

Can this Model Also Recognize Dogs? Zero-Shot Model Search from Weights

Jonathan Kahana¹ Or Nathan¹ Eliahu Horwitz¹ Yedid Hoshen¹

Abstract

With the increasing numbers of publicly available models, there are probably pretrained, online models for most tasks users require. However, current model search methods are rudimentary, essentially a text-based search in the documentation, thus users cannot find the relevant models. This paper presents ProbeLog, a method for retrieving classification models that can recognize a target concept, such as "Dog", without access to model metadata or training data. Differently from previous probing methods, ProbeLog computes a descriptor for each output dimension (logit) of each model, by observing its responses on a fixed set of inputs (probes). Our method supports both logit-based retrieval ("find more logits like this") and zero-shot, text-based retrieval ("find all logits corresponding to dogs"). As probing-based representations require multiple costly feedforward passes through the model, we develop a method, based on collaborative filtering, that reduces the cost of encoding repositories by $3\times$. We demonstrate that ProbeLog achieves high retrieval accuracy, both in real-world and fine-grained search tasks and is scalable to full-size repositories. <https://jonkahana.github.io/probelog>

1. Introduction

Neural networks have revolutionized fields like computer vision (He et al., 2016; Dosovitskiy, 2020; Redmon, 2016; Radford et al., 2021; Li et al., 2023; Rombach et al., 2022) and natural language processing (Touvron et al., 2023; Devlin, 2018; Vaswani, 2017), becoming indispensable tools for many real-world classification tasks. However, their high training cost leaves users with two suboptimal options: i) invest heavily in computational resources for training or fine-tuning a model, ii) settle for a general-purpose model that may not suit their task. Now, imagine that instead, one could simply search online for the most accurate model

¹School of Computer Science and Engineering
The Hebrew University of Jerusalem, Israel. Correspondence to:
Jonathan Kahana <jonathan.kahana@mail.huji.ac.il>.

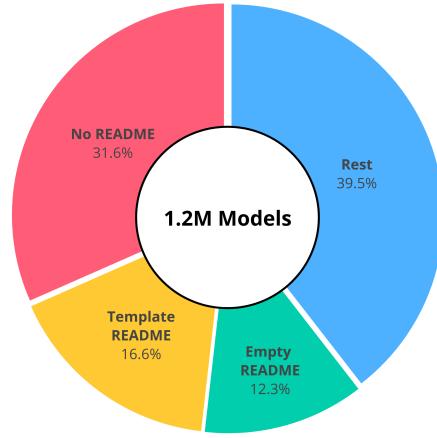


Figure 1. Hugging Face Documentation. We analyze the model cards of 1.2M Hugging Face models. We discover that the majority of models are either undocumented or poorly documented.

for their specific task and use it directly without additional training. With the rise of large public model repositories, this is becoming feasible. For instance, Hugging Face, the largest existing model repository, hosts over 1 million models, with more than 100,000 models added monthly. This significantly increases the likelihood of finding a suitable public model for most user tasks. The main challenge, however, lies in retrieving the right model for each task. While current model search methods (Shen et al., 2024; Luo et al., 2024) rely on provided metadata or text descriptions, most models in practice are either undocumented or have very limited descriptions (See Fig. 1), which severely limits the ability of these search methods to retrieve suitable models.

We aim to search for new models based on their weights, without assuming access to their training data or metadata, as these are often unavailable. More precisely, the goal is to retrieve all classification models capable of recognizing a particular concept, such as "Dog". For a solution to be effective and practical, it must meet several requirements: i) identifying models that recognize the target concept regardless of the other concepts they can detect, ii) being invariant to model output class order, iii) scaling to large model repositories, and iv) supporting text-based search. Using a single representation to describe models is suboptimal for this task, as the target concept may only account for a small part of

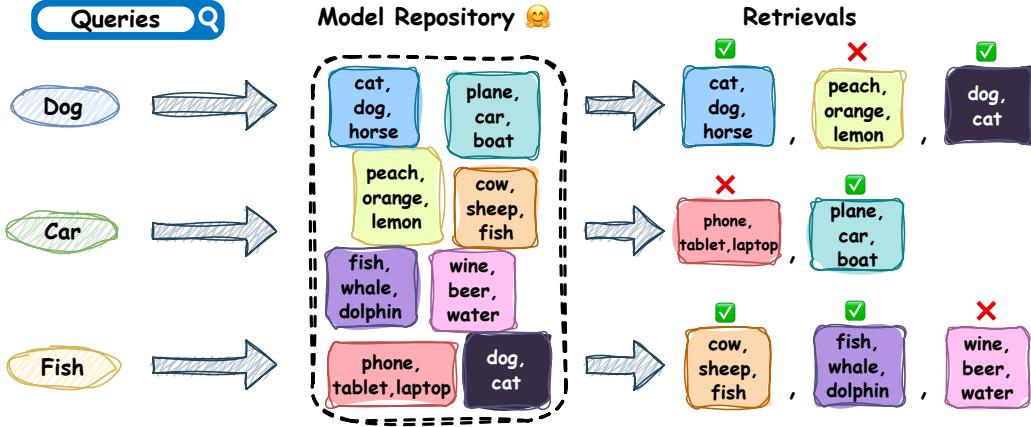


Figure 2. Classification Model Search. We present a new task of Classification Model Search, where the goal is to find classifiers that can recognize a target concept. Concretely, given an input prompt, such as “Dog”, we wish to retrieve all classifiers that one of their classes is “Dog”. The search space is a large model repository, that contains many models and concepts to search from. The retrieved models can replace model training, increasing accuracy, reducing cost and environmental impact.

the representation. Model-level representations are often overly large, suffer from permutation variation and may be insensitive to the target concept.

In this paper, we introduce *ProbeLog*, a probing based logit-level descriptor especially designed for model search. Since our goal is to identify a functional property of the model (what it does), the descriptor is a functional representation (Herrmann et al., 2024), essentially it describes what the logit does. To compute the ProbeLog descriptor for a specific logit in a given model, we first query the model with a fixed set of pre-determined input samples (probes) and monitor its responses in the specific output dimension. By normalizing the response vector across all probes, we obtain the ProbeLog descriptor. Its dimension is equal to the number of probes. An illustration of ProbeLog descriptors is provided in Fig. 3. Crucially, unlike prior methods for analyzing neural network weights (Lu et al., 2023), our approach represents logits rather than the models, which are more suitable for search.

ProbeLog representations enable searching by logit (“more like this”), but do not allow searching for unseen concepts (“find models that recognize ‘dogs’”). Probably the most convenient way to achieve such zero-shot concept search is to incorporate text. Therefore, we propose to use a text alignment model (e.g., CLIP (Radford et al., 2021)) between the probes and target concept name to compute a zero-shot ProbeLog representation. After suitable domain normalization, this approach achieves accurate zero-shot search. To make ProbeLog practical for model search, we must address several questions. How does the choice of probes affect the representations? How can we choose effective probes suitable for various concepts? What similarity criteria should be used to compare between ProbeLog representations from

two separate models? To answer these questions, in Sec. 5.4 we conduct a thorough study of these questions and propose strategies to address them. As another core contribution, we present Collaborative Probing, a method to significantly reduce the cost of creating representations for a repository. Instead of probing all models with all probes, we only use a random selection of the probes for each model. We then complete the missing information with matrix-factorization based collaborative filtering. This results in greatly improved performance for low probe numbers.

