Sparse Autoencoder Neural Operators: Model Recovery in Function Spaces

Bahareh Tolooshams[1,2,3*] **Ailsa Shen**[4*] **Anima Anandkumar**[4]
[1]University of Alberta [2]Alberta Machine Intelligence Institute (Amii)
[3]Neuroscience and Mental Health Institute (NMHI), University of Alberta
[4] California Institute of Technology (Caltech)

Abstract

We frame the problem of unifying representations in neural models as one of sparse model recovery and introduce a framework that extends sparse autoencoders (SAEs) to lifted spaces and infinite-dimensional function spaces, enabling mechanistic interpretability of large neural operators (NO). While the Platonic Representation Hypothesis suggests that neural networks converge to similar representations across architectures, the representational properties of neural operators remain underexplored despite their growing importance in scientific computing. We compare the inference and training dynamics of SAEs, lifted-SAE, and SAE neural operators. We highlight how lifting and operator modules introduce beneficial inductive biases, enabling faster recovery, improved recovery of smooth concepts, and robust inference across varying resolutions, a property unique to neural operators.

1 Introduction

A growing body of work reports that neural networks, despite architectural differences, converge to unifying representations across contexts such as brain–model comparisons [1, 2], alignment with large language models [3], brain–computer interfaces [4], and universality in dynamics [5]. To study this, methods for representation similarity have been developed [6–8], building on representational similarity analysis [9]. The Platonic Representation Hypothesis (PRH) [10] formalizes why and where such convergence arises, framing it as a unifying principle across architectures. Recent work suggests it may arise from implicit regularization in stochastic gradient descent, which favours solutions with minimal gradient norms [11]. These advances raise a central question: under what conditions do different architectures learn equivalent representations, and how can this be formalized?

If convergence is truly universal, it should also extend beyond Euclidean spaces to neural operators (NO) [12–14], which learn mappings between infinite-dimensional function spaces. Neural operators are now central to data-driven scientific modelling [15] and have recently been applied in neuroscience to capture brain representations [16, 17]. While their predictive power is well established across applications [18–20], their representational properties and interpretability remain largely unknown, leaving open whether universal representation principles extend naturally to function spaces.

To study the PRH in this broader setting, we adopt the perspective of concept learning [21], focusing on sparse model recovery [22, 23]. Rather than asking whether two models learn the same representations, we ask whether they recover the same underlying concepts, an approach aligned with mechanistic interpretability, where SAEs extract interpretable features from large networks [24–34]. By framing representation learning as estimating latent sparse codes and dictionaries, we unify comparisons across architectures, training dynamics, and inductive biases. This leads us to three central questions: i) Can recovery generalize from Euclidean to functional spaces? ii) Under what

*Equal contribution.

Preprint.

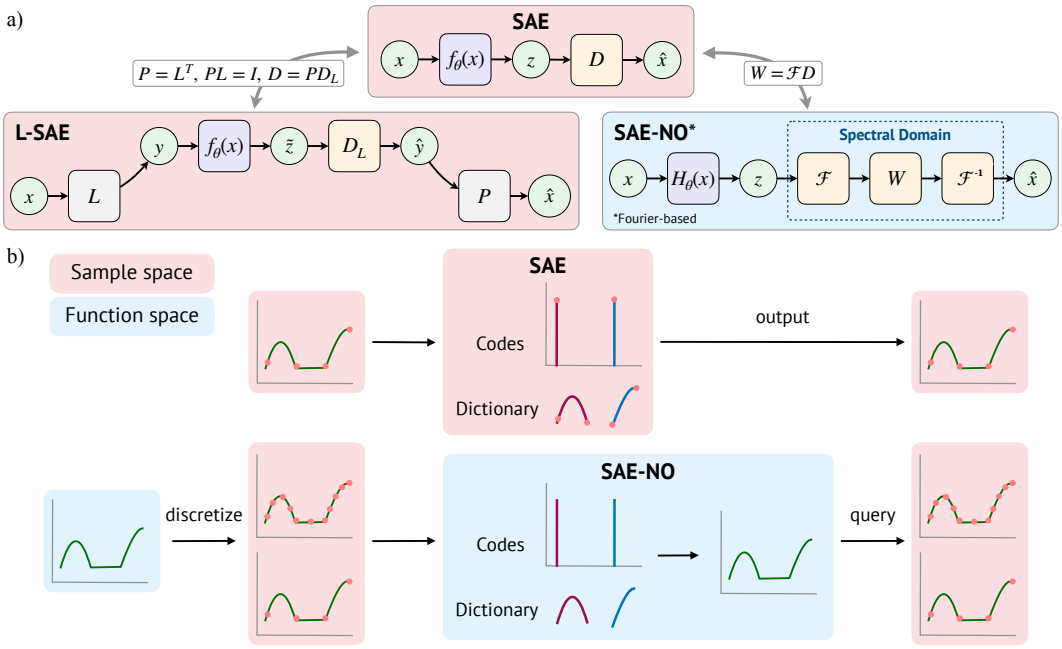


Fig. 1: **Model Recovery with SAEs.** a) Architectural comparison of SAE, lifted SAE, and SAE Neural Operators. b) Learning in sampled Euclidean spaces vs. function spaces.

conditions do networks and operators recover equivalent representations, and when do operators offer advantages? iii) How does lifting affect recovery dynamics?

Our Contributions We address these questions by extending SAEs to lifted SAEs (L-SAEs), which learn concepts in a lifted space, and SAE neural operators (SAE-NOs), which formalize model recovery in function spaces. We show that lifting enables recovery while acting as a preconditioner that accelerates learning, and identify conditions where the inference and dynamics of L-SAEs reduce to those of standard SAEs. We further introduce SAE-NOs, demonstrating that i) lifting-induced preconditioning extends to L-SAE-NOs, ii) truncated modes confer an inductive bias that is advantageous for recovering smooth concepts, iii) SAE-NOs recover representations robustly under resolution changes where SAEs fail, and iv) with full-frequency filters and matched spatial support, their dynamics reduce to those of SAE-CNN.

Preliminaries The sparse linear generative model (Def. 1.1), also known as sparse coding [22, 23], has a rich history in inverse problems [35], deeply studied in statistics [36] and in compressed sensing [37–40]. Representing data $\boldsymbol{x}$ under a *sparse generative model* involves learning a dictionary $\boldsymbol{D}^*$ to infer a sparse code $\boldsymbol{z}$. We focus on the case where $\boldsymbol{D}^*$ is unknown (a.k.a. *dictionary learning*) [41–43] and refer to it as *sparse model recovery*, formulated as a bilevel problem (Def. 1.2) [44].

Definition 1.1 (Sparse Generative Models)**.** *A data sample $\boldsymbol{x} \in \mathbb{R}^m$ is said to follow a sparse generative model if there exists a sparse latent representation $\boldsymbol{z} \in \mathbb{R}^p$ (with supp$(\boldsymbol{z}) \leq k \ll p$) and an overcomplete dictionary $\boldsymbol{D}^* \in \mathbb{R}^{m \times p}$ (with $p \gg m$) that model the data as $\boldsymbol{x} = \boldsymbol{D}^*\boldsymbol{z}$ in the noiseless setting (see Def. B.1 for sparse convolutional generative models).*

Definition 1.2 (Sparse Model Recovery)**.** *Under the sparse generative model (Def. 1.1), the goal of sparse model recovery is to estimate the underlying dictionary $\boldsymbol{D}^*$ by solving the following problem:*

$$\textit{(outer)} \min_{\boldsymbol{D} \in \mathcal{D}} \ \tfrac{1}{2} \|\boldsymbol{x} - \boldsymbol{D}\boldsymbol{z}\|_2^2 + \lambda \mathcal{R}(\boldsymbol{z}) \quad \textit{s.t.} \quad \textit{(inner)}\ \boldsymbol{z} = f_\theta(\boldsymbol{x}) = \arg\min_{\boldsymbol{v} \in \mathbb{R}^p} \mathcal{L}(\boldsymbol{x}, \boldsymbol{v}, \theta), \tag{1}$$

where f_θ is a forward map from data $\boldsymbol{x}$ to latent code $\boldsymbol{z}$, defined as the minimizer of the inner loss $\mathcal{L}(\boldsymbol{x}, \boldsymbol{v}, \theta)$. The outer-level optimizes $\boldsymbol{D}$, and model recovery is successful *when the learned concept $\boldsymbol{D}$ approximates the true underlying dictionary $\boldsymbol{D}^*$.*

The connection between bilevel optimization and neural networks is well established through SAEs [45–48]. The inner and outer levels in Def. 1.2 mirror the encoder–decoder structure of a neural network [44], where the encoder f_θ maps data $\boldsymbol{x}$ to sparse codes $\boldsymbol{z}$, and the decoder reconstructs $\boldsymbol{x}$ via $\boldsymbol{D}\boldsymbol{z}$ using learned weights $\boldsymbol{D}$ (see Def. 1.3).

Definition 1.3 (Sparse Autoencoders for Model Recovery)**.** *The goal of model recovery with SAEs is to infer a sparse code $\boldsymbol{z} \in \mathbb{R}^p$, representing the data $\boldsymbol{x} \in \mathbb{R}^m$, with a linear combination of a dictionary $\boldsymbol{D}$: (encoder) $\boldsymbol{z} = f_\theta(\boldsymbol{x})$, (decoder) $\hat{\boldsymbol{x}} = \boldsymbol{D}\boldsymbol{z}$, where θ denotes the set of encoder parameters. Model recovery refers to learning the dictionary $\boldsymbol{D}$ (see Def. B.1 for SAE-CNN).*

Sparse model recovery has attracted interest from two communities: a) *unrolled learning*, using optimization-inspired architectures for network design [46–54]; and b) *sparse interpretability* or mechanistic interpretability, employing SAEs to extract concepts from large models [24–34]. To extend SAEs to function spaces, we first cover basics of neural operators [12–14]. Unlike neural networks, which operate on finite-dimensional Euclidean spaces, neural operators learn mappings between infinite-dimensional function spaces over bounded domains. They have gained traction in data-driven scientific modelling [15], particularly for solving partial differential equations (PDEs), and can generalize across discretizations. Neural operators consist of the following three modules of lifting, kernel integration, and projection (see Appendix and [14]).

Fourier Neural Operator (FNO) [13] models the kernel integral operator with a convolution operator parameterized in Fourier space (Def. 1.4, see also Def. B.3 for Fourier transform). The convolution operator parameterizes the kernel $\kappa(x, y) = \kappa(x - y)$ as a complex function $\kappa : D \to \mathbb{C}^{d_w \times d_v}$ (Def. B.2). FNO is used with truncated frequency modes in practice; this has been shown to improve performance and lower sensitivity to change (decreasing) the discretization sampling [13, 14].

Definition 1.4 (Fourier integral operator $\mathcal{K}$ (restated from [13, 14]))**.** *Define the Fourier integral operator by $(\mathcal{K}(\phi)\boldsymbol{v})(\boldsymbol{x}) = \mathcal{F}^{-1}\big(R_\phi \cdot (\mathcal{F}\boldsymbol{v})\big)(\boldsymbol{x})$, $\forall \boldsymbol{x} \in D$, where $R_\phi := \mathcal{F}(\kappa)$ is the Fourier transform of a periodic function $\kappa : D \to \mathbb{C}^{d_w \times d_v}$ parameterized by ϕ.*

To extend SAEs to function spaces, we leverage the link between convolution in Euclidean space and multiplication in the Fourier domain. Building on [55] showing SAE-induced inductive biases, we investigate when networks and operators recover similar or distinct concepts. Section 2 formalizes sparse model recovery with neural operators [12–14] for concept learning in function spaces.

2 SAE Neural Operators for Model Recovery in Function Spaces

We extend SAEs, which are typically implemented with MLP layers (SAE-MLPs), to neural operators to enable concept recovery in function spaces. Based on operator learning, we formulate a sparse functional generative model of data (Def. 2.1), and its corresponding optimization for model recovery on function spaces (Def. 2.2).

Definition 2.1 (Sparse Functional Generative Model)**.** *Consider a linear operator $\mathcal{G}_{\boldsymbol{D}^*} : \mathcal{Z} \to \mathcal{X}$ in function spaces, where $\mathcal{Z}$ and $\mathcal{X}$ are Banach spaces of functions defined on bounded domains $D_z \subset \mathbb{R}^p$ and $D_x \subset \mathbb{R}^m$, respectively. The data sample $\boldsymbol{x} \in \mathcal{X}$ is said to follow a sparse functional generative model if there exists a sparse latent representation $\boldsymbol{z}$ drawn from the support $\mathcal{Z}$ of a probability measure μ, where $\boldsymbol{x} = \mathcal{G}_{\boldsymbol{D}^*}(\boldsymbol{z})$.*

Definition 2.2 (Sparse Functional Model Recovery)**.** *Under the sparse functional generative model (Def. 2.1), the goal of sparse model recovery in function spaces is to estimate the underlying operator $\mathcal{G}_{\boldsymbol{D}^*}$ by solving the following bilevel optimization problem:*

$$\min_{\boldsymbol{D} \in \mathcal{D}} \quad \mathbb{E}_{\boldsymbol{z}\sim\mu} \|\mathcal{G}_{\boldsymbol{D}^*}(\boldsymbol{z}) - \mathcal{G}_{\boldsymbol{D}}(\hat{\boldsymbol{z}})\|_{\mathcal{X}}^2 \quad \textit{s.t.} \quad \hat{\boldsymbol{z}} = \arg\min_{\boldsymbol{v}} \mathbb{E}_{\boldsymbol{z}\sim\mu} \|\boldsymbol{v} - \mathcal{H}_\theta(\mathcal{G}_{\boldsymbol{D}^*}(\boldsymbol{z}))\|_{\mathcal{Z}}^2, \tag{2}$$

where the operator $\mathcal{H}_\theta : \mathcal{X} \to \mathcal{Z}$ aims to approximate the inverse of the operator $\mathcal{G}_{\boldsymbol{D}^*}$. Model recovery is successful when the learned concepts $\boldsymbol{D}$ in $\mathcal{G}_{\boldsymbol{D}}$ closely approximate $\boldsymbol{D}^*$ in $\mathcal{G}_{\boldsymbol{D}^*}$.

