Visual Scene Representation with Hierarchical Equivariant Sparse Coding

Christian Shewmake SHEWMAKE@BERKELEY.EDU
Domas Buracas DOMINYKAS@BERKELEY.EDU
Hansen Lillemark HLILLEMARK@BERKELEY.EDU
Jinho Shin JHS0640@BERKELEY.EDU
Erik Bekkers E.J.BEKKERS@UVA.NL
Nina Miolane NINAMIOLANE@UCSB.EDU
Bruno Olshausen BAOLSHAUSEN@BERKELEY.EDU

Editors: Sophia Sanborn, Christian Shewmake, Simone Azeglio, Nina Miolane

Abstract

We propose a hierarchical neural network architecture for unsupervised learning of equivariant part-whole decompositions of visual scenes. In contrast to the global equivariance of group-equivariant networks, the proposed architecture exhibits equivariance to part-whole transformations throughout the hierarchy, which we term hierarchical equivariance. The model achieves these structured internal representations via hierarchical Bayesian inference, which gives rise to rich bottom-up, top-down, and lateral information flows, hypothesized to underlie the mechanisms of perceptual inference in visual cortex. We demonstrate these useful properties of the model on a simple dataset of scenes with multiple objects under independent rotations and translations.

Keywords: group equivariance, sparse coding, scene decomposition, unsupervised representation learning, geometric deep learning, disentanglement, bayesian generative model

1. Introduction

Understanding how to represent the rich structure in visual scenes has been a longstanding challenge in both deep learning and visual neuroscience. In the natural world, scenes can be described in terms of object identities and their poses (e.g. position, orientation, etc). Objects can be defined in terms of the relative poses of their parts, parts in terms of arrangements of sub-parts, and so on. Each of these components can also undergo rigid transformations or articulation of their parts. These families of *group transformations* can naturally be used to describe poses and relationships between parts and the objects to which they belong. Often considered a nuisance to object recognition, these variations instead carry important information for understanding and meaningfully interacting with the world. Thus, a rich *compositional hierarchy* that is compatible with group actions is essential for forming visual representations (Mumford and Desolneux, 2010; Olshausen et al., 1993; Hinton, 2022). However, since this generative structure is not directly observable, it must be *inferred* from sensory data and prior knowledge in order to form internal representations. For example, though objects and their parts have definite coordinates in a scene, this is not explicit in raw image pixel values.

Models such as Capsule Networks and Slot Attention leverage this perspective (Sabour et al., 2017; Ribeiro et al., 2022; Locatello et al., 2020; Biza et al., 2023). Capsule networks parse scenes into parts and wholes in a supervised context. Slot Attention is an unsupervised object-centric method which extracts objects into a fixed number of latent slots using self-attention and an iterative update. Yet, these methods have notable shortcomings. The proposed routing mechanism of standard Capsule networks has no clear normative basis, has prohibitive inference time, and requires supervision during training. The Slot Attention model represents all objects at the same level of the hierarchy, and stipulates a predefined number of object slots. This introduces challenges when presented with images varying in the number of objects.

Contribution. In this work, we propose a neural network architecture, Hierarchical Equivariant Sparse Coding (HESC), for unsupervised learning of visual scene representations that leverage **group transformations**, **hierarchical composition**, and **Bayesian inference**. We demonstrate that the model (1) learns a hierarchical decomposition of parts and wholes in terms of relative configurations of parts, (2) is equivariant to group actions throughout the visual hierarchy, and (3) forms sparse, explicit representations of parts and wholes via the explaining away machinery of Bayesian inference.

2. Proposed Framework

We seek a formal description of hierarchical transformations in visual scenes. We introduce group theory and its utility in modeling visual scenes, group equivariance in neural networks, and the need for hierarchical equivariance. We then describe how hierarchical part-whole decomposition motivates latent inference with Bayesian generative models.

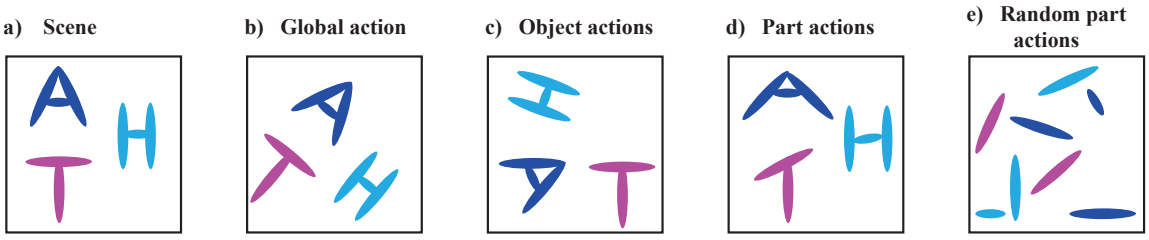


Figure 1: **(a)** Visual scenes are composed of objects $\{\mathcal{O}_i\}$ with corresponding group transformations $\{g_i\}$ applied **(b)** Global transformation g' applied to I distributes down to objects and their parts **(c)** Local object transformations h are applied to objects **(d)** A subset h' of local part transformations are applied, preserving object identity **(e)** A random subset of local part transformations h'' are applied. Full version in the appendix as Figure 6

2.1. Defining Global, Local, and Hierarchical Group Actions

As a guiding example, let us consider the simple scene in Figure 1. The configuration of each of the three objects in the scene can be described in terms of its pose: transformation from a default, canonical instance to its current position $\mathbf{x} = (x, y)$ and orientation θ. The set of transformations we can apply to a given object has the structure of a *group*. A group is a set of elements G, an identity element e, and a way of combining elements via a binary operation $\cdot$ which satisfies the *group axioms* (see Appendix A). For this discussion

we consider only the group of translations $(\mathbb{R}^2, +)$ and rotations $SO(2)$, the *rototranslation group*, defined here as the semidirect product $G = \mathbb{R}^2 \rtimes SO(2)$ [1]. Elements of this group, $g \in G$ have the form $g = (\mathbf{x}, \theta)$. The group's composition operator is:

$$g \cdot g' = (\mathbf{x}, \theta) \cdot (\mathbf{x}', \theta') = \left(R_\theta\left(\mathbf{x}'\right) + \mathbf{x}, \theta + \theta' \bmod 2\pi\right), \qquad \forall g, g' \in G$$

