}(\mathbf{x}_t, t) - \frac{1}{2}g(t)^2\nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t)\right\}}_{:=\tilde{\mathbf{f}}(\mathbf{x}_t, t)}\mathrm{d}t \tag{27}$$

for the toy model where $\mathbf{f}(\mathbf{x}_t, t) = 0$.

Log-likelihood. Furthermore, if we first take the logarithm of Eq. (25) where

$$\frac{\partial \log p_t(\mathbf{x}_t)}{\partial t} = \frac{1}{p_t(\mathbf{x}_t)}\frac{\partial p_t(\mathbf{x}_t)}{\partial t} = -\sum_{i=1}^{N} \frac{\partial \tilde{f}_i(\mathbf{x}_t, t)}{\partial x_i} = \nabla \cdot \tilde{\mathbf{f}}(\mathbf{x}_t, t) \tag{28}$$

we can compute the log-likelihood of $p_0(\mathbf{x}_0)$ using the following equation

$$\log p_0(\mathbf{x}_0) = \log p_T(\mathbf{x}_T) + \int_0^T \nabla \cdot \tilde{\mathbf{f}}(\mathbf{x}_t, t)\mathrm{d}t \tag{29}$$

where $\nabla \cdot \tilde{\mathbf{f}}(\mathbf{x}_t, t)$ is parameterized as $\nabla \cdot \tilde{\mathbf{f}}_\theta(\mathbf{x}_t, t)$ since $s_\theta(\mathbf{x}_t, t) = \nabla_{\mathbf{x}_t}\log p_t(\mathbf{x}_t; \theta)$. Moreover, in general the term $\nabla \cdot \tilde{\mathbf{f}}_\theta(\mathbf{x}_t, t)$ is computed via the Hutchinson trace estimator (Hutchinson, 1989),

$$\nabla \cdot \tilde{\mathbf{f}}_\theta(\mathbf{x}_t, t) = \mathbb{E}_{p(\epsilon)}\big[\epsilon^\top \nabla_{\mathbf{x}}\tilde{\mathbf{f}}_\theta(\mathbf{x}_t, t)\epsilon\big] \tag{30}$$

where $\nabla_{\mathbf{x}}\tilde{\mathbf{f}}_\theta(\mathbf{x}_t, t)$ denotes the Jacobian of $\tilde{\mathbf{f}}_\theta(\mathbf{x}_t, t)$ and the random variable ϵ satisfies $\mathbb{E}_{p(\epsilon)}[\epsilon] = 0$ and $\mathrm{Cov}_{p(\epsilon)}[\epsilon] = \mathbf{I}$. However, for the toy model in Section (3), due to its low dimensionality we find that it is possible to not utilize the Hutchinson trace estimator at all, and instead opt to compute the trace of the Jacobian without using the noise estimators.

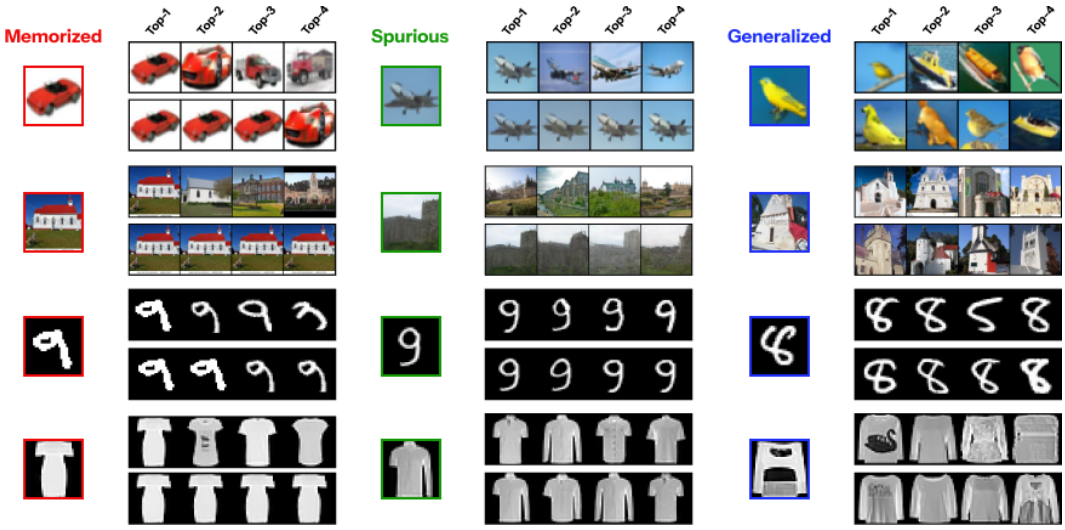

Figure 4: Examples of memorized, spurious, and generalized samples in their respective columns for four datasets. For each target image (shown on the left), its ***top-4*** nearest neighbors from the training set ***(top row)*** and the synthetic set ***(bottom row)*** are shown to highlight the novelty or the commonality of the target image with respect to training and synthetic sets. Memorized samples are duplicates of the training points. Spurious samples are copies of the synthetic set that do not appear in the training set. Generalized samples do not belong to either of the two sets, indicating that they are genuinely novel samples that are not generated repeatedly even if the generation process is performed multiple times.