To recover concepts, we formalize SAE-NO (Def. 2.3) with an encoder approximating the inverse generative operator and a decoder that captures the underlying operator and its learnable concepts.

Definition 2.3 (Sparse Autoencoder Neural Operators (SAE-NOs) for Model Recovery)**.** *The goal of model recovery with SAE-NO in function spaces is to infer a sparse code $\boldsymbol{z}$ as a sample from the function space $\mathcal{Z}$, representing the data sample from the function space $\mathcal{X}$ via the operator $\mathcal{G}_{\boldsymbol{D}}$: (encoder) $\boldsymbol{z} = \mathcal{H}_\theta(\boldsymbol{x})$, (decoder) $\hat{\boldsymbol{x}} = \mathcal{G}_{\boldsymbol{D}}(\boldsymbol{z})$, where $\mathcal{H}_\theta$ is the encoder operator, and model recovery involves estimating the concepts $\boldsymbol{D}$.*

We consider Fourier parametrization and define SAE Fourier Neural Operators (*SAE-FNOs*) as $\mathcal{G}_{\boldsymbol{D}}(\boldsymbol{z}) := \mathcal{F}^{-1}\big(\sum_{c=1}^{C} \boldsymbol{W}_c \cdot (\mathcal{F}\boldsymbol{z}_c)\big)$, where $\boldsymbol{W}_c = \mathcal{F}\boldsymbol{D}_c$ is the Fourier transform of the convolution kernel. This reduces sparse functional model recovery to recovery of a sparse convolutional generative

model (Def. B.1), implemented as a sum of convolutions between $\boldsymbol{z}$ and $\boldsymbol{D}$. This formulation generalizes SAEs to operator and Fourier domains, enabling concept learning in function spaces. We further extend this approach to model recovery in lifted spaces (Def. 2.4), incorporating *lifting* and *projection* modules that are crucial components of neural operators [12–14].

Definition 2.4 (Lifted SAE-NO for Model Recovery)**.** *The goal of model recovery with lifted SAE-NO in function spaces is to infer a sparse code $\boldsymbol{z}$ as a sample from the function space $\mathcal{Z}$, representing the data from $\mathcal{X}$ via the operator $\mathcal{G}_{\boldsymbol{D_L}}$:*

$$\begin{aligned} \textit{(lifting and encoder)} \quad & \boldsymbol{z} = \mathcal{H}_\theta(\boldsymbol{y}), \quad \boldsymbol{y} = \boldsymbol{L}\boldsymbol{x} \\ \textit{(decoder and projecting)} \quad & \hat{\boldsymbol{x}} = \boldsymbol{P}\hat{\boldsymbol{y}}, \quad \hat{\boldsymbol{y}} = \mathcal{G}_{\boldsymbol{D_L}}(\boldsymbol{z}), \end{aligned} \tag{3}$$

where $\mathcal{H}_\theta$ encodes the lifted data y, and model recovery is done via the learned lifted concepts $\boldsymbol{D_L}$.

Finally, we informally define *architectural inference equivalence*, i.e., when two architectures can encapsulate the same underlying model and produce the same latent representations of an input.

3 Results

We generate data under the sparse convolutional generative model, and study the impact of lifting and operator on model recovery on SAE-CNN, L-SAE-CNN, SAE-FNO, and L-SAE-FNO (see Appendix). The concepts in SAE-CNN (or L-SAE-CNN), SAE-FNO (or L-SAE-FNO) are parameterized as a convolution and a Fourier-based operator, respectively.

Lifting We examined model recovery in the lifted space and found: i) models can be recovered in the lifted space (Fig. 4); ii) lifting can act as a preconditioner (Prop. 3.1), accelerating recovery (Fig. 4) by reducing dictionary atom correlation, promoting a more isotropic loss landscape (Fig. 6); and iii) when lifting and projection are tied and orthogonal, the architectural inference (Props. 3.2 and D.1) and training dynamics (Props. 3.1 and D.2) of lifted SAEs reduces to SAEs (Fig. 5).

Proposition 3.1 (Training Dynamics of Lifting)**.** *The training dynamics of the lifted-SAE (L-SAE) $\boldsymbol{D}_L^{(k+1)} = \boldsymbol{D}_L^{(k)} + \eta_L \boldsymbol{P}^\top(\boldsymbol{x} - \boldsymbol{P}\boldsymbol{D}_L^{(k)}\boldsymbol{z})\boldsymbol{z}^\top$, with lifting $\boldsymbol{L}$ and projection $\boldsymbol{P}$, has the effective update in the original space, expressed as: $\boldsymbol{D}^{(k+1)} = \boldsymbol{D}^{(k)} + \eta_L(\boldsymbol{L}^\top\boldsymbol{L})(\boldsymbol{x} - \boldsymbol{D}^{(k)}\boldsymbol{z})\boldsymbol{z}^\top$, where $\boldsymbol{L}^\top\boldsymbol{L}$ acts as a preconditioner, potentially accelerating learning by inducing a more isotropic update. If they satisfy the equivalent architectural inference (Prop. 3.2), then the dynamics are equivalent to those of an SAE: $\boldsymbol{D}^{(k+1)} = \boldsymbol{D}^{(k)} + \eta(\boldsymbol{x} - \boldsymbol{D}^{(k)}\boldsymbol{z})\boldsymbol{z}^\top$ (see Prop. D.2 for L-SAE-CNN).*

Proposition 3.2 (Architectural Inference Equivalence of Lifting in SAE)**.** *The architectural inference of a SAE is equivalent to a lifted-SAE if the projection is tied to the lifting operator ($\boldsymbol{P} = \boldsymbol{L}^\top$), and the lifting operator is orthogonal ($\boldsymbol{L}^\top\boldsymbol{L} = \boldsymbol{I}$), i.e., under the same learned model $\boldsymbol{D} = \boldsymbol{P}\boldsymbol{D}_L$, the representation dynamics and output of the networks are equivalent (see Prop. D.1 for SAE-CNN and Prop. D.6 for SAE-FNO).*

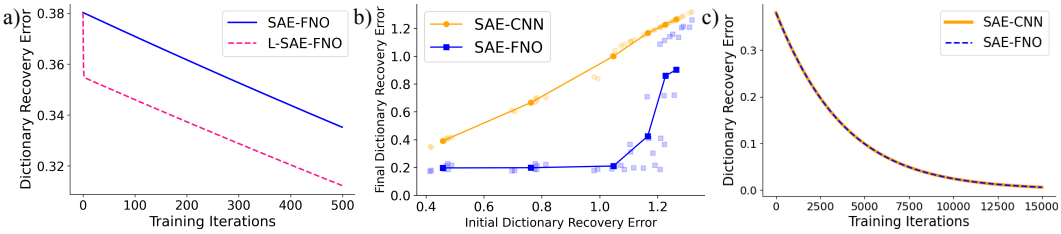
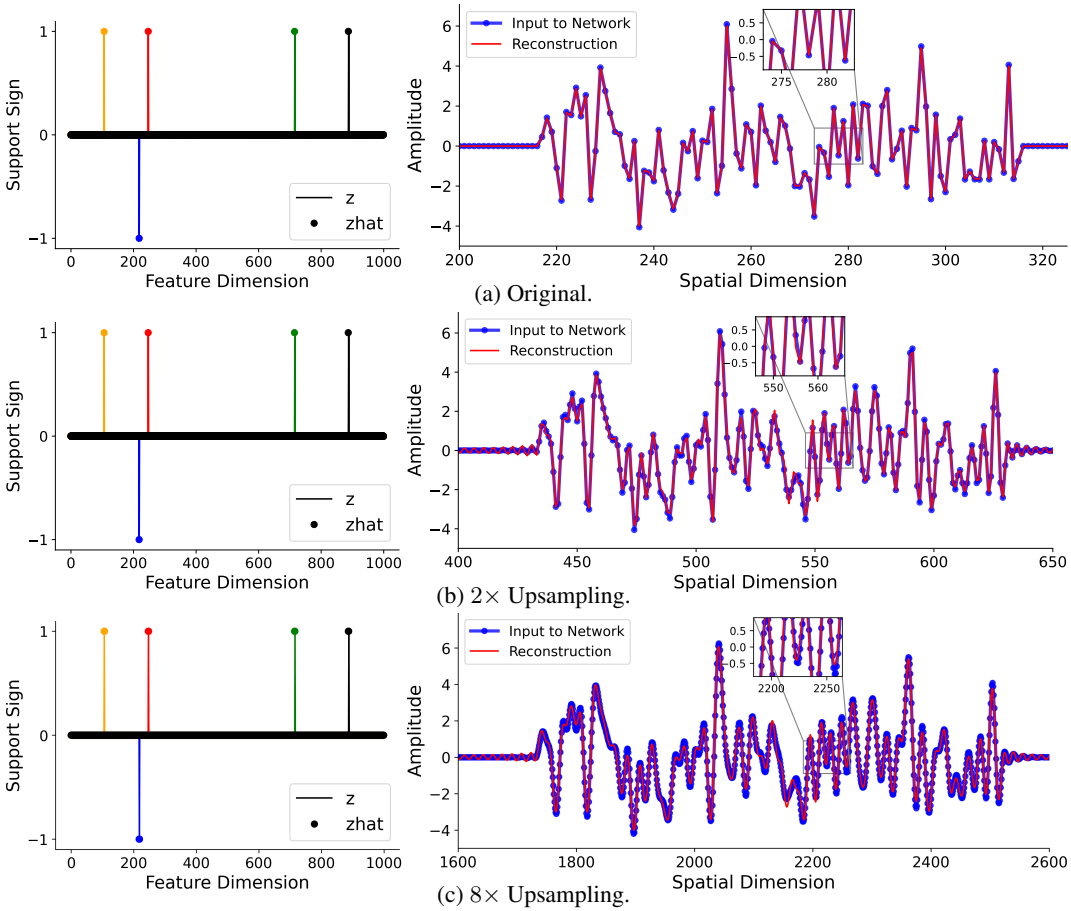


Fig. 2: **SAE-CNN vs. SAE-FNO.** a) Lifting accelerates learning. b) SAE-FNO's superiority in recovering smooth concepts via truncated Fourier modes. c) Equivalent learning when SAE-FNO uses all Fourier modes and matched spatial receptive field of SAE-CNNs.

SAE-FNO We examined model recovery in function spaces. Our results show that: i) the lifting-induced preconditioning effect extends to L-SAE-FNO (Fig. 2a, Prop. D.3); ii) SAE-FNO with truncated modes exhibits an inductive bias that favours recovery of smooth concepts, which is desirable when the underlying data concepts are smooth (Fig. 2b); iii) SAE-FNOs can successfully infer representations from higher-resolution data, whereas SAEs fail to recover the true underlying representation when data resolution changes (Fig. 3); and iv) when the integral operator is implemented as a convolution with weights spanning the full frequency modes and constrained to the same spatial support, the architectural inference and training dynamics of SAE-FNO (Prop. 3.3) reduce to those of SAE-CNN, with weights related by a Fourier transform (Prop. 3.4, Fig. 2c).

Proposition 3.3 (Training Dynamics of SAE-FNO)**.** *Consider an SAE-FNO whose integral operator is implemented as a convolution, with frequency-domain weights $\boldsymbol{W}_c$ defined as the Fourier transform of the SAE-CNN's spatial kernels, i.e., $\boldsymbol{W}_c = \mathcal{F}\boldsymbol{D}_c$. The training dynamics of the SAE-FNO are then expressed as: $\boldsymbol{D}_c^{(k+1)} = \boldsymbol{D}_c^{(k)} + \frac{\eta}{M}(\boldsymbol{x} - \frac{1}{\sqrt{M}}\mathcal{F}^{-1}[\sum_{i=1}^{C}\mathcal{F}\boldsymbol{D}_i^{(k)} \odot \mathcal{F}\boldsymbol{z}_{i,t}]) \star \boldsymbol{z}_{c,t}$ where M is the full number of modes, $\odot$ denotes element-wise multiplication, and $\star$ denotes correlation. If the SAE-FNO's filters span the full frequency range and are constrained to have the same spatial support as in the SAE-CNN, then its dynamics reduce to those of the SAE-CNN (see Prop. 3.4).*

Proposition 3.4 (Training Dynamics of SAE-CNN and SAE-FNO)**.** *The training dynamics of the SAE-FNO's frequency-domain weights $\boldsymbol{W}_c^{(k+1)} = \boldsymbol{W}_c^{(k)} + \frac{\eta}{\sqrt{M}}\mathcal{F}\left(\boldsymbol{x} - \hat{\boldsymbol{x}}^{(k)}\right) \odot \mathcal{F}(\boldsymbol{z}_{c,t})$ has an effective update in the original space, expressed as: $\boldsymbol{D}_c'^{(k+1)} = \boldsymbol{D}_c'^{(k)} + \frac{\eta}{M}\left(\boldsymbol{x} - \hat{\boldsymbol{x}}^{(k)}\right) \star \boldsymbol{z}_{c,t}$. If the architectures satisfy the inference equivalence condition (Prop. D.4, see Prop. D.5 for lifting), then the dynamics are equivalent to that of a SAE-CNN: $\boldsymbol{D}_c^{(k+1)} = \boldsymbol{D}_c^{(k)} + \eta(\boldsymbol{x} - \sum_{i=1}^{C}\boldsymbol{D}_i^{(k)} * \boldsymbol{z}_{i,t}) \star \boldsymbol{z}_{c,t}$.*



(a) Original.

(b) 2× Upsampling.

(c) 8× Upsampling.

Fig. 3: **SAE-FNO Upsampling Robustness Across Resolutions.** SAE-FNO successfully infers the underlying sparse representations and reconstructs data across multiple discretization levels. The left panels show inference of 1-sparse code supports across 5 kernels, and the right panels display spatial-domain signal reconstruction (see also Fig. 7).