The scene I in Figure 1 can be fully described by specifying the collection of objects $(\mathcal{O}_1, \mathcal{O}_2, \mathcal{O}_3)$ and how they are transformed by group elements (g_1, g_2, g_3). When group elements transform other mathematical objects, they are referred to as *group actions* (see Appendix A). We denote the action of g_i on $\mathcal{O}_i$ as $T_{g_i}\mathcal{O}_i$. Thus, the scene is written as:

$$I = (T_{g_1}\mathcal{O}_1, T_{g_2}\mathcal{O}_2, T_{g_3}\mathcal{O}_3)$$

To transform all objects in the scene together, as seen in Figure 1(b), we apply a group transformation $g' \in G$ to the scene, which distributes to each of the objects. The composition of two group elements g' and g_i acting on $\mathcal{O}$ can be denoted as $T_{g'}T_{g_i}\mathcal{O}_i = T_{g'g_i}\mathcal{O}_i$.

$$T_{g'}I = (T_{g'g_1}\mathcal{O}_1, T_{g'g_2}\mathcal{O}_2, T_{g'g_3}\mathcal{O}_3)$$

We call this a *global action* because it acts equally on all objects in the scene. More generally, we can describe independent transformations on each object in the scene by transforming each object with its own group action, defined as $h = (h_1, h_2, h_3)$, as shown in Figure 1(c).

$$T_h I = (T_{h_1 g_1}\mathcal{O}_1, T_{h_2 g_2}\mathcal{O}_2, T_{h_3 g_3}\mathcal{O}_3)$$

We call this a *local action* because it acts independently on each object in the scene. Note that all global actions are a special case of local actions, in which the transformation on each object is the same, i.e. $h = (h_1, h_1, h_1)$. This new structure which combines multiple group elements into a tuple is called a *product group*, denoted $G^K = \oplus_{i=1}^{K} G$, where an element $h \in G^K$ is a K-tuple of group elements, $h = (h_1, h_2, ..., h_K)$. Product groups have a natural composition operation, i.e. given two elements $h, g \in G^K$: $h \cdot g = (h_1 g_1, h_2 g_2, ..., h_K g_K)$ which satisfies the group axioms. Note that both the transformations h of the scene as well as the configuration of all objects in the scene, g, are elements of the product group G^K. Just as scenes are composed of objects, the relative configuration of parts is what defines an object, and an object can be written as its parts $\{\mathcal{P}_i\}$[2] in an analogous way to the scene I:

$$\mathcal{O}_1 = (T_{g_1}\mathcal{P}_1, T_{g_2}\mathcal{P}_2, ..., T_{g_K}\mathcal{P}_K)$$

Using the same formulation as above, we can generalize this concept to any level in the visual hierarchy, relating images to objects, objects to parts, parts to subparts, and so on. There is a space of actions on parts of an object which deform the object, but preserve the relative configuration so it is still identifiable (Figure 1(d)), however this is a small subspace of all actions on parts (Figure 1(e)), demonstrating the combinatorial explosion of unstructured scenes. For simplicity and to avoid discussion of sub-parts and sub-sub-parts, we use the terms 'objects' and 'parts' to represent any arbitrary layer and sub-layer in the hierarchy. We refer to this extension of actions defined over an arbitrary number of compositional layers as *hierarchical actions*.

1. Groups, however, are general mathematical objects which can describe a large family of variations in data beyond simple spatial transformations.
2. Our model represents parts $\mathcal{P}_j$ and objects $\mathcal{O}_i$ in canonical poses with dictionary elements in different layers: $\phi_1^j(e)$ and $\phi_2^i(e)$, respectively.

2.2. Global and Hierarchical Group Equivariance

We have framed the problem of scene representation in terms of product groups and hierarchical group transformations. How should a neural network's activations $f(I)$ change when the input scene I changes? A desirable property, *group equivariance*, is that group transformations on the scene are represented by group transformations on activations:

$$f(T_g I) = T'_g f(I) \quad \forall g \in G$$

where T_g and T'_g are different actions of the same $g \in G$. In other words, the action T_g of the input image I should be represented by an analogous transformation (or homomorphism) T'_g on the model's latent representation. A model is said to be *group invariant* if a transformation on the image leaves the model's output unchanged:

$$f(T_g I) = f(I) \quad \forall g \in G$$

The most prevalent strategy for constructing group equivariant neural networks is through the use of *group convolutions*, an extension of the standard 2D translational CNNs to groups such as rotation and scaling, termed GCNNs (Cohen and Welling, 2016). Though these methods have seen success on a variety of tasks, so far they have been limited to the very specific case of *global* group actions on the domain of the input data, e.g., translations or rotations of an *entire* image $T_g f(I(x)) = f(I(g^{-1}x))$. As demonstrated above, much of the variation in natural images cannot be described with global actions directly on the image pixel values. This motivates the need to construct models which are equivariant to local and hierarchical actions, or *hierarchically equivariant*.

2.3. Hierarchical Bayesian Inference

Using the notion of local and hierarchical equivariance, we aim to construct models with equivariance to part-whole transformations. In order to form internal representations of multiple objects and their poses, a model must solve the *what-where* decomposition problem for all components of the scene. Further, as an image is merely a collection of pixel values, the presence and pose of an object are latent variables which have to be inferred from pixel data given prior knowledge. How does the brain solve this problem? Evidence from visual neuroscience suggests that the brain leverages both bottom-up and top-down dynamics to form visual representations, a perspective known as *perception as inference*. Lee and Mumford, among others, suggested Hierarchical Bayesian Inference as a mechanism for neural scene decomposition (Lee and Mumford, 2003; Olshausen and Lewicki, 2014). Mirroring the structure of visual cortex, they proposed a model with multiple layers interconnected in a hierarchy. Each layer receives inputs from the layer below as well as the layer above, in addition to recurrent connections within a layer.

What is the functional role of these three pathways? Consider an image of a face where the right side of the face is well lit, but the left side is shadowed and has low contrast with the background. For a given layer, the role of the bottom-up pathway is to provide proposals for candidate parts in a scene. Using only feed-forward bottom-up information, there would be strong evidence for face parts on the right side, but not on the left. However if the next layer has prior knowledge of faces as objects, it can be combined with information from the bottom-up pathway to gather evidence for the existence of a whole face. This can then be passed back to the lower layer as a top-down signal that can strengthen the previously weak

evidence for parts on the left side of the face. Finally, there is the problem of some parts having multiple probable explanations given evidence from both bottom-up and top-down information. This role is taken up by lateral interactions within a layer, which provide a mechanism for *explaining away*, which precludes one probable hypothesis in favor of another, more parsimonious explanation.