D Detection Metrics

The main goal of our work is to establish the existence of spurious states in diffusion models trained on natural datasets, not just toy model problems. With this goal in mind we design detection metrics that can classify any generated sample into one of the three categories: memorized sample, spurious sample, and generalized sample. Fundamentally, these metrics defined below rely on two datasets: S — the training dataset used to train the model G_{θ^*} and S' — the synthetic dataset generated from G_{θ^*}, where θ^* is defined according to Eq. (3). The size of the synthetic set S' is assumed to be much bigger than the training set size.

The core intuition comes from the typical energy landscape around these three distinct kinds of generated samples, see Fig. (3). Memorized samples are elements of the synthetic set S', which have duplicates in the training set S. Spurious samples are elements of the synthetic set S', which have duplicates in the synthetic set S', but do not have duplicates in S. In other words, the same spurious sample appears in the synthetic set more than once. Finally, generalized samples are elements of S', which do not have duplicates in both the synthetic set S' and the training set S.

With this intuition in mind we examine two histograms, see Fig. (5). First, for every element of the synthetic set S' the distance to the closest nearest neighbor from the training set S is shown by the olive histogram. Second, for every element of the synthetic set S' the distance to the closest nearest neighbor in the synthetic set S' is shown by the grey histogram. Both histograms exhibit a clear bimodal shape, indicating that at least two groups of samples are present in the synthetic set. This bimodal structure makes it possible to chose two thresholds δ_m ("m" stands for memorized) and δ_s ("s" stands for spurious) to separate the samples forming the two peaks in each histogram. The left peak of the olive histogram is composed of synthetic samples that are a small distance away from the training set can therefore be considered as memorized samples. The left peak of the gray histogram contains synthetic samples that have duplicates in the synthetic set, indicating they can be memorized if the same sample appears in the training set, or spurious, otherwise. The right peak of the gray histogram contains generalized samples, and the right peak of the olive histogram contains generalized and spurious samples. These observations make it possible to formally define the following detection metrics.

Metric D.1 (**Memorization Detection**). Following Yoon et al. (2023), we define the memorization detection metric $\mathcal{M}$. Given a target pattern $\hat{\mathbf{x}} \in S'$ and its first nearest neighbor $\mathbf{x}_1$ extracted from

the training set S according to a distance measure $d(\cdot, \cdot)$, where a small distance corresponds to high similarity, metric $\mathcal{M}$ detects sample $\hat{\mathbf{x}}$ as memorized if $d(\hat{\mathbf{x}}, \mathbf{x}_1)$ is small.

$$\mathcal{M}(\hat{\mathbf{x}}, S) = \mathbb{I}\Big(d(\hat{\mathbf{x}}, \mathbf{x}_1) \leq \delta_m\Big) \tag{31}$$

where $\mathbb{I}$ represents the indicator function and $\delta_m \in \mathbb{R}$ is a threshold defined by the distance histograms (see Fig. (5)).

Metric D.2 (Spurious Detection). Given a synthetic sample $\hat{\mathbf{x}}$ and its first nearest neighbor from the synthetic set $\mathbf{x}'_1 \in S'$, the spurious pattern detection metric $\mathcal{S}$ identifies instances where the model generates outputs that do not belong to the training set but have high similarity with samples from the synthetic set (left peak of the gray histogram in Fig. (5) after the exclusion of memorized samples)

$$\mathcal{S}(\hat{\mathbf{x}}, S, S') = \mathbb{I}\Big(d(\hat{\mathbf{x}}, \mathbf{x}'_1) \leq \delta_s\Big) \wedge \neg\mathcal{M}(\hat{\mathbf{x}}, S) \tag{32}$$

where $\delta_s \in \mathbb{R}$ is a threshold value chosen based on the metric $d(\cdot, \cdot)$ and the gray distance histogram (see Fig. (5)).

Metric D.3 (Generalization Detection). The sample $\hat{\mathbf{x}}$ is generalized if it is neither memorized nor spurious

$$\mathcal{G}(\hat{\mathbf{x}}, S, S') = \neg\mathcal{M}(\hat{\mathbf{x}}, S) \wedge \neg\mathcal{S}(\hat{\mathbf{x}}, S, S') \tag{33}$$