Conclusion We introduced a unified framework for sparse model recovery to analyze representations across finite- and infinite-dimensional settings. Our analysis shows that the SAE-FNO's inductive biases enable superior recovery of smooth concepts and generalization across resolutions where standard models fail. This provides a rigorous foundation for both model recovery in function spaces and mechanistic interpretability of large neural operators. While validated here on synthetic data, critical next steps include applying this framework to real-world settings, where based on the prior success of neural operators, we speculate that SAE-FNO will be highly effective in extracting meaningful concepts, including learning harmonic representations useful in scientific modelling and capturing global concepts or frequency-based coding in neuroscience.

References

[1] M. Schrimpf, J. Kubilius, H. Hong, N. J. Majaj, R. Rajalingham, E. B. Issa, K. Kar, P. Bashivan, J. Prescott-Roy, F. Geiger, *et al.*, "Brain-score: Which artificial neural network for object recognition is most brain-like?," *BioRxiv*, p. 407007, 2018.

[2] B. Lin and N. Kriegeskorte, "The topology and geometry of neural representations," *Proceedings of the National Academy of Sciences*, vol. 121, no. 42, p. e2317881121, 2024.

[3] A. Doerig, T. C. Kietzmann, E. Allen, Y. Wu, T. Naselaris, K. Kay, and I. Charest, "High-level visual representations in the human brain are aligned with large language models," *Nature Machine Intelligence*, pp. 1–15, 2025.

[4] A. D. Degenhart, W. E. Bishop, E. R. Oby, E. C. Tyler-Kabara, S. M. Chase, A. P. Batista, and B. M. Yu, "Stabilization of a brain–computer interface via the alignment of low-dimensional spaces of neural activity," *Nature biomedical engineering*, vol. 4, no. 7, pp. 672–685, 2020.

[5] N. Maheswaranathan, A. H. Williams, M. D. Golub, S. Ganguli, and D. Sussillo, "Universality and individuality in neural dynamics across large populations of recurrent networks," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[6] A. H. Williams, "Equivalence between representational similarity analysis, centered kernel alignment, and canonical correlations analysis," in *Proceedings of UniReps: the Second Edition of the Workshop on Unifying Representations in Neural Models*, pp. 10–23, PMLR, 2024.

[7] M. Khosla and A. H. Williams, "Soft matching distance: A metric on neural representations that captures single-neuron tuning," in *Proceedings of UniReps: the First Workshop on Unifying Representations in Neural Models*, pp. 326–341, PMLR, 2024.

[8] S. Barannikov, I. Trofimov, N. Balabin, and E. Burnaev, "Representation topology divergence: A method for comparing neural network representations.," in *International Conference on Machine Learning*, pp. 1607–1626, PMLR, 2022.

[9] N. Kriegeskorte, M. Mur, and P. A. Bandettini, "Representational similarity analysis–connecting the branches of systems neuroscience," *Frontiers in Systems Neuroscience*, vol. 2, p. 4, 2008.

[10] M. Huh, B. Cheung, T. Wang, and P. Isola, "Position: The platonic representation hypothesis," in *Forty-first International Conference on Machine Learning*, 2024.

[11] L. Ziyin and I. Chuang, "Proof of a perfect platonic representation hypothesis," *arXiv preprint arXiv:2507.01098*, 2025.

[12] Z. Li, N. Kovachki, K. Azizzadenesheli, B. Liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, "Neural operator: Graph kernel network for partial differential equations," *arXiv preprint arXiv:2003.03485*, 2020.

[13] Z. Li, N. B. Kovachki, K. Azizzadenesheli, B. liu, K. Bhattacharya, A. Stuart, and A. Anandkumar, "Fourier neural operator for parametric partial differential equations," in *International Conference on Learning Representations*, 2021.

[14] N. Kovachki, Z. Li, B. Liu, K. Azizzadenesheli, K. Bhattacharya, A. Stuart, and A. Anandkumar, "Neural operator: Learning maps between function spaces with applications to pdes," *Journal of Machine Learning Research*, vol. 24, no. 89, pp. 1–97, 2023.

[15] K. Azizzadenesheli, N. Kovachki, Z. Li, M. Liu-Schiaffini, J. Kossaifi, and A. Anandkumar, "Neural operators for accelerating scientific simulations and design," *Nature Reviews Physics*, pp. 1–9, 2024.

[16] B. Tolooshams, L. Lydia, T. Callier, J. Wang, S. Pal, A. Chandrashekar, C. Rabut, Z. Li, C. Blagden, S. L. Norman, K. Azizzadenesheli, C. Liu, M. G. Shapiro, R. A. Andersen, and A. Anandkumar, "Vars-fusi: Variable sampling for fast and efficient functional ultrasound imaging using neural operators," *bioRxiv*, pp. 2025–04, 2025.

[17] L. Ghafourpour, V. Duruisseaux*, B. Tolooshams*, P. H. Wong, C. A. Anastassiou, and A. Anandkumar, "Noble–neural operator with biologically-informed latent embeddings to capture experimental variability in biological neuron models," *arXiv:2506.04536*, 2025.

[18] J. Pathak, S. Subramanian, P. Harrington, S. Raja, A. Chattopadhyay, M. Mardani, T. Kurth, D. Hall, Z. Li, K. Azizzadenesheli, *et al.*, "Fourcastnet: A global data-driven high-resolution weather model using adaptive fourier neural operators," *arXiv preprint arXiv:2202.11214*, 2022.

[19] Z. Li, N. Kovachki, C. Choy, B. Li, J. Kossaifi, S. Otta, M. A. Nabian, M. Stadler, C. Hundt, K. Azizzadenesheli, *et al.*, "Geometry-informed neural operator for large-scale 3d pdes," *Advances in Neural Information Processing Systems*, vol. 36, 2024.

[20] A. S. Jatyani, J. Wang, Z. Wu, M. Liu-Schiaffini, B. Tolooshams, and A. Anandkumar, "Unifying subsampling pattern variations for compressed sensing mri with neural operators," *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

[21] B. Kim, M. Wattenberg, J. Gilmer, C. Cai, J. Wexler, F. Viegas, *et al.*, "Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav)," in *International conference on machine learning*, pp. 2668–2677, PMLR, 2018.

[22] B. A. Olshausen and D. J. Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," *Nature*, vol. 381, no. 6583, pp. 607–609, 1996.

[23] B. A. Olshausen and D. J. Field, "Sparse coding with an overcomplete basis set: A strategy employed by v1?," *Vision research*, vol. 37, no. 23, pp. 3311–3325, 1997.

[24] N. Elhage, T. Hume, C. Olsson, N. Schiefer, T. Henighan, S. Kravec, Z. Hatfield-Dodds, R. Lasenby, D. Drain, C. Chen, *et al.*, "Toy models of superposition," *arXiv:2209.10652*, 2022.

[25] R. Huben, H. Cunningham, L. R. Smith, A. Ewart, and L. Sharkey, "Sparse autoencoders find highly interpretable features in language models," in *The Twelfth International Conference on Learning Representations*, 2023.

[26] T. Bricken, A. Templeton, J. Batson, B. Chen, A. Jermyn, T. Conerly, N. Turner, C. Anil, C. Denison, A. Askell, *et al.*, "Towards monosemanticity: Decomposing language models with dictionary learning," *Transformer Circuits Thread*, vol. 2, 2023.

[27] S. Rajamanoharan, T. Lieberum, N. Sonnerat, A. Conmy, V. Varma, J. Kramár, and N. Nanda, "Jumping ahead: Improving reconstruction fidelity with jumprelu sparse autoencoders," *arXiv preprint arXiv:2407.14435*, 2024.

[28] K. Park, Y. J. Choe, and V. Veitch, "The linear representation hypothesis and the geometry of large language models," in *International Conference on Machine Learning*, pp. 39643–39666, PMLR, 2024.

[29] A. Templeton, T. Conerly, J. Marcus, J. Lindsey, T. Bricken, B. Chen, A. Pearce, C. Citro, E. Ameisen, A. Jones, H. Cunningham, N. L. Turner, C. McDougall, M. MacDiarmid, C. D. Freeman, T. R. Sumers, E. Rees, J. Batson, A. Jermyn, S. Carter, C. Olah, and T. Henighan, "Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet," *Transformer Circuits Thread*, 2024.

[30] T. Lieberum, S. Rajamanoharan, A. Conmy, L. Smith, N. Sonnerat, V. Varma, J. Kramar, A. Dragan, R. Shah, and N. Nanda, "Gemma scope: Open sparse autoencoders everywhere all at once on gemma 2," in *The 7th BlackboxNLP Workshop*, 2024.

[31] L. Gao, T. D. la Tour, H. Tillman, G. Goh, R. Troll, A. Radford, I. Sutskever, J. Leike, and J. Wu, "Scaling and evaluating sparse autoencoders," in *The Thirteenth International Conference on Learning Representations*, 2025.

[32] T. Fel, E. S. Lubana, J. S. Prince, M. Kowal, V. Boutin, I. Papadimitriou, B. Wang, M. Wattenberg, D. Ba, and T. Konkle, "Archetypal sae: Adaptive and stable dictionary learning for concept extraction in large vision models," *arXiv preprint arXiv:2502.12892*, 2025.

[33] S. Marks, C. Rager, E. J. Michaud, Y. Belinkov, D. Bau, and A. Mueller, "Sparse feature circuits: Discovering and editing interpretable causal graphs in language models," in *The Thirteenth International Conference on Learning Representations*, 2025.

[34] A. Karvonen, C. Rager, J. Lin, C. Tigges, J. I. Bloom, D. Chanin, Y.-T. Lau, E. Farrell, C. S. McDougall, K. Ayonrinde, D. Till, M. Wearden, A. Conmy, S. Marks, and N. Nanda, "SAEBench: A comprehensive benchmark for sparse autoencoders in language model interpretability," in *Forty-second International Conference on Machine Learning*, 2025.

[35] C. W. Groetsch and C. Groetsch, *Inverse problems in the mathematical sciences*, vol. 52. Springer, 1993.

[36] T. Hastie, R. Tibshirani, and M. Wainwright, *Statistical learning with sparsity: the lasso and generalizations*. CRC press, 2015.

[37] D. L. Donoho, "Compressed sensing," *IEEE Transactions on information theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

[38] E. J. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Transactions on information theory*, vol. 52, no. 2, pp. 489–509, 2006.

[39] E. J. Candès and M. B. Wakin, "An introduction to compressive sampling," *IEEE signal processing magazine*, vol. 25, no. 2, pp. 21–30, 2008.

[40] M. Lopes, "Estimating unknown sparsity in compressed sensing," in *International Conference on Machine Learning*, pp. 217–225, PMLR, 2013.

[41] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," in *Proceedings of the 26th annual international conference on machine learning*, pp. 689–696, 2009.

[42] A. Agarwal, A. Anandkumar, P. Jain, and P. Netrapalli, "Learning sparsely used overcomplete dictionaries via alternating minimization," *SIAM Journal on Optimization*, vol. 26, no. 4, pp. 2775–2799, 2016.

[43] N. S. Chatterji and P. L. Bartlett, "Alternating minimization for dictionary learning: Local convergence guarantees," *arXiv preprint arXiv:1711.03634*, 2017.

[44] B. Tolooshams, *Deep Learning for Inverse Problems in Engineering and Science*. PhD thesis, Harvard University, 2023.

[45] K. Gregor and Y. LeCun, "Learning fast approximations of sparse coding," in *Proceedings of international conference on international conference on machine learning*, pp. 399–406, 2010.

[46] P. Ablin, T. Moreau, M. Massias, and A. Gramfort, "Learning step sizes for unfolded sparse coding," in *Proceedings of Advances in Neural Information Processing Systems*, vol. 32, pp. 1–11, 2019.

[47] B. Malézieux, T. Moreau, and M. Kowalski, "Understanding approximate and unrolled dictionary learning for pattern recovery," in *International Conference on Learning Representations*, 2022.

[48] B. Tolooshams and D. E. Ba, "Stable and interpretable unrolled dictionary learning," *Transactions on Machine Learning Research*, 2022.

[49] T. V. Nguyen, R. K. Wong, and C. Hegde, "On the dynamics of gradient descent for autoencoders," in *Proceedings of International Conference on Artificial Intelligence and Statistics*, pp. 2858–2867, PMLR, 2019.

[50] S. Arora, R. Ge, T. Ma, and A. Moitra, "Simple, efficient, and neural algorithms for sparse coding," in *Proceedings of Conference on Learning Theory* (P. Grünwald, E. Hazan, and S. Kale, eds.), vol. 40 of *Proceedings of Machine Learning Research*, (Paris, France), pp. 113–149, PMLR, 03–06 Jul 2015.

[51] X. Chen, J. Liu, Z. Wang, and W. Yin, "Theoretical linear convergence of unfolded ista and its practical weights and thresholds," in *Proceedings of Advances in Neural Information Processing Systems*, vol. 31, pp. 1–11, 2018.

[52] A. Rangamani, A. Mukherjee, A. Basu, A. Arora, T. Ganapathi, S. Chin, and T. D. Tran, "Sparse coding and autoencoders," in *Proceedings of IEEE International Symposium on Information Theory (ISIT)*, pp. 36–40, 2018.

[53] B. Tolooshams, A. Song, S. Temereanca, and D. Ba, "Convolutional dictionary learning based auto-encoders for natural exponential-family distributions," in *Proceedings of the 37th International Conference on Machine Learning* (H. D. III and A. Singh, eds.), vol. 119 of *Proceedings of Machine Learning Research*, pp. 9493–9503, PMLR, 7 2020.

[54] S. Rambhatla, X. Li, and J. Haupt, "Noodl: Provable online dictionary learning and sparse coding," in *Proceedings of International Conference on Learning Representations*, pp. 1–11, 2018.

[55] S. S. R. Hindupur, E. S. Lubana, T. Fel, and D. Ba, "Projecting assumptions: The duality between sparse autoencoders and concept geometry," *arXiv preprint arXiv:2503.01822*, 2025.

[56] M. Elad, *Sparse and redundant representations: from theory to applications in signal and image processing*. Springer Science & Business Media, 2010.

[57] M. Aharon, M. Elad, and A. Bruckstein, "K-svd: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Transactions on Signal Processing*, vol. 54, no. 11, pp. 4311–4322, 2006.

[58] B. Cleary, L. Cong, A. Cheung, E. S. Lander, and A. Regev, "Efficient generation of transcriptomic profiles by random composite measurements," *Cell*, vol. 171, no. 6, pp. 1424–1436.e18, 2017.