There is ubiquitous evidence in perceptual psychology and visual neuroscience for the presence and effects of these three pathways in the visual cortex. However, current feed-forward deep learning architectures lack the computational primitives for hierarchical Bayesian inference as described above. These three interactions, implicated in forming part-whole representations—bottom-up, top-down, and lateral interactions—emerge directly from *hierarchical sparse coding*, a Bayesian generative model which provides a normative theory for the formation of visual representations (Olshausen et al., 2014). We begin by describing the classical sparse coding model, follow with its geometric extension *equivariant sparse coding*, and then introduce *hierarchical equivariant sparse coding*.

3. Hierarchical Equivariant Sparse Coding

3.1. Sparse Coding

Sparse coding was originally proposed as a model for how neurons in primary visual cortex represent image data coming from the retina. As a linear **generative model**, it assumes that images can be represented as $I = \sum_i \phi^i a^i + \epsilon$, a sparse linear combination of learned basis functions, or dictionary elements $\Phi = \{\phi^1, \phi^2, ..., \phi^D\}$, with noise ϵ (e.g. Gaussian) and a sparsity-inducing **prior** on the latent variables a (e.g. Laplace). The energy function of the model arises from computing the maximum a posteriori (MAP) estimate using the likelihood and the prior, which is equivalent to computing the mode of the posterior distribution, $P(a|I;\Phi)$: $\min_a \frac{1}{2}||I - \hat{I}||^2_{X,2} + \lambda||a||_1$. Inference gives rise to recurrent dynamics in latent variables a that implement *explaining away*. Popular inference methods include ISTA (Daubechies et al., 2003), FISTA (Beck and Teboulle, 2009), and LCA (Rozell et al., 2008).

3.2. Group Equivariant Sparse Coding

The traditional sparse coding model provides no mechanism to reason about *configurations* or *transformations* of components in a scene, e.g. position, orientation, and scale, since the dictionary elements ϕ_i form an unordered collection of vectors (see Figure 2(b)).

However, by equipping the generative model with *manifold* structure via continuous group actions, dictionary elements and coefficients are endowed with explicit pose information, shown by (Shewmake et al., 2023). Specifically, one can replace the unordered set of dictionary elements in traditional sparse coding with a set of C *canonical dictionary elements* $\{\phi^1(e), \phi^2(e), ..., \phi^C(e)\}$, where each $\phi^c(e) : \mathbb{R}^2 \to \mathbb{R}$ is a function in image space. The full dictionary Φ is generated via *group actions* on each canonical dictionary element, written formally as $T_g\phi^c(e)(x) = \phi^c(e)(g^{-1}x)$[3], for $g \in G$ and the identity transformation denoted as $e \in G$.

3. Evaluating a g-transformed function at a point x requires evaluating the original function at point $g^{-1}x$

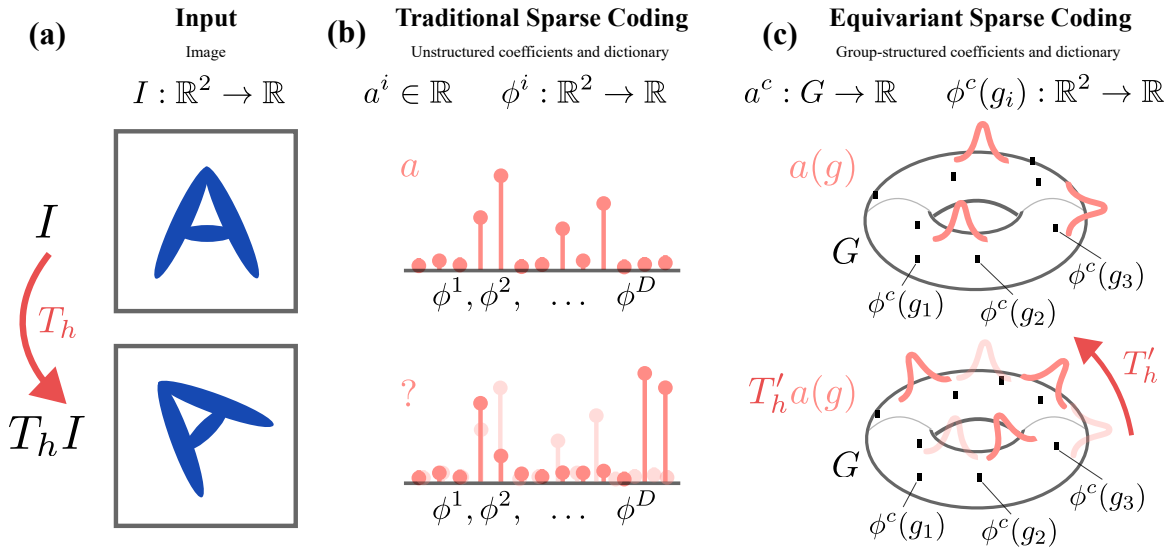


Figure 2: **(a)**: An input image is transformed by group action T_h. **(b)**: Traditional SC has unstructured coefficients. **(c)**: ESC activations transform equivariantly (via action T'_h) over the manifold G, shown on the torus for visualization.

The group G can denote translation, rotation, scaling, or a more general family of transformations. Here, we denote a dictionary element with its coordinate g for simplicity: $\phi^c(g) = T_g \phi^c(e)$. The full generated dictionary is $\Phi = \{\phi^c(g) : \forall c \in \{1, \ldots, C\}, \forall g \in G\}$.

Importantly, each latent variable corresponds to a dictionary element, and thus the coefficients inherit the same group structure, denoted now as $a^c(g)$. The manifold geometry of coefficient activity and a visualization of equivariance is shown in Figure 2(c). Images are now generated by a linear combination of dictionary elements with coordinates $g \in G$.

$$I(x) = \sum_{c=1}^{C} \sum_{g \in G} \phi^c(g)(x) a^c(g) + \epsilon(x) \quad \forall x \in \mathbb{R}^2 \tag{1}$$

3.3. Hierarchical Equivariant Sparse Coding

The first layer of equivariant sparse coding performs a decomposition of a scene into a sparse set of primitive parts (canonical dictionary elements) and their corresponding poses. This representation provides a natural way to combine lower-level parts into objects in the next layer to achieve hierarchical scene decomposition. Similar to the one-layer case, we define a set of C_2 canonical dictionary elements in the second layer $\{\phi_2^1(e), \phi_2^2(e), ..., \phi_2^{C_2}(e)\}$[4]. Dictionary elements are again generated by group actions, and are thus equivariant to global, local, and hierarchical transformations (see Figure 3). The input to this layer is a set of group coordinates from layer 1, so each dictionary element is defined over G instead of the $\mathbb{R}^2$ image space: $\phi_2^{c_2}(g) : G \to \mathbb{R}^{C_1}$.

