Hierarchical Concept Embedding & Pursuit for Interpretable Image Classification

Nghia Nguyen
University of Pennsylvania
nghianhh@seas.upenn.edu

Tianjiao Ding
University of Pennsylvania
tjding@seas.upenn.edu

René Vidal
University of Pennsylvania
vidalr@seas.upenn.edu

Abstract

Interpretable-by-design models are gaining traction in computer vision because they provide faithful explanations for their predictions. In image classification, these models typically recover human-interpretable concepts from an image and use them for classification. Sparse concept recovery methods leverage the latent space of vision-language models to represent image embeddings as a sparse combination of concept embeddings. However, because such methods ignore the hierarchical structure of concepts, they can produce correct predictions with explanations that are inconsistent with the hierarchy. In this work, we propose Hierarchical Concept Embedding and Pursuit (HCEP), a framework that induces a hierarchy of concept embeddings in the latent space and uses hierarchical sparse coding to recover the concepts present in an image. Given a semantic hierarchy of concepts, we construct a corresponding hierarchy of concept embeddings and, assuming the correct concepts for an image form a rooted path in the hierarchy, derive desirable conditions for identifying them in the embedded space. We show that hierarchical sparse coding reliably recovers hierarchical concept embeddings, whereas vanilla sparse coding fails. Our experiments demonstrate that HCEP outperforms baselines on real-world datasets in concept precision and recall while maintaining competitive classification accuracy. Moreover, when the number of samples is limited, HCEP achieves superior classification accuracy and concept recovery. These results suggest that incorporating hierarchical structures into sparse coding yields more reliable and interpretable image classification models.

1. Introduction

Machine learning has been adopted in many applications, including image classification, question answering, and recommendation systems [16, 32, 52]. While machine learning models have achieved accuracies comparable to or beyond human experts, the lack of interpretability in these models has raised concerns about their trustworthiness [18, 37, 54].

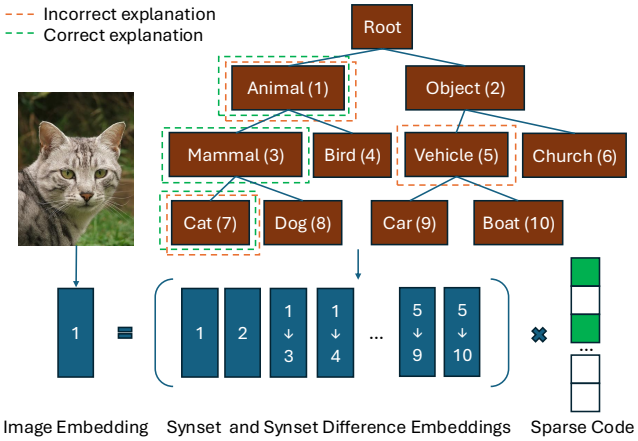

Figure 1. Overview of our Hierarchical Concept Embedding and Pursuit (HCEP) framework for interpretable image classification. For each classification task, we adopt a semantic hierarchy (where the classes are the leaves) to create a dictionary with hierarchical structures. Then, given an image embedding, we recover a sparse representation that respects the hierarchy, which is the unique path in the hierarchy that provides a correct explanation (shown in green).

Interpretable models in computer vision generally fall in two categories: *post-hoc explanation* and *interpretable-by-design*. Post-hoc explanation methods aim to provide insights into the decision-making process of pre-trained black-box models [39, 53, 56, 57]. However, such methods often suffer from a lack of faithfulness and stability of the explanations to the original pre-trained models [1, 5]. Interpretable-by-design methods build interpretability directly into the model training process [2, 10, 29]. Such models usually consist of two steps: (1) extracting human-interpretable concepts from the input and (2) using only these concepts for downstream tasks such as classification or regression. For instance, given an image of a cat, an interpretable-by-design model would first extract concepts such as `animal`, `furry`, and `short muzzles`, and then classify the image using only these concepts.

Recent methods for recovering human-interpretable concepts from images differ in how supervision is used and whether they can handle unseen concepts. In *fully su-*

pervised concept recovery, a model is trained to predict a predefined set of concepts [28, 29]. While effective, this approach does not generalize to unseen concepts, and requires annotations that are often costly for large datasets with many concepts. In *concept-specific supervised recovery*, the model receives both an image and a concept and is trained to predict whether the concept is present in the image. This also requires annotated data, but by leveraging vision-language embeddings such as CLIP [52], it may generalize to new concepts at inference time [9]. *Zero-shot concept recovery* methods rely solely on pre-trained vision-language models to predict concepts without requiring additional annotations [43]. Finally, *sparse concept recovery* methods start with a predefined set of concepts and identify those present in an image by representing the image embedding as a sparse linear combination of the concept embeddings [4, 8]. This approach is scalable as it focuses on selecting a small number of the most relevant concepts.

That being said, concept recovery methods in interpretable-by-design models overlook the fact that semantic synsets [1] often possess *hierarchical* relationships, e.g., hypernyms and hyponyms. As illustrated in Figure 1, we consider the hierarchy of concepts to take the form of a tree whose leaves are the object classes. Thus, there is a unique path connecting a class to the root of the tree, and the concepts along that rooted path can be viewed as an explanation for the class. For example, the explanation for `cat` in Figure 1 is `mammal`, `mammal` $\rightarrow$ `animal`, and `animal` $\rightarrow$ `cat`. However, the aforementioned methods do not account for the hierarchy, and consequently, may recover concepts that are inconsistent with the hierarchy, leading to false explanations and predictions.

Contributions. To address these issues, this paper proposes Hierarchical Concept Embedding & Pursuit (HCEP), a framework that leverages the hierarchical relationships between concepts to improve concept recovery for interpretable image classification. We summarize HCEP as follows and highlight our contributions.

- *Hierarchical Concept Embedding (§3):* Given semantic concepts under a hierarchy, how can we embed them according to the semantic hierarchy to facilitate concept recovery? **We propose ideal geometric properties for hierarchical concept embeddings**, namely embeddings of descendant synsets should be close to that of the parent synsets, while embeddings of sibling synsets should be well-separated. We further embody hierarchical orthogonality inspired by the intriguing study in large language models [50]. These properties are theoretically analyzed and empirically verified on vision-language models.

[1] A synset is a categorical concept that groups all synonyms together (e.g., `cats`). A concept is a higher level abstraction that includes synsets and differences of synsets.

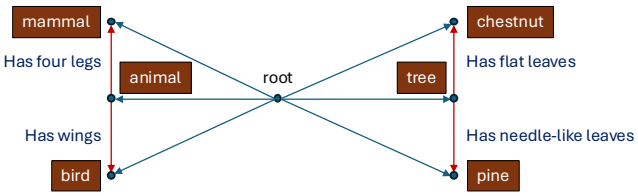

Figure 2. Illustration of the hierarchical latent data model in $\mathbb{R}^2$. The root node spawns two child nodes, `animal` and `tree`, each having their own children. There are two desirable conditions for concept identifiability: (1) The children cluster around the parent while being separated themselves; (2) The difference between a child and its parent (shown as red arrows) is orthogonal to the parent, and the differences between the children of a parent form a simplex (which is a line in $\mathbb{R}^2$). The difference vectors capture the characteristics that distinguish each child from its parent. See §3 for more details.

- *Hierarchical Concept Pursuit (§4):* Based on the geometric properties above, we propose a concept recovery procedure that proceeds in two steps. It first **constructs a hierarchical dictionary from pretrained vision-language embeddings**, which are the differences of the embeddings of parent and child synsets. Then, it leverages hierarchical sparse coding to effectively recover a rooted path in the hierarchy, thus recovering the concepts for interpretable classification.
- *Interpretable Image Classification (§5):* Our experiments on synthetic and real-world datasets demonstrate that **HCEP outperforms sparse concept recovery baselines in concept recovery while maintaining competitive classification accuracy**. In few-shot settings, HCEP outperforms all interpretable baselines in terms of both classification accuracy and concept recovery. We also show that for existing datasets without a predefined hierarchy, we can construct a meaningful hierarchy using taxonomy induction methods [24, 36, 63] and still achieve improved concept recovery using our framework.

2. Preliminaries

2.1. Interpretable-by-design models

Common interpretable-by-design classification models consist of two steps: (1) extracting human-interpretable concepts from the input and (2) using this concept-based representation as the input to a simple classifier (e.g., a linear classifier) to predict. Since the classifier has to be simple for interpretability, the main design space lies in concept extraction. Initial work used supervision to learn concept extractors; however, they are not scalable because they require extra labeled data [29]. A more recent approach is to use pre-trained embeddings to extract concepts from data, with the goal of representing an image embedding as a sparse

linear combination of concept embeddings [4, 8]. In this work, we focus on concept extraction on pre-trained image embeddings via the sparse coding objective.

2.2. Sparse coding for concept extraction

Sparse coding aims to represent data as a sparse linear combination of basis elements, typically chosen from an overcomplete dictionary [21]. Given an input signal $\mathbf{x} \in \mathbb{R}^d$, sparse coding seeks to find a sparse vector $\mathbf{z} \in \mathbb{R}^k$ and a dictionary $\mathbf{D} \in \mathbb{R}^{d \times k}$ such that $\mathbf{x} \approx \mathbf{D}\mathbf{z}$, and the sparsity of $\mathbf{z}$ encourages the model to use only a few dictionary elements to reconstruct the input signal. In interpretable image classification, the signal $\mathbf{x}$ is typically the embedding of an image obtained from a pre-trained model, and the dictionary $\mathbf{D}$ consists of text embeddings that correspond to human-interpretable concepts [4, 8].

The goal is to find a sparse representation $\mathbf{z}$ that captures the most relevant concepts in the image. This can be achieved by solving the following optimization problem:

$$\min_{\mathbf{z}} \|\mathbf{x} - \mathbf{D}\mathbf{z}\|_2^2 + \lambda \|\mathbf{z}\|_0, \tag{1}$$

where $\|\cdot\|_0$ denotes the ℓ_0 norm and λ is a regularization parameter that controls the trade-off between reconstruction accuracy and sparsity.[2] Eq. (1) can be solved using various algorithms, such as orthogonal matching pursuit (OMP) [51] or basis pursuit (ℓ_1 relaxation) [11]. For interpretable image classification, we will focus on OMP due to its connection to the information pursuit framework for interpretability [8]. See App. A.2 for a brief review of sparse coding methods.

The concept extraction step, formulated in Eq. (1), assumes that concepts describing an object are independent of each other; however, in reality, concepts are often hierarchically structured, as objects can be categorized into broader synsets and sub-synsets. For example, the synset `vehicle` can be further divided into `car`, `truck`, and `motorcycle`, each of which can be further divided into more specific synsets. The concepts, which *implicitly* describe the difference between a fine-grained synset and its parent synset, form the edges of the synset hierarchy. To capture this hierarchical structure, we need to extend the sparse coding framework to incorporate hierarchical relationships among concepts. In the next section, we will formalize this idea by proposing a hierarchical concept embedding model.

3. Hierarchical Concept Embedding Model

Given semantic synsets under a hierarchy, how can we represent them as vectors corresponding to the semantic hierarchy so as to facilitate concept recovery? In this section, we describe ideal geometric conditions for hierarchical concept embeddings, which are empirically verified on vision-language models. These conditions later drive the design of the concept recovery algorithm in the next section (§4).

[2] Although not the initial motivation, the validity of this approach is supported by the linear representation and superposition hypotheses [20, 48, 55].

