FOUNDATION MODELS SECRETLY UNDERSTAND NEURAL NETWORK WEIGHTS: ENHANCING HYPERNETWORK ARCHITECTURES WITH FOUNDATION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Large pre-trained models, or foundation models, have shown impressive performance when adapted to a variety of downstream tasks, often out-performing specialized models. Hypernetworks, neural networks that generate some or all of the parameters of another neural network, have become an increasingly important technique for conditioning and generalizing implicit neural representations (INRs), which represent signals or objects such as audio or 3D shapes using a neural network. However, despite the potential benefits of incorporating foundation models in hypernetwork methods, this research direction has not been investigated, likely due to the dissimilarity of the weight generation task with other visual tasks. To address this gap, we (1) show how foundation models can improve hypernetworks with Transformer-based architectures, (2) provide an empirical analysis of the benefits of foundation models for hypernetworks through the lens of the generalizable INR task, showing that leveraging foundation models improves performance, generalizability, and data efficiency across a variety of algorithms and modalities. We also provide further analysis in examining the design space of foundation model-based hypernetworks, including examining the choice of foundation models, algorithms, and the effect of scaling foundation models.

1 INTRODUCTION

Foundation models, models that are pre-trained using self-supervision on diverse large-scale datasets and are readily adaptable to a wide variety of downstream tasks (Bommasani et al., 2021), have revolutionized AI as these models have formed the backbone for state-of-the-art models in many tasks across a wide range of modalities, such as zero-shot image classification (Radford et al., 2021) and segmentation (Kirillov et al., 2023). However, the use of foundation models has not been investigated for many downstream tasks where they may be useful.

Hypernetworks, which are neural networks that produce or adapt some or all of the weights of another neural network, have been investigated as a way to create adaptive layers (Ha et al., 2016; Ba et al., 2016; Goyal et al., 2019), perform neural architecture search (Brock et al., 2017; Zhang et al., 2018a), meta-learning (Andrychowicz et al., 2016; Zhao et al., 2020) and multi-task learning (Tay et al., 2020), continual learning (Von Oswald et al., 2019), and more. One major area of hypernetwork research is using hypernetworks as a means of conditioning or creating generalizable implicit neural representations (INRs) (Mescheder et al., 2019; Sitzmann et al., 2019; 2020b; Chen & Wang, 2022; Gu et al., 2023; Kim et al., 2023; Lee et al., 2024). INRs, also known as coordinate-based neural networks or neural fields, represent signals or objects using a neural network and have emerged as a continuous and memory-efficient alternative to traditional discrete representations (Sitzmann et al., 2020b). Typically, INRs are trained to represent a single object from many partial sensor observations of that object. Generalizable INRs improve this training framework by leveraging additional data to improve INR quality (Tancik et al., 2021; Chen & Wang, 2022; Gu et al., 2023; Kim et al., 2023; Lee et al., 2024), training efficiency (Tancik et al., 2021), and speed (Hong et al., 2023) as well as allow the generation of INRs for objects that would otherwise have insufficient partial observations (Hong et al., 2023).

Despite the success of foundation models for other tasks, there has been little investigation into adapting foundation models to improve hypernetworks and generalizable INRs, as none of the state-of-the-art methods (Chen & Wang, 2022; Gu et al., 2023; Kim et al., 2023; Lee et al., 2024) leverage foundation models. We believe that this is due to modality gap between neural network weights, the output of the hypernetwork task, which differs significantly from the outputs of more typical tasks such as image classification. We address this gap in the existing literature by answering the following questions: Do foundation models improve hypernetwork performance on the generalizable INR task? In which ways do they improve hypernetworks? And if they do, how should one design a hypernetwork from a foundation model? To answer these questions, we first augment Transformer-based generalizable INR architectures (Chen & Wang, 2022; Gu et al., 2023; Kim et al., 2023) with foundation models and show that adaptation via fine-tuning improves downstream task performance, generalization to unseen classes, and data efficiency. We also show that the foundation model approach can outperform existing approaches even when the foundation model features are frozen and only linear heads and extra tokens are trained on top of these frozen features to produce the weights of each layer. In addition, we provide an analysis of the design space of adapting foundation models to hypernetworks through targeted experiments exploring 1) the choice of foundation model, 2) the choice of algorithm, and 3) scaling properties. Finally, we show that the performance is robust by examining different modalities.