4. For multiple layers, we subscript for layer index l, and superscript for canonical dictionary index c_l: $\phi_l^{c_l}$

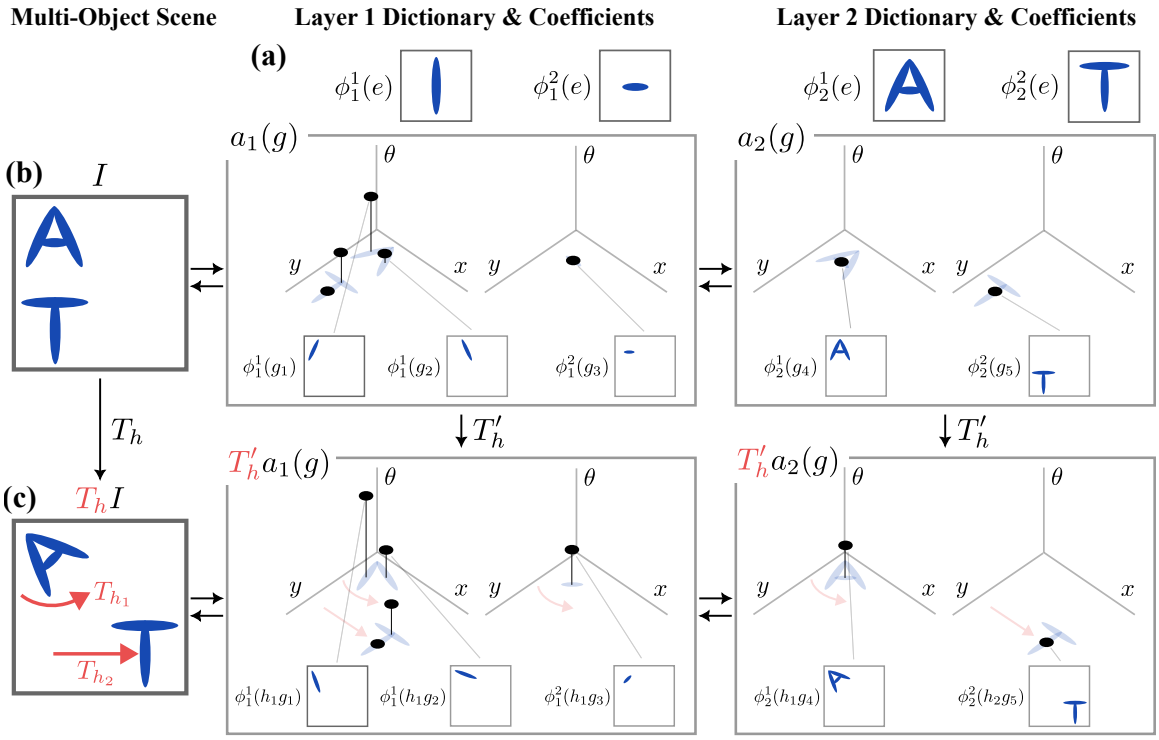


Figure 3: **(a)** Illustration of the model's dictionary elements and latent representation. Dictionary elements correspond to parts (ϕ_1^1, ϕ_1^2) and objects (ϕ_2^1, ϕ_2^2) of the scene. **(b)** The model infers a latent representation of parts and objects, decomposing a multi-object scene. Each non-zero coefficient is a point with a pose in the rototranslation group. **(c)** After an action $T_h = (T_{h_1}, T_{h_2})$ is applied to the multi-object scene, inferred coefficients transform equivariantly via $T'_h = (T'_{h_1}, T'_{h_2})$.

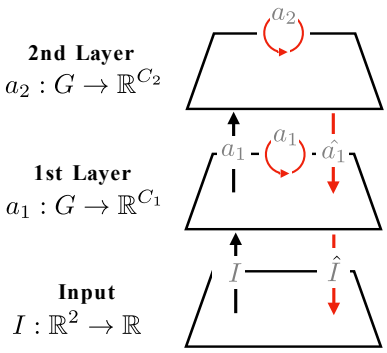
Images are generated by a linear combination of hierarchical dictionary elements with explicit coordinates $g \in G$.

$$\text{First-level } (C_1 \text{ Parts}): \quad I(x) = \sum_{c_1=1}^{C_1} \sum_{g \in G} \phi_1^{c_1}(g)(x) a_1^{c_1}(g) + \epsilon(x) \quad \forall x \in \mathbb{R}^2 \tag{2}$$

$$\text{Second-level } (C_2 \text{ Objects}): \quad a_1(g) = \sum_{c_2=1}^{C_2} \sum_{g' \in G} \phi_2^{c_2}(g')(g) a_2^{c_2}(g') + \epsilon(g) \quad \forall g \in G \tag{3}$$

The corresponding layer 2 coefficient activations $a_2^{c_2}(g)$ inherit the same group structure as corresponding dictionary elements $\phi_2^{c_2}(g)$. This could be extended to an arbitrary layer l, which would have the same structure for $\phi_l^{c_l}(g)$ and $a_l^{c_l}(g)$ based on input from layer $l-1$. We generate reconstructions $\hat{I}(x)$ and $\hat{a}_1(g)$ using the generative model in (2) and (3).

Inference. The inference mechanism follows directly from computing the gradient of the hierarchical energy function E, shown in Figure 4. The implementation of the architecture and inference is provided in Appendix C.



$$E = \frac{1}{2}\|I - \hat{I}\|_2^2 + \lambda_1\|a_1\|_1 + \frac{1}{2}\|a_1 - \hat{a}_1\|_2^2 + \lambda_2\|a_2\|_1$$

$$\nabla_{a_1} E = -\underbrace{\Phi_1^T I}_{\text{bottom-up}} + \underbrace{\Phi_1^T\Phi_1 a_1 + \lambda_1 sign(a_1)}_{\text{lateral}} - \underbrace{\Phi_2 a_2 + a_1}_{\text{top-down}}$$

$$\nabla_{a_2} E = -\underbrace{\Phi_2^T a_1}_{\text{bottom-up}} + \underbrace{\Phi_2^T\Phi_2 a_2 + \lambda_2 sign(a_2)}_{\text{lateral}}$$

Figure 4: Hierarchical Energy and Gradients: **Bottom-up** terms activate coefficients corresponding to dictionary elements matching the image well. **Lateral** terms implement interactions between coefficients representing similar or overlapping parts while pushing coefficients towards zero. This results in a sparse part decomposition. **Top-down** term(s) promote the presence of parts in lower layers compatible with wholes in the above layers

4. Results

The assumption of our model is that natural images can be efficiently described in terms of a hierarchical decomposition of wholes into parts, where components are configured according to group actions. Here we show that, when trained on a dataset containing objects under independent group transformations, the model **(1)** successfully recovers the ground truth hierarchical dictionary, **(2)** is equivariant to hierarchical group actions, and **(3)** implements explaining away to produce sparse latent object representations.