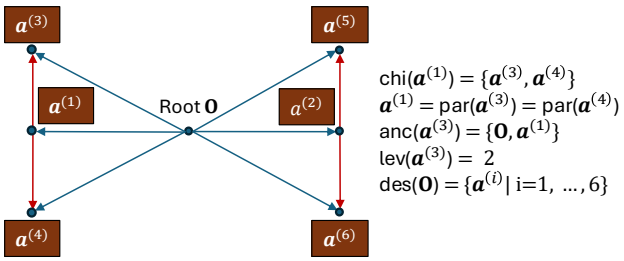

Figure 3. An example hierarchy with $L = 2$ levels, branching factor $b = 2$, and $N_L = 6$ nodes in total. Each node $i \in \mathcal{A}$ has an associated vector representation $\boldsymbol{a}^{(i)} \in \mathbb{R}^d$, where $\mathcal{A} = \{1, \ldots, N_L\}$ is the set of node indices. Let $\mathrm{par}(\cdot), \mathrm{chi}(\cdot), \mathrm{anc}(\cdot), \mathrm{desc}(\cdot) : \mathcal{A} \to \mathcal{P}(\mathcal{A})$ respectively be functions returning the set of parent, children, ancestors, and descendants of a node. The level of a node is the number of its ancestors, that is $\mathrm{lev}(i) = |\,\mathrm{anc}(i)|$.

We first describe some desiderata for a hierarchical representation that allows for concept identifiability:

- *Well-clustered synsets (§3.1):* Sibling synsets should be well-separated in the embedding space to ensure that they can be easily distinguished from each other. Also, child siblings should cluster around their parents. As a consequence, concepts with different ancestries would be well-separated and can be reliably recovered.
- *Hierarchical independence [50] (§3.2):* The concepts that distinguish a child synset from its parent are independent of the parent synset itself. The intuition is that if we make an object more mammal-like, it should not alter the relative probability of it being a dog vs a cat. This condition ensures that the semantic meaning of hierarchical synsets is preserved in the embedding space, thereby corresponding to real-world synsets.

To formalize these desiderata, we describe notations for representing hierarchies in a vector space in Fig. 3.

3.1. Well-clustered synset embeddings

To ensure that each node in the hierarchy can be uniquely assigned to its parent, we impose geometric constraints on the embedding space. The following proposition formalizes conditions that guarantee well-separated subtrees and unambiguous parent identification.

Proposition 3.1 (Well-clustered hierarchy ensures unique parent assignment)**.** *Suppose the following geometric conditions hold for all nodes in the hierarchy:*

1. ***Subtree containment:*** *Each subtree rooted at node i is contained in a cone with vertex $\boldsymbol{a}^{(i)}$ and half-angle*

$\theta_{\text{lev}(i)}$:

$$\max_{j \in \text{desc}(i)} \angle(\boldsymbol{a}^{(i)}, \boldsymbol{a}^{(j)}) \leq \theta_{\text{lev}(i)}, \quad i = 1, \ldots, N_L. \quad (2)$$

2. ***Sibling-cone disjointness:*** *For any parent node i and any pair of distinct children $j, j' \in \text{chi}(i)$, their corresponding subtree cones do not intersect:*

$$\angle(\boldsymbol{a}^{(j)}, \boldsymbol{a}^{(j')}) > \theta_{\text{lev}(j)} + \theta_{\text{lev}(j')} = 2\,\theta_{\text{lev}(i)-1}. \quad (3)$$

Then the subtrees rooted at any two sibling nodes are disjoint, and every node has a unique parent.

All proofs are provided in App. B. A sufficient way to satisfy both conditions in Proposition 3.1 is through a geometric half-angle decreasing schedule, as presented next.

Proposition 3.2 (Geometric half-angle decreasing schedule)**.** *If the half-angles satisfy*

$$\theta_{l+1} \leq \min\{r, 1/b\}\,\theta_l, \quad r \in (0, 1/2), \quad (4)$$

then both conditions in Proposition 3.1 are satisfied.

Intuitively, the geometric decrease in half-angles ensures that as we go down the hierarchy, sibling cones remain disjoint and contained within their parent cone. However, as the hierarchy deepens, these angles can become small, making it harder to distinguish between sibling synset embeddings. This is an inherent limitation of hierarchical embeddings in Euclidean space, which might require alternative geometries (e.g., hyperbolic space) for more faithful embeddings of deep hierarchies [46]. Extending the framework to non-Euclidean geometries is an interesting direction for future work.

3.2. Hierarchical Orthogonality and Simplex Structure

Inspired from the geometric conditions on the concept embeddings in large language models [50], we further recall hierarchical orthogonality and simplex conditions on the concept embeddings:

- The difference between a child and its parent is orthogonal to the parent. For a parent node $\boldsymbol{a}^{(i)}$, any child node $\boldsymbol{a}^{(j)} \in \text{chi}(i)$ satisfies $(\boldsymbol{a}^{(j)} - \boldsymbol{a}^{(i)})^\top \boldsymbol{a}^{(i)} = 0$.
- The difference between the b children of node i that are independent semantically *given the parent* must form a $(b-1)$-simplex. Formally, $\{\boldsymbol{a}^{(j)} - \boldsymbol{a}^{(i)}\}_{j \in \text{chi}(i), |\,\text{chi}(i)| = b}$ forms a $(b-1)$-simplex.

As we shall see soon in the next section, these conditions will guide use to define a dictionary that will be used for sparse coding. Nevertheless, for these two conditions to hold, we need a minimum dimension requirement for the embedding space, as stated in the following proposition.

Proposition 3.3 (Depth–dimension necessity)**.** *Suppose that at every non-leaf node up to depth L, the children satisfy hierarchical orthogonality and their differences form a regular $(b-1)$-simplex, i.e., Eqs. (30) and (32). Then the ambient dimension must satisfy the depth–dimension condition: $d \geq L + b$.*

Intuitively, hierarchical orthogonality imposes $l+1$ independent affine constraints at depth l, restricting children to a $(d-l-1)$-dimensional feasible subspace; embedding a regular $(b-1)$-simplex within that subspace requires $(d-l-1) \geq (b-1)$ for all depths, yielding $d \geq L + b$.

4. Hierarchical Concept Pursuit

Given the concept embedding model described in §3, we now turn to the problem of recovering the sparse representation of a signal generated from this model. To do so, we first construct a hierarchical dictionary that captures the hierarchical structure of the synsets and concepts (§4.1). We then propose a hierarchical sparse coding algorithm that leverages this structure to enhance the recovery of sparse representations (§4.2).

4.1. Hierarchical Dictionary Construction

First, we define the hierarchical dictionary $\boldsymbol{D}$ as

$$\left[\boldsymbol{a}^{(1)}, \ldots, \boldsymbol{a}^{(b)},\ \boldsymbol{a}^{(j_1)} - \boldsymbol{a}^{(\text{par}(j_1))}, \ldots, \boldsymbol{a}^{(j_k)} - \boldsymbol{a}^{(\text{par}(j_k))}\right] \quad (5)$$

where $(j_1, \ldots, j_k) = (j \in \mathcal{A} : j > b)$. The first b columns contain the root child atoms, while the remaining columns contain the differences between synsets and their parents for non-root child nodes.

In the context of interpretable image classification, $\boldsymbol{a}^{(i)}$ represents the embedding of synsets within the WordNet [45] hierarchy.[3] while the difference $\boldsymbol{a}^{(i)} - \boldsymbol{a}^{(\text{par}(i))}$ is the embedding of concepts that distinguish the synset from its parent. As an example, if $\boldsymbol{a}^{(i)}$ is the embedding of the synset bear and $\boldsymbol{a}^{(j)} - \boldsymbol{a}^{(i)}$ is the embedding of the concept `the color white`, then $\boldsymbol{a}^{(j)}$ is the embedding of the synset `polar bear`. This construction of the dictionary in Eq. 5 avoids the trivial solution to the sparse formulation of concept recovery (i.e., the sparsest explanation for an image of a `cat` is just that it is a `cat`) Note that the difference embeddings have a grounded and interpretable meaning on their own, as they represent the direction that differentiates a child synset from its parent. However, we can also interpret these difference embeddings using textual description with vision-language models (e.g., CLIP [52]) and large language models (e.g., GPT-5 [47]), as described in §5.3.

With this model, we can verify that a signal is a sparse linear combination of the atoms corresponding to the nodes

[3] We use WordNet as a concrete example, but this analysis generalizes to other hierarchies.

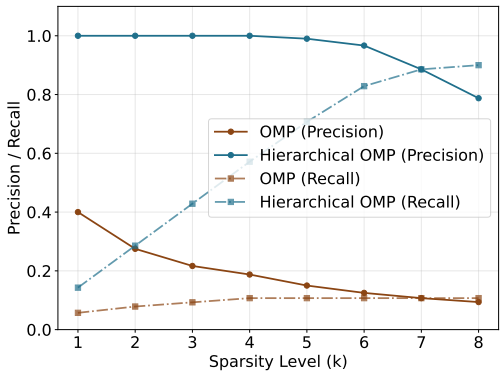

Figure 4. Hierarchical OMP has improved precision and recall compared to standard sparse coding methods on synthetic data.

along the path from the root to the leaf node i plus some Gaussian noise:

$$\boldsymbol{x} = \boldsymbol{a}^{(i)} + \epsilon = \boldsymbol{a}^{(\pi_1(i))} + \sum_{\substack{j \in \text{anc}(i) \\ j \neq \pi_1(i)}} \left(\boldsymbol{a}^{(j)} - \boldsymbol{a}^{(\text{par}(j))}\right) + \epsilon. \quad (6)$$

Here $i \in \mathcal{A}$, $\epsilon \sim \mathcal{N}(0, \sigma^2 \boldsymbol{I})$, and $\pi_1(i)$ denotes the first ancestor of i.

4.2. Hierarchical Orthogonal Matching Pursuit

We now propose a version of Hierarchical OMP [27, 34] that uses the hierarchical structure and beam search to explore multiple hierarchically sparse hypotheses at each iteration. This approach allows us to maintain a diverse set of candidate solutions while correctly navigating the hierarchical structure of the dictionary.

There are two main modifications from OMP: (1) Extending the sparse support only by children of the deepest node explored so far; (2) Using beam search to manage the hypothesis set. The overall procedure is detailed in Algorithm 1. (For completeness, we describe OMP in Algorithm 5). Note that for clarity, which level a synset is at, a synset embedding $\boldsymbol{a}^{(i)}$ in the algorithm is referred to as $\boldsymbol{a}_{l,i}$ where l is the depth of node i. Also, except for the root children atoms, all other atoms are not the synset themselves but the *differences* of the synsets from their parents. We formalize the theoretical advantages of this hierarchical approach in the following proposition.

Proposition 4.1 (Informal)**.** *Suppose that the atoms chosen by Hierarchical OMP are in the correct support at each iteration. Then, Hierarchical OMP yields a strictly larger ERC [61] success region than OMP on the full dictionary.*

This result supports our choice of using beam search to explore multiple hypotheses, as it increases the likelihood of recovering the correct hierarchical support.