We showcase ProbeLog’s effectiveness on two real-world datasets that we curate: one based on models that we train and the other containing models that we download from Hugging Face. Our method is scalable and can handle large models with high effectiveness and efficiency. It achieves high retrieval accuracy, reaching over 40% top-1 accuracy when predicting whether a model can recognize an ImageNet (Deng et al., 2009) target concept from text. As the retrieval accuracy of a random method only scores 0.1% (since there are 1,000 possible classes), our method’s performance is significant. Furthermore, we establish the strong performance of our Collaborative Probing approach.

Our main contributions are:

1. Introducing ProbeLog, an effective logit-level model representation based on probing.
2. Developing a method to extend this representation to zero-shot representations, enabling text-based model search.
3. Proposing Collaborative Probing to reduce the number of probes required for gallery encoding.

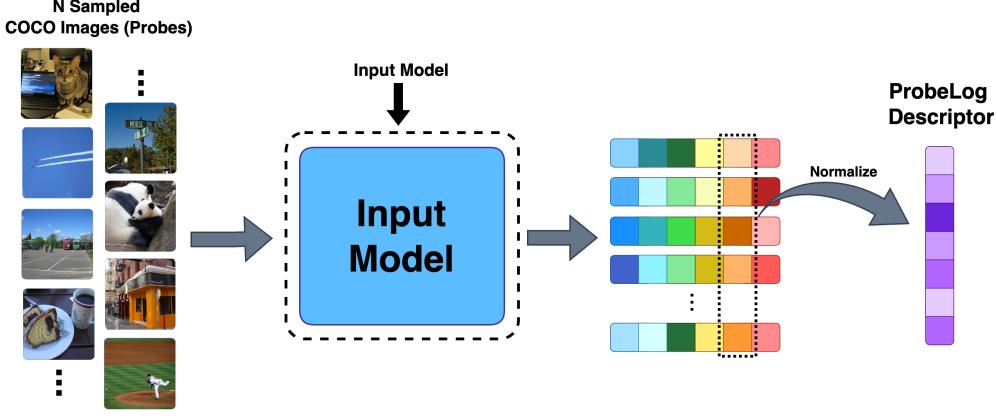


Figure 3. ProbeLog Descriptors. Our method generates a descriptor for individual output dimensions (logits) of models. First, we sample and a set of inputs (e.g., from the COCO dataset), and fix them as our set of probes. Then, to create a new ProbeLog descriptor for a model logit, we feed the set of ordered probes into the model and observe their outputs. Finally, we take all values of the logit we wish to represent, and normalize them. We use this representation to accurately retrieve model logits associated with similar concepts. In Fig. 5, we extend this idea to zero-shot concept descriptors.

2. Related Works

2.1. Weight-Space Learning

While neural networks can learn effective representations for many traditional data modalities, effective representations for neural networks are still work in progress. As a first step, Unterthiner et al. (2020) proposed to observe simple statistics of weights, and (Ke et al., 2017) on them. Others proposed to encode the weights by modeling the connections between the neurons (Navon et al., 2023; De Luigi et al., 2023; Schürholz et al., 2024; 2021; Eilertsen et al., 2020; Lim et al., 2024; Zhou et al., 2024a; Tran et al., 2024; Dupont et al., 2022; Horwitz et al., 2024a). Recent methods (Kofinas et al., 2024; Zhou et al., 2024b; Lim et al., 2023; Kalogeropoulos et al., 2024) model a network as a graph where every neuron is a node, and train permutation-equivariant architectures (Gilmer et al., 2017; Kipf & Welling, 2016; Vaswani, 2017; Diao & Loynd, 2022) on these graphs. Probing is an alternative paradigm that encodes the network by observing its outputs to a fixed set of inputs (probes) (Kahana et al., 2024; Herrmann et al., 2024; Carlini et al., 2024; Tahan et al., 2024; Choshen et al., 2022; Kofinas et al., 2024; Huang et al., 2024; Dravid et al., 2023; Bau et al., 2017). Differently from these approaches, we propose a probing-based method for zero-shot classification model search.

2.2. Other Applications of Model Weights

Learning on model weights has found many applications. Several approaches demonstrated advanced generation abilities using the weights (Dravid et al., 2024; Erkoç et al., 2023; Dravid et al., 2024; Shah et al., 2023), and others proposed to compress the weights to a smaller, more com-

pact representation (Ha et al., 2016; Ashkenazi et al., 2022; Peebles et al., 2022). A different line of research explored the relations between the weights for recovering the model graph (Horwitz et al., 2024c; Yax et al., 2025) or for merging (Yadav et al., 2024; Gueta et al., 2023; Izmailov et al., 2018; Wortsman et al., 2022; Ramé et al., 2023). Recently, a few works proposed to recover the exact black-boxed weights (Horwitz et al., 2024b; Carlini et al., 2024) by having access to their fine-tuned versions or to an API. Finally, some relevant works search for new adapters for generative models (Shen et al., 2024; Luo et al., 2024; Lu et al., 2023), however these approaches either rely on available metadata or tailored for generative models. Here, we propose an approach to search for new discriminative models which are capable of detecting a specific concept among other unrelated concepts seen in training time.

3. Background and Motivation

3.1. Problem Definition: Model Search

We assume a model repository composed of m classifiers, f_1, f_2, \dots, f_m . Each classifier f_i can have multiple output dimensions (logits), and each corresponding to an unknown concept $c_{i,j}$. The user then inputs a text prompt containing some query concept, c_q , they wish to search for. Finally, the goal is to return a model f_i such that one of its classes matches the query concept. Formally, the set of all valid retrieval models, $R(c_q)$, is defined as:

$$R(c_q) = \{f_i \mid \exists j \text{ s.t. } c_{i,j} = c_q\} \quad (1)$$

As mentioned above, the retrieval algorithm does not know the class concepts of each model. We assume access to them solely for evaluation purposes.

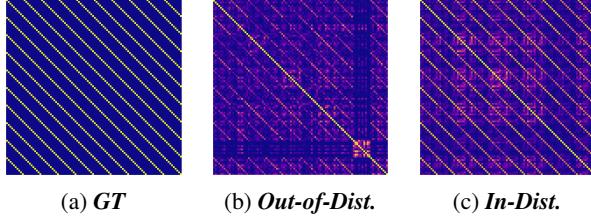


Figure 4. CIFAR10 Logit Similarities.(a) Ground truth label. (b) ProbeLog representations using 1,000 out-of-distribution COCO image probes. (c) ProbeLog representations using 1,000 in-distribution CIFAR10 image probes. Both find meaningful similarities, although in-distribution probes work better.