E Memorization-Generalization Transition

Using our detection metrics, we computed the fractions of memorized, spurious, and generalized samples for various sizes of training data for different datasets (see Fig. (2)). These datasets include MNIST (Deng, 2012), FASHION-MNIST (Xiao et al., 2017), CIFAR10 (Krizhevsky et al., 2014), and LSUN-CHURCH (Yu et al., 2015) scaled down to 64×64 resolution using center-crop and down-scale. For each dataset, we trained DDPM-based diffusion models (Ho et al., 2020) for $M = 38$ different training set sizes obtained by using a fixed random seed to split. The same training setting as DDPM was used with the exception of using no random flip, modifying batch size, and the channel multipliers in the Unet backbone to accommodate for the dataset's dimensionality. For each model $\alpha = 1, ..., M$, trained on the training set S_α, a synthetic set S'_α was generated for each model. To account for duplication, we ensured that each S'_α is four times the size of its corresponding S_α. Each sample $\hat{\mathbf{x}} \in S'_\alpha$ is classified as either memorized, spurious, or generalized using the above metrics (31), (32), and (33). The fractions of these three pattern types with respect to the size of S'_α are plotted in Fig. (2). For each dataset our smallest model was trained on the training set of $|S_1| = 2$ data points, and the largest model was trained on the entire original training set $S_M = S$. Please refer to Appendix (F) for more details on training, selection of data sizes, and detection metrics' hyperparameters for each dataset.

The results in Fig. (2) clearly demonstrate the transition from memorization to generalization as the dataset size increases. Meanwhile, the collected samples also show distinct characteristics in each of the considered phases (see Fig. (4)). Specifically, in the small data regime, we see in Fig. (2) that the diffusion model predominantly replicates the training data (see the memorized panel of Fig. (4), or additional samples in Fig. (9) of the Appendix). As the data size surpasses the memorization capacity, e.g., at $K = 3931$ for CIFAR10 in Fig. (2), we observe a critical transition where the memorization fraction declines and spurious patterns become prominent. Such patterns exhibit strong duplication in the synthetic set (see the spurious panel of Fig. (4) and additional samples in Fig. (10) of the Appendix). The emergence of such patterns aligns with the onset of generalization, as the model moves away from strictly reproducing the training set and starts generating novel combinations of learned features. Subsequent increase of the training set size leads to the decrease of the fractions of memorized and spurious samples, signaling the transition to the full generalization regime. At this state, the model completely loses its replication ability and no duplicate samples are detected in either training or synthetic sets (see the generalized panel of Fig. (4) and additional samples in Fig. (11) of the Appendix).

F Experimental Details on Transition Mapping

Points Selection. For the computation of Fig. (2), we follow the experiment setup of Yoon et al. (2023) and conduct a sparse search starting at data size 500 and doubling it all the way to the total data size $|S|$, i.e., $K = 500, 1000, 2000, \ldots, |S|$. We then identified two transitional critical points, A and B (see Table (1)), to conduct a more fine-grained search to elucidate the memorization-generalization transition. Point A indicates the initial drop in memorization while B signals the plateauing of memorization. To capture the details of the memorization-generalization transition, we perform a search of 30 inclusive linearly spaced points, from A to B. For regions outside of the transition, using linear spacing, we sample 5 points from $|S| = 2$ to point A, and another 5 points from B to the total dataset size $|S|$, inclusively. In other words, we train a separate model for each selected data size and generate the corresponding evaluation and synthetic sets. Then, for Fig. (2), we computed the memorization, spurious, and generalization fractions using equations (31), (32), and (33). The selection process for the values δ_m and δ_s is explained below.

Dataset	**Point A** (Data Size)	**Point B** (Data Size)	**Total** (Data Size)
CIFAR10	2,000	16,000	50,000
LSUN-CHURCH	2,000	16,000	126,227
FASHION-MNIST	4,000	16,000	60,000
MNIST	4,000	32,000	60,000

Table 1: Table showing the critical points A and B of the memorization-generalization transition for each dataset. We use 30 linearly spaced points between A and B to finely characterize the transition plots shown in Fig. (2).

Selection of the Distance Metric. For high-dimensional datasets, e.g., CIFAR10 and LSUN-CHURCH, we utilize LPIPS (Zhang et al., 2018) with the AlexNet (Krizhevsky et al., 2012) backbone, as the function $d(\cdot, \cdot)$ for both memorization and spurious detection metrics (31) and (32). We selected this approach since it is a commonly used perceptual metric that compares the similarity between two images based on their feature representations. Moreover, it has been shown to better align with human judgment of visual similarity, making it ideal for assessing the quality and diversity of generated samples in these high-dimensional image datasets. For simpler datasets like MNIST and FASHION-MNIST, where images are single-channel and less complex, we found that L_2-distance suffices for both memorization and spurious detection metrics.

Dataset	δ_m	δ_s
CIFAR10	0.03	0.018
LSUN-CHURCH	0.12	0.05
FASHION-MNIST	4.5	2.5
MNIST	4.5	3.5

Table 2: Table displaying memorized and spurious threshold values in the computation of the transition plots in Fig. (2).