Algorithm 1 Hierarchical OMP

Require: $\boldsymbol{x} \in \mathbb{R}^d$, dict $\boldsymbol{D}$, roots $\mathcal{R}$, child map $\text{chi}(\cdot)$, ancestry $\text{anc}(\cdot)$, tol ϵ, max steps T, beam B

Initialize with a null hypothesis: $\mathcal{H}^{(0)} \leftarrow \left\{(\emptyset, \boldsymbol{x}, \boldsymbol{0})\right\}$
for $t = 0, \ldots, T-1$ **do**
  if $\min_{h \in \mathcal{H}^{(t)}} \|\boldsymbol{r}_h\|_2 < \epsilon$ **then**
    break
  $\mathcal{H}_{\text{new}} \leftarrow \emptyset$
  for each hypothesis h in $\mathcal{H}^{(t)}$ **do**
    $\mathcal{D}_{\text{active}} \leftarrow \textsc{ExtendDict}(h, t)$ (Alg. 2)
    $c_i \leftarrow \left| \frac{\langle \boldsymbol{r}, \boldsymbol{d}^{(i)} \rangle}{\|\boldsymbol{r}\|_2 \, \|\boldsymbol{d}^{(i)}\|_2} \right|$ for all $\boldsymbol{d}^{(i)} \in \mathcal{D}_{\text{active}}$
    $\mathcal{C} \leftarrow$ top-B indices of c_i
    if $\mathcal{C} = \emptyset$ **then** ▷ leaf reached
      Add h to $\mathcal{H}_{\text{new}}$ ▷ keep hypothesis
      continue
    $\mathcal{H}_{\text{ext}} \leftarrow \textsc{ExtendHypo}(h, \mathcal{C}, \boldsymbol{x}, \boldsymbol{D})$ (Alg. 3)
    $\mathcal{H}_{\text{new}} \leftarrow \mathcal{H}_{\text{new}} \cup \mathcal{H}_{\text{ext}}$
  $\mathcal{H}^{(t+1)} \leftarrow \textsc{PruneBeam}(\mathcal{H}_{\text{new}}, B)$ (Alg. 4)
Return $\boldsymbol{z}_{h^\star}$ where $h^\star \in \arg\min_{h \in \mathcal{H}^{(t)}} \|\boldsymbol{r}_h\|_2$

Algorithm 2 Extend Active Dictionary

Require: iteration t, hypothesis $h = (\mathcal{S}, \boldsymbol{r}, i_{last})$, roots $\mathcal{R}$

if $t = 0$ **then**
  return $\{\boldsymbol{a}_{1,i} : i \in \text{chi}(\mathcal{R})\}$
else
  return $\{\boldsymbol{a}_{l+1,j} - \boldsymbol{a}_{l,i_{last}} : j \in \text{chi}(i_{last})\}$

Algorithm 3 Extend Hypothesis with Sparse Update

Require: hypothesis $h = (\mathcal{S}, \boldsymbol{r}, i_{last})$, candidates $\mathcal{C}$, signal $\boldsymbol{x}$, dict $\boldsymbol{D}$

$\mathcal{H}_{\text{ext}} \leftarrow \emptyset$
for each $i \in \mathcal{C}$ **do**
  $\mathcal{S}' \leftarrow \mathcal{S} \cup \{i\}$
  $\boldsymbol{z}' \leftarrow \arg\min_{\boldsymbol{w}} \|\boldsymbol{x} - \boldsymbol{D}_{\mathcal{S}'} \boldsymbol{w}\|_2^2$
  $\boldsymbol{r}' \leftarrow \boldsymbol{x} - \boldsymbol{D}_{\mathcal{S}'} \boldsymbol{z}'$
  Add $(\mathcal{S}', \boldsymbol{r}', i)$ to $\mathcal{H}_{\text{ext}}$
return $\mathcal{H}_{\text{ext}}$

Algorithm 4 Beam Pruning

Require: hypotheses $\mathcal{H}$, beam size B

$\mathcal{H}_{\text{sorted}} \leftarrow$ hypotheses in $\mathcal{H}$ sorted by $\|\boldsymbol{r}_h\|_2^2$
return top-$\min(B, |\mathcal{H}_{\text{sorted}}|)$ elements of $\mathcal{H}_{\text{sorted}}$

5. Experiments

In this section, we evaluate the performance of HCEP by comparing Hierarchical OMP on both synthetic (§5.1) and real-world datasets (§5.2). We compare our method with interpretable baselines in terms of concept recovery accuracy and classification performance.

5.1. Synthetic Experiments

We first compare the performance of Hierarchical OMP (Alg. 1) with standard OMP on synthetic data generated from the Hierarchical Concept Embeddings model in §3. We evaluate the reconstruction error and the recovery of the ground-truth sparse support (i.e., the path from the root to the leaf node) under varying noise levels and hierarchy depths.

We choose a branching factor of $b = 3$, hierarchy depth $L = 7$, and dimension $d = 50$. Note that this dimension satisfies the depth–dimension condition in Eq. (37) since $d = 50 \geq 7 + 3 = 10$. This gives us 2187 leaf synsets and 3280 atoms in the dictionary. We generate 5 samples per leaf for a total of $10,935$ samples. Further details on how the synthetic data is generated can be found in App. C, and the hyperparameters can be found in App. D.1.

See the results in Fig. 4. We observe that Hierarchical OMP consistently outperforms standard OMP in both precision and recall. This demonstrates the effectiveness of incorporating hierarchical structure into sparse coding for improved concept recovery.

5.2. Real-world Experiments

In this section, we evaluate HCEP on real-world image classification tasks. First, given a class (which is a leaf node), we estimate the class embeddings by taking the mean of the CLIP [52] image embeddings for images belonging to that class. For the non-leaf synsets, we estimate their embeddings using the mean of their children's embeddings. Next, we construct the hierarchical dictionary as described in Eq. (5). We keep this fixed dictionary for all experiments. We next provide the overall experiment settings; more details are in App. D.2.

Hierachical Orthogonality and Well-Clustered Synsets in Real-world Datasets. To test the validity of the hierarchical orthogonality [50] condition in real-world datasets, we measure the cosine similarity between the difference vectors of child-parent pairs and their parents, as seen in Fig. 5 for ImageNet (see Fig. 11 for CIFAR-100). We find that the average cosine similarity is close to zero. However, if child and random non-parent pairs are considered, the average cosine similarity is significantly different from zero. We also test the well-clustered synset condition (Prop. 3.1) on ImageNet in Fig. 6, showing that most branches are tightly clustered and well-separated from other branches.

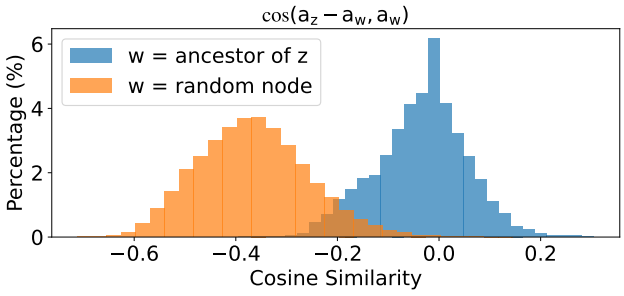

Figure 5. Observed Hierarchical Orthogonality on ImageNet for CLIP [52]. The cosine similarity between child-parent difference vectors and their parents is close to zero, while random non-parent pairs have significantly higher cosine similarity. This suggests that hierarchical orthogonality [50] holds even on contrastively trained vision models.

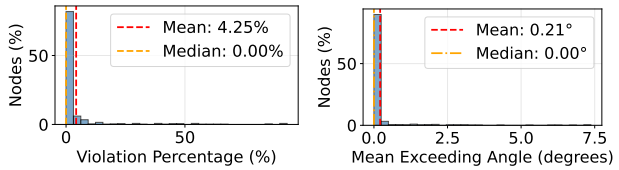

Figure 6. Clusters are tight on ImageNet for CLIP image embeddings. (Top): Fixed a node i, this is the proportion of non-descendants of i intersecting the cone of i. (Bottom): Given the same node i, we show the mean angle violation (non-violating angles are counted as 0).

Datasets. We evaluate on three datasets: (1) ImageNette; (2) ImageNet [14]; (3) CIFAR-100 [31]. For ImageNet-based datasets, we can use the WordNet hierarchy directly. For CIFAR-100, we use taxonomy induction methods [63] to construct a hierarchy over the classes.

Baselines. We compare HCEP with the following baselines: (1) OMP [8, 51] using the full dictionary; (2) Concept Bottleneck Models [29] that use supervised concept annotations to learn a concept extractor; (3) Nearest Neighbor (NN) classifier using the synset embeddings directly (this can be thought of as using a black box zero-shot classifier); (4) Hierarchical NN that traverses the hierarchy using nearest neighbor search at each level.

Evaluation Metrics. We evaluate the models based on (1) classification accuracy; (2) support precision and recall, which measure how well the recovered sparse support matches the ground-truth path in the hierarchy.

Classification Procedure. For each image, we extract its CLIP embedding. Next, the different methods extract the sparse representation: (1) The sparse-coding methods (Hierarchical OMP and Standard OMP) recover a sparse representation using their respective algorithms; (2) Hierarchical NN recovers the sparse code by traversing the hierarchy us-

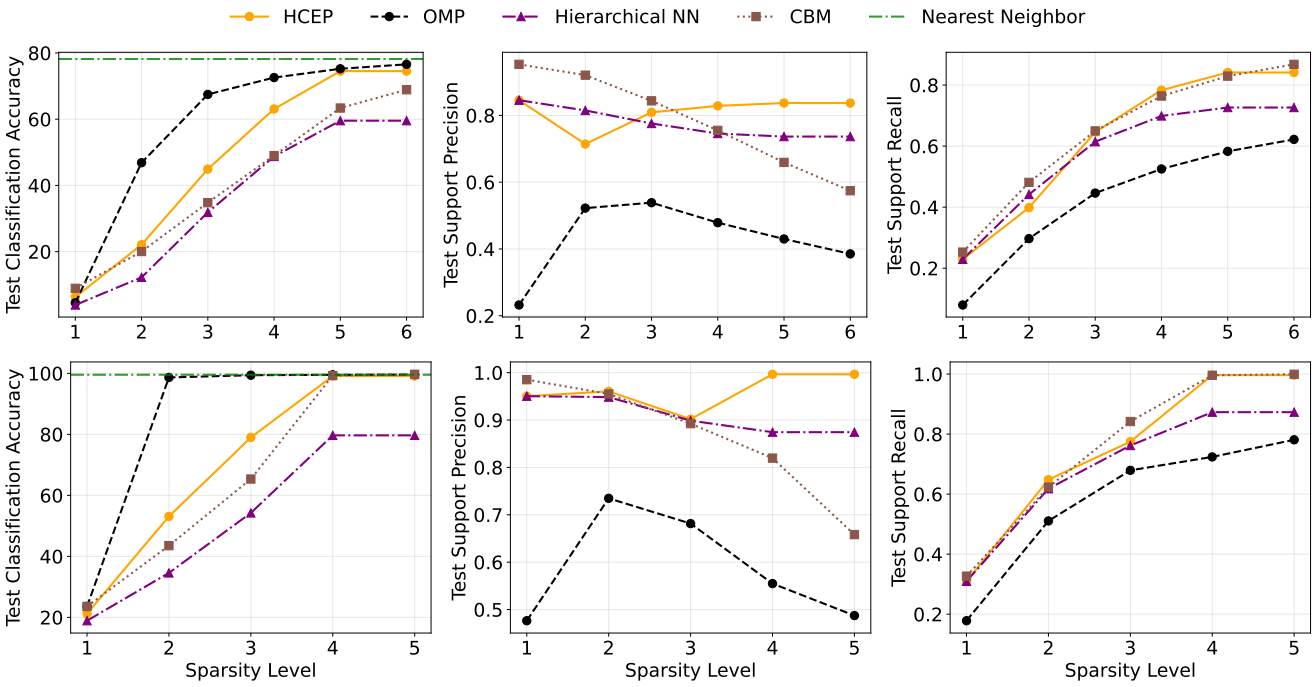

Figure 7. Interpretable image classification on on CIFAR-100 (top) and ImageNette (bottom). HCEP has state of the art support precision/recall while maintaining comparable classification accuracy.

Table 1. As we reduce the number of images per class in ImageNet, HCEP consistently improves test classification accuracy, support precision, and support recall over all baselines. Result at sparsity level 14.