[59] B. Cleary, B. Simonton, J. Bezney, E. Murray, S. Alam, A. Sinha, E. Habibi, J. Marshall, E. S. Lander, F. Chen, *et al.*, "Compressed sensing for highly efficient imaging transcriptomics," *Nature Biotechnology*, pp. 1–7, 2021.

[60] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[61] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM review*, vol. 43, no. 1, pp. 129–159, 2001.

[62] N. Parikh and S. Boyd, "Proximal algorithms," *Foundations and Trends in optimization*, vol. 1, no. 3, pp. 127–239, 2014.

[63] I. Daubechies, M. Defrise, and C. De Mol, "An iterative thresholding algorithm for linear inverse problems with a sparsity constraint," *Communications on Pure and Applied Mathematics*, vol. 57, no. 11, pp. 1413–1457, 2004.

[64] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM journal on imaging sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[65] M. a. Ranzato, C. Poultney, S. Chopra, and Y. Cun, "Efficient learning of sparse representations with an energy-based model," in *Advances in Neural Information Processing Systems*, vol. 19, MIT Press, 2007.

[66] M. a. Ranzato, Y.-l. Boureau, and Y. Cun, "Sparse feature learning for deep belief networks," in *Proceedings of Advances in Neural Information Processing Systems* (J. Platt, D. Koller, Y. Singer, and S. Roweis, eds.), vol. 20, 2008.

[67] J. R. Hershey, J. L. Roux, and F. Weninger, "Deep unfolding: Model-based inspiration of novel deep architectures," *preprint arXiv:1409.2574*, 2014.

[68] V. Monga, Y. Li, and Y. C. Eldar, "Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing," *IEEE Signal Processing Magazine*, vol. 38, no. 2, pp. 18–44, 2021.

[69] V. Papyan, Y. Romano, and M. Elad, "Convolutional neural networks analyzed via convolutional sparse coding," *Journal of Machine Learning Research*, vol. 18, no. 83, pp. 1–52, 2017.

[70] V. Papyan, J. Sulam, and M. Elad, "Working locally thinking globally: Theoretical guarantees for convolutional sparse coding," *IEEE Transactions on Signal Processing*, vol. 65, no. 21, pp. 5687–5701, 2017.

[71] D. Ba, "Deeply-sparse signal representations (ds2p)," *IEEE Transactions on Signal Processing*, vol. 68, pp. 4727–4742, 2020.

[72] F. Doshi-Velez and B. Kim, "Towards a rigorous science of interpretable machine learning," *preprint arXiv:1702.08608*, 2017.

A Appendix - Acknowledgments

A.S. conducted this work as a Dale and Suzanne Burger SURF Fellow through the Summer Undergraduate Research Fellowship (SURF) program at Caltech and gratefully acknowledges its funding. A.A. was supported by the Bren Chair Professorship and AI2050 Senior Fellowship.

B.T. proposed, designed, and led the project. A.S. implemented the methods and executed all experiments. A.S. and B.T. wrote the paper together. A.A. provided supervision, valuable feedback, and editorial comments.

B Appendix - Additional Background

Notations We denote scalars as non-bold, lower-case a, vectors as bold-lower-case $\boldsymbol{a}$, and matrices as upper-case letters $\boldsymbol{A}$.

Sparse model recovery The bi-level optimization (Def. 1.2) takes a general form. When the inner level objective $\mathcal{L}(\boldsymbol{x}, \boldsymbol{v}, \theta) = \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{D}\boldsymbol{v}\|_2^2 + \mathcal{R}(\boldsymbol{v})$, this problem is reduced to the classical alternating-minimization dictionary learning [42, 43]. In this classical form, sparse coding is widely applied in science and engineering; e.g., in signal processing [56], image denoising [57], and extracting interpretable gene modules [58, 59]. When the model parameters $\boldsymbol{D}$ are known, the problem is reduced to recovering the sparse representation by solving $\min_{\boldsymbol{z}\in\mathbb{R}^p} \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{D}\boldsymbol{z}\|_2^2 + \lambda\mathcal{R}(\boldsymbol{z})$, where $\mathcal{R}(\boldsymbol{z})$ is a sparse regularizer; setting $\mathcal{R}(\boldsymbol{z}) = \|\boldsymbol{z}\|_1$, the optimization problem is reduced down to lasso [60], or referred to as basis pursuit [61], solving via proximal gradient descent [62] such as iterative shrinkage-thresholding algorithm (ISTA) [63] and its fast momentum-based version [64].

Sparse ReLU autoencoders For shallow ReLU autoencoders [26], the formulation reduces to

$$\begin{aligned} \min_{\boldsymbol{D}\in\mathcal{D}} \quad & \tfrac{1}{2}\|\boldsymbol{x} - (\boldsymbol{D}\boldsymbol{z} + \boldsymbol{b}_{\text{pre}})\|_2^2 + \lambda\|\boldsymbol{z}\|_1 \\ \text{s.t} \quad & \boldsymbol{z} = \text{ReLU}(\boldsymbol{W}^\top(\boldsymbol{x} - \boldsymbol{b}_{\text{pre}}) + \boldsymbol{b}_{\text{enc}}) \end{aligned} \tag{4}$$

This connection between bilevel optimization and neural networks, highlighted in [44, 55], has been explored extensively in the context of sparse autoencoders. Prior works have studied the gradient dynamics of variants of ReLU networks when the encoder is shallow [49, 50, 52] or deep, as in learned ISTA architectures [48]. Others have analyzed model recovery when the encoder is deep and iterative with hard-thresholding nonlinearity (JumpReLU [27]) [54].

Overall, optimizing or recovering sparse generative linear models has attracted sustained attention from two main deep learning communities over the past years; a) *Unrolling learning*: using optimization formulations to design and theoretically study neural architectures [46–54], and b) *Sparse interpretability* or mechanistic interpretability: leveraging sparse models to interpret and analyze the internal representations of complex larger networks [24–34]. We briefly expand on both works.

Unrolled learning The early connections between sparse coding and deep learning trace back to sparse energy-based deep models [65, 66] and the pioneering work of LISTA [45] in constructing sparsifying recurrent neural networks. This line of work, known as unfolding [67] or unrolling [68], designs neural network layers based on iterations of an algorithm that solves an optimization problem. This connection has enabled researchers to use optimization models as a proxy to theoretically study accelerated convergence [51, 46], gradient dynamics [50, 52, 49, 53], and model recovery in both shallow [50] and deep neural networks [54, 48]. Furthermore, several works have explored the theoretical connections between convolutional neural networks and convolutional sparse coding [69, 70], or more generally, between deeply sparse signal representations and deep neural networks [71]. An example of a deep unrolled JumpReLU neural network for sparse model recovery can be formulated as the following bilevel optimization problem.

$$\min_{\boldsymbol{D}\in\mathcal{D}} \quad \|\boldsymbol{x} - \boldsymbol{D}\boldsymbol{z}_T\|_2^2 \quad \text{s.t.} \quad \boldsymbol{z}_t = \text{JumpReLU}_\lambda(\boldsymbol{z}_{t-1} - \alpha\boldsymbol{D}^\top(\boldsymbol{D}\boldsymbol{z}_{t-1} - \boldsymbol{x})) \tag{5}$$

for $t = 1, \dots, T$, where $\boldsymbol{z}_0 = \boldsymbol{0}$, α is the step size, and $\text{JumpReLU}_\lambda(\boldsymbol{z}) = \boldsymbol{z} \cdot \mathbb{1}_{\boldsymbol{z}>\lambda}$, where $\mathbb{1}$ is the indicator function, and λ controls the sparsity level of the representation. The inner mapping is now a deep/iterative encoder, which for simplicity we denote by $\boldsymbol{z}_T = f_\theta(\boldsymbol{x})$. The outer objective enforces the structure of the decoder and the loss function. The bilevel optimization in eq. (5) can be mapped to the following neural network autoencoder architecture, which uses a recurrent and residual encoder.

The synthetic experiments in this paper use this encoder in its architecture for sparse model recovery.

$$\begin{aligned} &\text{(encoder)} \quad \boldsymbol{z}_t = \text{JumpReLU}_\lambda(\boldsymbol{z}_{t-1} - \alpha \boldsymbol{D}^\top(\boldsymbol{D}\boldsymbol{z}_{t-1} - \boldsymbol{x})), \quad \text{for} \quad t = 1, \dots, T \\ &\text{(decoder)} \quad \hat{\boldsymbol{x}} = \boldsymbol{D}\boldsymbol{z}_T. \end{aligned} \tag{6}$$

For experiments recovering the sparse convolutional generative model, we use the architecture that takes convolution blocks, as shown below.

$$\begin{aligned} &\text{(encoder)} \quad \boldsymbol{z}_{c,t} = \text{JumpReLU}_\theta(\boldsymbol{z}_{c,t-1} - \alpha \boldsymbol{D}_c \star (\sum_{c=1}^{C} \boldsymbol{D}_c * \boldsymbol{z}_{c,t-1} - \boldsymbol{x})), \quad \text{for} \quad t = 1, \dots, T \\ &\text{(decoder)} \quad \hat{\boldsymbol{x}} = \boldsymbol{D} * \boldsymbol{z}_T \end{aligned} \tag{7}$$

for $t = 1, \dots, T$ and $c = 1, \dots, C$, where $\boldsymbol{z}_0 = \boldsymbol{0}$, α is the step size, $*$ is convolution operator, and $\star$ is a correlation operator.

Sparse Interpretability Interpretability in deep learning [72], particularly through extracting concepts learned by deep neural networks [21], has gained prominence in the era of large models. Motivated by the linear representation hypothesis (LRH) [24, 28], sparse autoencoders (SAEs) are widely used to recover interpretable structures from internal representations of large language models [25–27, 31] and vision models [32]. Recent studies show that the architecture of SAEs imposes inductive biases on which concepts are recoverable [55]. Building on this, we investigate conditions under which neural networks and neural operators recover similar concepts, and we explore their respective inductive biases. Our formulation of sparse model recovery in function spaces enables the use of SAEs operating on functions to interpret large neural operators.

Convolutional Setting Sparse generative models can be extended to *sparse convolutional generative models* (Def. B.1), where the concepts are locally and sparsely appearing in the data.

Definition B.1 (Sparse Convolutional Generative Models)**.** *A data example $\boldsymbol{x} \in \mathbb{R}^m$ is said to follow a sparse convolutional generative model if there exists a sparse latent representation $\boldsymbol{z} = \{\boldsymbol{z}_c \in \mathbb{R}^p\}_{c=1}^C$ (with supp$(\boldsymbol{z}_c) \le k \ll p$) and a set of C localized dictionary $\{\boldsymbol{D}_c^* \in \mathbb{R}^h\}_{c=1}^C$ (with $h \ll m$) such that $\boldsymbol{x} = \sum_{c=1}^C \boldsymbol{D}_c^* * \boldsymbol{z}_c$ in the noiseless setting, where $*$ denotes the convolution operator.*

Neural operators Neural operators consist of the following three modules:

- **Lifting**: This is a fully local operator which we model by a matrix $\boldsymbol{L} : \mathbb{R}^m \to \mathbb{R}^{d_{v_0}}$. It maps the input $\{\boldsymbol{x} : D_x \to \mathbb{R}^m\}$ into a latent representation $\{\boldsymbol{v}_0 : D_0 \to \mathbb{R}^{d_{v_0}}\}$, where $h > m$.
- **Kernel Integration**: For $t = 0, \dots, T-1$, this is a non-local integral kernel operator that maps representation at one layer $\{\boldsymbol{v}_t : D_t \to \mathbb{R}^{d_{v_t}}\}$ to the next $\{\boldsymbol{v}_{t+1} : D_{t+1} \to \mathbb{R}^{d_{v_{t+1}}}\}$ (see Def. B.2).
- **Projection**: This is a fully local operator, similar to the lifting operator. It maps the filtered lifted data to the output function, i.e., $\boldsymbol{P} : \mathbb{R}^{d_{v_T}} \to \mathbb{R}^m$, where $d_{v_T} > m$.

where the kernel integral operator is used to map $\boldsymbol{x}$ to an estimate of the representation $\boldsymbol{z}$, the input/output domains would be D_x and D_z, respectively. Moreover, when the kernel integral operator is used to refine the latent representation $\boldsymbol{z}$ from one layer to another, the input/output domain of the operator would be both D_z.

Definition B.2 (Kernel integral operator $\mathcal{K}$ (restated from [13, 14]))**.** *Define the kernel integral operator by*

$$(\mathcal{K}_t(\boldsymbol{v}_t))(x) := \int_{D_t} \kappa^{(t)}(x, y)\boldsymbol{v}_t(y) d\boldsymbol{v}_t(y), \quad \forall x \in D_{t+1} \tag{8}$$

where $\kappa^{(t)} : C(D_{t+1} \times D_t; \mathbb{R}^{dv_{t+1} \times dv_t})$ are the parameters of the kernels, modeled by a neural network, and v_t is a Borel measure on D_t, where C denotes the space of continuous functions.

Definition B.3 (Fourier transform)**.** *Let $\mathcal{F}$ be the Fourier transform of the function $\boldsymbol{v} : D \to \mathbb{R}^{d_v}$, whose inverse is denoted by $\mathcal{F}^{-1}$ on the function $\boldsymbol{w} : D \to \mathbb{C}^{d_w}$. We have*

$$\begin{aligned} (\mathcal{F}\boldsymbol{v})_j(k) &= \int_D \boldsymbol{v}_j(x) e^{-2i\pi\langle x,k\rangle} dx, \qquad j = 1, \dots, d_v \\ (\mathcal{F}^{-1}\boldsymbol{w})_j(x) &= \int_D \boldsymbol{w}_j(k) e^{2i\pi\langle x,k\rangle} dk, \qquad j = 1, \dots, d_w \end{aligned} \tag{9}$$

where $i = \sqrt{-1}$ is the imaginary unit.