Experimental Design. To test ground truth dictionary recovery, we construct a dataset with known hierarchical and group transformation structure. We use a single Gabor as a first-level part. Next, we construct two digits by arranging the first-level parts with poses in the rototranslation group. The ground-truth part and objects are shown in Figure 5. We create multi-object scenes by sampling one object choice per quadrant in a random pose with 5×5 possible spatial positions and 4 possible orientations. Using this model, there are $(|G| \times 2 \text{ objects})^4 = (5 \times 5 \times 4 \times 2)^4 = 1.6$ billion unique scenes with four objects. Our training set is a collection of 1000 such random multi-object scenes. We train a two layer model with one canonical dictionary element in layer one and two in layer two (i.e. $C_1 = 1, C_2 = 2$). The goal during training is for the model to successfully recover the canonical Gabor function in layer 1, and to recover each of the two digits in layer 2.

Findings. First, after unsupervised training, the model **(1) successfully recovers hierarchical dictionary elements**, as shown in Figure 5(c). Note that the layer 1's canonical dictionary element $\phi_1^1(e)$ converges to a component which sparsely decomposes the objects in the dataset, and the trained layer 2's canonical dictionary elements $\phi_2^1(e)$ and $\phi_2^2(e)$ recover the digits 1 and 3. Second, due to our unsupervised setup and hierarchical model, we are able to perform **(2) scene decomposition** on multi-object scenes, generalizing from extremely few samples. Note that through inference, the model has formed a what-where decomposition of all objects in the scene in parallel. This can be observed in the second

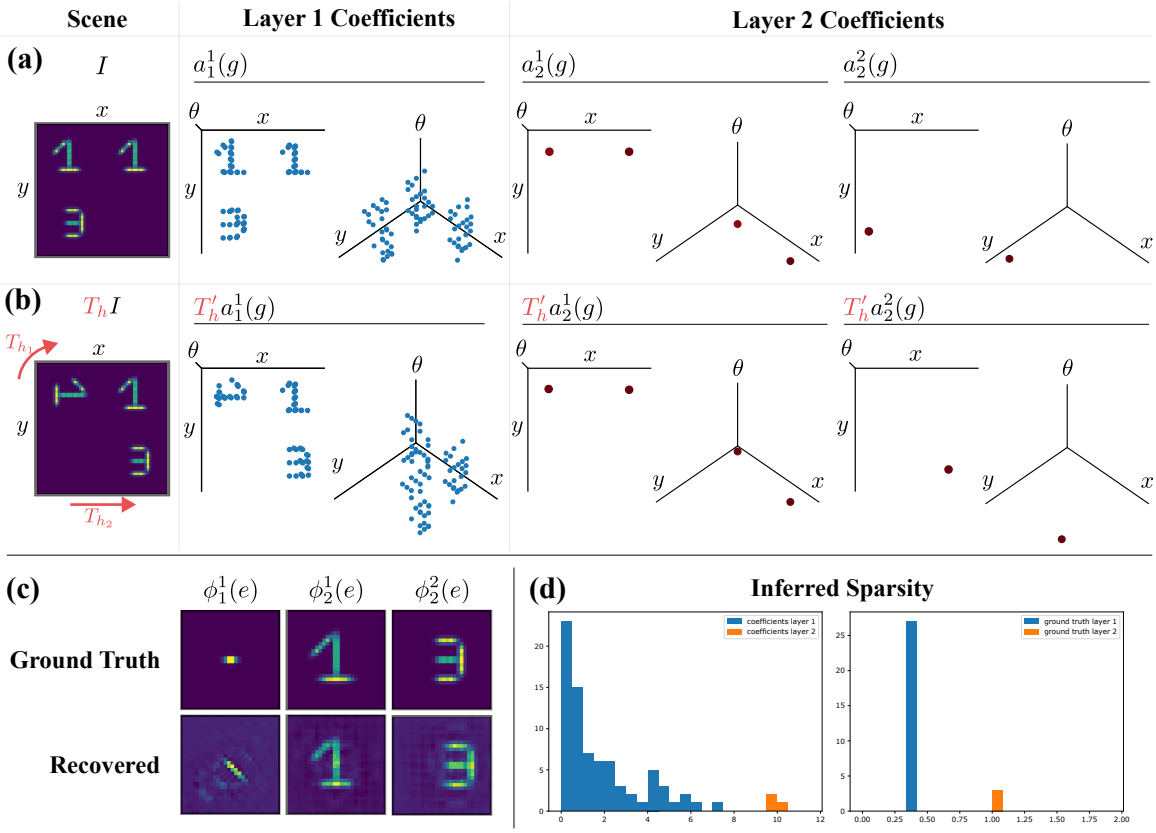


Figure 5: **(a)** Synthesized scene image. Inferred coefficients correspond to presences of parts and objects in the scenes. **(b)** Image transformation by T_h is composed of independent actions on objects in the scene. Inferred part and object coefficients transform equivariantly, via T_h'. **(c)** Recovered and ground truth dictionaries for layers 1 and 2. **(d)** Histograms of inferred and ground-truth dataset generation coefficient magnitudes.

layer's latent activations in Figure 5(a), which have excitations corresponding to each object and its corresponding pose in the rototranslation group. Further, observe that this same what-where decomposition occurs in the first layer as well, in which each Gabor function present in the scene is represented along with its pose. Third, the model is **(3) equivariant to hierarchical transformations**. Critically, when objects in the image undergo independent group transformations, the latent variables in both the first and second layers also undergo independent group transformations, shown in Figure 5(b). Finally, the latent representations are **(4) sparse in both layers due to explaining away**, with increasing sparsity in the second layer as compared to the first, shown in Figure 5(d).

5. Discussion and Future Work

Our current model provides a proof of concept, trained on a synthetic dataset to clearly demonstrate the learned dictionary and equivariance properties of the model's represen-

tation. The first steps going forward are to study the learned object decompositions and empirical equivariance when trained on larger datasets with deeper hierarchical structure, such as the Hangul characters, CIFAR, and natural images (Krizhevsky et al., 2009; Livezey et al., 2019; Sun et al., 2020). To do this, we will expand the architecture to deeper layers, more canonical dictionary elements, and more expressive group actions. With regard to applications, we expect the model's performance on a variety of tasks will benefit from meaningful latent representations such as classification, scene decomposition, object detection, visual question answering, and compression of images and video. We expect to compare favorably to existing models in terms of adversarial robustness, sample complexity, parameter count, and computational cost. Finally, we plan to explore the implications of our model on visual neuroscience, for example, in emergent phenomena such as perceptual grouping, the dynamics of bistable percepts, and contour completion.