Method	Classification Accuracy				Support Precision				Support Recall			
	12-shot	25-shot	50-shot	Full	12-shot	25-shot	50-shot	Full	12-shot	25-shot	50-shot	Full
OMP	45.8	52.9	57.4	70.7	15.6	16.3	16.6	17.5	36.6	37.9	38.3	41.0
Hierarch. NN	28.1	29.0	29.8	15.7	47.9	48.2	48.6	34.9	44.1	44.0	44.1	28.7
CBM	24.7	34.4	44.0	**78.5**	30.5	31.8	33.3	45.6	60.0	62.9	66.2	**97.3**
HCEP (Ours)	**52.1**	**57.3**	**61.3**	65.2	**71.2**	**72.2**	**73.0**	**70.1**	**71.8**	**72.7**	**73.4**	71.1

ing nearest neighbor search at each level; (3) CBMs get the code for all atoms at once by training a classifier on top of the CLIP image embeddings. The recovered sparse codes are then fed into a linear classifier trained on the training set to predict the class labels. For the Nearest Neighbor baseline, we directly use the synset embeddings to classify the images without any intermediate representation.

See the results on ImageNet (Tab. 1), CIFAR100/ImageNette (Fig. 7). We observe that Hierarchical OMP achieves higher support precision and recall compared to other baselines, indicating better recovery of relevant concepts. In the low-data setting, Hierarchical OMP outperforms all interpretable baselines in classification accuracy and support precision/recall, demonstrating its robustness in low-data settings.

5.3. Text Interpretation of Synset Differences

To qualitatively evaluate an alternative text meaning of the synset differences, which form the atoms in our hierarchical dictionary, we use CLIP text embeddings and GPT-5 [47]. First, for each pair of parent-child, we generate a text description of the difference between the parent and child synset using GPT-5. Then, we pool the text embeddings of these descriptions to form a set of candidate concept embeddings. Next, for each synset difference vector, we find the top-k neighbor in the candidate concept embeddings. Finally, we utilize GPT-5 to generate a summary of the top-k neighbor descriptions, which helps interpret the synset difference. See some example interpretations for some parent-child pairs in WordNet [45] (the hierarchy behind ImageNet) in Tab. 2.

Table 2. Example text interpretations of child–parent synset differences produced via CLIP embeddings and GPT summarization.

Parent→Child Pair	Text Interpretation
`bear` → `polar bear`	thick matte white fur blending with snow.
`container` → `basket`	open-top woven or perforated sides with handles.
`structure` → `lumbermill`	vertical log-sawing machines and plank conveyors.
`citrus` → `orange`	round, bright orange, pebbled rind.

6. Related Work

Interpretable-by-design models. Interpretable-by-design models aim to provide explanations for their predictions by using human-interpretable concepts as intermediate representations. Early works explored attribute-based classification for face verification [33] and learning to detect unseen object classes through attribute transfer [35]. Subsequent works include Concept Activation Vectors [28], which use linear classifiers to identify directions in the embedding space corresponding to specific concepts. Concept Bottleneck Models [29] extend this idea by training models to predict concepts before predicting. In an adjacent line of work, Information Pursuit [23] is used as a criterion to choose the most relevant concepts [7, 9, 30]. More recent works have explored leveraging pre-trained embeddings and sparse coding for identifying specific concept directions [4, 8]. Our work builds upon these foundations by introducing a concept embedding framework that captures the hierarchical relationships among synsets in interpretable image classification.

Sparse Recovery. Sparse recovery aims to recover a sparse signal from a set of observations, often using techniques such as Orthogonal Matching Pursuit (OMP) [51], Basis Pursuit [11]. Sparse coding has been widely used in image processing [40, 41], signal processing, and machine learning. Although there have been works on hierarchical sparse coding [25, 27, 34], they do not consider the hierarchical structure of concepts in the context of interpretable models or deep representation learning.

Geometric Structures of Meanings in Vector Embeddings. A notable example of geometric structures in vector embeddings is Word2Vec [44], where certain semantic relationships can be captured through vector arithmetic. More recent works have explored the linear structure [26, 49, 59, 60] and the hierarchical and categorical concepts in vector spaces [50]. Our work extends these ideas to the context of sparse coding for interpretable models, providing a foundation for hierarchical concept recovery.

7. Limitations

While our framework demonstrates clear advantages in concept recovery, there are several limitations:

Dimensionality Constraints. Our theoretical analysis (Proposition 3.3) establishes that embedding a hierarchy with depth L and branching factor b requires ambient dimension $d \geq L+b$. For deep hierarchies (large L) or highly branching structures (large b), this constraint becomes restrictive. Real-world embeddings from models like CLIP typically have fixed dimensions (e.g. $d = 768$), which limits the depth and complexity of hierarchies that can be faithfully represented. Moreover, as hierarchies deepen, the half-angles of the cones containing each subtree (Proposition 3.2) must decrease geometrically. As mentioned in § 3.1, this limitation may necessitate exploring alternative geometries (e.g., hyperbolic spaces) [15, 46] for more faithful hierarchical representations. This is an interesting direction for future work.

Hierarchy Quality Dependence. The performance of Hierarchical OMP critically depends on the quality of the predefined hierarchy. For ImageNet-based datasets, we leverage the well-curated WordNet hierarchy, which provides semantically meaningful relationships. However, for CIFAR-100, we must use taxonomy induction methods [63], which may produce hierarchies with inconsistencies or unclear relationships.

Computational Complexity. Hierarchical OMP with beam search (Algorithm 1) has complexity $O(TBK|\mathcal{D}_{\text{active}}|)$, where T is the number of iterations, B is the beam width, K is the branching factor, and $|\mathcal{D}_{\text{active}}|$ is the size of the active dictionary at each level. In contrast, OMP has complexity $O(T|\mathcal{D}|)$, where $|\mathcal{D}|$ is the size of the entire dictionary. For large branching factors or deep hierarchies, this can become computationally expensive. While we demonstrate better concept recovery accuracy over standard OMP, the computational cost remains a practical consideration for deployment at scale.

8. Conclusion

We introduced a geometric framework for hierarchical concept embeddings together with Hierarchical OMP, a pursuit algorithm that respects the structure of synset hierarchies. We analyze identifiability requirements through well-clustered cones, hierarchical orthogonality, and simplex structure along with algorithmic guarantees via an expanded ERC region for Hierarchical OMP. Empirically, the resulting codes deliver substantially better support precision and recall than interpretable baselines across synthetic and real-world benchmarks, particularly in low-data regimes where interpretability is often most valuable. Our findings highlight the promise of structured sparse coding as a scalable and flexible framework for interpretable machine learning.

References

[1] Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. In *NeurIPS*, pages 9505–9515, 2018. 1

[2] David Alvarez-Melis and Tommi S. Jaakkola. Towards robust interpretability with self-explaining neural networks. In *NeurIPS*, 2018. 1

[3] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009. 13

[4] Usha Bhalla, Alex Oesterling, Suraj Srinivas, Flavio P. Calmon, and Himabindu Lakkaraju. Interpreting CLIP with Sparse Linear Concept Embeddings (SpLiCE), 2024. arXiv:2402.10376 [cs]. 2, 3, 8

[5] Blair Bilodeau, Natasha Jaques, Pang Wei Koh, and Been Kim. Impossibility theorems for feature attribution. *Proceedings of the National Academy of Sciences*, 121(2): e2304406120, 2024. 1

[6] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2011. 13

[7] Aditya Chattopadhyay, Kwan Ho Ryan Chan, Benjamin D Haeffele, Donald Geman, and René Vidal. Variational information pursuit for interpretable predictions. In *The Eleventh International Conference on Learning Representations*, 2023. 8

[8] Aditya Chattopadhyay, Ryan Pilgrim, and René Vidal. Information Maximization Perspective of Orthogonal Matching Pursuit with Applications to Explainable AI. 2023. 2, 3, 6, 8

[9] Aditya Chattopadhyay, Kwan Ho Ryan Chan, and Rene Vidal. Bootstrapping variational information pursuit with large language and vision models for interpretable image classification. In *The Twelfth International Conference on Learning Representations*, 2024. 2, 8

[10] Chaofan Chen, Oscar Li, Daniel Tao, Karthik Barnett, and Cynthia Rudin. This looks like that: Deep learning for interpretable image recognition. In *NeurIPS*, 2019. 1

[11] S. Chen and D. Donoho. Basis pursuit. In *Proceedings of 1994 28th Asilomar Conference on Signals, Systems and Computers*, pages 41–44, 1994. 3, 8

[12] Scott Shaobing Chen, David L. Donoho, and Michael A. Saunders. Atomic decomposition by basis pursuit. *SIAM J. Sci. Comput.*, 20(1):33–61, 1998. 13

[13] Ingrid Daubechies, Michel Defrise, and Christine De Mol. An iterative thresholding algorithm for linear inverse problems. *Communications on Pure and Applied Mathematics*, 57(11):1413–1457, 2004. 13

[14] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 248–255, 2009. 6

[15] Karan Desai, Maximilian Nickel, Tanmay Rajpurohit, Justin Johnson, and Ramakrishna Vedantam. Hyperbolic Image-Text Representations, 2024. arXiv:2304.09172 [cs]. 8

[16] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019. 1

[17] David L. Donoho and Yoram Tsaig. Fast solution of ell_1-minimization problems when the solution may be sparse. *IEEE Transactions on Information Theory*, 54(11): 4789–4812, 2008. 13

[18] Finale Doshi-Velez and Been Kim. Towards a rigorous science of interpretable machine learning. *arXiv*, 2017. 1

[19] Bradley Efron, Trevor Hastie, Iain Johnstone, and Robert Tibshirani. Least angle regression. *Annals of Statistics*, 32 (2):407–499, 2004. 13

[20] Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Liao, Christopher Olah, Sam Heidenreich, Tom Hooker, Jonathan Weiss, Jared Zimmerman, Ben Mann, Neel Joseph, and Evan Hubinger. Toy Models of Superposition, 2022. arXiv:2209.10652 [cs]. 3

[21] Simon Foucart and Holger Rauhut. *An Invitation to Compressive Sensing*. Springer, 2013. 3, 13

[22] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software*, 33(1):1–22, 2010. 13

[23] Donald Geman and Bruno Jedynak. An active testing model for tracking roads in satellite images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(1):1–14, 2002. 8

[24] Marti A. Hearst. Automatic acquisition of hyponyms from large text corpora. In *COLING*, 1992. 2

[25] Rodolphe Jenatton, Julien Mairal, Guillaume Obozinski, and Francis Bach. Proximal methods for hierarchical sparse coding. *Journal of Machine Learning Research*, 12:2297–2334, 2011. 8

[26] Yibo Jiang, Goutham Rajendran, Pradeep Ravikumar, Bryon Aragam, and Victor Veitch. On the origins of linear representations in large language models. *arXiv preprint arXiv:2403.03867*, 2024. 8

[27] Philippe Jost, Pierre Vandergheynst, and Pascal Frossard. Tree-Based Pursuit: Algorithm and Properties. *IEEE Transactions on Signal Processing*, 54(12):4685–4697, 2006. 5, 8

[28] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, and Rory Sayres. Interpretability Beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors (TCAV). In *Proceedings of the 35th International Conference on Machine Learning*, pages 2668–2677, 2018. 2, 8

[29] Pang Wei Koh, Thao Nguyen, Yew Siang Tang, Stephen Mussmann, Emma Pierson, Been Kim, and Percy Liang. Concept Bottleneck Models. In *Proceedings of the 37th International Conference on Machine Learning*, pages 5338–5348, 2020. 1, 2, 6, 8

[30] Stefan Kolek, Aditya Chattopadhyay, Kwan Ho Ryan Chan, Hector Andrade-Loarca, Gitta Kutyniok, and René Vidal. Learning interpretable queries for explainable image classification with information pursuit. In *Proceedings of the*

IEEE/CVF International Conference on Computer Vision, pages 3947–3956, 2025. 8

[31] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. Technical report, University of Toronto, 2009. 6

[32] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, pages 1106–1114, 2012. 1

[33] Neeraj Kumar, Alexander C. Berg, Peter N. Belhumeur, and Shree K. Nayar. Attribute and simile classifiers for face verification. In *2009 IEEE 12th International Conference on Computer Vision*, pages 365–372. IEEE, 2009. 8