3.2. The Challenge

While a trivial solution is to create model-level representations, this idea encounters serious setbacks. First, representing models by their weights is difficult and computationally expensive due to their high dimensionality and complex symmetries (Kofinas et al., 2024; Kahana et al., 2024). Second, encoding an entire model is not suitable for functionality-based search. To illustrate, consider a classifier that separates between “Dog”, “Cat” and another one for “Dog” and “Lion”. Despite both including the target concept “Dog”, each of them will have a different encoding. Moreover, even classifiers with identical classes that are ordered differently (“Dog”-“Cat” vs. “Cat”-“Dog”) may produce distinct representations. To overcome this limitation and ensure invariance to other detected classes and their order, we propose to have a separate descriptor (representation) for each output dimension of each model.

3.3. Real Models are Poorly Documented

The existing solution for model search is text-based search in the user uploaded documentation. To understand the effectiveness of this solution, we explore the level of documentation of models in [Hugging Face](#), the largest model repository. For that, we analyzed all 1.2M model cards. As shown in Fig. 1, over 30% of all models have no model card at all. Moreover, there are another 28.9% of model cards that are either empty or include an empty automatic template with no information. The remaining 40% of model cards may include some information, however we cannot determine exactly how many of them include relevant information about the training data. As most models are poorly documented we conclude that searching models by weights alone is a practical and useful setting.

4. Method

4.1. ProbeLog: Logit-Level Descriptors

Our objective is to accurately and efficiently find relevant models in a large repository that can recognize a target con-

cept, e.g., “Dog”. Instead of using a single representation for the entire model, we represent each model output (logit) separately. Our method for extracting logit descriptors first presents each model with a set of n ordered, fixed input samples (probes). Intuitively, these are a set of standardized questions that we ask the model. In practice, we compose the list of probes by randomly sampling images (without replacement) from an image dataset. For generality, most of our results use the COCO dataset (Lin et al., 2014) which is highly diverse but also out-of-distribution to our models. We investigate the choice of probe dataset in Sec. 5.4. We input each probe x_j into the model f , obtaining the output $f(x_j)[i]$ for the model’s i^{th} logit. We define the ProbeLog descriptor for logit i of model f as the responses of all probes at this logit:

$$\phi(f, i) = [f(x_1)[i], f(x_2)[i], \dots, f(x_n)[i]] \quad (2)$$

Fig. 3 presents an overview of ProbeLog extraction.

To validate that logit responses to probes provide an effective description of the semantic function, we present a simple experiment. We take 10 different ViT foundation models, each trained via a different procedure and fine-tune them on the CIFAR10 (Krizhevsky et al., 2009) classification task (classifying small images into one of 10 object categories). We randomly sample 1,000 ImageNet (Deng et al., 2009) images as probes and run them through the model, computing the ProbeLog description of each logit in each model (100 in total). We then compute the correlation between all pairs of logits, and present the correlation matrix in Fig. 4. We observe that logit responses to probes are mostly correlated to those of logits with a matching semantic concept instead of, for example, logits from the same model.

4.2. A Discrepancy Measure for Logit-Level Descriptors

For downstream tasks such as model retrieval, we need to compute the discrepancy between pairs of logit-level ProbeLog descriptors. However, naive metrics such as Euclidean or correlation yield subpar results. We hypothesize that models are only reliable for probes they are confident about, while their responses exhibit high variance for the others. To mitigate this phenomenon, we propose focusing only on probes for which the query logit has high confidence about. We introduce an asymmetric discrepancy measure, specifically designed for logit-level comparisons. Given a query logit descriptor, ϕ , we sort its values (probe responses) from highest to lowest. Let $a = [a_1, a_2, \dots, a_n]$ be the indices of the sorted entries in descending order. We then reorder all gallery descriptors using the same index sequence a . Lastly, we compute the discrepancy between the query and each of the gallery descriptors by measuring the difference (in L_2) only over the top k probe entries of the sorted descriptors:

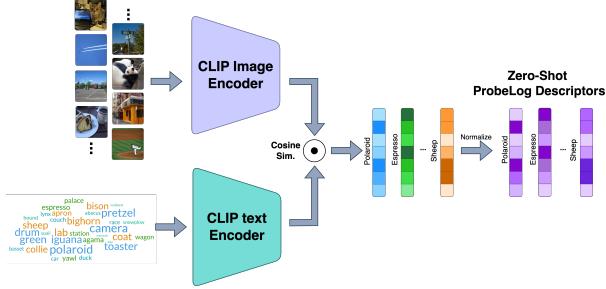


Figure 5. Text-Aligned ProbeLog Representation. We present a method to create ProbeLog-like representations for text prompts. We encode and store each of our ordered probes using the CLIP image encoder. At inference time, we embed the target text prompt, and compute its similarity with respect to the stored probe representations. We demonstrate that by normalizing this zero-shot ProbeLog descriptor, we can effectively search descriptors of real model logits, accurately retrieving similar concepts.

$$d(\phi, \phi') = \sqrt{\sum_{i=1}^k (\phi_{a_i} - \phi'_{a_i})^2} \quad (3)$$

where $d(\phi, \phi')$ is the discrepancy between the query descriptor ϕ and a gallery descriptor ϕ' . In Sec. 5.4 we show the importance of these design choices.

4.3. Text-Aligned ProbeLog Descriptors

The previous sections provided a way to search by logit, essentially finding similar logits to an existing one. This limits its applicability as it assumes the user already has such model. In this section, we extend our method to searching by text, thus allowing the user to search for concepts without already having such model, making it zero-shot. To do so, we present a method for generating ProbeLog descriptors from text alone. We use a multimodal text alignment model. For example, when the inputs are images, we choose CLIP, a joint text-image embeddings model. We use the multimodal model to extract embeddings from each probe a_i as well as from a user description α_{text} of the target concept. We define the zero-shot ProbeLog descriptor of the target concept as the vector of dot products between the embeddings of each probe and that of the target text:

$$\phi_{text} = [\alpha_i \cdot \alpha_{text}, \alpha_2 \cdot \alpha_{text}, \dots, \alpha_n \cdot \alpha_{text}] \quad (4)$$

Using our discrepancy measure between the logit and the zero-shot ProbeLog descriptors does not achieve good results as their numerical values are in different scales. To reduce this domain gap, we normalize each descriptor by its mean and standard deviation. The normalized ProbeLog descriptor is:

$$\phi(f, i) \leftarrow \frac{\phi(f, i) - \mu_{f,i}}{\sigma_{f,i}} \quad (5)$$

, where $\mu_{f,i}$ and $\sigma_{f,i}$ indicate the mean and standard deviation of $\phi(f, i)$ respectively. We illustrate the creation of our zero-shot ProbeLog descriptors in Fig. 5.