C Appendix - Additional Experimental Results

C.1 Experimental Setup

Sparse Generative Model The results shown in Fig. 6 are from a synthetic dataset that follows a sparse generative model (Def. 1.1). The atoms of the ground-truth dictionary $\boldsymbol{D}^* \in \mathbb{R}^{1000\times 1500}$ are drawn from a standard normal distribution and then ℓ_2-normalized. The dataset consists of 50,000 samples $\boldsymbol{x} \in \mathbb{R}^{1000}$. Each latent code $\boldsymbol{z} \in \mathbb{R}^{1500}$ has a total sparsity of 20. The amplitudes of the non-zero elements are drawn from a sub-Gaussian distribution $\mathcal{N}(15, 1)$.

SAE For SAE (SAE-MLP) and L-SAE (L-SAE-MLP) models, the encoder implements $f_\theta(\boldsymbol{x})$ following Def. 1.2 using a deep unrolled JumpReLU network (eqs. (5) and (6)) with $T = 50$ layers, as described in the sparse model recovery framework (Def. 1.2). The non-linearity is Hard-Thresholding with a threshold of $\lambda = 0.5$, and the algorithm's internal step-size is $\alpha = 0.2$. The decoder reconstructs data as $\hat{\boldsymbol{x}} = \boldsymbol{D}\boldsymbol{z}$ following Def. 1.3. The dictionary weights for SAE and L-SAE are initialized with a noisy ($\sigma = 0.02$) version of the ground-truth dictionary. For the L-SAE, the 1000-dimensional input is lifted to a 1200-dimensional space. Both models are trained to minimize the Mean Squared Error (MSE) loss using the ADAM optimizer with a learning rate of $\eta = 10^{-3}$.

Sparse Convolutional Generative Model The results shown in Fig. 2 are from a synthetic dataset that follows a sparse convolutional generative model (Def. B.1). The ground-truth dictionary $\boldsymbol{D}^*$ consists of $p = 5$ kernels with spatial support $h = 99$. Each kernel is a multi-channel signal with 64 input channels. The values for each kernel are drawn from a standard normal distribution and then normalized. The dataset consists of 50,000 samples $\boldsymbol{x} \in \mathbb{R}^{64\times 1000}$. Each sample is generated by convolving a single (i.e., sparsity of 1) randomly chosen kernel from the dictionary with a sparse feature map, whose non-zero amplitudes are drawn from a sub-Gaussian distribution.

SAE-CNN For SAE-CNN and L-SAE-CNN models, the encoder implements the convolutional version of the deep unrolled network (eq. (7)) with $T = 50$ layers, as described in the sparse model recovery framework (Def. 1.2). The non-linearity is Hard-Thresholding with a threshold of $\lambda = 10$, and the algorithm's internal step-size is $\alpha = 0.01$. The dictionary for both SAE-CNN and L-SAE-CNN are initialized with a noisy ($\sigma = 0.05$) version of the ground-truth kernels. For L-SAE-CNN, the 64-channel input is lifted to a 128-dimensional space. Both models are trained to minimize the Mean Squared Error (MSE) loss using the SGD optimizer with a learning rate of $\eta = 0.04$.

SAE-FNO For SAE-FNO and L-SAE-FNO models, the encoder implements a deep unrolled network with $T = 50$ layers, as described in the sparse functional model recovery framework (Def. 2.2). The non-linearity is Hard-Thresholding with a threshold of $\lambda = 10$, and the algorithm's internal step-size is $\alpha = 0.01$. The dictionary kernels are initialized with a noisy ($\sigma = 0.05$) version of the ground-truth. For L-SAE-FNO, the 64-channel input is lifted to a 128-dimensional space. Both SAE-FNO and L-SAE-FNO are trained to minimize the Mean Squared Error (MSE) loss using the SGD optimizer with a learning rate of $\eta_L = 20.04$. All experiments are based on these settings, except Fig. 2b and Fig. 3.

Fig. 2b uses a modified experimental setup from the sparse convolutional generative model described above. The data consists of single channel signal with 1-sparse codes ($p = 1$ kernel), where the amplitudes of non-zero elements are drawn from a sub-Gaussian distribution $\mathcal{N}(5, 1)$. Training uses threshold $\lambda = 0.2$, step size $\alpha = 0.05$, and ADAM optimizer with learning rates 2×10^{-5} (SAE-CNN) and 0.01 (SAE-FNO).

Fig. 3 uses the experimental setup of Sparse Convolutional Generative Model with $1\times$, $2\times$, $4\times$, and $8\times$ upsampling rates. The upsampling process is as follows: original low-resolution data ($m = 1000$) is first upsampled by inserting zeros at intervals corresponding to the upsampling rate (e.g., for $4\times$ upsampling, 3 zeros are inserted between every original sample, creating a signal of length 4000). This process introduces high-frequency spectral replicas of the original signal. The expanded signal is then processed through a 10th-order Butterworth low-pass filter using zero-phase filtering (filtfilt) to remove aliasing and ensure smooth interpolation.

C.2 Experimental Results

Additional experimental results are shown in Figs. 4 to 7.

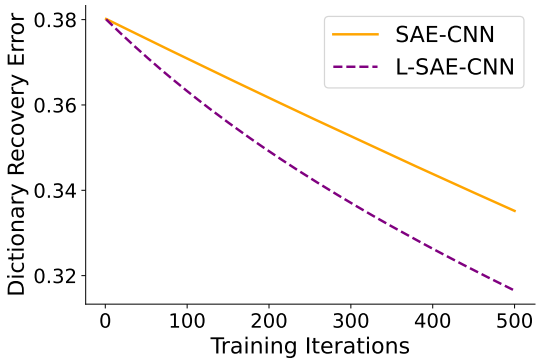


Fig. 4: **Lifting as a preconditioner.** Lifting accelerates learning.

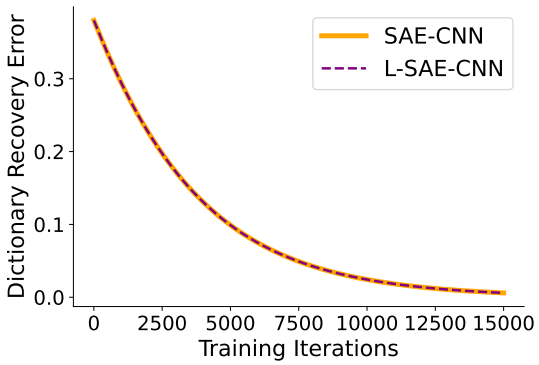
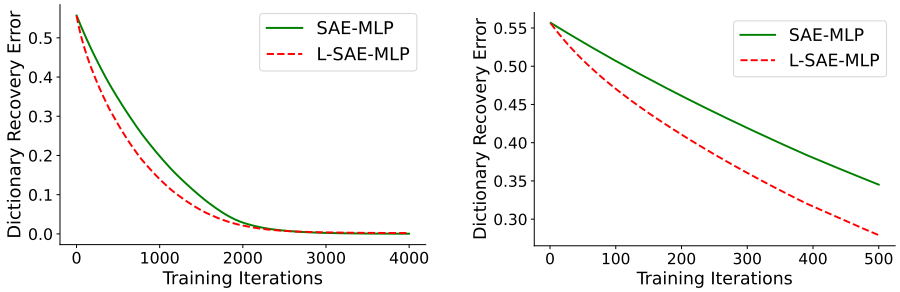
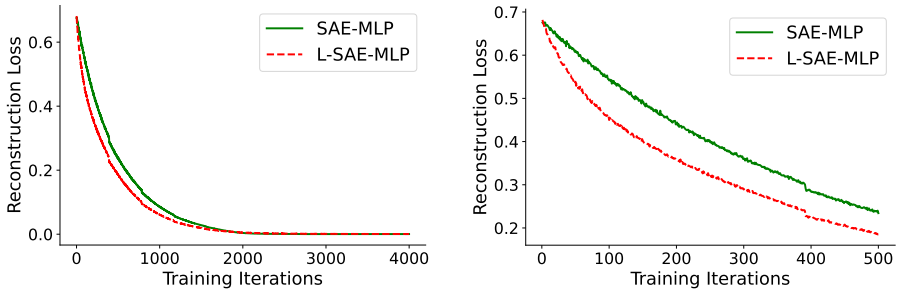


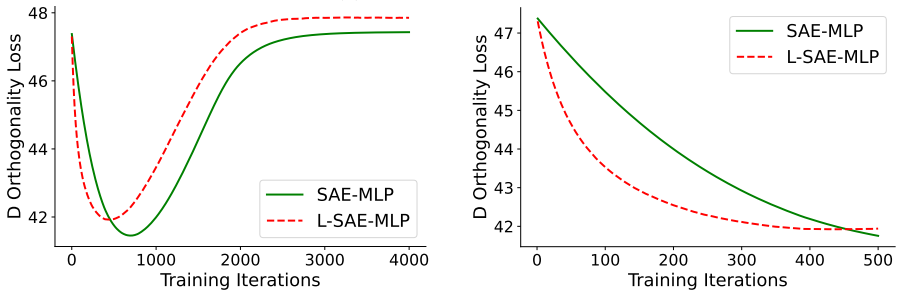
Fig. 5: **Lifting.** When the lifting operator satisfies the orthogonal condition $\boldsymbol{L}^\top \boldsymbol{L} = \boldsymbol{I}$, the lifted SAE-CNN (L-SAE-CNN) exhibits equivalent learning dynamics to the SAE-CNN .



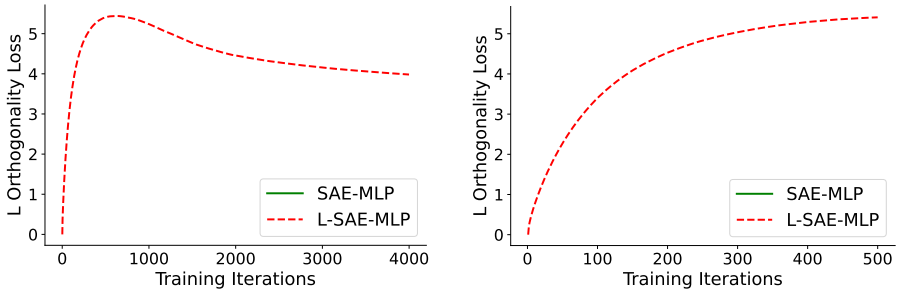
(a) Dictionary Recovery Error



(b) Reconstruction Loss

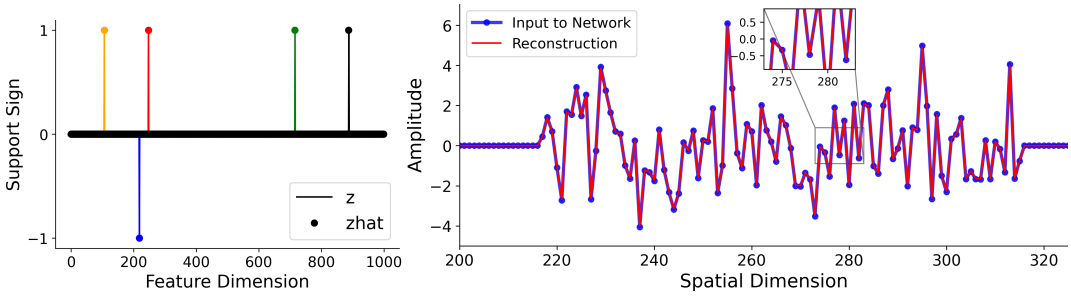


(c) D Orthogonality Loss

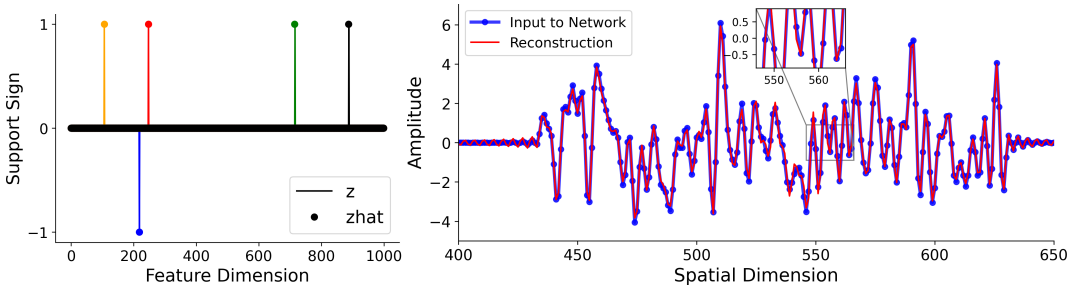


(d) L Orthogonality Loss

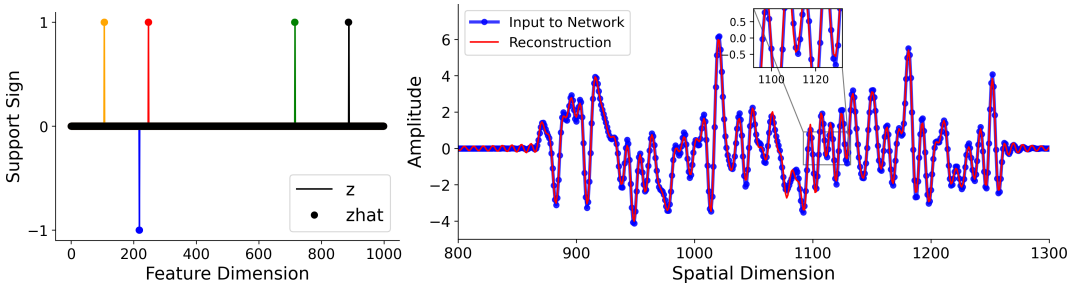
Fig. 6: **SAEs vs. L-SAEs.** Learning the lifting operator $\boldsymbol{L}$ accelerates model recovery. (a) Dictionary recovery error converges faster for L-SAE-MLP, confirming the preconditioning effect of lifting; (b) Reconstruction loss follows similar convergence trends as dictionary recovery error; (c) Lifting encourages the effective dictionary $\boldsymbol{D}$ to learn more orthogonal (less correlated) atoms early in training, creating a more isotropic loss landscape that accelerates recovery; (d) The lifting operator $\boldsymbol{L}$ initially becomes less orthogonal, allowing the dictionary $\boldsymbol{D}$ to become more orthogonal.