Acknowledgments

The authors thank their colleagues at the Redwood Center and Geometric Intelligence Lab, especially for their careful review of the original manuscript. CS acknowledges support from the NIH NEI Training Grant T32EY007043 and the NSF CISE Robust Intelligence Grant 22-631.

References

Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sci.*, 2:183–202, 2009. URL https://api.semanticscholar.org/CorpusID:3072879.

Ondrej Biza, Sjoerd van Steenkiste, Mehdi S. M. Sajjadi, Gamaleldin F. Elsayed, Aravindh Mahendran, and Thomas Kipf. Invariant slot attention: Object discovery with slot-centric reference frames, 2023.

Victor Boutin, Angelo Franciosini, Frederic Chavane, Franck Ruffier, and Laurent Perrinet. Sparse deep predictive coding captures contour integration capabilities of the early visual system. (arXiv:1902.07651), Oct 2019. doi: 10.48550/arXiv.1902.07651. URL http://arxiv.org/abs/1902.07651. arXiv:1902.07651 [cs].

Taco Cohen and Max Welling. Group equivariant convolutional networks. In *International conference on machine learning*, pages 2990–2999. PMLR, 2016.

Ingrid Daubechies, Michel Defrise, and Christine De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint, 2003.

Geoffrey Hinton. How to represent part-whole hierarchies in a neural network. *Neural Computation*, pages 1–40, 2022.

David M. Knigge, David W. Romero, and Erik J. Bekkers. Exploiting redundancy: Separable group convolutional networks on lie groups. In *International Conference on Machine Learning*, page 11359–11386. PMLR, 2022.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

T S Lee and D Mumford. Hierarchical bayesian inference in the visual cortex. *JOSA A*, 20 (7):1434–1448, 2003.

Jesse A Livezey, Ahyeon Hwang, and Kristofer E Bouchard. Hangul fonts dataset: a hierarchical and compositional dataset for interrogating learned representations. *arXiv preprint arXiv:1905.13308*, 2019.

Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. *ArXiv*, abs/2006.15055, 2020. URL https://api.semanticscholar.org/CorpusID:220127924.

David Mumford and Agnès Desolneux. *Pattern theory: the stochastic analysis of real-world signals*. CRC Press, 2010.

B A Olshausen and M S Lewicki. What natural scene statistics can tell us about cortical representation. *The New Visual Neurosciences*, 2014.

Bruno A. Olshausen, Charles H. Anderson, and David C. Van Essen. A neurobiological model of visual attention and invariant pattern recognition based on dynamic routing of information. *Journal of Neuroscience*, 13(11):4700–4719, 1993.

Bruno A Olshausen, GR Mangun, and MS Gazzaniga. Perception as an inference problem. *The cognitive neurosciences*, page 295, 2014.

Fabio De Sousa Ribeiro, Kevin Duarte, Miles Everett, Georgios Leontidis, and Mubarak Shah. Learning with capsules: A survey. *ArXiv*, abs/2206.02664, 2022. URL https://api.semanticscholar.org/CorpusID:249394871.

David W. Romero, Anna Kuzina, Erik J. Bekkers, Jakub M. Tomczak, and Mark Hoogendoorn. Ckconv: Continuous kernel convolution for sequential data. *arXiv preprint arXiv:2102.02611*, 2021.

Christopher J. Rozell, Don H. Johnson, Richard G. Baraniuk, and Bruno A. Olshausen. Sparse coding via thresholding and local competition in neural circuits. *Neural computation*, 20(10):2526–2563, 2008.

Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. *arXiv preprint arXiv:1710.09829*, 2017.

Christian Shewmake, Nina Miolane, and Bruno Olshausen. Group equivariant sparse coding. In *Geometric Science of Information (GSI)*, 2023.

Vincent Sitzmann, Julien N. P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *ArXiv*, abs/2006.09661, 2020. URL https://api.semanticscholar.org/CorpusID:219720931.

Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, Vijay Vasudevan, Wei Han, Jiquan Ngiam, Hang Zhao, Aleksei Timofeev, Scott Ettinger, Maxim Krivokon, Amy Gao, Aditya Joshi, Sheng Zhao, Shuyang Cheng, Yu Zhang, Jonathon Shlens, Zhifeng Chen, and Dragomir Anguelov. Scalability in perception for autonomous driving: Waymo open dataset, 2020.

Appendix A. Mathematical Background

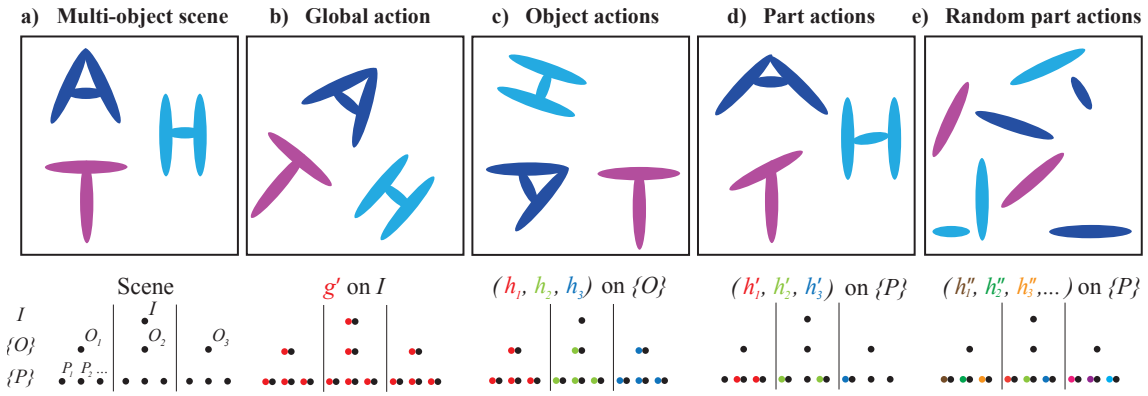


Figure 6: A version of Figure 1 with an additional parse tree visualization corresponding to the image. **(a)** Visual scenes are composed of objects $\{\mathcal{O}_i\}$ with corresponding group transformations g_i applied **(b)** Global transformation of $g' = (g, g, g)$ on I distributes down to objects and their parts. **(c)** Local object transformations $h = (h_1, h_2, h_3)$ are applied to objects. Transformations distribute down to parts, and relative part configurations are conserved. **(d)** A subset of local part transformations $h' = (h'_1, h'_2, h'_3)$ are applied which conserve relative part configurations enough to preserve object identity **(e)** A random subset of local part transformations $h'' = (h''_1, h''_2, h''_3)$ are applied which do not preserve object identity. The space of possible scenes without object and hierarchical structure is much larger than scenes with object and hierarchical structure.