The contributions of our paper are as follows: first, using a framework based on existing Transformer-based hypernetworks, we show that foundation models improve hypernetwork performance on the generalizable INR task on different modalities. Second, we perform additional experiments to analyze many different facets of performance, examining generalization to unseen data, data efficiency, and parameter efficient approaches. We also provide additional analysis on the design space of adapting foundation models to hypernetworks, the choice of algorithm, examining the choice of foundation model, how performance scales with the number of foundation model parameters.

2 Background and Setup

In this section, we first describe how hypernetworks can be used to generate generalizable INRs. We then describe the architecture of the foundation model-based hypernetwork we use, which is closely based on the Trans-INR (Chen & Wang, 2022), a Transformer-based hypernetwork architecture that produces an INR in one forward pass of the hypernetwork. We then discuss the design space of hypernetworks and describe how parameter-efficient fine-tuning can be done using prompt tuning (Jia et al., 2022).

Hypernetworks for INRs Define a signal $I : X \to Y$ as a function that maps coordinates X to a space of quantities Y. An implicit neural representation (INR) represents the signal I by parameterizing it with a neural network f_θ with weights θ: $I(x) \approx f_\theta(x), \forall x$. INRs are typically trained with only partial observations v of the signal I using a forward map F that maps the outputs y to the partial observations v', and are supervised with a reconstruction loss between v, v'. For example, in Neural Radiance Fields (NeRF) (Mildenhall et al., 2021) the signal parameterized is the radiance field of a 3D scene (I) and the partial observations v are 2D views of the scene. The forward map F is volume rendering, an operation which takes the radiance field and produces the predicted partial 2D view v'. The INR is then supervised with a reconstruction loss between v and v', which is the mean-squared error (MSE) in this example.

A hypernetwork is a neural network that produces the weights of another neural network. In our case, a hypernetwork g_ϕ produces the weights of an INR f_θ given some partial observations v of the signal I: $g_\phi(v) = \theta$. The hypernetwork model is then supervised by a reconstruction loss between the v and the predicted partial observation $v' = F(f_{g_\phi(v)}(x))$, where x are the coordinates corresponding to v and F is the forward map. Only the hyperparameter weights ϕ are optimized using backpropagation.

Foundation Model Framework for Hypernetworks We base the foundation model framework for analysis (Figure 1) on the Trans-INR architecture (Chen & Wang, 2022), a Transformer-based hypernetwork architecture for generalizable INR. It consists of four main components: (1) a pre-trained Transformer foundation model consisting of an embedding layer `Embed` and d-dimensional Transformer encoder `Enc` consisting of attention blocks $\{B_i\}_{i=1}^N$, (2) extra learnable input tokens

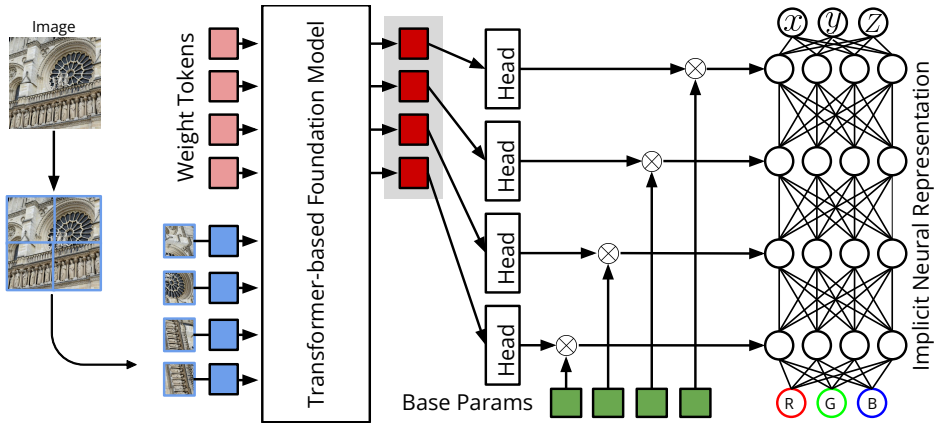


Figure 1: An overview of the hypernetwork-foundation model framework. First, an image is tokenized and concatenated with learnable weight tokens. Second, all tokens are encoded by a pre-trained foundation model encoder (Eq. 1). Tokens are then grouped, transformed using linear heads $\texttt{Head}_k$, and multiplied element-wise $\otimes$ with the base parameter $\texttt{BaseParam}_k$. (Eq. 2), and normalized (not shown). The resulting masked weights are then used to instantiate an implicit neural representation (INR). The INR can then be trained as usual.