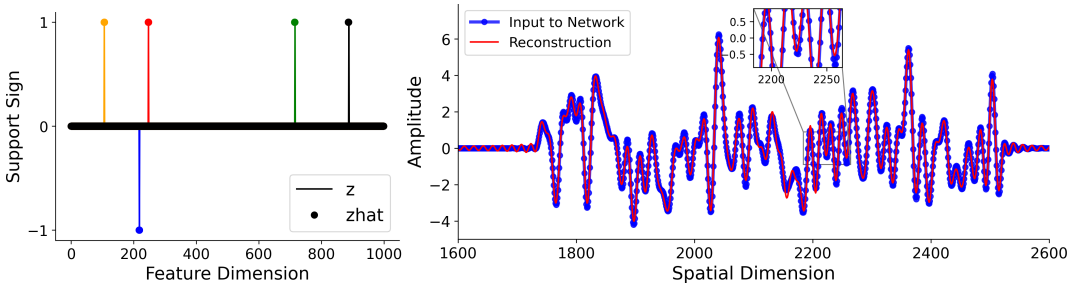
(a) Without Upsampling.



(b) $2\times$ Upsampling.



(c) $4\times$ Upsampling.



(d) $8\times$ Upsampling.

Fig. 7: **SAE-FNO Upsampling Robustness Across Resolutions.** SAE-FNO successfully infers the underlying sparse representations and reconstructs data across multiple discretization levels. The left panels show inference of 1-sparse code supports across 5 kernels, while the right panels display spatial-domain signal reconstruction. (a) Original resolution ($1\times$): Baseline performance at training resolution. (b-d) Higher upsampling rates ($2\times, 4\times, 8\times$): SAE-FNO maintains accurate code support inference and reconstruction as spatial resolution increases. Across all resolutions, the inferred support remains consistent with the ground truth, and reconstructions closely match the input signals. This resolution invariance property is unique to neural operators and represents a significant practical advantage over SAE-CNNs, which typically fail when input resolution differs from their training resolution. The function space of neural operators enables this generalization across discretizations.

D Appendix - Theoretical Analysis

This section provides the mathematical derivations for the training dynamics of the architectures discussed in the main text. We derive the gradient descent update rules for a single data sample, which are used to learn the dictionary parameters. These derivations consider the *analytic gradient*, i.e., the dictionary gradient is computed given the codes [47, 48].

Proposition 3.1 (Training Dynamics of Lifting)**.** *The training dynamics of the lifted-SAE (L-SAE) $\boldsymbol{D}_L^{(k+1)} = \boldsymbol{D}_L^{(k)} + \eta_L \boldsymbol{P}^\top(\boldsymbol{x} - \boldsymbol{P}\boldsymbol{D}_L^{(k)}\boldsymbol{z})\boldsymbol{z}^\top$, with lifting $\boldsymbol{L}$ and projection $\boldsymbol{P}$, has the effective update in the original space, expressed as: $\boldsymbol{D}^{(k+1)} = \boldsymbol{D}^{(k)} + \eta_L(\boldsymbol{L}^\top\boldsymbol{L})(\boldsymbol{x} - \boldsymbol{D}^{(k)}\boldsymbol{z})\boldsymbol{z}^\top$, where $\boldsymbol{L}^\top\boldsymbol{L}$ acts as a preconditioner, potentially accelerating learning by inducing a more isotropic update. If they satisfy the equivalent architectural inference (Prop. 3.2), then the dynamics are equivalent to those of an SAE: $\boldsymbol{D}^{(k+1)} = \boldsymbol{D}^{(k)} + \eta(\boldsymbol{x} - \boldsymbol{D}^{(k)}\boldsymbol{z})\boldsymbol{z}^\top$ (see Prop. D.2 for L-SAE-CNN).*

Proof. For the SAE, we learn a dictionary $\boldsymbol{D}$ by minimizing the reconstruction loss for the data sample $\boldsymbol{x}$. The loss function is:

$$\mathcal{L}(\boldsymbol{D}) = \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{D}\boldsymbol{z}\|_2^2 \tag{10}$$

We consider the case of SAE-MLP. The gradient of the loss for $\boldsymbol{D}$ is:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{D}} = -(\boldsymbol{x} - \boldsymbol{D}\boldsymbol{z})\boldsymbol{z}^\top \tag{11}$$

The analytic gradient update, given $\boldsymbol{z}$, for $\boldsymbol{D}$ at iteration k with a learning rate η is therefore:

$$\boldsymbol{D}^{(k+1)} = \boldsymbol{D}^{(k)} - \eta\frac{\partial \mathcal{L}}{\partial \boldsymbol{D}^{(k)}} = \boldsymbol{D}^{(k)} + \eta(\boldsymbol{x} - \boldsymbol{D}^{(k)}\boldsymbol{z})\boldsymbol{z}^\top \tag{12}$$

For lifted-SAE, we learn a dictionary $\boldsymbol{D}_L \in \mathbb{R}^{h\times p}$ in a higher-dimensional lifted space. The input $\boldsymbol{x}$ is lifted by $\boldsymbol{L} \in \mathbb{R}^{h\times m}$ and the reconstruction is projected back by $\boldsymbol{P} \in \mathbb{R}^{m\times h}$.

The loss function is defined in the original data space:

$$\mathcal{L}(\boldsymbol{D}) = \frac{1}{2}\|\boldsymbol{x} - \boldsymbol{P}\boldsymbol{D}_L\boldsymbol{z}\|_2^2 \tag{13}$$

The analytic gradient given $\boldsymbol{z}$ of the loss with respect to $\boldsymbol{D}_L$ is:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{D}} = -\boldsymbol{P}^\top(\boldsymbol{x} - \boldsymbol{P}\boldsymbol{D}_L\boldsymbol{z})\boldsymbol{z}^\top \tag{14}$$

The gradient update rule for the lifted dictionary $\boldsymbol{D}_L$ is:

$$\boldsymbol{D}_L^{(k+1)} = \boldsymbol{D}_L^{(k)} - \eta_L\frac{\partial \mathcal{L}}{\partial \boldsymbol{D}_L^{(k)}} = \boldsymbol{D}_L^{(k)} + \eta_L\boldsymbol{P}^\top(\boldsymbol{x} - \boldsymbol{P}\boldsymbol{D}_L\boldsymbol{z})(\boldsymbol{z})^\top \tag{15}$$

To understand the learning dynamics in the original space, we first incorporate the underlying model from L-SAEs into SAE using the relation $\boldsymbol{D}' = \boldsymbol{P}\boldsymbol{D}_L$. Then, we derive the update rule for the effective dictionary, $\boldsymbol{D}'$. Left-multiplying the update for $\boldsymbol{D}_L^{(k+1)}$ by $\boldsymbol{P}$, we have:

$$\boldsymbol{D}'^{(k+1)} = \boldsymbol{D}'^{(k)} + \eta_L(\boldsymbol{P}\boldsymbol{P}^\top)(\boldsymbol{x} - \boldsymbol{D}'^{(k)}\boldsymbol{z})(\boldsymbol{z})^\top \tag{16}$$

In our specific architecture, we constrain the projection matrix to be the transpose of the lifting matrix such that $\boldsymbol{P} = \boldsymbol{L}^\top$. Substituting this into the update rules yields:

$$\boldsymbol{D}_L^{(k+1)} = \boldsymbol{D}_L^{(k)} + \eta_L\boldsymbol{L}(\boldsymbol{x} - \boldsymbol{L}^\top\boldsymbol{D}_L\boldsymbol{z})(\boldsymbol{z})^\top \tag{17}$$

$$\boldsymbol{D}'^{(k+1)} = \boldsymbol{D}'^{(k)} + \eta_L(\boldsymbol{L}^\top\boldsymbol{L})(\boldsymbol{x} - \boldsymbol{D}'^{(k)}\boldsymbol{z})(\boldsymbol{z})^\top \tag{18}$$

Hence, when $\boldsymbol{L}^\top\boldsymbol{L} = \boldsymbol{I}$, the lifted-SAE shows equivalent learning dynamics as the SAE for model recovery. ■

Proposition 3.2 (Architectural Inference Equivalence of Lifting in SAE)**.** *The architectural inference of a SAE is equivalent to a lifted-SAE if the projection is tied to the lifting operator ($\boldsymbol{P} = \boldsymbol{L}^\top$), and the lifting operator is orthogonal ($\boldsymbol{L}^\top \boldsymbol{L} = \boldsymbol{I}$), i.e., under the same learned model $\boldsymbol{D} = \boldsymbol{P}\boldsymbol{D}_L$, the representation dynamics and output of the networks are equivalent (see Prop. D.1 for SAE-CNN and Prop. D.6 for SAE-FNO).*

Proof. We provide the proof for the sparse generative model case when the encoder implements a proximal gradient descent type of operation in an unrolled learning setting (e.g., 6). For the SAE, the representation refinement at every encoder layer follows:

$$\boldsymbol{z}_{t+1} = \mathcal{R}(\boldsymbol{z}_t - \alpha \boldsymbol{D}^\top (\boldsymbol{D}\boldsymbol{z}_t - \boldsymbol{x})) \tag{19}$$

By construction, we incorporate the underlying model from the lifted-SAE to the standard SAE: $\boldsymbol{D} = \boldsymbol{P}\boldsymbol{D}_L$, and re-write the encoder as:

$$\boldsymbol{z}_{t+1} = \mathcal{R}(\boldsymbol{z}_t - \alpha (\boldsymbol{P}\boldsymbol{D}_L)^\top (\boldsymbol{P}\boldsymbol{D}_L \boldsymbol{z}_t - \boldsymbol{x})) = \mathcal{R}(\boldsymbol{z}_t - \alpha \boldsymbol{D}_L^\top \boldsymbol{P}^\top (\boldsymbol{P}\boldsymbol{D}_L \boldsymbol{z}_t - \boldsymbol{x})) \tag{20}$$

Using the assumption on the lifting-projection,

$$\boldsymbol{z}_{t+1} = \mathcal{R}\left(\boldsymbol{z}_t - \alpha (\boldsymbol{D}_L)^\top (\boldsymbol{D}_L \boldsymbol{z}_t - \boldsymbol{L}\boldsymbol{x})\right) \tag{21}$$

This final expression is the iterative encoder update for $\boldsymbol{z}$ in the lifted space, where the dictionary is $\boldsymbol{D}_L$ and the input is lifted by $\boldsymbol{L}$. This completes the proof of architectural inference equivalence on the standard and lifted architectures. ■

Proposition D.1 (Architectural Inference Equivalence of Lifting in SAE-CNN)**.** *The architectural inference of a SAE-CNN is equivalent to a lifted SAE-CNN (L-SAE-CNN) if the projection is tied to the lifting operator ($\boldsymbol{P} = \boldsymbol{L}^\top$) that is orthogonal ($\boldsymbol{L}^\top \boldsymbol{L} = \boldsymbol{I}$), i.e., under the same learned model $\boldsymbol{D}_c = \boldsymbol{P}\boldsymbol{D}_{L,c}$, the representation dynamics and output of the networks are equivalent.*

Proof. We provide the proof for the sparse convolutional generative model case when the encoder implements a proximal gradient descent type of operation in an unrolled learning setting (e.g., 7). For the SAE-CNN, the representation refinement for the c-th feature map at every encoder layer follows:

$$\boldsymbol{z}_{c,t+1} = \mathcal{R}\left(\boldsymbol{z}_{c,t} - \alpha \left(\sum_{i=1}^{C} \boldsymbol{D}_i * \boldsymbol{z}_{i,t} - \boldsymbol{x}\right) \star \boldsymbol{D}_c\right) \tag{22}$$

By construction, we incorporate the underlying model from the lifted-SAE-CNN to the standard SAE-CNN: $\boldsymbol{D}_c^{(k)} = \boldsymbol{P}\boldsymbol{D}_{L,c}^{(k)}$, and re-write the encoder using the adjoint property:

$$\boldsymbol{z}_{c,t+1} = \mathcal{R}\left(\boldsymbol{z}_{c,t} - \alpha \left(\boldsymbol{P}^\top \left[\left(\sum_{i=1}^{C} \boldsymbol{P}\boldsymbol{D}_{L,i} * \boldsymbol{z}_{i,t}\right) - \boldsymbol{x}\right] \star \boldsymbol{D}_{L,c}\right)\right) \tag{23}$$

$$\boldsymbol{z}_{c,t+1} = \mathcal{R}\left(\boldsymbol{z}_{c,t} - \alpha \left(\sum_{i=1}^{C} (\boldsymbol{P}^\top \boldsymbol{P})\boldsymbol{D}_{L,i} * \boldsymbol{z}_{i,t} - \boldsymbol{P}^\top \boldsymbol{x}\right) \star \boldsymbol{D}_{L,c}\right) \tag{24}$$

Using the assumption on the lifting-projection,

$$\boldsymbol{z}_{c,t+1} = \mathcal{R}\left(\boldsymbol{z}_{c,t} - \alpha \left(\sum_{i=1}^{C} \boldsymbol{D}_{L,i} * \boldsymbol{z}_{i,t} - \boldsymbol{L}\boldsymbol{x}\right) \star \boldsymbol{D}_{L,c}\right) \tag{25}$$

This final expression is the iterative encoder update for $\boldsymbol{z}_c$ in the lifted space, where the dictionary is $\boldsymbol{D}_L$ and the input is lifted by $\boldsymbol{L}$. This completes the proof of architectural inference equivalence for SAE-CNN and L-SAE-CNN. ■

Proposition D.2. *[Training Dynamics of Lifted SAE-CNN] The training dynamics of the lifted-SAE-CNN (L-SAE-CNN) $\boldsymbol{D}_{L,c}^{(k+1)} = \boldsymbol{D}_{L,c}^{(k)} + \eta_L \boldsymbol{L}\left(\boldsymbol{x} - \boldsymbol{L}^\top \left(\sum_{i=1}^{C} \boldsymbol{D}_{L,i}^{(k)} * \boldsymbol{z}_{i,t}\right)\right) \star \boldsymbol{z}_{c,t}$, with a lifting operator $\boldsymbol{L}$ and projection $\boldsymbol{P}$, has the effective update in the original space, expressed as: $\boldsymbol{D}_c^{(k+1)} =$*

$\boldsymbol{D}_c^{(k)} + \eta_L(\boldsymbol{L}^\top \boldsymbol{L}) \left(\boldsymbol{x} - \sum_{i=1}^{C} \boldsymbol{D}_i^{(k)} * \boldsymbol{z}_{i,t} \right) \star \boldsymbol{z}_{c,t}$, *where* $\boldsymbol{L}^\top \boldsymbol{L}$ *acts as a preconditioner, potentially accelerating learning by inducing a more isotropic update. If the architectures satisfy the architectural inference equivalence condition (Prop. D.1), then the dynamics become equivalent to that of a SAE-CNN:* $\boldsymbol{D}_c^{(k+1)} = \boldsymbol{D}_c^{(k)} + \eta \left(\boldsymbol{x} - \sum_{i=1}^{C} \boldsymbol{D}_i^{(k)} * \boldsymbol{z}_{i,t} \right) \star \boldsymbol{z}_{c,t}$.