Selection of Threshold Values δ_m and δ_s. The detection thresholds, δ_m and δ_s, were set based on the chosen distance metric $d(\cdot, \cdot)$ and visual inspection of the distance histograms for bimodality as mentioned in Appendix (D) of the main text. Moreover, these thresholds were chosen to reflect the varying visual complexity and feature richness of the datasets, and their values for each dataset are shown in Table (2). For CIFAR10, we utilized the histograms obtained at $K = 7310$, which demonstrate strong bimodality shape, to tune our threshold values to minimize the number of spurious samples classified as memorized samples and vice versa, via manual inspection. For manual inspection, a general rule we followed — it is better to have more false positives in the generalized set than the memorized set since spurious samples can be seen as early-generalized samples. Thus, we observe the least-5 spurious samples and ensure that they do not resemble memorized samples. We performed the same process for memorized samples. The entire process is then repeated for the other three datasets. We utilized $K = 4896$ for LSUN-CHURCH, $K = 21379$ for MNIST, and $K = 7724$ for FASHION-MNIST to select δ_m and δ_s. For more details on the selected histograms, please refer to Figs. (5), (7), (6), and (8).

Dataset	Initial Latent	Channel Multipliers	Number of Parameters	Batch Size	Training Iterations
CIFAR10	128	(1, 2, 2, 2)	35.7M	128	500,000
LSUN-CHURCH	96	(1, 1, 2, 2, 4, 4)	61.7M	64	800,000
FASHION-MNIST	128	(1, 2, 2)	24.5M	128	400,000
MNIST	128	(1, 2, 2)	24.5M	128	400,000

Table 3: Table displaying both model and training configurations for each dataset.

Model and Training Details. For each point in our transition plots in Fig. (2), we train a DDPM-based diffusion model, where the score model is a PixelCNN++ based Unet (Van den Oord et al., 2016; Salimans et al., 2017). We keep the variances, $\beta_{\text{min}} = 10^{-4}$ and $\beta_{\text{max}} = 2 \times 10^{-2}$, timesteps $T = 1000$, and learning rate $lr = 2 \times 10^{-4}$ for all models and datasets. Each model has 2 residual blocks (He et al., 2016) for each down- and up- sampling layer, while an attention block is placed at 16x resolution. We only modified the channel multipliers for each model based on the complexity of the dataset, see Table (3). If the dataset size is smaller than the specified batch size, we take the batch size to be equal to that small dataset size. For generation or inference, we use the exponential moving average (EMA) of each trained model, as delineated in Ho et al. (2020), which was obtained with the decay value set as 0.9999 during training. We did not use random flipping in the training of our models since we want our measurements to reflect the training data size at best as random flipping implicitly increases the number of patterns that the models see during training. However, we did use dropout (of value 0.1) for the training of CIFAR10, MNIST, and FASHION-MNIST models. Lastly, for each of the training set S_α, where $\alpha = 1, \ldots, M$, they are split from the original dataset given a specific size, using the same random seed value of 3407.