4.4. Collaborative Probing

Creating the ProbeLog representations for an entire model repository can be very costly, as it requires computing many forward passes for millions of models. Reducing this number of probes is critical for making the method feasible. Therefore, we propose Collaborative Probing. For each model, we randomly sample $p\%$ of the probes, and compute the ProbeLog representation just on these probes, masking out the entries for the other probes. We can therefore describe the ProbeLog descriptors for the logits of all models in the repository as a sparse matrix X , with $1 - p\%$ of entries missing, where $X_{i,j}$ is the response of logit i to probe j . The core idea is to use missing data imputation methods to complete this matrix, thus cheaply computing the full ProbeLog representations while actually probing each model with only a small fraction of the probes.

To complete the matrix X , we use the truncated SVD algorithm (Koren et al., 2009) that famously won the Netflix prize for movie recommendation. The idea is to decompose matrix X into low-rank matrices U, V such that:

$$U^*, V^* = \arg \min_{U,V} |(U^T V - X) \odot M|^2 \quad (6)$$

where M is the mask matrix that has all ones except for zeros for masked entries of X . We solve this optimization problem using iterative optimization. This involves alternating between fixing U while optimizing V and vice versa until convergence. By the end of optimization, we compute $\tilde{X} = U^T V$, the completed matrix. Computing the zero-shot ProbeLog embedding does not require any modification, as the probe embeddings can be cached. At inference time, the text embedding requires a single forward pass, and the zero-shot ProbeLog descriptor requires a single matrix vector multiplication. The retrieval then proceeds normally.

5. Experiments

5.1. Experimental Setting

Datasets. As there are no suitable existing datasets for model search that include ground-truth data, we created 2 new ones, INet-Hub and HF-Hub. For each model in the INet-Hub, we sample a subset of ImageNet classes, a model architecture and foundation model initialization checkpoint. We then train the model on the selected data. The final dataset consists of 1,500 models, making a total of more than 85,000 logits (consisting of 1000 unique fine-grained concepts). For more details, see App. A. Our second hub, HF-Hub, is a set of 71 real-world models (400 logits) downloaded manually from HuggingFace. As these data were

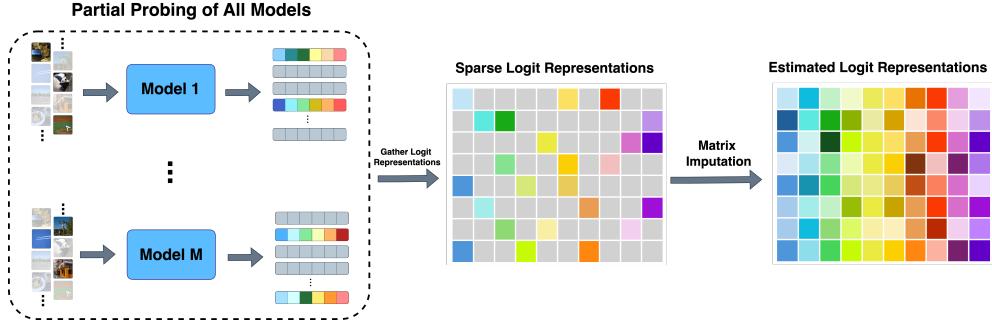


Figure 6. Collaborative Probing. We pass a random subset of probes through each model in the repository to obtain partial logit representations. By performing factorization based matrix imputation we can complete the missing information. This saves a substantial part of the computational resources needed to build our repository’s logit descriptors gallery.

Table 1. Retrieval Results. We evaluate the Top-1 and Top-5 retrieval accuracies of our method and the baselines for search-by-logit and search-by-text. All methods use COCO images as probes. For a fair comparison, all experiments are performed with 4,000 probes.

Retrieval	Method	INet → INet	INet → HF	HF → INet	text → HF	text → INet
Top-1 Accuracy	Full Query	59.9% \pm 0.2	14.8% \pm 0.1	15.3% \pm 0.8	22.6% \pm 0.5	16.9% \pm 0.2
	Model-Level	0% \pm 0.	13.9% \pm 1.0	21.0% \pm 1.8	17.8% \pm 1.5	0% \pm 0.0
	Ours (ProbeLog)	72.8% \pm 0.2	26.1% \pm 0.8	40.6% \pm 0.3	34.0% \pm 1.5	43.8% \pm 1.1
Top-5 Accuracy	Full Query	82.8% \pm 0.1	31.5% \pm 0.1	19.7% \pm 0.8	38.6% \pm 1.1	22.8% \pm 0.2
	Model-Level	0% \pm 0.	34.6% \pm 0.6	51.6% \pm 2.0	38.8% \pm 1.8	0% \pm 0.0
	Ours (ProbeLog)	92.6% \pm 0.1	43.6% \pm 0.5	58.6% \pm 0.9	53.7% \pm 1.9	68.0% \pm 0.6

created by real Hugging Face users, the concepts names might partially overlap (e.g., “Apple” vs. “Apples”). We therefore manually label the allowed retrievals with respect to this dataset and to ImageNet classes (see App. B).

Baselines. We test our retrieval algorithm against two baselines: (i) model-level, and (ii) direct logit comparison. The model-level approach averages all ProbeLog descriptors of the model’s logits, and searches for a similar logit descriptor to that model-level representation. The logit-level baseline does not use our discrepancy metric, but computes the Euclidean distance between a pair of logit representations.

Metrics. We evaluate the retrieval performance using standard metrics: top-k accuracy and precision (with $k \in [1, 5]$). Top-k accuracy measures the percentage of target logits that had a relevant result in any of their top-k retrieved logits. Top-k precision measures the percentage of all top-k retrievals across all target concepts that were relevant.

5.2. Model Search Results

We evaluate our method on 3 scenarios and present the results in Tab. 1. Collaborative Probing is evaluated separately in Sec. 5.3. Here, we report top-1/5 accuracies, for top-5 precision results see App. C.

In the first scenario, we evaluate our performance when

target models come from the same distribution as the repository models. To test this, we split the INet-Hub into 2 distinct subsets, and evaluate the retrieval performance. In this setting, ProbeLog achieves excellent accuracy, with a top-1 accuracy of 70% i.e., more than two thirds of target logits have the correct concept as their top retrieval result.

The second scenario is the more difficult case, where the queries are out-of-distribution to the repository. To test this, we search for real model logits (HF-Hub) in the INet-Hub and vice versa. This is especially difficult as the INet-Hub contains logits corresponding to ImageNet classes that are quite fine-grained. Still, ProbeLog obtains top-1 retrieval accuracy of 40.6% in the $HF \rightarrow INet$ task, compared to both baselines which are at 21%.

In the search-by-text evaluation, we search for the closest retrievals to a zero-shot text descriptor in either the INet-Hub or the HF-Hub. We can see that in both cases, our approach greatly exceeds the baselines, reaching an impressive top-1 accuracy of 43.8% on the INet-Hub. Moreover, when tested on the HF-Hub we can see that our method generalizes to real-world models, as it finds suitable matches for more than a third of the queries in the first search result, and for more than half of the queries within the first 5 retrievals. This shows that while simple, our approach can generalize to a real-world scenarios where user models are searched for using just a simple text prompt.