$\{w_j^0\}_{j=1}^q$, (3) an INR f, generally a ReLU MLP with positional encoding composed of layers L_k, and (4) a learnable linear head $\texttt{Head}_k$ for each layer L_i of the INR, and (5) a set of learnable base parameters for the INR, which will be modulated by the output of each linear head $\texttt{Head}_k$. This improves training compared to directly producing the weights (Ortiz et al., 2023). The main difference between our framework and Trans-INR (Chen & Wang, 2022) is that our framework uses a pre-trained Transformer-based foundation model, whereas Trans-INR uses a Transformer encoder trained from scratch. Given an input data instance, such as a 2D view in the example above, it is discretized and embedded into $\mathbb{R}^d$ by the embedding layer $\texttt{Embed}$ to get data tokens $[t_1^0, \ldots, t_m^0]$. Superscripts indicate the number of attention blocks B_i a token has passed through. For simplicity, unlike Trans-INR we do not use $\texttt{Embed}$ to tokenize task- or modality-specific auxiliary data such as camera pose information in the novel view synthesis task. The extra weight tokens are then concatenated to the data tokens and fed through the transformer encoder:

$$[t_1^i, \ldots, t_m^i, w_1^i, \ldots, w_q^i] = B_i([t_1^{i-1}, \ldots, t_m^{i-1}, w_1^{i-1}, \ldots, w_q^{i-1}]), 1 \leq i \leq N \tag{1}$$

Finally, only the output tokens corresponding to weight tokens are used to generate the weights of each layer:

$$L_k = \texttt{Norm}(\texttt{Head}_k([w_{a_1}^N, \ldots, w_{a_r}^N]) * \texttt{BaseParam}_k) \tag{2}$$

where $\texttt{Norm}$ is an operation that normalizes the weights to have unit L_2 norm. For computational efficiency, we keep the weight grouping scheme of Trans-INR, where each token only helps to generate the weights of a single layer, with the number of tokens r being the total number of parameters in the layer divided by some hyperparameter g. More details can be found in Chen & Wang (2022). The model is trained end-to-end by generating the weights of the INR using the encoder ($\texttt{Embed}$ and $\texttt{Enc}$), using the INR to predict the data instance (see the previous section and Figure 1), and supervising the training with a reconstruction loss (not shown in Figure 1).

Prompt Tuning Transformer-based Hypernetworks As our framework is based on the Transformer architecture, parameter-efficient fine-tuning (PEFT) methods for Transformers such as prompt tuning (Jia et al., 2022) can be used almost directly. Prompt tuning is particularly simple because our framework already has learnable prompt tokens in the form of the weight tokens (the red tokens in Figure 1), so (shallow) prompt tuning can be achieved by just freezing the weights of the pre-trained foundation model encoder, consisting of the embedding layer $\texttt{Embed}$ and Transformer encoder $\texttt{Enc}$, and fine-tuning the remaining weights, which consist of the learnable weight tokens w_j^0, INR weight-producing linear heads $\texttt{Head}_K$, and base parameters $\texttt{BaseParam}_k$. Unlike prompt tuning, the token input to the linear heads corresponds to the learnable prompt tokens and not the data tokens.

3 EXPERIMENTS

3.1 EXPERIMENTAL SETUP

Pre-trained Backbones We experiment with the following large pre-trained models: supervised ViT (Dosovitskiy et al., 2020) trained on ImageNet-21k (Deng et al., 2009), DeiT (Touvron et al., 2021), a supervised model trained on ImageNet-1k (Deng et al., 2009) using distillation, DINO (Caron et al., 2021), a self-supervised model pre-trained with self-distillation, DINO v2 (Oquab et al., 2023), which improves on DINO with additional curated training data and other improvements, CLIP (Radford et al., 2021), a large vision-language model trained using an image-text contrastive loss, and MAE (He et al., 2022), which is pre-trained with masked image modeling. For audio, we use Whisper (Radford et al., 2023), an encoder-decoder model that is self-supervised on a variety of speech tasks. For the Whisper model, we only use the encoder.