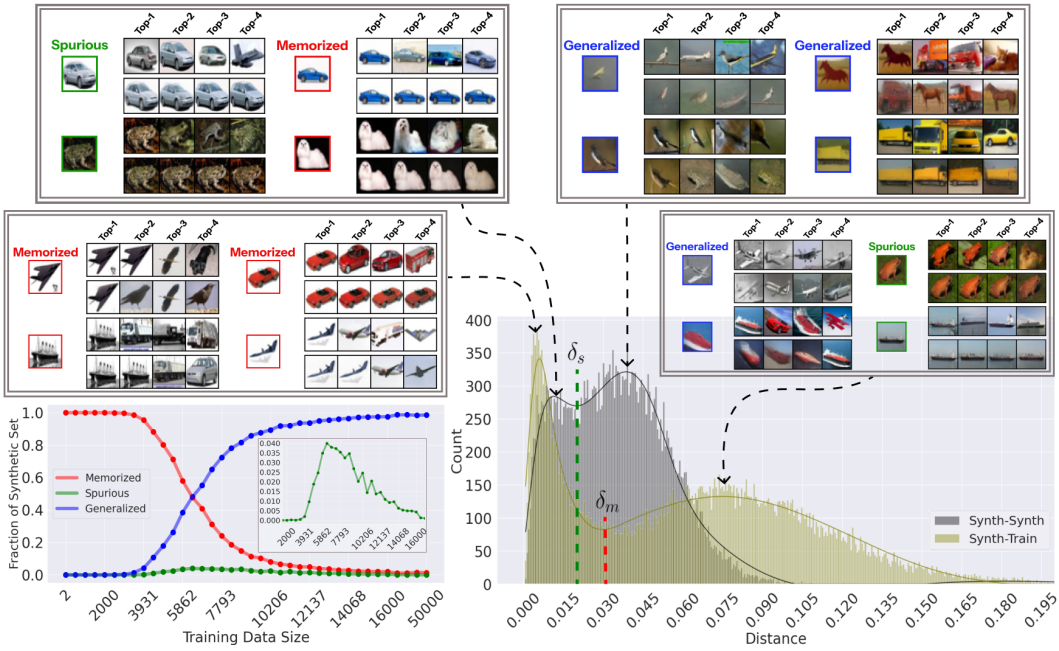

Figure 5: Different sample types across the memorization-to-generalization transition for CIFAR10. The grey histogram shows the distances between synthetic samples and their nearest neighbors from the synthetic set S'. The threshold δ_s is defined as a boundary between the two peaks. The olive histogram depicts the distances from the synthetic samples to their closest neighbor from the training set S, with threshold δ_m separating the two peaks. Memorized samples are located in the left peak of the olive histogram, below δ_m. In contrast, generalized and spurious samples appear to the right of δ_m (olive histogram). Examples of the generated samples forming each of the four peaks of the histograms are shown in the inset frames. For each generated sample top-4 nearest neighbor images from the training set are shown in the ***top row***, and top-4 nearest neighbors from the synthetic set are shown in the ***bottom row***. Training set size $K = 7310$ was used in this figure (peak of the frequency of spurious states), but phenomena discussed are generic and largely independent of this specific value. The fraction of the memorized, spurious, and generalized samples in the pool of all generated samples is shown in the bottom left panel as a function of the training set size. The inset shows amplified spurious fraction (green curve).

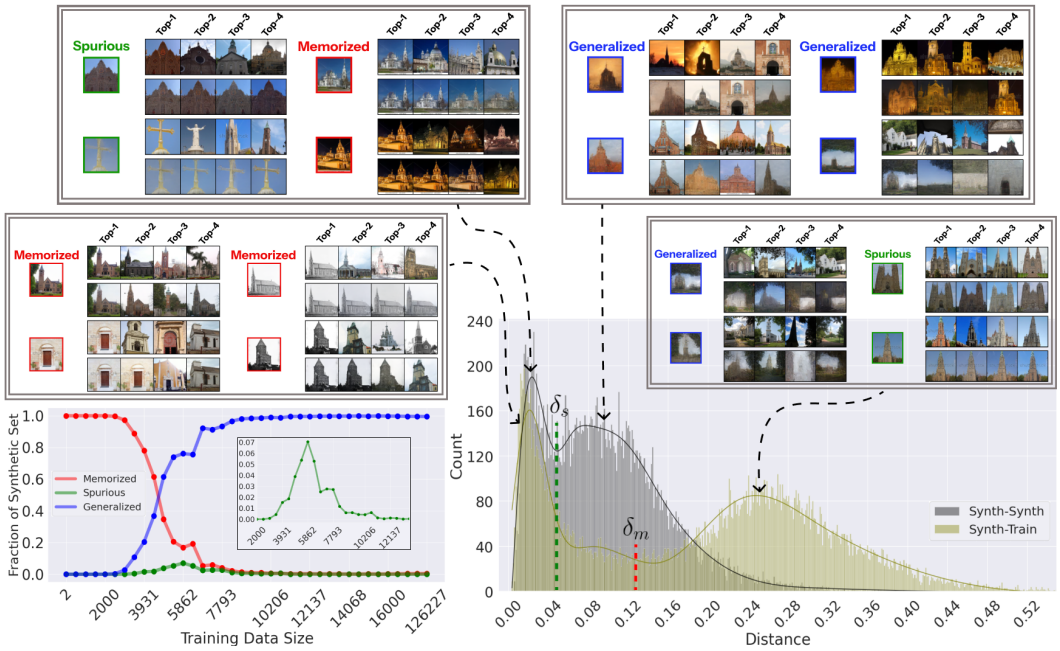

Figure 6: Different sample types across the memorization-to-generalization transition for LSUN-CHURCH, defined by the spurious and memorized thresholds, δ_s and δ_m. The grey histogram shows the distances between synthetic samples and their nearest neighbors from the synthetic set S'. The threshold δ_s is defined as a boundary between the two peaks. The olive histogram depicts the distances from the synthetic samples to their closest neighbor from the training set S, with threshold δ_m separating the two peaks. The threshold δ_m is chosen much stricter here such that it works well for all training dataset sizes via visual inspection. Memorized samples are located in the left peak of the olive histogram, below δ_m. In contrast, generalized and spurious samples appear to the right of δ_m (olive histogram). Examples of the generated samples forming each of the four peaks of the histograms are shown in the inset frames. For each generated sample top-4 nearest neighbor images from the training set are shown in the ***top row***, and top-4 nearest neighbors from the synthetic set are shown in the ***bottom row***. Training set size K = 4896 was used in this figure (at the peak of the frequency of spurious states), but phenomena discussed are generic and largely independent of this specific value. The fraction of the memorized, spurious, and generalized samples in the pool of all generated samples is shown in the bottom left panel as a function of the training set size. The inset shows amplified spurious fraction (green curve).