Table 2. Dataset Ablations. We compare both real and synthetic probe distributions. While distributions closer to the model’s training data lead to better results, even out-of-distribution probes sampled from the COCO dataset retrieve relevant logits with high accuracy.

Method	Top-1 Accuracy			Top-5 Accuracy		
	HF → INet	text → HF	text → INet	HF → INet	text → HF	text → INet
Dead-Leaves	1.3% \pm 0.7	1.6% \pm 1.4	1.0% \pm 0.2	5.9% \pm 1.6	6.8% \pm 1.3	3.8% \pm 0.2
Stable-Diffusion	51.4% \pm 1.0	36.9% \pm 0.9	47.0% \pm 0.6	69.8% \pm 0.9	56.2% \pm 0.9	73.3% \pm 0.9
ImageNet	57.8% \pm 1.3	33.1% \pm 1.2	55.4% \pm 1.1	71.4% \pm 1.3	55.1% \pm 0.9	80.4% \pm 0.9
COCO	40.6% \pm 0.3	34.0% \pm 1.5	43.8% \pm 1.1	58.6% \pm 0.9	53.7% \pm 1.9	68.0% \pm 0.6

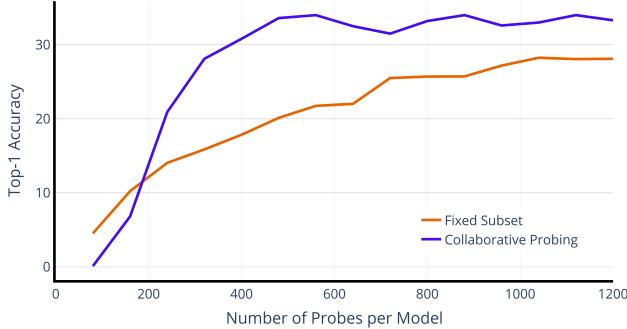


Figure 7. Collaborative Probing. We test our method using collaborative probing on the text → INet-Hub retrieval task. While the full size of the dataset is 8,000 COCO probes, we show cases where each model is probed by less than 15% of these probes. We can see that for the limited probe regime, collaborative probing can improve accuracy by as much as 2×.

5.3. Collaborative Probing

We compare Collaborative Probing, sampling a number of randomly selected probes for each model against simply using the same probes for all models. The results are presented in Fig. 7. While our collaborative probing technique requires around 400 probes per model to be effective, it can then substantially improve probing efficiency. Specifically, it reaches similar results as the standard approach with less than a third the number of probes. For example, having just 4% of all probes per model, is just as good as probing all models with 15% of all probes. This highlights the potential of our collaborative probing technique to significantly improve the efficiency of our search approach.

5.4. Ablation Studies

How to select the probe distribution? We showed (Sec. 5.2) that ProbeLog can generalize to real-world scenarios. Here, we conduct an ablation study, to test the effect of sampling probes from different distributions: (i) Dead-Leaves (Baradad Jurjo et al., 2021; Lee et al., 2001): a very coarse, hand-crafted generative model. (ii) ImageNet images. (iii) StableDiffusion (Rombach et al., 2022) samples using prompts of ImageNet-21K objects. (iv) COCO Images. Results, shown in Tab. 2, demonstrate a consistent

pattern: probes sampled from distributions that are closer to the target concept obtain more accurate retrievals. However, we note that even quite different probe distribution can yield high retrieval accuracies. E.g., even though COCO images are typically of scenes rather than objects, they are effective probes, reaching a top-5 accuracy of more than 60% when searching the INet-Hub by text. These results show that defining a general set of probes, which can retrieve a wide range of concepts is feasible. However, if a prior knowledge about the distribution of target concepts exists, then it is better to select in-distribution probes.

Which probes should be in the discrepancy metric? We proposed a discrepancy metric that compares the query and retrieved logits only on the probes that the query logit obtained large values on (Sec. 4.1). We ablate this choice of metric, comparing to several other probe selection criteria: lowest value probes, random sampling, uniform quantile sampling, highest value probes without normalization, and using all probes. The results, presented in Tab. 3, show that selecting the highest valued probes of the query logit is crucial for successful retrieval. We believe this is because logit values tend to be noisy, and highly confident values should be more consistent across logits of the same concept.

How many probes are enough? Fig. 8 presents results of text retrieval on INet-Hub using increasing numbers of probes. More probes lead to better results but with diminishing gains. For example, 4,000 COCO probes are enough for good performance of 43.8% top-1 accuracy, though it is possible to achieve a 47.8% using 8,000 probes.

6. Discussion

Non-random probe selection. We proposed an approach for searching models that can recognize a target concept. Our approach probes each model with 4,000 COCO images to produce the representation of each logit. However, we believe this number can be reduced substantially. For instance, while we chose the set of probes at random, it is likely that a smaller and more curated of probes exists. Specifically, core-set methods, which aim to reduce the number of training data, could potentially reduce this number.

Table 3. Logit Discrepancy Ablations. Our evaluation reveals: i) normalizing logit descriptors is necessary for accurate retrieval, especially for search-by-text. ii) choosing the most confident probes of the query logit is crucial, no other approach achieved comparable accuracy.

Selected Probes	Top-1 Accuracy			Top-5 Accuracy		
	HF→INet	text→HF	text→INet	HF→INet	text→HF	text→INet
Top- k + No Norm.	1.9% \pm 0.4	0.1% \pm 0.1	0% \pm 0.0	5.0% \pm 0.9	0.5% \pm 0.2	0.6% \pm 0.1
Bottom- k	0.8% \pm 0.3	1.3% \pm 0.9	2.3% \pm 0.5	1.2% \pm 0.3	6.8% \pm 0.5	7.9% \pm 0.9
Random	8.6% \pm 1.2	2.5% \pm 1.4	6.3% \pm 0.5	16.2% \pm 1.5	8.6% \pm 1.2	16.7% \pm 0.7
Quantiles	7.7% \pm 2.0	5.9% \pm 1.0	5.8% \pm 0.5	17.9% \pm 3.8	15.3% \pm 1.8	16.4% \pm 1.7
All	15.3% \pm 0.8	22.6% \pm 0.5	16.9% \pm 0.2	19.7% \pm 0.8	38.6% \pm 1.1	22.8% \pm 0.2
Top-k (Ours)	40.6%\pm0.3	34.0%\pm1.5	43.8%\pm1.1	58.6%\pm0.9	53.7%\pm1.9	68.0%\pm0.6

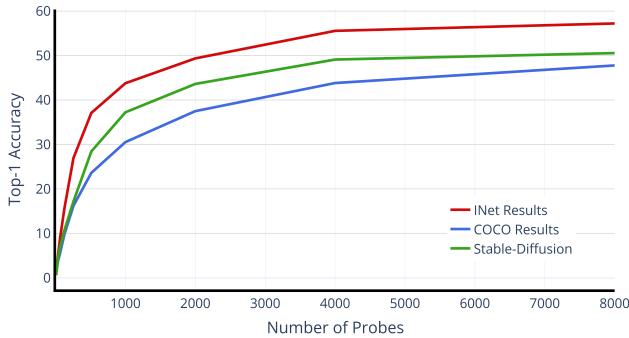


Figure 8. Number of Probes. We test our zero-shot retrieval approach on INet-Hub with increasing numbers of probes. While more probes lead to higher accuracy, the gains are diminishing.