[34] Chinh La and Minh N. Do. Tree-Based Orthogonal Matching Pursuit Algorithm for Signal Reconstruction. In *2006 International Conference on Image Processing*, pages 1277–1280, Atlanta, GA, 2006. IEEE. 5, 8

[35] Christoph H. Lampert, Hannes Nickisch, and Stefan Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 951–958. IEEE, 2009. 8

[36] Matthew Le, Stephen Roller, Laetitia Papaxanthos, Douwe Kiela, and Maximilian Nickel. Inferring Concept Hierarchies from Text Corpora via Hyperbolic Embeddings. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3231–3241, Florence, Italy, 2019. Association for Computational Linguistics. 2

[37] Zachary C. Lipton. The mythos of model interpretability. *Communications of the ACM*, 61(10):36–43, 2018. 1

[38] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 16

[39] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, 2017. 1

[40] Julien Mairal, Guillermo Sapiro, and Michael Elad. Learning multiscale sparse representations for image and video restoration. *Multiscale Modeling & Simulation*, 7(1):214–241, 2008. 8

[41] Julien Mairal, Francis Bach, Jean Ponce, and Guillermo Sapiro. Online learning for matrix factorization and sparse coding. *Journal of Machine Learning Research*, 11:19–60, 2010. 8

[42] Stephane Mallat and Zhifeng Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Trans. Signal Process.*, 41(12):3397–3415, 1993. 12

[43] Sachit Menon and Carl Vondrick. Visual Classification via Description from Large Language Models, 2022. arXiv:2210.07183 [cs]. 2

[44] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S. Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 3111–3119, 2013. 8

[45] George A. Miller. Wordnet: A lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995. 4, 7

[46] Maximillian Nickel and Douwe Kiela. Poincaré embeddings for learning hierarchical representations. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 6338–6347, 2017. 4, 8

[47] OpenAI. Gpt-5. Model documentation, 2025. Accessed 2025-11-10. 4, 7, 16

[48] Kiho Park, Yo Joong Choe, and Victor Veitch. The Linear Representation Hypothesis and the Geometry of Large Language Models. In *Proceedings of the 41st International Conference on Machine Learning*. PMLR, 2024. arXiv:2311.03658 [cs]. 3

[49] Kiho Park, Yo Joong Choe, and Victor Veitch. The Linear Representation Hypothesis and the Geometry of Large Language Models, 2024. arXiv:2311.03658 [cs]. 8

[50] Kiho Park, Yo Joong Choe, Yibo Jiang, and Victor Veitch. The Geometry of Categorical and Hierarchical Concepts in Large Language Models, 2025. arXiv:2406.01506 [cs]. 2, 3, 4, 6, 8

[51] Y. C. Pati, R. Rezaiifar, and P. S. Krishnaprasad. Orthogonal matching pursuit: Recursive function approximation with applications to wavelet decomposition. In *Proceedings of 27th Asilomar Conference on Signals, Systems and Computers*, pages 40–44, 1993. 3, 6, 8, 12

[52] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision, 2021. arXiv:2103.00020 [cs]. 1, 2, 4, 6, 16

[53] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier. In *KDD*, pages 1135–1144, 2016. 1

[54] Cynthia Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nature Machine Intelligence*, 1:206–215, 2019. 1

[55] Baturay Saglam, Paul Kassianik, Blaine Nelson, Sajana Weerawardhena, Yaron Singer, and Amin Karbasi. Large Language Models Encode Semantics in Low-Dimensional Linear Subspaces. 3

[56] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *ICCV*, pages 618–626, 2017. 1

[57] Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *ICML*, 2017. 1

[58] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996. 13

[59] Matthew Trager, Pramuditha Perera, Luca Zancato, Alessandro Achille, Rahul Bhotika, and Stefano Soatto. Linear spaces of meanings: Compositional structures in vision-language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 15395–15405, 2023. 8

[60] Matthew Trager, Alessandro Achille, Pramuditha Perera, Luca Zancato, and Stefano Soatto. Compositional structures

in neural embedding and interaction decompositions, 2024. arXiv:2407.08934. 8

[61] Joel A. Tropp. Greed is good: Algorithmic results for sparse approximation. *IEEE Trans. Inf. Theory*, 50(10):2231–2242, 2004. 5, 12, 15

[62] John Wright and Yi Ma. *High-Dimensional Data Analysis with Low-Dimensional Models: Principles, Computation, and Applications*. Cambridge University Press, 2022. 13

[63] Qingkai Zeng, Yuyang Bai, Zhaoxuan Tan, Shangbin Feng, Zhenwen Liang, Zhihan Zhang, and Meng Jiang. Chain-of-layer: Iteratively prompting large language models for taxonomy induction from limited examples. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*, pages 3093–3102, 2024. 2, 6, 8, 16

[64] Tong Zhang. Sparse recovery with orthogonal matching pursuit: Sharp sufficient conditions and worst-case examples. *IEEE Trans. Inf. Theory*, 57(9):6219–6229, 2011. 12

Contents

A. Preliminaries

A.1. Canonical regular simplex

Define

$$\tilde{\boldsymbol{s}}_j \;=\; \boldsymbol{e}_j \;-\; \frac{1}{b}\,\mathbf{1}, \qquad j = 1, \dots, b, \tag{7}$$

where $\{\boldsymbol{e}_j\}_{j=1}^{b}$ are the standard basis vectors of $\mathbb{R}^b$ and $\mathbf{1} \in \mathbb{R}^b$ is the all-ones vector. These centred vertices satisfy $\sum_{j=1}^{b} \tilde{\boldsymbol{s}}_j = \mathbf{0}$ and $\tilde{\boldsymbol{s}}_j^\top \tilde{\boldsymbol{s}}_k = \begin{cases} 1, & j = k, \\ -\dfrac{1}{b-1}, & j \neq k, \end{cases}$ i.e. they form a regular $(b-1)$-simplex of unit edge length in $\mathbb{R}^{b-1}$.

A.2. Sparse Recovery

We briefly recall two classical sparse recovery approaches that motivate our hierarchical construction: greedy pursuit via Orthogonal Matching Pursuit (OMP) and convex relaxation via Basis Pursuit (BP).

Consider the linear model

$$\boldsymbol{x} = \boldsymbol{D}\boldsymbol{z}, \qquad \boldsymbol{D} \in \mathbb{R}^{d \times k},\ \boldsymbol{z} \in \mathbb{R}^k \text{ sparse}, \tag{8}$$

with normalized columns (atoms) of $\boldsymbol{D}$. We focus on the exact linear model for clarity. The goal is to recover the unknown sparse coefficient vector $\boldsymbol{z}$ from $\boldsymbol{x}$. We use the shorthand $[k] := \{1, 2, \dots, k\}$ for column indices.

Matching Pursuit (MP). The classical MP algorithm greedily selects at each step the dictionary atom most correlated with the current residual and updates the residual by removing its component along that atom, without re-fitting over the accumulated support [42]. OMP below is the orthogonalized variant that re-solves on the active set at every iteration.

Orthogonal Matching Pursuit (OMP). OMP is a greedy algorithm that orthogonalizes over the active set at each step (See OMP in Algorithm 5; for comparison, we also provide the full details of Hierarchical OMP in Algorithm 6), contrasting with our hierarchical variant. The restricted least-squares update used at each iteration is

$$\boldsymbol{z}_S \;=\; \arg\min_{\boldsymbol{w} \in \mathbb{R}^{|S|}} \|\boldsymbol{x} - \boldsymbol{D}_S \boldsymbol{w}\|_2^2, \qquad \boldsymbol{z}_{S^c} = \mathbf{0}. \tag{9}$$

The orthogonal projection step ensures previously selected atoms are regressed jointly in Eq. (9) (hence "orthogonal"). A classic guarantee is that if $\boldsymbol{z}$ is s-sparse and the dictionary coherence

$$\mu(\boldsymbol{D}) = \max_{i \neq j} |\boldsymbol{d}_i^\top \boldsymbol{d}_j| \tag{10}$$

is small enough so that $s < \frac{1}{2}\big(1 + \mu(\boldsymbol{D})^{-1}\big)$, then OMP exactly recovers the support in s steps [61, 64][4].

[4] The OMP algorithm was introduced by Pati et al. [51]. The mutual coherence exact recovery bound $s < \frac{1}{2}(1 + 1/\mu)$ was first proved rigorously by Tropp [61] and refined with sharp sufficient conditions and worst-case examples by Zhang [64].

Algorithm 5 Orthogonal Matching Pursuit (OMP)

Require: signal $\boldsymbol{x} \in \mathbb{R}^d$, dictionary $\boldsymbol{D} \in \mathbb{R}^{d\times k}$ with normalized columns $\{\boldsymbol{d}_j\}_{j\in[k]}$, sparsity budget s (optional)

Initialize support $S^{(0)} \leftarrow \emptyset$, residual $\boldsymbol{r}^{(0)} \leftarrow \boldsymbol{x}$, coefficient vector $\boldsymbol{z} \leftarrow \boldsymbol{0}$, iteration index $t \leftarrow 0$
for $t = 0, \dots, s-1$ **do**
  $j^* \leftarrow \arg\max_{j\in[k]} \left| \frac{\boldsymbol{d}_j^\top \boldsymbol{r}^{(t)}}{\|\boldsymbol{d}_j\|_2 \, \|\boldsymbol{r}^{(t)}\|_2} \right|$ ▷ Select highest cosine similarity
  $S^{(t+1)} \leftarrow S^{(t)} \cup \{j^*\}$
  $\boldsymbol{z}_{S^{(t+1)}} \leftarrow \arg\min_{\boldsymbol{w}\in\mathbb{R}^{|S^{(t+1)}|}} \|\boldsymbol{x} - \boldsymbol{D}_{S^{(t+1)}}\boldsymbol{w}\|_2^2$; $\boldsymbol{z}_{(S^{(t+1)})^c} \leftarrow \boldsymbol{0}$ ▷ Restricted least squares (Eq. 9)
  $\boldsymbol{r}^{(t+1)} \leftarrow \boldsymbol{x} - \boldsymbol{D}\boldsymbol{z}$ ▷ Residual update
  if $\|\boldsymbol{r}^{(t+1)}\|_2 = 0$ **then**
    break
return $\boldsymbol{z}$

Basis Pursuit (BP). BP replaces the combinatorial ℓ_0 objective with an ℓ_1 minimization [12]:

$$\min_{\boldsymbol{z}\in\mathbb{R}^k} \|\boldsymbol{z}\|_1 \quad \text{s.t.} \quad \boldsymbol{D}\boldsymbol{z} = \boldsymbol{x}. \tag{11}$$

This convex program promotes sparsity via soft-thresholding effects. Under RIP or incoherence assumptions similar to those for OMP, BP provably recovers the sparsest solution when $\boldsymbol{x}$ lies in the range of a sparse $\boldsymbol{z}$ [12, 21]. Efficient solvers include coordinate descent [22], proximal gradient methods (ISTA/FISTA) [3, 13], homotopy [17], and ADMM [6]. See [21, 62] for a survey. A related penalized least-squares formulation is LASSO [19, 58].