Groups. A *group* $(G, \cdot)$ is a set G with a binary operation $\cdot$, which we can generically call the *product*. The notation $a \cdot b$ denotes the product of two elements in the set; however, it is standard to omit the operator and write simply ab. Concretely, a group G may define a class of transformations, such as two-dimensional translations or rotations in the plane. The elements of the group $g \in G$ define *particular* transformations, such as *rotation by* 30 or *rotation by* 90. The binary operation $\cdot$ provides a means for combining two particular transformations —for example, first rotating by 30 and then rotating by 90. For a set of transformations G to be a group under the operation $\cdot$, the four following axioms must hold:

1. *Closure*: The product of any two elements of the group is also an element of the group, i.e. for all $a, b \in G$, $ab \in G$.

2. *Associativity*: The grouping of elements under the operation does not change the outcome, so long as the order of elements is preserved, i.e. $(ab)c = a(bc)$.

3. *Identity*: There exists a "do-nothing" *identity* element e that such that the product of e with any other element g returns g, i.e. $ge = eg = g$ for all $g \in G$.

4. *Inverse*: For every element g, there exists an *inverse* element g^{-1} such that the product of g and g^{-1} returns the identity, i.e. $gg^{-1} = g^{-1}g = e$.

Homomorphisms. Two groups $(G, \cdot)$ and $(H, *)$ are *homomorphic* if there exists a correspondence between elements of the groups that respect the group operation. Concretely, a *homomorphism* is a map $\rho : G \to H$ such that $\rho(u \cdot v) = \rho(u) * \rho(v)$. An *isomorphism* is a bijective homomorphism.

Product Groups. A *product group* as a set is the cartesian product $G \times H$ of two given groups G and H. The new group product is given by $(g_1, h_1) \cdot (g_2, h_2) = (g_1 g_2, h_1 h_2)$.

Commutativity. A group $(G, +)$ is *commutative* or *abelian* if the order of operations does not matter, i.e. $ab = ba$. If this does not hold for all elements of the group, then the group is called *non-commutative*. The classification of finite commutative groups says that each such group is a product of cyclic groups.

Group Actions. A *group action* is a map $T : G \times X \to X$ that maps (g, x) pairs to elements of X. We say a group G *acts* on a space X if the following properties of the action T hold:

1. The identity $e \in G$ maps an element of $x \in X$ to itself, i.e. $T(e, x) = x$
2. Two elements $g_1, g_2 \in G$ can be combined before or after the map to yield the same result, i.e. $T(g_1, T(g_2, x)) = T(g_1 g_2, x)$

For simplicity, we will use the shortened notation $T_g x$ to denote $T(g, x)$, often expressed by saying that a point x maps to gx $(= T_g(x))$.

Invariance. A function $\phi : X \mapsto Y$ is *G-invariant* if $\phi(x) = \phi(gx)$ for all $g \in G$ and $x \in X$. This means that group actions on the input space have no effect on the output.

Equivariance. A function $f : X \mapsto Y$ is *G-equivariant* if $f(gx) = g' f(x)$ for all $g \in G$ and $x \in X$, with $g' \in G'$, a group homomorphic to G that acts on the output space. This means that a group action on the input space results in a corresponding group action on the output space.

Orbits. Given a point $x \in X$, the *orbit* Gx of x is the set $\{gx : g \in G\}$. In the context of image transformations, the orbit defines the set of all transformed versions of a canonical image—for example, if G is the group of translations, then the orbit contains all translated versions of that image.

Appendix B. Group Convolution: Theory and Computation

Group Convolutions (Lifting) Consider an image I defined on a domain X on which a group G acts. A neural network convolutional filter is a map $\psi : X \to \mathbb{R}^c$ defined with the same domain X and codomain $\mathbb{R}^c$ as the image. A G-convolutional layer is defined by a set of filters $\{\psi_1, ..., \psi_K\}$. For a given filter k, the layer performs a *G-convolution* with the input signal I:

$$\Theta_k(g) = (\psi_k * I)(g) = \int_{x \in X} \psi_k(T_{g^{-1}}(x)) I(x) dx, \quad \forall g \in G, \tag{4}$$

by taking the dot product in $\mathbb{R}^c$ of the signal with a transformed version of the filter. In practice, the domain X of the signal is discretized, such that the G-convolutional layer becomes:

$$\Theta_k(g) = \sum_{x \in X} \psi_k(T_{g^{-1}}(x)) I(x), \quad \forall g \in G. \tag{5}$$

The output of one filter k is therefore a map $\Theta_k : G \to \mathbb{R}$, while the output of the whole layer with K filters is $\Theta : G \to \mathbb{R}^K$ defined as $\Theta(g) = [\Theta_1(g), \ldots, \Theta_K(g)]$ for all $g \in G$. The G-convolution therefore outputs a signal Θ whose domain has necessarily become the group $X = G$ and whose number of channels is the number of convolutional filters K. We call this convolution a lifting G-convolution. The G-convolution is *equivariant* to the action of the group on the domain of the image I (Cohen and Welling, 2016).

Group Convolutions (Homogeneous) After the first layer, the signal has become a function $f : G \to \mathbb{R}^K$. Thus, the definition of the G-convolution slightly changes: we call it a homogeneous group convolution. For a given filter defined with a domain being a group $\phi_k : G \to \mathbb{R}^K$, we have:

$$\Theta'_k(g) = (\psi_k * \Theta)(g) = \int_{h \in G} \psi_k(g^{-1}h)\Theta(h)dh, \quad \forall g \in G, \tag{6}$$

where instead of using the action of the group G on the domain, we use its "natural" action on itself: the binary operation $\cdot$ of the group. The discretized version is written:

$$\Theta'_k(g) = \sum_{h \in G} \psi_k(g^{-1}h)\Theta(h), \quad \forall g \in G. \tag{7}$$

Appendix C. Architecture

Dictionary Elements. We parameterize canonical dictionary elements in the first and second layers of the network using images and Sinusoidal Representation Networks (SIREN) (Sitzmann et al., 2020), similar to (Romero et al., 2021) and (Knigge et al., 2022). In the first layer, a set of canonical dictionary element images with learnable weights Θ are billinearly sampled at input coordinates $x \in \mathbb{R}^2$, returning that coordinate's pixel value $\text{INTERPNET}^1_\Theta(x) \in \mathbb{R}^{C_0}$, where $C_0 = 1$ for grayscale and $C_0 = 3$ for RGB. Here, $\text{SIREN}^1_\Theta : \mathbb{R}^2 \to \mathbb{R}^{C_0}$. Dictionary elements in the second layer are generated by a SIREN over the group $\text{SIREN}^2_\Theta : G \to \mathbb{R}^{C_1}$.

$$\phi_1(e)(x) = \text{INTERPNET}^1_\Theta(x) \qquad \phi_2(e)(g) = \text{SIREN}^2_\Theta(g)$$

Both first and second layer dictionary elements share the same support as the image domain and the group domain, $H \times W$ and $|G|$, respectively.