Scaling-up to entire repositories. While our model hubs already have up to 1,500 large models, including ViTs (Dosovitskiy, 2020) and RegNet-Ys (Radosavovic et al., 2020), large model repositories may contain a millions of models. We chose to test our approach on smaller hubs mainly because we did not have the resources to probe a million models. However, given ProbeLog representations for all models, the search should be fast and well within the capabilities of most researchers. The descriptors are lightweight compared to actual model weights, and storing them is quite cheap. E.g., our INet-Hub models require 400GB of memory, but their logit 8,000 probes descriptors only consume 1.4GB. Also, our search algorithm operates in a space of a few tens of dimensions, where retrieval from even a billion entries is possible (Johnson et al., 2019; Jayaram Subramanya et al., 2019; Chen et al., 2021).

Improved collaborative probing. We showed that a simple collaborative filtering approach can significantly reduce the probing cost for repository. There are several ways to improve it. One direction is to develop a more sophisticated method for selecting which probes to sample for each model. This can be in an adaptive way i.e., sampling the first few prompts can inform the choice for the next probes. Another direction is to use improved collaborative filtering ideas which take into account the statistics of logit values. We

believe this is a fruitful avenue for future research.

7. Limitations

Extension beyond classification models. Our proposed method embeds each logit of each model on its own. This will require modification for generative models where the output dimensions do not explicitly encode the learned concepts. While some works attempted to search for generative adapters (Lu et al., 2023), they typically required many more (50,000) probes as their descriptors summarize the distribution of probe outputs. We believe that our methodology, where the inputs are ordered and fixed for all models, can reduce the number of probes substantially.

Out-of-distribution concepts. To enable search for diverse concepts we chose sampled probes from the COCO dataset (Lin et al., 2014) which does not just contain centered objects but also entire scenes. Still, this probe distribution does not represent all concepts, e.g. it does not include medical concepts. Successfully searching for such far OOD concepts will probably require selecting a probe distribution that is better aligned to the target concepts.

8. Conclusion

In this paper we propose an approach for searching for models in large repositories that can recognize a target concept. We first probe all models with a fixed, ordered set of probes, and define the values from each output dimension (logit) across all probes as a ProbeLog descriptor. We find that by normalizing these descriptors, we can compare them across different models, and even to zero-shot classifiers such as CLIP. With the pairwise discrepancy measure, we propose a method for searching models by text. We also present Collaborative Probing to significantly reduce the number of required probes at the same accuracy. We evaluate our approach on real-world models, and show it generalizes well to In-the-Wild models collected from HuggingFace. We ablated our design choices and showed they are crucial for effective model search.

References

- Ashkenazi, M., Rimon, Z., Vainshtein, R., Levi, S., Richardson, E., Mintz, P., and Treister, E. Nern–learning neural representations for neural networks. *arXiv preprint arXiv:2212.13554*, 2022.
- Baradad Jurjo, M., Wulff, J., Wang, T., Isola, P., and Torralba, A. Learning to see by looking at noise. *Advances in Neural Information Processing Systems*, 34:2556–2569, 2021.
- Bau, D., Zhou, B., Khosla, A., Oliva, A., and Torralba, A. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6541–6549, 2017.
- Carlini, N., Paleka, D., Dvijotham, K. D., Steinke, T., Hayase, J., Cooper, A. F., Lee, K., Jagielski, M., Nasr, M., Conmy, A., et al. Stealing part of a production language model. *arXiv preprint arXiv:2403.06634*, 2024.
- Chen, Q., Zhao, B., Wang, H., Li, M., Liu, C., Li, Z., Yang, M., and Wang, J. Spann: Highly-efficient billion-scale approximate nearest neighborhood search. *Advances in Neural Information Processing Systems*, 34:5199–5212, 2021.
- Choshen, L., Venezian, E., Don-Yehia, S., Slonim, N., and Katz, Y. Where to start? analyzing the potential value of intermediate models. *arXiv preprint arXiv:2211.00107*, 2022.
- De Luigi, L., Cardace, A., Spezialetti, R., Ramirez, P. Z., Salti, S., and Di Stefano, L. Deep learning on implicit neural representations of shapes. *arXiv preprint arXiv:2302.05438*, 2023.
- Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee, 2009.
- Devlin, J. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- Diao, C. and Loynd, R. Relational attention: Generalizing transformers for graph-structured tasks. *arXiv preprint arXiv:2210.05062*, 2022.
- Dosovitskiy, A. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.
- Dravid, A., Gandelsman, Y., Efros, A. A., and Shocher, A. Rosetta neurons: Mining the common units in a model zoo. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1934–1943, 2023.
- Dravid, A., Gandelsman, Y., Wang, K.-C., Abdal, R., Wetstein, G., Efros, A. A., and Aberman, K. Interpreting the weight space of customized diffusion models. *arXiv preprint arXiv:2406.09413*, 2024.
- Dupont, E., Kim, H., Eslami, S., Rezende, D., and Rosenbaum, D. From data to functa: Your data point is a function and you can treat it like one. *arXiv preprint arXiv:2201.12204*, 2022.
- Eilertsen, G., Jönsson, D., Ropinski, T., Unger, J., and Ynnerman, A. Classifying the classifier: dissecting the weight space of neural networks. In *ECAI 2020*, pp. 1119–1126. IOS Press, 2020.
- Erkoç, Z., Ma, F., Shan, Q., Nießner, M., and Dai, A. Hyperdiffusion: Generating implicit neural fields with weight-space diffusion. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 14300–14310, 2023.
- Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. Neural message passing for quantum chemistry. In *International conference on machine learning*, pp. 1263–1272. PMLR, 2017.
- Gueta, A., Venezian, E., Raffel, C., Slonim, N., Katz, Y., and Choshen, L. Knowledge is a region in weight space for fine-tuned language models. *arXiv preprint arXiv:2302.04863*, 2023.
- Ha, D., Dai, A., and Le, Q. V. Hypernetworks. *arXiv preprint arXiv:1609.09106*, 2016.
- He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016.
- Herrmann, V., Faccio, F., and Schmidhuber, J. Learning useful representations of recurrent neural network weight matrices. *arXiv preprint arXiv:2403.11998*, 2024.
- Horwitz, E., Cavia, B., Kahana, J., and Hoshen, Y. Representing model weights with language using tree experts. *arXiv preprint arXiv:2410.13569*, 2024a.
- Horwitz, E., Kahana, J., and Hoshen, Y. Recovering the pre-fine-tuning weights of generative models. In *ICML*, 2024b. URL <https://openreview.net/forum?id=761UxjOTHB>.
- Horwitz, E., Shul, A., and Hoshen, Y. On the origin of llamas: Model tree heritage recovery. *arXiv preprint arXiv:2405.18432*, 2024c.
- Huang, Q., Song, J., Xue, M., Zhang, H., Hu, B., Wang, H., Jiang, H., Wang, X., and Song, M. Lg-cav: Train any