Algorithm 6 Hierarchical OMP

Require: $\boldsymbol{x} \in \mathbb{R}^d$, dict $\boldsymbol{D}$, roots $\mathcal{R}$, child map $\mathrm{chi}(\cdot)$, ancestry $\mathrm{anc}(\cdot)$, tol ϵ, max steps T, beam B

Initialize with a null hypothesis: $\mathcal{H}^{(0)} \leftarrow \left\{ \left(\emptyset, \boldsymbol{x}, \boldsymbol{0}\right) \right\}$
for $t = 0, \dots, T-1$ **do**
  if $\min_{h\in\mathcal{H}^{(t)}} \|\boldsymbol{r}_h\|_2 < \epsilon$ **then**
    break
  $\mathcal{H}_{\text{new}} \leftarrow \emptyset$
  for each hypothesis $h = (\mathcal{S}, \boldsymbol{r}, i_{last})$ in $\mathcal{H}^{(t)}$ **do**
    if $t = 0$ **then**
      $\mathcal{D}_{\text{active}} \leftarrow \{\boldsymbol{a}_{1,i} : i \in \mathcal{R}\}$
    else
      $\mathcal{D}_{\text{active}} \leftarrow \{\boldsymbol{a}_{l+1,j} - \boldsymbol{a}_{l,i} : j \in \mathrm{chi}(i)\}$ ▷ l is the depth of node i
    Compute $c_i \leftarrow \left| \frac{\langle \boldsymbol{r}, \boldsymbol{d}^{(i)} \rangle}{\|\boldsymbol{r}\|_2 \, \|\boldsymbol{d}^{(i)}\|_2} \right|$ for all $\boldsymbol{d}^{(i)} \in \mathcal{D}_{\text{active}}$
    $\mathcal{C} \leftarrow$ top-$\min(B, |\mathcal{D}_{\text{active}}|)$ indices of c_i
    if $\mathcal{C} = \emptyset$ **then**
      Add h to $\mathcal{H}_{\text{new}}$ ▷ leaf reached; keep hypothesis
      continue
    for each $i \in \mathcal{C}$ **do**
      $\mathcal{S}' \leftarrow \mathcal{S} \cup \{i\}$; ▷ extend path
      $\boldsymbol{z}' \leftarrow \arg\min_{\boldsymbol{w}} \|\boldsymbol{x} - \boldsymbol{D}_{\mathcal{S}'}\boldsymbol{w}\|_2^2$
      $\boldsymbol{r}' \leftarrow \boldsymbol{x} - \boldsymbol{D}_{\mathcal{S}'}\boldsymbol{z}'$
      Add $h' \leftarrow (\mathcal{S}', \boldsymbol{r}', i)$ to $\mathcal{H}_{\text{new}}$
  Prune: keep top-$\min(B, |\mathcal{H}_{\text{new}}|)$ hypotheses with smallest $\|\boldsymbol{r}'\|_2^2$
  $\mathcal{H}^{(t+1)} \leftarrow$ pruned set
Return $\boldsymbol{z}_{h^\star}$ where $h^\star \in \arg\min_{h\in\mathcal{H}^{(t)}} \|\boldsymbol{r}_h\|_2$

B. Proofs

B.1. Proof of Proposition 3.1

Statement. If subtree containment (Eq. (2)) and sibling-cone disjointness (Eq. (3)) hold, then the subtrees rooted at sibling nodes do not overlap.

Proof. We show that these two conditions are sufficient to guarantee that sibling subtrees are disjoint. Suppose node k is a descendant of node j, which is a child of parent i.

Subtree containment (Eq. (2)): Since $k \in \mathrm{desc}(j)$, Eq. (2) gives

$$\angle(\boldsymbol{a}^{(j)}, \boldsymbol{a}^{(k)}) \le \theta_{\mathrm{lev}(j)}. \tag{12}$$

Thus, every descendant of j lies within the cone of half-angle $\theta_{\mathrm{lev}(j)}$ rooted at $\boldsymbol{a}^{(j)}$, so in particular k is confined to this cone.

Sibling-cone disjointness (Eq. (3)): Consider any sibling $j' \in \mathrm{chi}(i)$ with $j' \neq j$. To derive a contradiction, assume that k also lies in the subtree of j', i.e., $k \in \mathrm{desc}(j')$. Then, by subtree containment applied to j', we similarly obtain

$$\angle(\boldsymbol{a}^{(j')}, \boldsymbol{a}^{(k)}) \le \theta_{\mathrm{lev}(j')}. \tag{13}$$

Consider the spherical triangle formed by the unit vectors $\boldsymbol{a}^{(j)}/\|\boldsymbol{a}^{(j)}\|$, $\boldsymbol{a}^{(k)}/\|\boldsymbol{a}^{(k)}\|$, and $\boldsymbol{a}^{(j')}/\|\boldsymbol{a}^{(j')}\|$ on the unit sphere. By the spherical triangle inequality, the angle between any two vertices is at most the sum of the angles to the third vertex:

$$\begin{aligned} \angle(\boldsymbol{a}^{(j)}, \boldsymbol{a}^{(j')}) &\le \angle(\boldsymbol{a}^{(j)}, \boldsymbol{a}^{(k)}) + \angle(\boldsymbol{a}^{(k)}, \boldsymbol{a}^{(j')}) \\ &\le \theta_{\mathrm{lev}(j)} + \theta_{\mathrm{lev}(j')}, \end{aligned} \tag{14}$$

which contradicts Eq. (3). Thus, no node k can simultaneously belong to the subtrees of two siblings j and j', so the subtrees rooted at sibling nodes do not overlap. □

B.2. Proof of Proposition 3.2

Statement. If the half-angles satisfy the geometric decrease $\theta_{l+1} \le r\,\theta_l$ with $r \in (0, 1/2)$, then *subtree containment* (Eq. (2)) . If *sibling-cone disjointness* (Eq. (3)), then $\theta_{l+1} \le \frac{1}{b}\,\theta_l$ holds. Thus there exists a placement of the nodes such that if the half-angles satisfy $\theta_{l+1} \le \min\{r, 1/b\}\,\theta_l$ with $r \in (0, 1/2)$, then *subtree containment* (Eq. (2)) and *sibling-cone disjointness* (Eq. (3)) hold.

Proof. We first show that the subtree containment condition (Eq. (2)) holds. A necessary and sufficient condition to Eq. (2) is that the cumulative half-angles of all the lower levels do not exceed the half-angle budget we have for this level

$$\sum_{k=l+1}^{L} \theta_k \le \theta_l, \quad \forall i \in \{1, \dots, N_L\},\ l = \mathrm{lev}(i). \tag{15}$$

Now assume that the half-angles satisfy the geometric decrease with rate $r \in (0, 1/2)$:

$$\theta_{l+1} \le r\,\theta_l. \tag{16}$$

Then, for any level l we have that

$$\sum_{k=l+1}^{L} \theta_k \le \sum_{k=1}^{L-l} \theta_l\, r^k \text{(by Eq. (16))} \tag{17}$$

$$= \theta_l \sum_{k=1}^{L-l} r^k \tag{18}$$

$$= \theta_l\, r \sum_{k=0}^{L-l-1} r^k \tag{19}$$

$$\le \theta_l\, r\, \frac{1 - r^{L-l-1}}{1 - r} \text{(sum of a geometric series).} \tag{20}$$

Taking $L \to \infty$ yields

$$\theta_l\, r \frac{1 - r^{L-l-1}}{1 - r} \to \theta_l\, \frac{r}{1 - r} \le \theta_l, \qquad r < \frac{1}{2},$$

which proves Eq. (15) and hence the subtree cone condition Eq. (2).

For the sibling cones under a parent cone of half-angle θ_l to be disjoint, it is necessary that each sibling cone's half-angle obeys

$$\theta_{l+1} \le \frac{1}{b}\,\theta_l. \tag{21}$$

In any 2D plane containing the cone axis, a cone of half-angle α appears as a planar angle of magnitude 2α. Packing b child cones of angle $2\theta_{l+1}$ inside the parent angle $2\theta_l$ requires $b\theta_{l+1} \le \theta_l$. □

B.3. Proof of Proposition 3.3

Statement. Under the hierarchical orthogonality constraints in Eq. (30) and the regular-simplex difference condition in Eq. (32) at every internal node up to depth L in ambient space $\mathbb{R}^d$, it is necessary that the ambient dimension satisfies the depth–dimension condition $d \ge L + b$.

Proof. Fix a level $l \ge 0$ and consider the path of ancestors $\boldsymbol{A}_l = \{\boldsymbol{a}^{(\pi_0)}, \dots, \boldsymbol{a}^{(\pi_l)}\}$. The hierarchical orthogonality constraints Eq. (30) define the affine feasible set for child candidates as

$$\mathcal{V}_l = \{\boldsymbol{x} \in \mathbb{R}^d : \boldsymbol{A}_l^\top \boldsymbol{x} = \boldsymbol{h}_l\},$$

which is Eq. (35). When the ancestor vectors are linearly independent (the generic case, since each level introduces a new non-collinear direction), we have

$$\dim \mathcal{V}_l = d - (l + 1).$$

The regular-simplex difference condition Eq. (32) requires placing b child points whose differences relative to a feasible origin in $\mathcal{V}_l$ form a regular $(b-1)$-simplex. This simplex has affine hull of dimension $b-1$; therefore it can be embedded in $\mathcal{V}_l$ only if

$$\dim \mathcal{V}_l \ \ge\ b - 1.$$

Combining the two displays yields, for every level l, the necessary inequality $d - (l + 1) \ge b - 1 \iff d \ge l + b$. Requiring this to hold up to the deepest level L gives $d \ge L + b$. □

B.4. Intermediate results for Proposition 4.1

Lemma B.1 (Column normalization equivalence)**.** *Let $\boldsymbol{D} = [\boldsymbol{d}_1, \dots, \boldsymbol{d}_k] \in \mathbb{R}^{d \times k}$ with arbitrary nonzero column norms ($\|\boldsymbol{d}_j\|_2 > 0$ for all $j \in [k]$), and define the diagonal matrix $\boldsymbol{W} := \mathrm{diag}(\|\boldsymbol{d}_1\|_2, \dots, \|\boldsymbol{d}_k\|_2)$ and the column-normalized dictionary $\widehat{\boldsymbol{D}} := \boldsymbol{D}\,\boldsymbol{W}^{-1}$. For any s-sparse $\boldsymbol{z} \in \mathbb{R}^k$ with support S, set $\widehat{\boldsymbol{z}} := \boldsymbol{W}\boldsymbol{z}$. Then $\boldsymbol{x} = \boldsymbol{D}\boldsymbol{z} = \widehat{\boldsymbol{D}}\,\widehat{\boldsymbol{z}}$ and $\mathrm{supp}(\widehat{\boldsymbol{z}}) = S$. Moreover, OMP run on $\boldsymbol{D}$ with the selection rule*

$$j^\star \in \arg\max_j \frac{\big|\langle \boldsymbol{r}, \boldsymbol{d}_j \rangle\big|}{\|\boldsymbol{d}_j\|_2} = \arg\max_j \frac{\big|\langle \boldsymbol{r}, \boldsymbol{d}_j \rangle\big|}{\|\boldsymbol{d}_j\|_2\, \|\boldsymbol{r}\|_2} \tag{22}$$

is identical (same index picked at every iteration) to OMP run on $\widehat{\boldsymbol{D}}$ with the usual (unnormalized) correlation rule. Equivalently, this selects the atom with the highest absolute cosine similarity to the residual.

Proof. Immediate from $\widehat{\boldsymbol{d}}_j = \boldsymbol{d}_j / \|\boldsymbol{d}_j\|_2$ and $\langle \boldsymbol{r}, \widehat{\boldsymbol{d}}_j \rangle = \langle \boldsymbol{r}, \boldsymbol{d}_j \rangle / \|\boldsymbol{d}_j\|_2$, together with $\boldsymbol{x} = \boldsymbol{D}\boldsymbol{z} = \boldsymbol{D}\,\boldsymbol{W}^{-1}\boldsymbol{W}\boldsymbol{z} = \widehat{\boldsymbol{D}}\,\widehat{\boldsymbol{z}}$. Since diagonal rescaling of the columns in $\boldsymbol{D}_S$ leaves the column span unchanged, the orthogonal projector onto

$\mathrm{span}(\boldsymbol{D}_S)$ is invariant to such rescaling. Therefore, once the same index is selected, the least-squares updates utilize the same projector, and the residuals match at every step. □

Definition B.2 (ERC on normalized dictionary)**.** [5] For a support S with $\widehat{\boldsymbol{D}}_S$ full column rank, define

$$\mathrm{ERC}(\widehat{\boldsymbol{D}}; S) := \left\| \widehat{\boldsymbol{D}}_S^{\dagger} \widehat{\boldsymbol{D}}_{S^c} \right\|_{\infty}, \tag{23}$$

$$\mathrm{ERC}(\widehat{\boldsymbol{D}}; S \,|\, T) := \left\| \widehat{\boldsymbol{D}}_S^{\dagger} \widehat{\boldsymbol{D}}_{T \setminus S} \right\|_{\infty}, \tag{24}$$

for any $T \supseteq S$.