Proof. For the SAE-CNN, we learn a dictionary $\boldsymbol{D}$ by minimizing the reconstruction loss for the data sample $\boldsymbol{x}$. The loss function is:

$$\mathcal{L}(\boldsymbol{D}) = \frac{1}{2} \left\| \boldsymbol{x} - \sum_{c=1}^{C} \boldsymbol{D}_c * \boldsymbol{z}_c \right\|_2^2 \tag{26}$$

The gradient of the loss with respect to the c-th kernel $\boldsymbol{D}_c$ is:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{D}_c} = - \left(\boldsymbol{x} - \sum_{i=1}^{C} \boldsymbol{D}_i * \boldsymbol{z}_i \right) \star \boldsymbol{z}_c \tag{27}$$

where $\star$ denotes cross-correlation.

The analytic gradient update, given $\boldsymbol{z}$, for $\boldsymbol{D}_c$ at iteration k with a learning rate η is:

$$\boldsymbol{D}_c^{(k+1)} = \boldsymbol{D}_c^{(k)} + \eta \left(\boldsymbol{x} - \sum_{i=1}^{C} \boldsymbol{D}_i^{(k)} * \boldsymbol{z}_{i,t} \right) \star \boldsymbol{z}_{c,t} \tag{28}$$

For L-SAE-CNN, we learn a dictionary $\boldsymbol{D}_L = \{\boldsymbol{D}_{L,c}\}_{c=1}^{C}$ that operates in a higher-dimensional lifted space. The input signal $\boldsymbol{x}$ is first lifted by $\boldsymbol{L}$ (acts on the channel dimension) to $\tilde{x}$ and the reconstruction $\hat{\boldsymbol{x}}$ is obtained by projecting the lifted reconstruction $\hat{\tilde{\boldsymbol{x}}} = \sum_c \boldsymbol{D}_{L,c} * \boldsymbol{z}_c$ back by $\boldsymbol{P}$.

The loss function is defined in the original data space:

$$\mathcal{L}(\boldsymbol{D}) = \frac{1}{2} \left\| \boldsymbol{x} - \boldsymbol{P} \left(\sum_{c=1}^{C} \boldsymbol{D}_{L,c} * \boldsymbol{z}_c \right) \right\|_2^2 \tag{29}$$

The analytic gradient of the loss, given $\boldsymbol{z}$, with respect the c-th lifted kernel $\boldsymbol{D}_{L,c}$ is:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{D}_{L,c}} = -\boldsymbol{P}^\top \left(\boldsymbol{x} - \boldsymbol{P} \left(\sum_{i=1}^{C} \boldsymbol{D}_{L,i} * \boldsymbol{z}_i \right) \right) \star \boldsymbol{z}_c \tag{30}$$

The gradient update rule for $\boldsymbol{D}_{L,c}$ is:

$$\boldsymbol{D}_{L,c}^{(k+1)} = \boldsymbol{D}_{L,c}^{(k)} + \eta_L \boldsymbol{P}^\top \left(\boldsymbol{x} - \boldsymbol{P} \left(\sum_{i=1}^{C} \boldsymbol{D}_{L,i}^{(k)} * \boldsymbol{z}_{i,t} \right) \right) \star \boldsymbol{z}_{c,t} \tag{31}$$

To understand the learning dynamics in the original space, we derive the update rule for the effective dictionary $\boldsymbol{D}'_c = \boldsymbol{P}\boldsymbol{D}_{L,c}$, i.e., the case where both frameworks contain the same underlying model but in a different form. Left-multiplying the update for $\boldsymbol{D}_{L,c}^{(k+1)}$ update rule by $\boldsymbol{P}$, we get:

$$\boldsymbol{D}_c'^{(k+1)} = \boldsymbol{D}_c'^{(k)} + \eta_L (\boldsymbol{P}\boldsymbol{P}^\top) \left(\boldsymbol{x} - \sum_{i=1}^{C} \boldsymbol{D}_i'^{(k)} * \boldsymbol{z}_{i,t} \right) \star \boldsymbol{z}_{c,t} \tag{32}$$

In our specific architecture, we constrain the projection matrix to be the transpose of the lifting matrix such that $\boldsymbol{P} = \boldsymbol{L}^\top$. Substituting this into the update rules yields:

$$\boldsymbol{D}_{L,c}^{(k+1)} = \boldsymbol{D}_{L,c}^{(k)} + \eta_L \boldsymbol{L} \left(\boldsymbol{x} - \boldsymbol{L}^\top \left(\sum_{i=1}^{C} \boldsymbol{D}_{L,i}^{(k)} * \boldsymbol{z}_{i,t} \right) \right) \star \boldsymbol{z}_{c,t} \tag{33}$$

$$\boldsymbol{D}_c'^{(k+1)} = \boldsymbol{D}_c'^{(k)} + \eta_L(\boldsymbol{L}^\top \boldsymbol{L}) \left(\boldsymbol{x} - \sum_{i=1}^{C} \boldsymbol{D}_i'^{(k)} * \boldsymbol{z}_{i,t} \right) \star \boldsymbol{z}_{c,t} \tag{34}$$

Hence, when $\boldsymbol{L}^\top \boldsymbol{L} = \boldsymbol{I}$, the lifted-SAE-CNN shows equivalent learning dynamics as the standard SAE-CNN for model recovery. ■

Proposition D.3. *[Training Dynamics of Lifted SAE-FNO] The training dynamics of the Lifted-SAE-FNO (L-SAE-FNO)'s frequency domain weights* $\boldsymbol{W}_{L,c}^{(k+1)} = \boldsymbol{W}_{L,c}^{(k)} + \frac{\eta_L}{\sqrt{M}} \mathcal{F}\left(\boldsymbol{L}\left(\boldsymbol{x} - \hat{\boldsymbol{x}}^{(k)}\right)\right) \odot \mathcal{F}(\boldsymbol{z}_{c,t})$*, with lifting operator* $\boldsymbol{L}$ *and projection* $\boldsymbol{P}$*, has an effective update in the original space, expressed as:* $\boldsymbol{D}_c^{(k+1)} = \boldsymbol{D}_c^{(k)} + \frac{\eta_L}{M}\left((\boldsymbol{L}^\top \boldsymbol{L})\left(\boldsymbol{x} - \hat{\boldsymbol{x}}^{(k)}\right)\right) \star \boldsymbol{z}_{c,t}$*, where* $\boldsymbol{L}^\top \boldsymbol{L}$ *acts as a preconditioner, potentially accelerating learning by inducing a more isotropic update. If the architectures satisfy the architectural inference equivalence condition (Prop. D.6), then the dynamics are equivalent to that of an SAE-FNO:* $\boldsymbol{D}_c'^{(k+1)} = \boldsymbol{D}_c'^{(k)} + \frac{\eta}{M}\left(\boldsymbol{x} - \frac{1}{\sqrt{M}} \mathcal{F}^{-1}\left(\sum_{i=1}^{C} \mathcal{F}(\boldsymbol{D}_i^{(k)}) \odot \mathcal{F}(\boldsymbol{z}_{i,t})\right)\right) \star \boldsymbol{z}_{c,t}$.

Proof. For the SAE-FNO, we learn frequency-domain weights $\boldsymbol{W}_c = \mathcal{F}\boldsymbol{D}_c$ by minimizing the reconstruction loss for the data sample $\boldsymbol{x}$. Taking normalization into account, the reconstruction operator is $\hat{\boldsymbol{x}} = \mathcal{G}_{\boldsymbol{W}}(\boldsymbol{z}) = \frac{1}{\sqrt{M}} \mathcal{F}^{-1}\big(\sum_{c=1}^{C} \boldsymbol{W}_c \cdot (\mathcal{F}\boldsymbol{z}_c)\big)$, where $\boldsymbol{W}_c = \mathcal{F}\boldsymbol{D}_c$, where M is the number of modes. The loss function is:

$$\mathcal{L}(\boldsymbol{W}) = \frac{1}{2} \mathbb{E}_{\boldsymbol{z}\sim\mu}\left[\|\boldsymbol{x} - \mathcal{G}_{\boldsymbol{W}}(\boldsymbol{z})\|_{\mathcal{X}}^2\right] \tag{35}$$

where $\boldsymbol{x} = \mathcal{G}_{\boldsymbol{D}^*}(\boldsymbol{z})$ is the ground-truth signal.

The gradient of the loss with respect to the c-th frequency-domain kernel $\boldsymbol{W}_c$ is:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_c} = -\frac{1}{\sqrt{M}} \mathcal{F}\left(\boldsymbol{x} - \hat{\boldsymbol{x}}\right) \odot \mathcal{F}(\boldsymbol{z}_c) \tag{36}$$

The analytic gradient update, given $\boldsymbol{z}$, for $\boldsymbol{W}_c$ at iteration k with a learning rate η is:

$$\boldsymbol{W}_c^{(k+1)} = \boldsymbol{W}_c^{(k)} + \frac{\eta}{\sqrt{M}} \mathcal{F}\left(\boldsymbol{x} - \hat{\boldsymbol{x}}^{(k)}\right) \odot \mathcal{F}(\boldsymbol{z}_{c,t}) \tag{37}$$

Applying the inverse Fourier transform and the cross-correlation theorem, we obtain the effective update rule for the spatial dictionary $\boldsymbol{D}_c'$:

$$\boldsymbol{D}_c'^{(k+1)} = \boldsymbol{D}_c'^{(k)} + \frac{\eta}{M}\left(\boldsymbol{x} - \hat{\boldsymbol{x}}^{(k)}\right) \star \boldsymbol{z}_{c,t} = \boldsymbol{D}_c'^{(k)} + \frac{\eta}{M}\left(\boldsymbol{x} - \frac{1}{\sqrt{M}} \mathcal{F}^{-1} \sum_{i=1}^{C} \mathcal{F}\boldsymbol{D}_i^{(k)} \odot \mathcal{F}\boldsymbol{z}_{i,t}\right) \star \boldsymbol{z}_{c,t} \tag{38}$$

In the lifted scenario, we learn a dictionary $\boldsymbol{W}_L = \{\boldsymbol{W}_{L,c}\}_{c=1}^{C}$ that operates in a higher-dimensional lifted frequency domain. The input signal $\boldsymbol{x}$ is lifted by $\boldsymbol{L}$, and the reconstruction is projected back by $\boldsymbol{P}$. The reconstruction operator is $\hat{\boldsymbol{x}} = \boldsymbol{P}\left(\mathcal{G}_{\boldsymbol{W}_L}(\boldsymbol{z})\right) = \boldsymbol{P}\left(\frac{1}{\sqrt{M}} \mathcal{F}^{-1}\left(\sum_{c=1}^{C} \boldsymbol{W}_{L,c} \odot \mathcal{F}(\boldsymbol{z}_c)\right)\right)$.

The loss function is defined in the original spatial space:

$$\mathcal{L}(\boldsymbol{W}) = \frac{1}{2} \mathbb{E}_{\boldsymbol{z}\sim\mu}\left[\|\boldsymbol{x} - \boldsymbol{P}\left(\mathcal{G}_{\boldsymbol{W}_L}(\boldsymbol{z})\right)\|_{\mathcal{X}}^2\right] \tag{39}$$

The gradient of the loss with respect to $\boldsymbol{W}_{L,c}$, given $\boldsymbol{z}$, is:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_{L,c}} = -\frac{1}{\sqrt{M}} \mathcal{F}\left(\boldsymbol{P}^\top \left(\boldsymbol{x} - \hat{\boldsymbol{x}}^{(k)}\right)\right) \odot \mathcal{F}(\boldsymbol{z}_c) \tag{40}$$

The analytic gradient update update for $\boldsymbol{W}_{L,c}$ at iteration k with a learning rate η_L is:

$$\boldsymbol{W}_{L,c}^{(k+1)} = \boldsymbol{W}_{L,c}^{(k)} + \frac{\eta_L}{\sqrt{M}} \mathcal{F}\left(\boldsymbol{P}^\top \left(\boldsymbol{x} - \hat{\boldsymbol{x}}^{(k)}\right)\right) \odot \mathcal{F}(\boldsymbol{z}_{c,t}) \tag{41}$$

To understand the learning dynamics in the original spatial-domain, we derive the update rule for the effective lifted spatial dictionary $\boldsymbol{D}'_{L,c}$. Taking the inverse Fourier, we get:

$$\boldsymbol{D}'^{(k+1)}_{L,c} = \boldsymbol{D}'^{(k)}_{L,c} + \frac{\eta_L}{M}\left(\boldsymbol{P}^\top\left(\boldsymbol{x} - \hat{\boldsymbol{x}}^{(k)}\right)\right) \star \boldsymbol{z}_{c,t} \tag{42}$$

We also derive the update rule for the effective spatial dictionary in the original space $\boldsymbol{D}'_c$. Left-multiplying the update for $\boldsymbol{D}'_{L,c}$ by $\boldsymbol{P}$, we have:

$$\boldsymbol{D}'^{(k+1)}_{c} = \boldsymbol{D}'^{(k)}_{c} + \frac{\eta_L}{M}\left((\boldsymbol{P}\boldsymbol{P}^\top)\left(\boldsymbol{x} - \hat{\boldsymbol{x}}^{(k)}\right)\right) \star \boldsymbol{z}_{c,t} \tag{43}$$