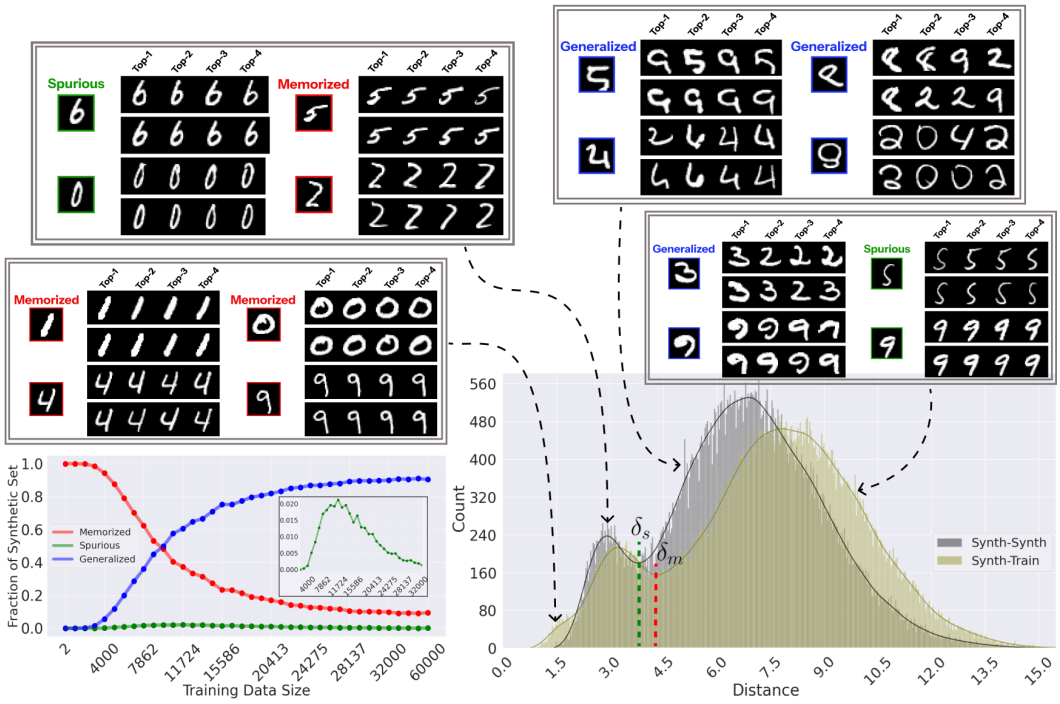

Figure 7: Different sample types across the memorization-to-generalization transition for MNIST, defined by the spurious and memorized thresholds, δ_s and δ_m. The grey histogram shows the distances between synthetic samples and their nearest neighbors from the synthetic set S'. The threshold δ_s is defined as a boundary between the two peaks. The olive histogram depicts the distances from the synthetic samples to their closest neighbor from the training set S, with threshold δ_m separating the two peaks. Memorized samples are located in the left peak of the olive histogram, below δ_m. In contrast, generalized and spurious samples appear to the right of δ_m (olive histogram). Examples of the generated samples forming each of the four peaks of the histograms are shown in the inset frames. For each generated sample top-4 nearest neighbor images from the training set are shown in the ***top row***, and top-4 nearest neighbors from the synthetic set are shown in the ***bottom row***. Training set size K = 21379 was used in this figure, which is slightly past the peak of the frequency of spurious states. The phenomena discussed are generic and largely independent of this specific value. The fraction of the memorized, spurious, and generalized samples in the pool of all generated samples is shown in the bottom left panel as a function of the training set size. The inset shows amplified spurious fraction (green curve).