Baselines In addition to our base framework, which is based on Trans-INR (Chen & Wang, 2022), we also examine two state-of-the-art extensions which are easily adapted to our framework by replacing their Transform backbone with pre-trained foundation models. PONP (Gu et al., 2023), representative of neural process-based/probabilistic methods, improves on Trans-INR by adapting the neural process (NP) (Garnelo et al., 2018b) meta-learning algorithm for generalizable INR learning. PONP learns a probabilistic INR instead of the deterministic one used by Trans-INR, with the output layer of the INR producing mean and variance predictions instead of point predictions. Instead of using MSE as a reconstruction loss, PONP uses the maximum-likelihood loss of conditional NPs (Garnelo et al., 2018a). Instance Pattern Composers (IPC) (Kim et al., 2023), representative of weight-sharing methods, improves on Trans-INR by using low-rank weight modulation to modulate just one weight matrix of one layer of the INR to instance-specific parameters while sharing all other weights of the INR among all data instances.

Training Training is done in one of three settings: 1) *randomly initialized*, where all the weights of the model are randomly initialized and then trained, 2) *fine-tuned*, *foundation model* or *FM*, where the Transformer encoder in our framework is initialized with the weights of a foundation model and the whole model is fine-tuned, and 3) *frozen* or *prompt tuned*, which corresponds to our prompt tuning approach where the Transformer encoder is initialized with foundation model weights and frozen.

3.2 TASKS

We experiment on the following datasets and tasks:

Novel view synthesis We use the novel view synthesis (NVS) dataset of LearnIt (Tancik et al., 2021), which consists of 50 rendered views of shapes in the *cars*, *chairs*, and *lamps* categories of the ShapeNet (Chang et al., 2015) 3D object dataset. Given a set of views of an object and a new viewing direction, the objective is to generate a view that best matches the ground truth view in that viewing direction. To fairly compare among different pre-trained models, unless otherwise stated we only examine models using the ViT-B/16 architecture. We restrict to the case where a single input view of the object is given. Unlike previous works (Tancik et al., 2021; Chen & Wang, 2022; Guo et al., 2023; Gu et al., 2023; Kim et al., 2023), unless otherwise stated we train on all categories at once instead of training a separate model for each category, a much harder task, and also evaluate using the average performance on each category. We numerically evaluate all methods with four metrics that cover different aspects of image similarity: peak signal-to-noise ratio (PSNR), SSIM (Wang et al., 2004), LPIPS (Zhang et al., 2018b), and FID (Heusel et al., 2017).

Audio reconstruction For audio reconstruction, following IPC (Kim et al., 2023) we use the LibriSpeech-clean audio dataset (Panayotov et al., 2015). The framework is trained on randomly cropped audio, while test audio is trimmed to 1s for evaluation (Kim et al., 2023), which is done with PSNR. For this method, we only benchmark the IPC algorithm.

3.3 MAIN RESULTS

Our main results can be found in Tables 1, 2, 3. We find that:

Table 1: Comparison of different foundation models using the ViT-B/16 architecture as backbones on the NVS task. All backbones examined outperform random initalization except for MAE (He et al., 2022), which we hypothesize is due to the lack of global image representation learning.

Backbone	PSNR (↑)	SSIM (↑)	LPIPS (↓)	FID (↓)
Randomly initialized	20.862	0.8357	0.1511	0.2751
MAE (He et al., 2022)	20.701	0.8312	0.1753	0.2866
Supervised (Dosovitskiy et al., 2020)	21.324	0.8501	0.0966	0.1516
DeiT (Touvron et al., 2021)	21.587	0.8530	0.1125	0.1860
DINO (Caron et al., 2021)	21.737	0.8555	0.1101	0.1810
CLIP (Radford et al., 2021)	21.770	0.8556	0.1126	0.1834
DINOv2 (Oquab et al., 2023)	22.095	0.8609	0.1063	0.1705