In our specific architecture, we constrain the projection matrix to be the transpose of the lifting matrix such that $\boldsymbol{P} = \boldsymbol{L}^\top$. Substituting this into the update rules yields:

$$\boldsymbol{W}^{(k+1)}_{L,c} = \boldsymbol{W}^{(k)}_{L,c} + \frac{\eta_L}{\sqrt{M}}\mathcal{F}\left(\boldsymbol{L}\left(\boldsymbol{x} - \hat{\boldsymbol{x}}^{(k)}\right)\right) \odot \mathcal{F}(\boldsymbol{z}_{c,t}) \tag{44}$$

$$\boldsymbol{D}'^{(k+1)}_{L,c} = \boldsymbol{D}'^{(k)}_{L,c} + \frac{\eta_L}{M}\left(\boldsymbol{L}\left(\boldsymbol{x} - \hat{\boldsymbol{x}}^{(k)}\right)\right) \star \boldsymbol{z}_{c,t} \tag{45}$$

$$\boldsymbol{D}'^{(k+1)}_{c} = \boldsymbol{D}'^{(k)}_{c} + \frac{\eta_L}{M}\left((\boldsymbol{L}^\top\boldsymbol{L})\left(\boldsymbol{x} - \hat{\boldsymbol{x}}^{(k)}\right)\right) \star \boldsymbol{z}_{c,t} \tag{46}$$

Substituting for $\hat{\boldsymbol{x}}^{(k)}$, the full effective spatial-domain update is:

$$\boldsymbol{D}'^{(k+1)}_{c} = \boldsymbol{D}'^{(k)}_{c} + \frac{\eta_L}{M}\left((\boldsymbol{L}^\top\boldsymbol{L})\left(\boldsymbol{x} - \frac{1}{\sqrt{M}}\mathcal{F}^{-1}\left(\sum_{i=1}^{C}\mathcal{F}(\boldsymbol{D}'^{(k)}_i) \odot \mathcal{F}(\boldsymbol{z}_{i,t})\right)\right)\right) \star \boldsymbol{z}_{c,t} \tag{47}$$

Hence, when $\boldsymbol{L}^\top\boldsymbol{L} = \boldsymbol{I}$, the lifted-SAE-FNO shows equivalent learning dynamics as the standard SAE-FNO for model recovery. ■

Proposition D.4 (Architectural Inference Equivalence of SAE-CNN and SAE-FNO)**.** *The architectural inference of an SAE-CNN is equivalent to a sparse autoencoder neural operator (SAE-NO) if the integral operator of SAE-FNO and parameters of SAEs are defined as a convolution where the SAE-FNO's frequency-domain weights $\boldsymbol{W}_c$ are the Fourier transform of SAE-CNN's spatial domain kernels: $\boldsymbol{W}_c = \mathcal{F}\boldsymbol{D}_c$.*

Proof. We provide the proof for an encoder that implements a proximal gradient descent type of operation in an unrolled learning setting. For the SAE-CNN, the representation refinement for the c-th feature map at every encoder layer follows:

$$\boldsymbol{z}_{c,t+1} = \mathcal{R}\left(\boldsymbol{z}_{c,t} - \alpha\left(\sum_{i=1}^{C}\boldsymbol{D}_i * \boldsymbol{z}_{i,t} - \boldsymbol{x}\right) \star \boldsymbol{D}_c\right) \tag{48}$$

By construction, we incorporate the underlying model from the SAE-FNO to the SAE-CNN: $\boldsymbol{W}_c = \mathcal{F}\boldsymbol{D}_c$, and re-write the gradient of the encoder using Correlation and Convolution Theorems:

$$\mathcal{F}^{-1}\left(\mathcal{F}\left(\sum_{i=1}^{C}\boldsymbol{D}_i * \boldsymbol{z}_{i,t} - \boldsymbol{x}\right) \odot \mathcal{F}\boldsymbol{D}_c^H\right) = \mathcal{F}^{-1}\left(\sum_{i=1}^{C}\boldsymbol{W}_i \odot \mathcal{F}\boldsymbol{z}_{i,t} - \mathcal{F}\boldsymbol{x}\right) \odot (\boldsymbol{W}_c)^H \tag{49}$$

Substituting this back into eq. (48) gives:

$$\boldsymbol{z}_{c,t+1} = \mathcal{R}\left(\boldsymbol{z}_{c,t} - \alpha\mathcal{F}^{-1}\left(\left(\sum_{i=1}^{C}\boldsymbol{W}_i \odot \mathcal{F}\boldsymbol{z}_{i,t}\right) - \mathcal{F}\boldsymbol{x}\right) \odot (\boldsymbol{W}_c)^H\right) \tag{50}$$

This final expression is the SAE-FNO's iterative encoder update for $\boldsymbol{z}_c$ in the spectral domain, where the dictionary is $\boldsymbol{W}$. This completes the proof of architectural inference equivalence. ■

Proposition 3.4 (Training Dynamics of SAE-CNN and SAE-FNO)**.** *The training dynamics of the SAE-FNO's frequency-domain weights* $\boldsymbol{W}_c^{(k+1)} = \boldsymbol{W}_c^{(k)} + \frac{\eta}{\sqrt{M}}\mathcal{F}\left(\boldsymbol{x} - \hat{\boldsymbol{x}}^{(k)}\right) \odot \mathcal{F}(\boldsymbol{z}_{c,t})$ *has an effective update in the original space, expressed as:* $\boldsymbol{D}_c'^{(k+1)} = \boldsymbol{D}_c'^{(k)} + \frac{\eta}{M}\left(\boldsymbol{x} - \hat{\boldsymbol{x}}^{(k)}\right) \star \boldsymbol{z}_{c,t}$*. If the architectures satisfy the inference equivalence condition (Prop. D.4, see Prop. D.5 for lifting), then the dynamics are equivalent to that of a SAE-CNN:* $\boldsymbol{D}_c^{(k+1)} = \boldsymbol{D}_c^{(k)} + \eta(\boldsymbol{x} - \sum_{i=1}^{C} \boldsymbol{D}_i^{(k)} * \boldsymbol{z}_{i,t}) \star \boldsymbol{z}_{c,t}$.

Proof. For a standard SAE-CNN, we learn a dictionary $\boldsymbol{D}$ by minimizing the reconstruction loss for the data sample $\boldsymbol{x}$. The loss function is:

$$\mathcal{L}(\boldsymbol{D}) = \frac{1}{2}\left\|\boldsymbol{x} - \sum_{c=1}^{C} \boldsymbol{D}_c * \boldsymbol{z}_c\right\|_2^2 \tag{51}$$

The gradient of the loss with respect to the c-th kernel $\boldsymbol{D}_c$ is:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{D}_c} = -\left(\boldsymbol{x} - \sum_{i=1}^{C} \boldsymbol{D}_i * \boldsymbol{z}_i\right) \star \boldsymbol{z}_c \tag{52}$$

where $\star$ denotes cross-correlation.

The analytic gradient update, given $\boldsymbol{z}$, for $\boldsymbol{D}_c$ at iteration k with a learning rate η is:

$$\boldsymbol{D}_c^{(k+1)} = \boldsymbol{D}_c^{(k)} + \eta\left(\boldsymbol{x} - \sum_{i=1}^{C} \boldsymbol{D}_i^{(k)} * \boldsymbol{z}_{i,t}\right) \star \boldsymbol{z}_{c,t} \tag{53}$$

For a standard SAE-FNO, we learn frequency-domain weights $\boldsymbol{W}_c = \mathcal{F}\boldsymbol{D}_c$ by minimizing the reconstruction loss for the data sample $\boldsymbol{x}$. Taking normalization into account, the reconstruction operator is $\hat{\boldsymbol{x}} = \mathcal{G}_{\boldsymbol{W}}(\boldsymbol{z}) = \frac{1}{\sqrt{M}}\mathcal{F}^{-1}\big(\sum_{c=1}^{C} \boldsymbol{W}_c \cdot (\mathcal{F}\boldsymbol{z}_c)\big)$, where $\boldsymbol{W}_c = \mathcal{F}\boldsymbol{D}_c$, where M is the number of modes. The loss function is:

$$\mathcal{L}(\boldsymbol{W}) = \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim\mu}\left[\|\boldsymbol{x} - \mathcal{G}_{\boldsymbol{W}}(\boldsymbol{z})\|_{\mathcal{X}}^2\right] \tag{54}$$

where $\boldsymbol{x} = \mathcal{G}_{\boldsymbol{D}^*}(\boldsymbol{z})$ is the ground-truth signal.

The gradient of the loss with respect to the c-th frequency-domain kernel $\boldsymbol{W}_c$ is:

$$\frac{\partial \mathcal{L}}{\partial \boldsymbol{W}_c} = -\frac{1}{\sqrt{M}}\mathcal{F}\left(\boldsymbol{x} - \hat{\boldsymbol{x}}\right) \odot \mathcal{F}(\boldsymbol{z}_c) \tag{55}$$

The analytic gradient update, given $\boldsymbol{z}$, for $\boldsymbol{W}_c$ at iteration k with a learning rate η is:

$$\boldsymbol{W}_c^{(k+1)} = \boldsymbol{W}_c^{(k)} + \frac{\eta}{\sqrt{M}}\mathcal{F}\left(\boldsymbol{x} - \hat{\boldsymbol{x}}^{(k)}\right) \odot \mathcal{F}(\boldsymbol{z}_{c,t}) \tag{56}$$

Applying the inverse Fourier transform and the cross-correlation theorem, we obtain the effective update rule for the spatial dictionary $\boldsymbol{D}_c'$:

$$\boldsymbol{D}_c'^{(k+1)} = \boldsymbol{D}_c'^{(k)} + \frac{\eta}{M}\left(\boldsymbol{x} - \hat{\boldsymbol{x}}^{(k)}\right) \star \boldsymbol{z}_{c,t} \tag{57}$$

This is equivalent to the gradient update of SAE-CNN with a scaling factor of $\frac{1}{M}$ on the learning rate, which arises from the inverse Fourier normalization conventions. ■

Proposition D.5 (Architectural Inference Equivalence of L-SAE-CNN and L-SAE-FNO)**.** *The architectural inference of a L-SAE-CNN is equivalent to a L-SAE-FNO if the integral operator of L-SAE-FNO and parameters of SAEs are defined as a convolution where the L-SAE-FNO's lifted frequency-domain weights* $\boldsymbol{W}_{L,c}$ *are the Fourier transform of L-SAE-CNN's lifted spatial domain kernels:* $\boldsymbol{W}_{L,c} = \mathcal{F}\boldsymbol{D}_{L,c}$.

Proof. We provide proof for an encoder that implements a proximal gradient descent type of operation in an unrolled learning setting. For the L-SAE-CNN, the representation refinement for the c-th feature map at every encoder layer follows:

$$\boldsymbol{z}_{c,t+1} = \mathcal{R}\left(\boldsymbol{z}_{c,t} - \alpha\left(\sum_{i=1}^{C} \boldsymbol{D}_{L,i} * \boldsymbol{z}_{i,t} - \boldsymbol{L}\boldsymbol{x}\right) \star \boldsymbol{D}_{L,c}\right) \tag{58}$$

By construction, we incorporate the underlying model from L-SAE-FNO to L-SAE-CNN: $\boldsymbol{W}_{L,c} = \mathcal{F}\boldsymbol{D}_{L,c}$, and re-write the gradient of the encoder using Correlation and Convolution Theorems:

$$\mathcal{F}^{-1}\left(\mathcal{F}\left(\sum_{i=1}^{C} \boldsymbol{D}_{L,i}^{(k)} * \boldsymbol{z}_{i,t} - \boldsymbol{L}\boldsymbol{x}\right) \odot \mathcal{F}\boldsymbol{D}_{L,c}^{H}\right) = \mathcal{F}^{-1}\left(\sum_{i=1}^{C} \boldsymbol{W}_{L,i}^{(k)} \odot \mathcal{F}\boldsymbol{z}_{i,t} - \mathcal{F}(\boldsymbol{L}\boldsymbol{x})\right) \odot (\boldsymbol{W}_{L,c})^{H} \tag{59}$$

Substituting this back into eq. (58) gives:

$$\boldsymbol{z}_{c,t+1} = \mathcal{R}\left(\boldsymbol{z}_{c,t} - \alpha\mathcal{F}^{-1}\left[\left(\left(\sum_{i=1}^{C} \boldsymbol{W}_{L,i} \odot \mathcal{F}(\boldsymbol{z}_{i,t})\right) - \mathcal{F}(\boldsymbol{L}\boldsymbol{x})\right) \odot (\boldsymbol{W}_{L,c})^{H}\right]\right) \tag{60}$$

This final expression is the L-SAE-FNO's iterative update for $\boldsymbol{z}_c$ in the lifted spectral domain, where the dictionary is $\boldsymbol{W}_{L,c}$. This completes the proof of architectural inference equivalence. ■

Proposition D.6 (Architectural Inference Equivalence of Lifting in SAE-FNO)**.** *The architectural inference of an SAE-FNO is equivalent to a Lifted SAE-FNO (L-SAE-FNO) if the projection is tied to the lifting operator ($\boldsymbol{P} = \boldsymbol{L}^{\top}$) that is orthogonal ($\boldsymbol{L}^{\top}\boldsymbol{L} = \boldsymbol{I}$), i.e., under the same learned model $\boldsymbol{W}_c = \boldsymbol{P}\boldsymbol{W}_{L,c}$, the representation dynamics and output of the networks are equivalent.*

Proof. We prove this equivalence by establishing a chain of architectural inference equivalences that connects the SAE-FNO to the Lifted SAE-FNO, leveraging the propositions previously established:

1. From Proposition D.4, the architectural inference of an SAE-FNO is equivalent to SAE-CNN.
2. From Proposition D.1, the architectural inference of an SAE-CNN is equivalent to L-SAE-CNN.
3. From Proposition D.5, the architectural inference of a L-SAE-CNN is equivalent to L-SAE-FNO.

By the transitive property of these equivalences, we can establish a direct architectural inference equivalence between SAE-FNO and L-SAE-FNO under the same lifting-projection conditions. ■