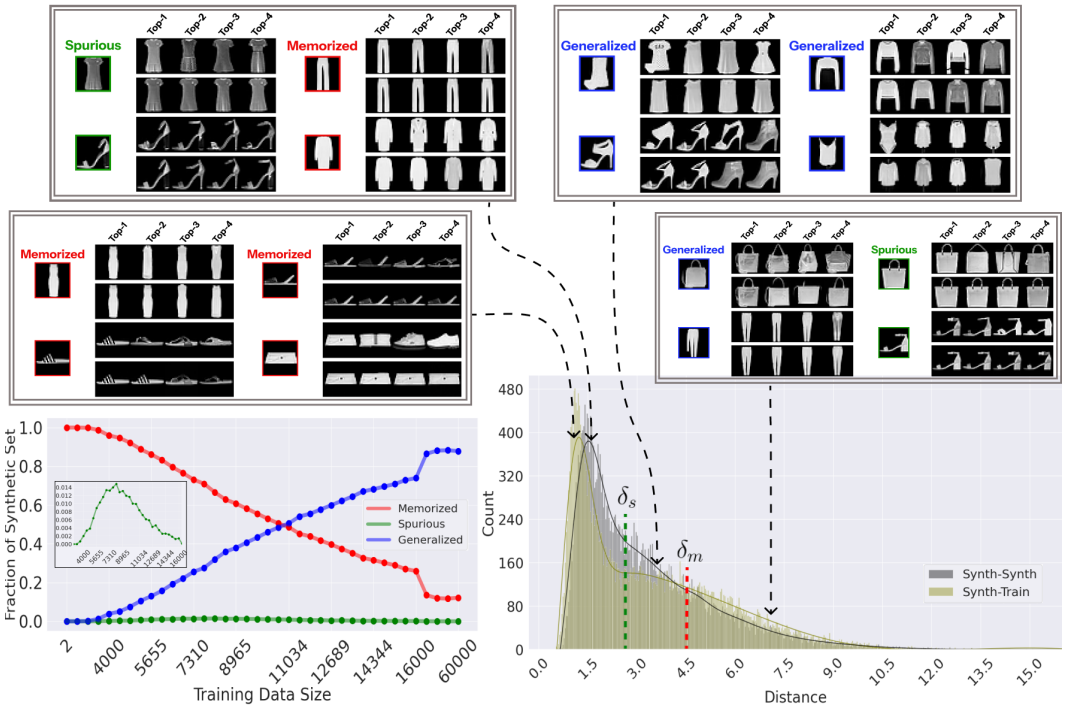

Figure 8: Different sample types across the memorization-to-generalization transition for FASHION-MNIST, defined by the spurious and memorized thresholds, δ_s and δ_m. The grey histogram shows the distances between synthetic samples and their nearest neighbors from the synthetic set S'. The threshold δ_s is defined as a boundary between the two peaks. The olive histogram depicts the distances from the synthetic samples to their closest neighbor from the training set S, with threshold δ_m separating the two peaks. Memorized samples are located in the left peak of the olive histogram, below δ_m. In contrast, generalized and spurious samples appear to the right of δ_m (olive histogram). Examples of the generated samples forming each of the four peaks of the histograms are shown in the inset frames. For each generated sample top-4 nearest neighbor images from the training set are shown in the ***top row***, and top-4 nearest neighbors from the synthetic set are shown in the ***bottom row***. Training set size K = 7724 was used in this figure (at the peak of the frequency of spurious states), but phenomena discussed are generic and largely independent of this specific value. The fraction of the memorized, spurious, and generalized samples in the pool of all generated samples is shown in the bottom left panel as a function of the training set size. The inset shows amplified spurious fraction (green curve).

G ADDITIONAL VISUALIZATIONS OF THE THREE SAMPLE TYPES

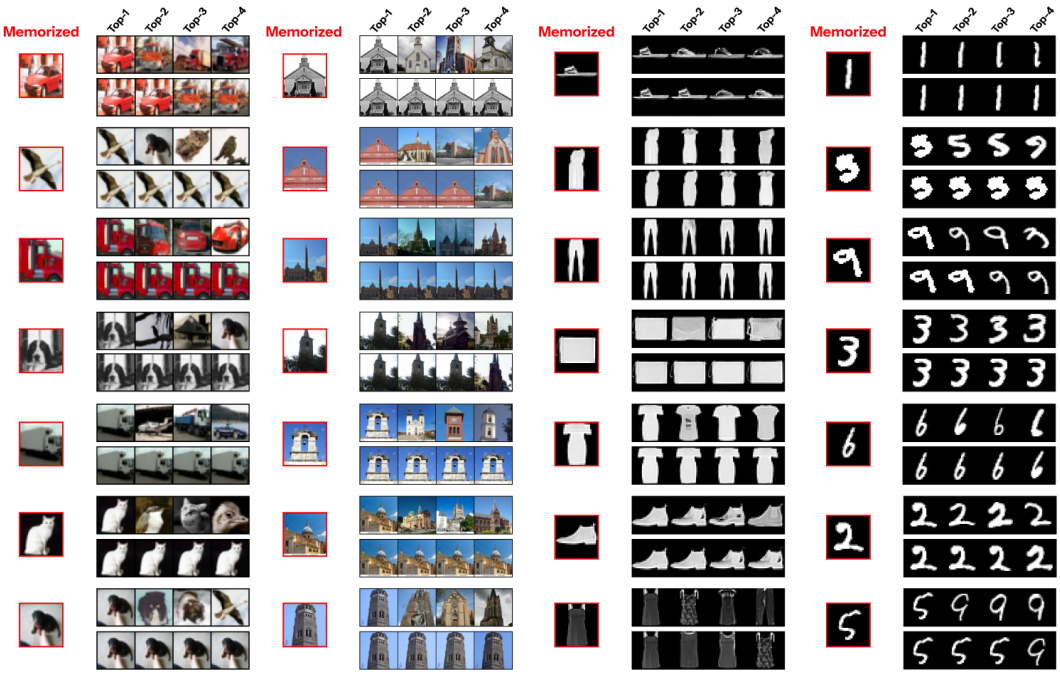

Figure 9: Visualizations of additional memorized patterns and their ***top-4*** nearest neighbors for different datasets. The ***top row*** illustrates nearest neighbors from the training set while the ***bottom row*** depicts those from the synthetic set. Memorized samples are duplicates of the training set. During the strong memorization phase, duplicates are also found within the synthetic set. Note, even though our memorized detection metric (Eq. (31)) does not utilize the synthetic set, we are showing the nearest neighbors obtained from it, for consistency.