1. **In general, hypernetworks with large pre-trained models as backbones outperform hypernetworks with the same architecture trained from scratch, but the choice of pre-trained model matters** (§3.4). Initializing hypernetwork weights from large pre-trained model improves performance in general, although not all foundation models lead to improvements due to differences in pre-training strategy. In particular, learning a good global image representation seems to be crucial.
2. **Foundation models improve generalization to classes unseen during training** (§3.5), but full fine-tuning may cause some forgetting of generalizable foundation model features.
3. **Hypernetworks with frozen foundation model backbones have at least comparable performance to hypernetworks with the same architecture trained from scratch** (§3.7), while using significantly fewer learnable parameters (100K vs 87M parameters).
4. **Foundation model-based hypernetworks scale** (§3.6, §3.8) Hypernetworks augmented with foundation models are both more data efficient (§3.6) and perform better with larger foundation models (§3.8).
5. **The effects of foundation models are robust over different algorithms and different modalities** (§3.9) We find that hypernetwork performance increases across different algorithms, including as neural process-based/probabilistic methods (Gu et al., 2023) and weight-sharing methods (Kim et al., 2023), as well as over different modalities (3D objects and audio).

3.4 Foundation models increase hypernetwork performance

Table 1 shows that fine-tuning our foundation model framework for hypernetworks outperforms training from a random initialization for almost all of the investigated foundation models, with the exception of MAE (He et al., 2022). We hypothesize that the poor performance of MAE is due to its masked image modeling self-supervision, which learns good mid-level interaction between image patches (Li et al., 2022a), but fails to learn good global features Liang et al. (2022). We find that the three best foundation models are CLIP (Radford et al., 2021), DINO (Caron et al., 2021), and DINOv2 (Oquab et al., 2023), which we hypothesize is due to these methods learning strong global image representations during pre-training. The contrastive pre-training objective of CLIP promotes the learning of global image representations (Li et al., 2024), whereas the DINO self-distillation objective (Caron et al., 2021) encourages the `[CLS]` token of both DINO and DINOv2 to learn a global image representation (Li et al., 2024). We hypothesize that learning good global image representations is crucial for the NVS and generalizable INR tasks.

Qualitative results can be found in Figure 4. We find that the foundation model approach is better at learning the shape of objects, such as the curved back of a chair.

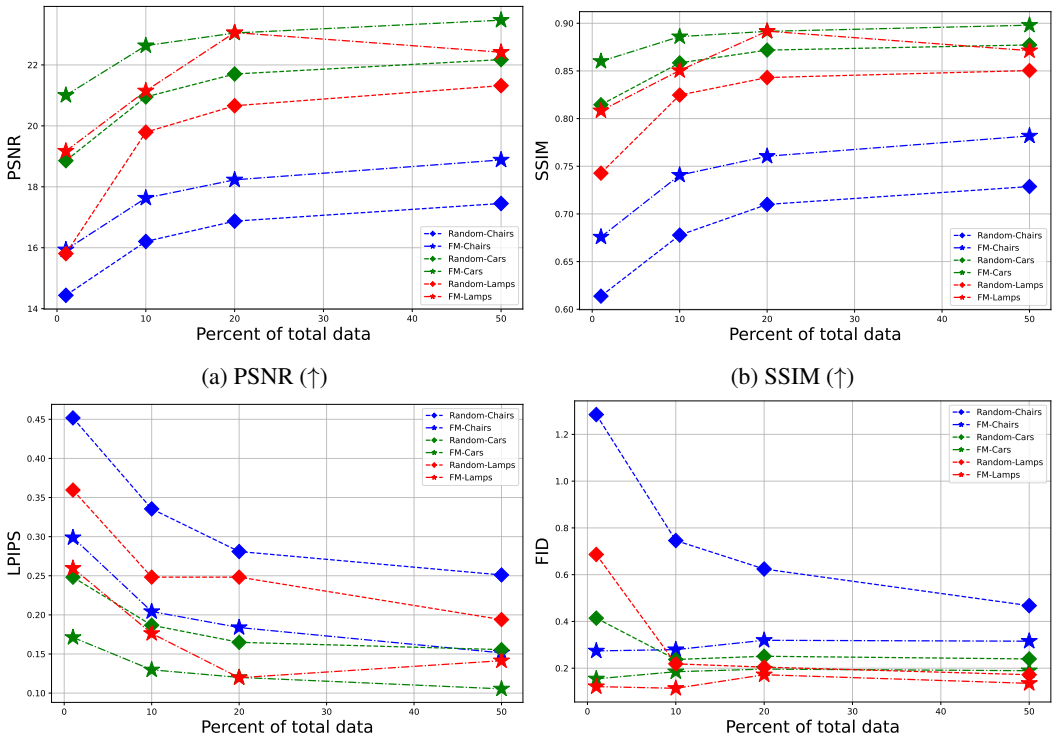
3.5 Foundation models improve hypernetwork generalizability to unseen classes