- concept activation vector with language guidance. *arXiv preprint arXiv:2410.10308*, 2024.
- Izmailov, P., Podoprikin, D., Garipov, T., Vetrov, D., and Wilson, A. G. Averaging weights leads to wider optima and better generalization. *arXiv preprint arXiv:1803.05407*, 2018.
- Jayaram Subramanya, S., Devvrit, F., Simhadri, H. V., Krishnawamy, R., and Kadekodi, R. Diskann: Fast accurate billion-point nearest neighbor search on a single node. *Advances in Neural Information Processing Systems*, 32, 2019.
- Johnson, J., Douze, M., and Jégou, H. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547, 2019.
- Kahana, J., Horwitz, E., Shuval, I., and Hoshen, Y. Deep linear probe generators for weight space learning. *arXiv preprint arXiv:2410.10811*, 2024.
- Kalogeropoulos, I., Bouritsas, G., and Panagakis, Y. Scale equivariant graph metanetworks. *arXiv preprint arXiv:2406.10685*, 2024.
- Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T.-Y. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017.
- Kipf, T. N. and Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016.
- Kofinas, M., Knyazev, B., Zhang, Y., Chen, Y., Burghouts, G. J., Gavves, E., Snoek, C. G., and Zhang, D. W. Graph neural networks for learning equivariant representations of neural networks. *arXiv preprint arXiv:2403.12143*, 2024.
- Koren, Y., Bell, R., and Volinsky, C. Matrix factorization techniques for recommender systems. *Computer*, 42(8):30–37, 2009.
- Krizhevsky, A., Hinton, G., et al. Learning multiple layers of features from tiny images. 2009.
- Lee, A. B., Mumford, D., and Huang, J. Occlusion models for natural images: A statistical study of a scale-invariant dead leaves model. *International Journal of Computer Vision*, 41:35–59, 2001.
- Li, J., Li, D., Savarese, S., and Hoi, S. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pp. 19730–19742. PMLR, 2023.
- Lim, D., Maron, H., Law, M. T., Lorraine, J., and Lucas, J. Graph metanetworks for processing diverse neural architectures. *arXiv preprint arXiv:2312.04501*, 2023.
- Lim, D., Gelberg, Y., Jegelka, S., Maron, H., et al. Learning on loras: GL-equivariant processing of low-rank weight spaces for large finetuned models. *arXiv preprint arXiv:2410.04207*, 2024.
- Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6–12, 2014, Proceedings, Part V 13*, pp. 740–755. Springer, 2014.
- Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., and Xie, S. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11976–11986, 2022.
- Lu, D., Wang, S.-Y., Kumari, N., Agarwal, R., Tang, M., Bau, D., and Zhu, J.-Y. Content-based search for deep generative models. In *SIGGRAPH Asia 2023 Conference Papers*, pp. 1–12, 2023.
- Luo, M., Wong, J., Trabucco, B., Huang, Y., Gonzalez, J. E., Chen, Z., Salakhutdinov, R., and Stoica, I. Stylus: Automatic adapter selection for diffusion models. *arXiv preprint arXiv:2404.18928*, 2024.
- Navon, A., Shamsian, A., Achituve, I., Fetaya, E., Chechik, G., and Maron, H. Equivariant architectures for learning in deep weight spaces. In *International Conference on Machine Learning*, pp. 25790–25816. PMLR, 2023.
- Peebles, W., Radosavovic, I., Brooks, T., Efros, A. A., and Malik, J. Learning to learn with generative models of neural network checkpoints. *arXiv preprint arXiv:2209.12892*, 2022.
- Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.
- Radosavovic, I., Kosaraju, R. P., Girshick, R., He, K., and Dollár, P. Designing network design spaces. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10428–10436, 2020.
- Ramé, A., Ahuja, K., Zhang, J., Cord, M., Bottou, L., and Lopez-Paz, D. Model ratatouille: Recycling diverse models for out-of-distribution generalization. In *International Conference on Machine Learning*, pp. 28656–28679. PMLR, 2023.

- Redmon, J. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.
- Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10684–10695, 2022.
- Schürholz, K., Kostadinov, D., and Borth, D. Self-supervised representation learning on neural network weights for model characteristic prediction. *Advances in Neural Information Processing Systems*, 34:16481–16493, 2021.
- Schürholz, K., Mahoney, M. W., and Borth, D. Towards scalable and versatile weight space learning. *arXiv preprint arXiv:2406.09997*, 2024.
- Shah, V., Ruiz, N., Cole, F., Lu, E., Lazebnik, S., Li, Y., and Jampani, V. Ziplora: Any subject in any style by effectively merging loras. *arXiv preprint arXiv:2311.13600*, 2023.
- Shen, Y., Song, K., Tan, X., Li, D., Lu, W., and Zhuang, Y. Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face. *Advances in Neural Information Processing Systems*, 36, 2024.
- Tahan, S. A., Gera, A., Sznajder, B., Choshen, L., Dor, L. E., and Shnarch, E. Label-efficient model selection for text generation. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 8384–8402, 2024.
- Tan, M. and Le, Q. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pp. 6105–6114. PMLR, 2019.
- Tolstikhin, I. O., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Steiner, A., Keysers, D., Uszkoreit, J., et al. Mlp-mixer: An all-mlp architecture for vision. *Advances in neural information processing systems*, 34:24261–24272, 2021.
- Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.
- Tran, V.-H., Vo, T. N., The, A. N., Huu, T. T., Nguyen-Nhat, M.-K., Tran, T., Pham, D.-T., and Nguyen, T. M. Equivariant neural functional networks for transformers. *arXiv preprint arXiv:2410.04209*, 2024.
- Unterthiner, T., Keysers, D., Gelly, S., Bousquet, O., and Tolstikhin, I. Predicting neural network accuracy from weights. *arXiv preprint arXiv:2002.11448*, 2020.
- Vaswani, A. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.
- Wightman, R. Pytorch image models. <https://github.com/rwightman/pytorch-image-models>, 2019.
- Wortsman, M., Ilharco, G., Gadre, S. Y., Roelofs, R., Gontijo-Lopes, R., Morcos, A. S., Namkoong, H., Farhadi, A., Carmon, Y., Kornblith, S., et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International conference on machine learning*, pp. 23965–23998. PMLR, 2022.
- Yadav, P., Tam, D., Choshen, L., Raffel, C. A., and Bansal, M. Ties-merging: Resolving interference when merging models. *Advances in Neural Information Processing Systems*, 36, 2024.
- Yax, N., Oudeyer, P.-Y., and Palminteri, S. Phylolm: Inferring the phylogeny of large language models and predicting their performances in benchmarks. 2025.
- Zhou, A., Yang, K., Burns, K., Cardace, A., Jiang, Y., Sokota, S., Kolter, J. Z., and Finn, C. Permutation equivariant neural functionals. *Advances in neural information processing systems*, 36, 2024a.
- Zhou, A., Yang, K., Jiang, Y., Burns, K., Xu, W., Sokota, S., Kolter, J. Z., and Finn, C. Neural functional transformers. *Advances in neural information processing systems*, 36, 2024b.