Group Actions on Dictionary Elements. This parameterization simplifies the generation of transformed instances of the canonical dictionary elements, as the group action can be applied to the INTERPNET or SIREN's input coordinate grid to generate a smooth family of transformed dictionary elements without need for interpolation.

$$\phi_1(g)(x) = \text{INTERPNET}^1_\Theta(g^{-1}x) \qquad \phi_2(g')(g) = \text{SIREN}^2_\Theta(g'^{-1}g)$$

In all experiments, we use the rototranslation group $G = \mathbb{R}^2 \rtimes SO(2)$. We compute the full dictionary in layers one and two by transforming each of the canonical dictionary elements with a uniform grid of group actions in G, denoted $\bar{G}_1$ and $\bar{G}_2$, respectively.

$$\Phi_1 = \{\phi_1^{c_1}(g)(x) : \forall c_1 \in \{1, \dots, C_1\}, \ \forall g \in \bar{G}_1\}$$

$$\Phi_2 = \{\phi_2^{c_2}(g)(x) : \forall c_2 \in \{1, \dots, C_2\}, \ \forall g \in \bar{G}_2\}$$

We do this for $N_{x,1}, N_{x,2}$ horizontal translations, $N_{y,1}, N_{y,2}$ vertical translations, and $N_{\theta,1}, N_{\theta,2}$ rotations, thus $|\bar{G}_1| = N_{x,1} \times N_{y,1} \times N_{\theta,1}$ and $|\bar{G}_2| = N_{x,2} \times N_{y,2} \times N_{\theta,2}$. This is functionally equivalent to group convolution, described in Appendix B.

Hierarchical Inference. For fast inference, we use an adapted version of FISTA (Beck and Teboulle, 2009). Our adaptation updates all layers' activations simultaneously over a given number of iterations, consistent with Lee and Mumford (2003). We explain this in more detail in Appendix D.

Dictionary Learning. As in traditional sparse coding, we alternate inference of the latents a and a dictionary update step for both layers using back-propagation through the dictionary function parameterization.

Appendix D. Model Training and Inference Details

For the experiments described in this paper, we used a two-layer hierarchical group equivariant sparse coding model. We see the hierarchical generative model (Equations 2, 3), energy, and gradients (Figure 4). Here we describe details of the FISTA algorithm for hierarchical sparse coding inference, the parameterization of our canonical dictionary functions using SIREN implicit networks, and hyperparameters we used.

D.1. Hierarchical Inference with FISTA

For the inference of our algorithm, we use an adapted version of the FISTA (Beck and Teboulle, 2009; Boutin et al., 2019) optimizer, due to its fast rate of convergence relative to others. Our adaptation couples the inference of both layers by including a top-down gradient term, as well as updating each layer in a simultaneous fashion.

Our hierarchical loss objective is:

$$\begin{aligned}\min_a L(I, \Phi, a) &= \min_a -\log(P(a|I:\Phi)) \\ &= \min_a \frac{1}{2}\|I - \Phi_1 a_1\|_2^2 + \lambda_1\|a_1\|_1 + \frac{1}{2}\|a_1 - \Phi_2 a_2\|_2^2 + \lambda_2\|a_2\|_1\end{aligned}$$

Its gradients are:

$$\nabla_{a_1} L(I, \Phi, a) = \underbrace{\Phi_1^T(\Phi_1 a_1 - I)}_{\text{reconstruction gradient}} + \underbrace{a_1 - \Phi_2 a_2}_{\text{top-down gradient}} + \underbrace{\lambda_1 sign(a_1)}_{\text{sparsity gradient}}$$

$$\nabla_{a_2} L(\Phi, a) = \underbrace{\Phi_2^T(\Phi_2 a_2 - a_1)}_{\text{reconstruction gradient}} + \underbrace{\lambda_2 sign(a_2)}_{\text{sparsity gradient}}$$

The resulting adapted FISTA algorithm is shown in Algorithm 1

Algorithm 1: Hierarchical inference with FISTA

Input: I in dataset, sparsity parameters λ_1, λ_2, step sizes η_1, η_2, number of steps T

$a^0, a^1 \leftarrow 0$;

$m_1 \leftarrow 1$;

for $t = (1, 2, ..., T-1)$ **do**

  $m_{t+1} \leftarrow \frac{1+\sqrt{1+4m_t^2}}{2}$; $\beta \leftarrow \frac{m_t - 1}{m_{t+1}}$;

  $u \leftarrow a^t + \beta(a^t - a^{t-1})$;

  $a_1^{t+1} \leftarrow \mathcal{T}_{\lambda_1 \eta_1}(a_1^t - \eta_1 \nabla_{a_1} L(I, \Phi, u))$;

  $a_2^{n+1} \leftarrow \mathcal{T}_{\lambda_2 \eta_2}(a_2^t - \eta_2 \nabla_{a_2} L(\Phi, u))$;

end

Result: a^N

The nonnegative soft-thresholding operator is $\mathcal{T}_{\lambda\eta}(x) = relu(x - \lambda\eta)$

D.2. Hyperparameters

We use 1 canonical dictionary element for layer 1, $C_1 = 1$, and 2 canonical dictionary elements for layer 2, $C_2 = 2$. The first layer's dictionary is paramaterized by a 28x28 pixel grid, which is billinearly sampled to produce transformed instances of the canonical image. The second layer's dictionary elements are parameterized by a three-layer SIREN with fully-connected layers of width 512. We train unsupervised with learning rate of $5e{-}3$ for 10 epochs on 60,000 examples per dataset. For inference our layers have sparsity penalty $\lambda_1 = 1.6$ and $\lambda_2 = 1.7$, and run joint inference for 200 iterations.