Lemma B.3 (Monotone ERC improvement under subtree restriction)**.** *Let $\boldsymbol{D} \in \mathbb{R}^{d \times k}$ have arbitrary nonzero column norms and let $\boldsymbol{z}$ be s-sparse with support S. Let $T_0 \supset T_1 \supset \cdots \supset T_L$ be a nested sequence with $T_0 = [k]$ and $S \subseteq T_\ell$ for all $\ell = 0, \ldots, L$. Assume $\widehat{\boldsymbol{D}}_S$ has full column rank. Then the ERC decreases monotonically along the restriction:*

$$\mathrm{ERC}(\widehat{\boldsymbol{D}}; S \,|\, T_L) \le \mathrm{ERC}(\widehat{\boldsymbol{D}}; S \,|\, T_{L-1}) \tag{25}$$

$$\le \cdots \le \mathrm{ERC}(\widehat{\boldsymbol{D}}; S \,|\, T_0) \tag{26}$$

$$= \mathrm{ERC}(\widehat{\boldsymbol{D}}; S). \tag{27}$$

Proof. The quantity $\widehat{\boldsymbol{D}}_S^{\dagger}$ is fixed, and shrinking T only removes columns from $\widehat{\boldsymbol{D}}_{T \setminus S}$, so the maximum defining the ERC is taken over a subset and therefore cannot increase. □

Lemma B.4 (ERC threshold implies restricted OMP success)**.** *Under the assumptions of Lemma B.3, if $\mathrm{ERC}(\widehat{\boldsymbol{D}}; S \,|\, T_L) < 1$, then OMP run on $\boldsymbol{D}$ with the normalized selection rule*

$$j^* \in \arg\max_j \frac{|\langle \boldsymbol{r}, \boldsymbol{d}_j \rangle|}{\|\boldsymbol{d}_j\|_2} = \arg\max_j \frac{|\langle \boldsymbol{r}, \boldsymbol{d}_j \rangle|}{\|\boldsymbol{d}_j\|_2 \, \|\boldsymbol{r}\|_2}, \tag{28}$$

restricted to T_L, recovers S in s steps. Equivalently, OMP on $\widehat{\boldsymbol{D}}_{T_L}$ with the standard rule succeeds in s iterations.

Proof. Lemma B.1 shows that the normalized-selection rule on $\boldsymbol{D}$ matches standard OMP on $\widehat{\boldsymbol{D}}$. The classical noiseless ERC theorem of Tropp [61] applied to the restricted dictionary $\widehat{\boldsymbol{D}}_{T_L}$ then yields exact support recovery in s iterations whenever $\|\widehat{\boldsymbol{D}}_S^{\dagger} \widehat{\boldsymbol{D}}_{T_L \setminus S}\|_{\infty} < 1$. □

B.5. Proof of Proposition 4.1

Statement. There exist instances with $\mathrm{ERC}(\widehat{\boldsymbol{D}}; S) \ge 1$ yet $\mathrm{ERC}(\widehat{\boldsymbol{D}}; S \,|\, T_L) < 1$ for some nested $T_0 \supset T_1 \supset \cdots \supset T_L$ satisfying the right-subtree assumption $S \subseteq T_\ell$. Consequently, hierarchical OMP yields a strictly larger ERC-certified success region than global OMP on the full dictionary.

[5] This follows the classical exact recovery coefficient (ERC) of Tropp [61].

Proof. If the maximizer(s) contributing to $\mathrm{ERC}(\widehat{\boldsymbol{D}}; S)$ lie outside T_L, pruning them ensures $\mathrm{ERC}(\widehat{\boldsymbol{D}}; S \,|\, T_L) < \mathrm{ERC}(\widehat{\boldsymbol{D}}; S)$, so the restricted value can fall below 1 while the global one remains at least 1. Whenever this happens, Lemma B.4 certifies exact recovery for Hierarchical OMP on T_L, whereas the ERC test for OMP on the full dictionary fails. Thus, the subtree-restricted algorithm possesses a strictly larger guaranteed support-recovery region. □

C. Step-by-step construction of a Hierarchical Concept Embedding

Assume we are at depth $l > 0$ of the hierarchy. The path from the root to the *current parent* $\boldsymbol{a}^{(\pi_l)} \in \mathbb{R}^d$ consists of the $l+1$ ancestor vectors

$$\boldsymbol{A}_l = \{\boldsymbol{a}^{(\pi_0)}, \boldsymbol{a}^{(\pi_1)}, \ldots, \boldsymbol{a}^{(\pi_l)}\}, \qquad \pi_0 < \pi_1 < \cdots < \pi_l. \tag{29}$$

We must construct b children $\{\boldsymbol{a}^{(j)}\}_{j=1}^{b} \subset \mathbb{R}^d$ that satisfy:

(i) **Hierarchical Orthogonality**:

$$(\boldsymbol{a}^{(j)} - \boldsymbol{a}^{(\pi_k)})^{\top} \boldsymbol{a}^{(\pi_k)} = 0, \quad k = 0, \ldots, l, \tag{30}$$

$$j = 1, \ldots, b. \tag{31}$$

Note that the current parent $\boldsymbol{a}^{(\pi_l)}$ satisfies this condition through induction.

(ii) **Regular $(b-1)$-simplex structure**:

$$(\boldsymbol{a}^{(j)} - \boldsymbol{g}_l)^{\top} (\boldsymbol{a}^{(k)} - \boldsymbol{g}_l) = \begin{cases} \lambda_l^2, & j = k, \\ -\dfrac{\lambda_l^2}{b-1}, & j \ne k, \end{cases} \tag{32}$$

where $\boldsymbol{g}_l$ is any point satisfying all $l+1$ equations in Eq. (30) and λ_l is the scaling factor for the simplex so that $\angle(\boldsymbol{a}^{(j)}, \boldsymbol{a}^{(\pi_l)}) = \theta_l$.

(iii) **Cone condition w.r.t. the current parent**:

$$\angle(\boldsymbol{a}^{(j)}, \boldsymbol{a}^{(\pi_l)}) = \theta_l$$

$$\iff \|\boldsymbol{a}^{(j)} - \boldsymbol{a}^{(\pi_l)}\| = \|\boldsymbol{a}^{(\pi_l)}\| \tan\theta_l, \tag{33}$$

$$j = 1, \ldots, b.$$

Note that this iff condition is true because $\langle \boldsymbol{a}^{(j)} - \boldsymbol{a}^{(\pi_l)}, \boldsymbol{a}^{(\pi_l)} \rangle = 0$ due to Eq. (30).

C.1. Feasible Subspace induced by Hierarchical Orthogonality

Let

$$\boldsymbol{A}_l = \begin{bmatrix} \boldsymbol{a}^{(\pi_0)} & \boldsymbol{a}^{(\pi_1)} & \ldots & \boldsymbol{a}^{(\pi_l)} \end{bmatrix} \in \mathbb{R}^{d \times (l+1)}. \tag{34}$$

The $l+1$ hyperplanes in (30) intersect in the affine subspace

$$\mathcal{V}_l = \{\boldsymbol{x} \in \mathbb{R}^d : \boldsymbol{A}_l^{\top} \boldsymbol{x} = \boldsymbol{h}_l\}, \tag{35}$$

where $\boldsymbol{h}_l = [\|\boldsymbol{a}^{(\pi_0)}\|^2, \ldots, \|\boldsymbol{a}^{(\pi_l)}\|^2]^\top$. If the ancestor columns of $\boldsymbol{A}_l$ are linearly independent[6], then

$$\dim \mathcal{V}_l = d - (l+1). \tag{36}$$

To be able to embed a regular $(b-1)$-simplex for all depths we therefore require the *depth-dimension condition*

$$d \geq L + b. \tag{37}$$

Equation (37) quantifies the depth–dimension trade-off: one ambient degree of freedom is lost per additional ancestor constraint, while $(b-1)$ directions are always needed to accommodate the regular simplex of conditionally independent children.

Our goal is to construct the children of a node $\boldsymbol{a}^{(\pi_l)}$ at level l.

1. **Find one feasible origin.** Solve the linear system $A_l^\top \mathbf{x} = \mathbf{h}_l$ to obtain any particular solution $\boldsymbol{g}_l \in \mathcal{V}_l$. If we take into account the cone half-angle condition, we can further reduce the set of solutions to the intersection of $\mathcal{V}_l$ and the cone centered at the parent $\boldsymbol{a}^{\pi_l}$ with the half-angle θ_l (which we defined in Section 3).

   One solution that gives us the most half-angle budget is choosing the current parent $\boldsymbol{a}^{(\pi_l)}$ as the origin, i.e. $\boldsymbol{g}_l = \boldsymbol{a}^{(\pi_l)}$. By construction, every node is orthogonal to all of its ancestors. Hence for each $k \in \{0, \ldots, l\}$

$$(\boldsymbol{a}^{(\pi_l)} - \boldsymbol{a}^{(\pi_k)})^\top \boldsymbol{a}^{(\pi_k)} = 0 \tag{38}$$

$$\implies \boldsymbol{a}^{(\pi_l)\top} \boldsymbol{a}^{(\pi_k)} = \|\boldsymbol{a}^{(\pi_k)}\|^2. \tag{39}$$

2. **Basis for the difference linear space.** Compute an orthonormal basis

$$\boldsymbol{U}_l \in \mathbb{R}^{d \times (d-l-1)}, \tag{40}$$

$$\boldsymbol{A}_l^\top \boldsymbol{U}_l = \mathbf{0}, \tag{41}$$

$$\boldsymbol{U}_l^\top \boldsymbol{U}_l = I_{d-l-1}, \tag{42}$$

   e.g. by taking the $d-(l+1)$ bottom left singular vectors of $\boldsymbol{A}_l$, denoted as $\boldsymbol{U}_{l+2:d}$.

3. **Canonical regular simplex in $\mathbb{R}^{b-1}$.** Use the centred construction $\tilde{\boldsymbol{d}}_i = \boldsymbol{e}_i - \frac{1}{b}\mathbf{1},\ i = 1, \ldots, b$ (cf. Eq. (7)).

4. **Scale $\tilde{\boldsymbol{d}}_j$ to satisfy the cone condition.**

$$\lambda_l = \|\boldsymbol{a}^{(\pi_l)}\| \tan\theta_l, \tag{43}$$

$$\boldsymbol{d}_j = \lambda_l \tilde{\boldsymbol{d}}_j. \tag{44}$$

5. **Embed and translate.**

$$\boldsymbol{a}^{(j)} = \boldsymbol{a}^{(\pi_l)} + \boldsymbol{U}_{l+2:d}\, \boldsymbol{d}_j, \qquad j = 1, \ldots, b. \tag{45}$$

[6]This is typical because every level adds a new non-collinear vector.

Choosing the scale factor

$$\lambda_l = \|\boldsymbol{a}^{(\pi_l)}\| \tan\theta_l, \tag{46}$$

forces $\|\boldsymbol{a}^{(j)} - \boldsymbol{a}^{(\pi_l)}\| = \|\boldsymbol{a}^{(\pi_l)}\| \tan\theta_l$ for all j, and therefore $\angle\big(\boldsymbol{a}^{(j)}, \boldsymbol{a}^{(\pi_l)}\big) = \theta_l$, which is precisely the requirement in Eq. (33).