A. INet-Hub Dataset Details

To simulate a model hub with many classifiers, we train 1,500 classifier models on different subsets of ImageNet classes. Each classifier is trained on a subset of between 15 and 200 classes, where the classes are chosen at random separately for each model. 90% of the classifiers are initialized from a foundation model, and the rest 10% are trained from scratch. The pre-training weights are selected from a set of 49 different models spanning various architectures including ViTs (Dosovitskiy, 2020), ResNets (He et al., 2016), RegNet-Ys (Radosavovic et al., 2020), MLP Mixers (Tolstikhin et al., 2021), EfficientNets (Tan & Le, 2019), ConvNets (Liu et al., 2022) and more. Each model is then trained for 2 – 5 epochs. This process results in a model hub with over 85,000 different logits to search for and 1,000 different fine-grained concepts. Below we list the possible pre-training weights of each model. All pre-training weights are taken from the timm library (Wightman, 2019).

- vit_base_patch32_clip_quickgelu_224.laion400m_e32
- vit_base_patch32_clip_224.laion400m_e32
- vit_base_patch32_clip_224.laion2b
- vit_base_patch32_clip_224.datacompxl
- convnext_base.clip_laiona
- convnext_base.clip_laion2b
- vit_base_patch32_clip_quickgelu_224.metaclip_400m
- vit_base_patch32_clip_quickgelu_224.metaclip_2pt5b
- vit_base_patch32_clip_224.metaclip_400m
- vit_base_patch32_clip_224.metaclip_2pt5b
- vit_base_patch32_clip_224.openai
- seresnextaa101d_32x8d.sw_in12k
- resmlp_24_224.fb_dino
- resmlp_12_224.fb_dino
- mixer_l16_224.goog_in21k
- mixer_b16_224.miil_in21k
- mixer_b16_224.goog_in21k
- resnetv2_152x2_bit.goog_in21k
- resnetv2_101x1_bit.goog_in21k
- resnetv2_50x1_bit.goog_in21k
- regnety_320.seer
- regnety_160.sw_in12k
- regnety_120.sw_in12k
- swin_tiny_patch4_window7_224.ms_in22k
- swin_base_patch4_window7_224.ms_in22k
- convnext_small.in12k
- convnext_tiny.in12k
- convnext_tiny.fb_in22k
- convnext_small.fb_in22k
- convnext_nano.in12k
- convnext_base.fb_in22k
- eca_nfnet_l0
- vit_base_patch16_224.dino
- vit_small_patch16_224.dino
- vit_base_patch16_224.mae
- vit_base_patch16_224.orig_in21k
- vit_base_patch32_224.orig_in21k
- vit_tiny_r_s16_p8_224.augreg_in21k
- vit_small_r26_s32_224.augreg_in21k
- vit_tiny_patch16_224.augreg_in21k
- vit_small_patch32_224.augreg_in21k
- vit_small_patch16_224.augreg_in21k
- vit_base_patch32_224.augreg_in21k
- vit_base_patch16_224_miil.in21k
- vit_base_patch16_224.augreg_in21k
- tf_efficientnetv2_s.in21k
- tf_efficientnetv2_m.in21k
- tf_efficientnetv2_l.in21k
- tf_efficientnetv2_b3.in21k

B. HF-Hub Dataset Details

In order to test our method on real-world data, we collected 71 classifiers uploaded by users to hugging face. Classifiers have between 2 and 82 classes they were trained on. Overall there are more than 400 possible logits in the dataset. The models are trained on a diverse set of models, and class names are given by free text. Hence, class names may not align perfectly as each user spells concept a bit differently (e.g., “Apple” vs. “Apples’). Moreover, some classifiers have different levels of granularity, such as “Car” vs. a specific car model “Toyota”. We therefore created a label mapping where we manually annotated to which classes each logit can be mapped. We follow these rules to allow mappings between labels: (i) Different spelling map to each other. (ii) An object can be mapped to a specific type of it, e.g. ”cat” - \downarrow ”siamese cat”. (iii) a specific type of object can be mapped to its super-class e.g. ”siamese cat” - \downarrow ”cat”. (iv) object of the same level of granularity that share a super class cannot be mapped to each other. For example, a ”Golden Retriever” is not a good match for a ”Husky”. Additionally, we created an additional mapping which matches each class to its corresponding ImageNet concept when available.

C. Additional Results

We present additional results with the baselines and the Top-5 precision metric as well, in Tab.4

Table 4. Retrieval Results. We provide the additional Top-5 retrieval precisions of our method and the baselines, over several scenarios. All methods use COCO images as probes.

Retrieval	Method	INet \rightarrow INet	INet \rightarrow HF	HF \rightarrow INet	text \rightarrow HF	text \rightarrow INet
Top-1 Accuracy	Full Query	59.9% \pm 0.2	14.8% \pm 0.1	15.3% \pm 0.8	22.6% \pm 0.5	16.9% \pm 0.2
	Model-Level	0% \pm 0.	13.9% \pm 1.0	21.0% \pm 1.8	17.8% \pm 1.5	0% \pm 0.0
	Ours (ProbeLog)	72.8% \pm 0.2	26.1% \pm 0.8	40.6% \pm 0.3	34.0% \pm 1.5	43.8% \pm 1.1
Top-5 Accuracy	Full Query	82.8% \pm 0.1	31.5% \pm 0.1	19.7% \pm 0.8	38.6% \pm 1.1	22.8% \pm 0.2
	Model-Level	0% \pm 0.	34.6% \pm 0.6	51.6% \pm 2.0	38.8% \pm 1.8	0% \pm 0.0
	Ours (ProbeLog)	92.6% \pm 0.1	43.6% \pm 0.5	58.6% \pm 0.9	53.7% \pm 1.9	68.0% \pm 0.6
Top-5 Precision	Full Query	61.2% \pm 0.1	9.7% \pm 0.1	13.6% \pm 0.3	13.1% \pm 0.3	14.1% \pm 0.1
	Model-Level	0% \pm 0.0	8.9% \pm 0.2	18.0% \pm 0.5	9.9% \pm 0.5	0% \pm 0.0
	Ours (ProbeLog)	71.2% \pm 0.1	15.4% \pm 0.4	39.4% \pm 0.9	20.8% \pm 0.4	39.7% \pm 0.8