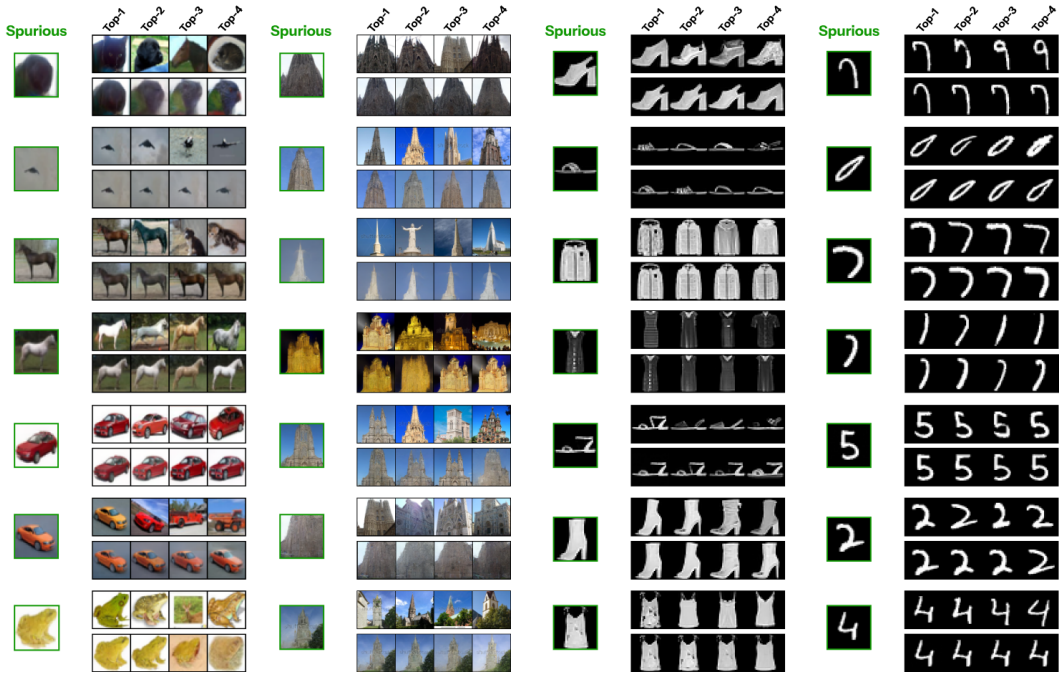

Figure 10: Visualizations of additional spurious patterns and their ***top-4*** nearest neighbors for different datasets. The ***top row*** depicts nearest neighbors from the training set, while the ***bottom row*** shows those from the synthetic set. Spurious patterns are demonstrated to arise from the onset of generalization where the mixing of training data points begins. Since the model's generalization is at its infancy, duplicates of spurious patterns sometimes appear several times in the synthetic set S', much like memorized patterns. Additionally, these samples lack the uniqueness to be considered as generalized samples as the model has yet to fully learn the underlying data distribution.

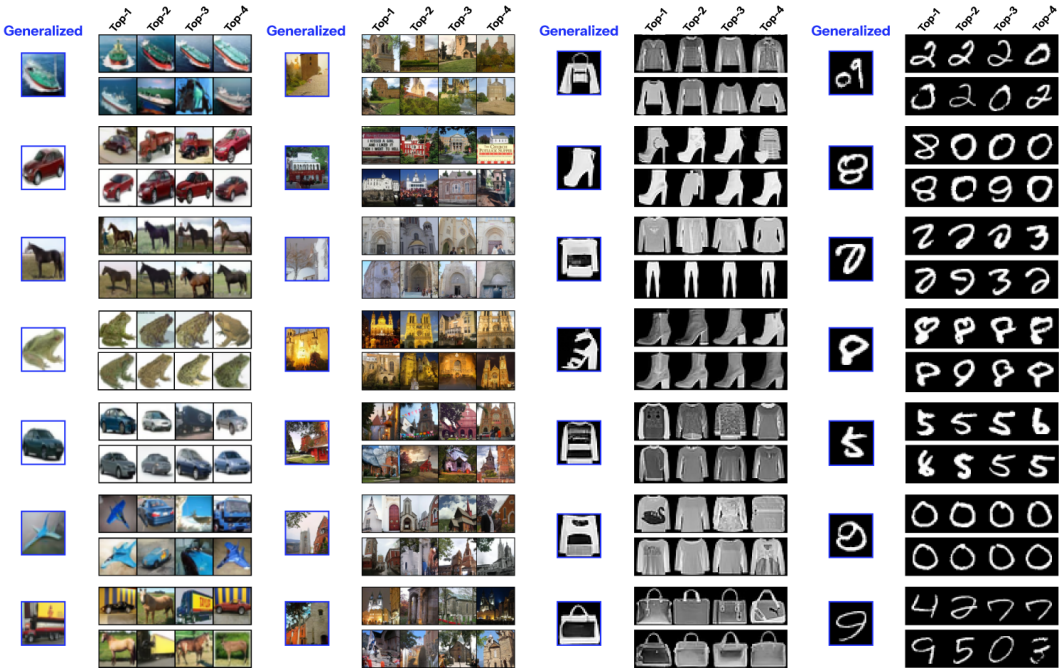

Figure 11: Visualization of generalized patterns and their ***top-4*** nearest neighbors for different datasets. The ***top row*** illustrates nearest neighbors from the training set while the ***bottom row*** depicts those from the synthetic set. Generalized samples are novel samples, which have little to no resemblance to their nearest neighbors in training and synthetic sets.