D. Additional Experimental Results

D.1. Additional Synthetic Experiment Details

Branching factor $b = 3$, hierarchy depth $L = 7$, dimension $d = 50$. Initial cone half angle = 85 degrees. Initial vector norm = 0.8. Geometric reduction factor = 0.4. Total leaf nodes = 2187. Total nodes = number of atoms = 3280. Gaussian noise for each leaf for data generation $\sigma^2 = 10^{-5}$. Generate 5 samples per leaf for a total of 10,935 samples.

D.2. Additional Real-Data Experiment Details

Model Architecture and Training Details. We use CLIP-ViT-L/14 as the backbone. To train the linear classifier, we use the AdamW optimizer [38] with a weight decay of 10^{-4} and a learning rate of 10^{-1}. To train the CBM model, we use the Adam optimizer with a learning rate of 10^{-1} and train for 500 epochs. We provide the detailed hyperparameters in Tab. 3.

Synset Difference Interpretations. We use CLIP text embeddings [52] and GPT-5 [47] to generate the text interpretations of the synset differences. We provide the top-10 concepts for each parent-child pair in Tab. 4. We also use the GPT-5 prompt in App. D.2.

Ablation Study on the Beam Size. We perform an ablation study on the beam size for Hierarchical OMP. We vary the beam size from 1 to 8 and evaluate the concept recovery accuracy on ImageNette. We provide the results in Fig. 10.

Taxonomy Generation Prompt. We use the taxonomy generation prompt from Zeng et al. [63] in App. D.2 to generate the taxonomy on CIFAR100.

Table 3. Key hyperparameters used in experiments for each dataset.

Hyperparameter	ImageNette	CIFAR100	ImageNet
Batch size	4096	4096	16384
Classification training epochs	500	500	1000
HCEP beam size	8	16	32

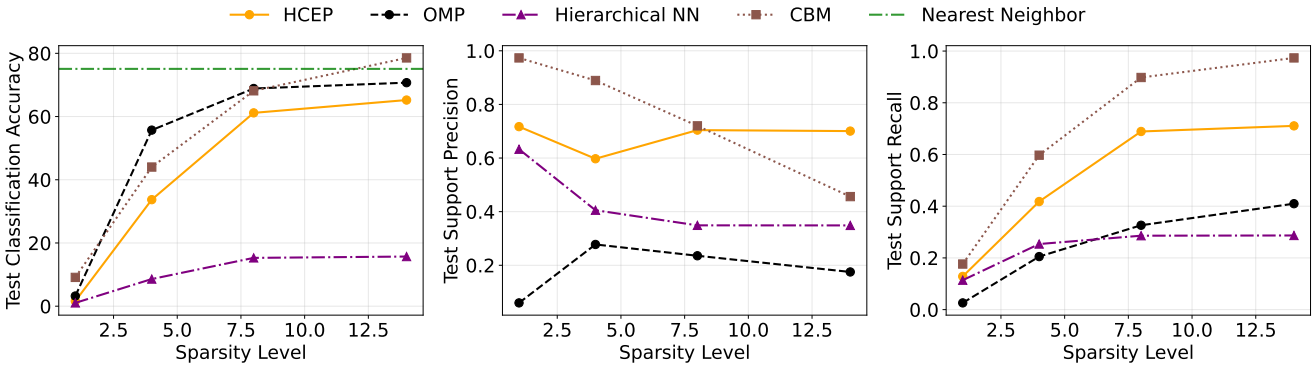

Figure 8. On ImageNet, HCEP achieves competitive accuracy while having higher concept precision/recall than sparse concept prediction baselines (OMP, Hierarchical NN).

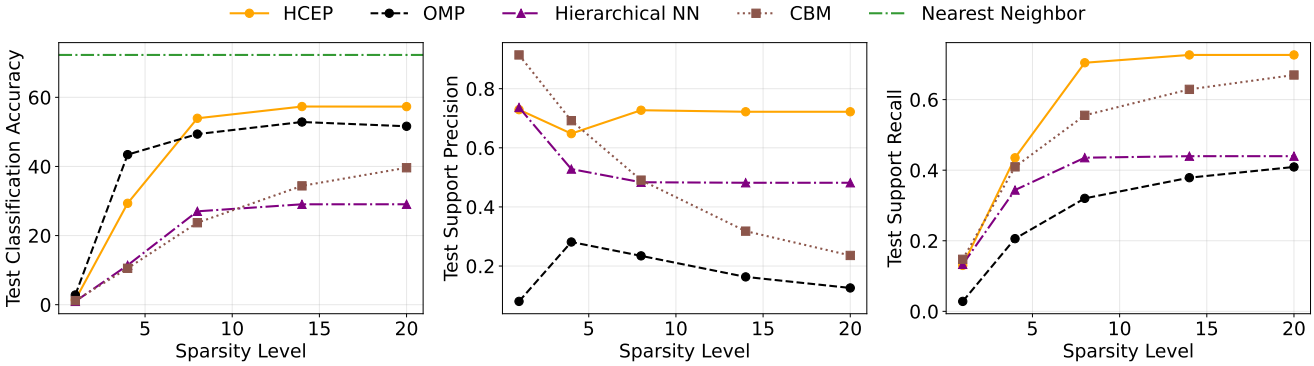

Figure 9. When we restrict ImageNet training set to 25 images per class, HCEP outperforms all interpretable baselines.

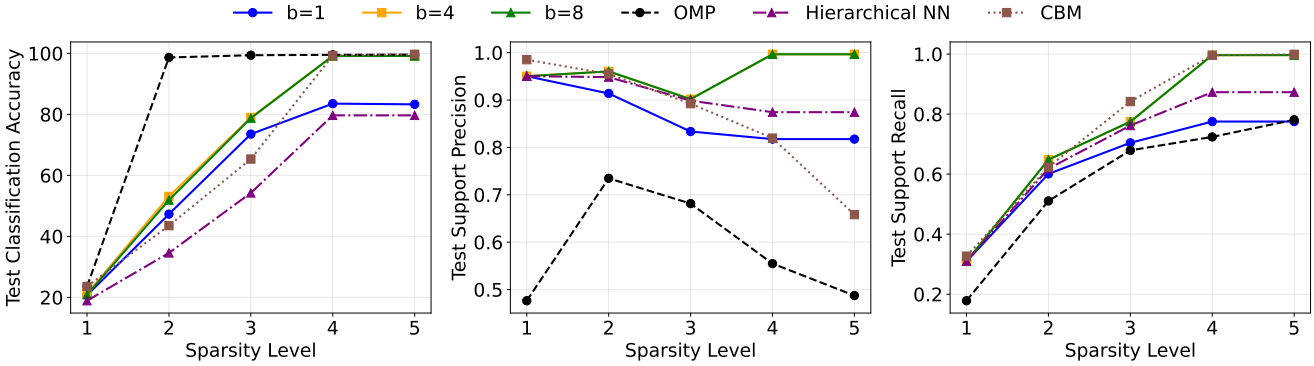

Figure 10. We vary the beam size in the range of (1, 4, 8) and evaluate on ImageNette.

Table 4. Example text interpretations of child–parent synset differences produced via CLIP embeddings and GPT summarization, with the top-10 contributing concepts.

Parent→Child Pair	Top-10 Concepts	Text Interpretation
`bear` → `polar bear`	1. Matte white texture 2. Thick white winter fur coat 3. White plumage in winter season 4. White fur with cream patches 5. Long, silky white coat 6. White fur blending with surroundings 7. Pure white fluffy coat 8. White coat with lemon markings 9. White cue ball 10. Long, corded white coat	thick matte white fur blending with snow.
`container` → `basket`	1. Wicker baskets filled with baguettes 2. Rectangular shopping baskets 3. Stacked woven baskets 4. Rear storage basket 5. Rectangular open-top basket 6. Woven rattan backrest 7. Plastic shopping baskets 8. Perforated cutlery basket in lower rack 9. Woven rattan seating surfaces 10. Rectangular basket frame	open-top woven or perforated sides with handles.
`structure` → `lumbermill`	1. Vertical log slicing machines 2. Massive log cutting machines 3. Metallic sawmill machinery 4. Heavy-duty sawmill frames 5. Conveyor belts with wood pieces 6. Wooden board sorting systems 7. Exposed wooden axles 8. Wooden log feeding chutes 9. Stacks of cut wooden planks 10. Narrow wooden steering wheel	vertical log-sawing machines and plank conveyors.

Taxonomy Generation Prompt

Given root concept `<root>` and leaf concepts `<leaves>`, generate a detailed hierarchical taxonomy that organizes these leaves under the root. Create multiple levels of intermediate category hierarchies to build a rich, fine-grained classification structure. Use as many hierarchical levels as needed to create meaningful semantic groupings and subgroupings.
The format is: 1. Parent Concept 1.1 Child Concept 1.1.1 Grandchild Concept.
CRITICAL: Every single leaf concept from the list must appear exactly as given in the taxonomy as the deepest level nodes. You may and should add multiple levels of intermediate concepts but do not add new leaf concepts. Before finishing, verify that each leaf concept from `<leaves>` appears in your taxonomy. Aim for depth and semantic richness in the hierarchy.

Taxonomy Generation Prompt

TASK: Generate a concise phrase (3-10 words) that describes what distinguishes "<child>" from its parent category "<parent>". This is for a hierarchical sparse model. A residual represents the visual difference between a child and parent category. Most correlated visual concepts (from CLIP embeddings): <concepts_string> REQUIREMENTS:
1. Generate ONE short phrase (3-10 words) that captures the key distinguishing features
2. Base your phrase on the correlated concepts provided above
3. Focus on the most salient visual features
4. Be specific and concrete, not vague or generic
5. Use natural language that a human would use to describe the difference
6. IMPORTANT: Do NOT use the synset names ("<parent>" or "<child>") in your phrase
7. IMPORTANT: Describe only the visual features, not the category name
EXAMPLES OF GOOD PHRASES: - "tawny coat with distinctive facial markings" (for a specific dog breed) - "long curved neck and pink coloration" (for flamingo vs bird) - "striped pattern and elongated body" (for a specific fish) - "metallic surface with cylindrical shape" (for a lighter)

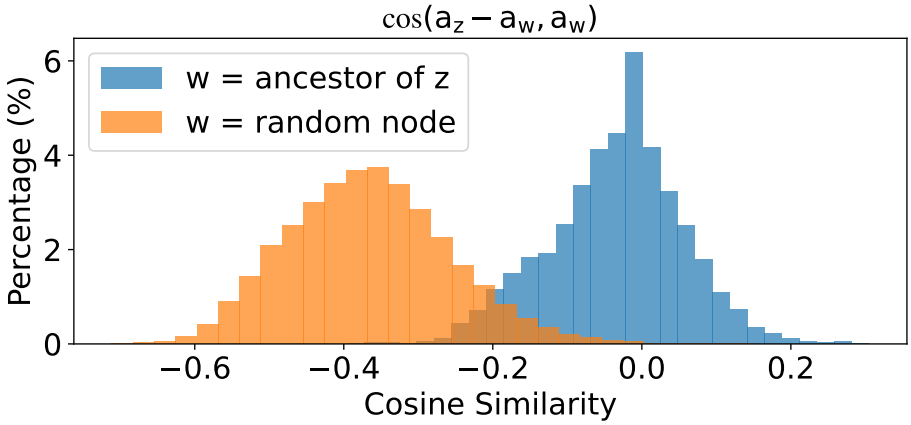

Figure 11. Observed Hierarchical Orthogonality test on CIFAR100. The cosine similarity between child-parent difference vectors and their parents is close to zero, while random non-parent pairs have significantly lower cosine similarity.