Due to their large pre-training datasets, we hypothesize that fine-tuning from foundation model features should also lead to better zero-shot generalization to unseen classes. To test this, we train on

Table 2: Comparison of hypernetwork generalizability to classes unseen during training using random initialization, fine-tuning from DINOv2 (Oquab et al., 2023), and prompt tuning with frozen DINOv2. Each method was trained with only two of the classes in the ShapeNet NVS dataset and evaluated on the third, unseen class. The best metrics are highlighted in **bold**.

Method	Training → Test	PSNR (↑)	SSIM (↑)	LPIPS (↓)	FID (↓)
Randomly initialized	cars, chairs → lamps	17.377	0.7959	**0.1898**	0.1179
Frozen	cars, chairs → lamps	**18.346**	0.7972	0.2941	0.3033
Fine-tuned	cars, chairs → lamps	17.474	**0.7977**	0.1903	**0.0956**
Randomly initialized	cars, lamps → chairs	13.163	0.6212	**0.3751**	1.0199
Frozen	cars, lamps → chairs	**13.536**	0.6112	0.4279	1.0017
Fine-tuned	cars, lamps → chairs	13.322	**0.6238**	0.3845	**0.9680**
Randomly initialized	chairs, lamps → cars	15.431	0.7521	0.2987	0.3276
Frozen	chairs, lamps → cars	**15.503**	0.7465	0.3607	0.3623
Fine-tuned	chairs, lamps → cars	15.382	**0.7692**	**0.2310**	**0.1548**

two of the three classes in the ShapeNet NVS dataset and evaluate on the third class, which is unseen during training, using the *random* and *fine-tune* strategies. Furthermore, to see if full fine-tuning is degrading the generalizability of foundation model features through catastrophic forgetting, we train additional models where the foundation model is frozen. In Table 2, we find that the full fine-tuning of foundation models improves zero-shot generalization to unseen classes over training from scratch, and that reconstructions (as measured by PSNR) can be improved even further if the foundation model is frozen instead of fine-tuned, indicating that some of the generalizability of the pre-trained features is lost during full fine-tuning.



(a) PSNR (↑) (b) SSIM (↑)

(c) LPIPS (↓) (d) FID (↓)

Figure 2: Plots showing performance vs the amount of training data for both the randomly initialized (Random) and foundation model (FM) strategies.

Table 3: Comparison of the three different training strategies using DINO (Caron et al., 2021) on the NVS task. We find that models with a frozen DINO encoder perform better than the same model randomly initialized on PSNR, while remaining close on the other metrics with a fraction of the parameters. Full fine-tuning results in a significant increase in performance for all metrics.

Method	Trainable Parameters	PSNR (↑)	SSIM (↑)	LPIPS (↓)	FID (↓)
Randomly initialized	87.3M (100%)	20.862	0.8357	0.1511	0.2751
Prompt-tuned	100K (0.11%)	21.035	0.8335	0.1767	0.3212
Fine-tuned	87.3M (100%)	21.737	0.8555	0.1101	0.1810

Table 4: Comparison of the effectiveness of foundation models for different hypernetwork algorithms on the novel view synthesis task. We find that regardless of the algorithm type, using a foundation model significantly improves performance. The best performing models are **bolded**.

Method	PSNR (↑)	SSIM (↑)	LPIPS (↓)	FID (↓)
Trans-INR (Chen & Wang, 2022)	20.850	0.8346	0.1586	0.2680
Fine-tuned Trans-INR	**22.095**	**0.8609**	**0.1063**	**0.1705**
PONP (Gu et al., 2023)	20.878	0.8357	0.1584	0.2795
Fine-tuned PONP	**21.993**	**0.8591**	**0.1083**	**0.1765**
Instance Pattern Composers (Kim et al., 2023)	20.102	**0.8344**	0.1811	0.2655
Fine-tuned Instance Pattern Composers	**20.672**	0.8324	**0.1450**	**0.1971**

3.6 Foundation models improve hypernetwork data efficiency

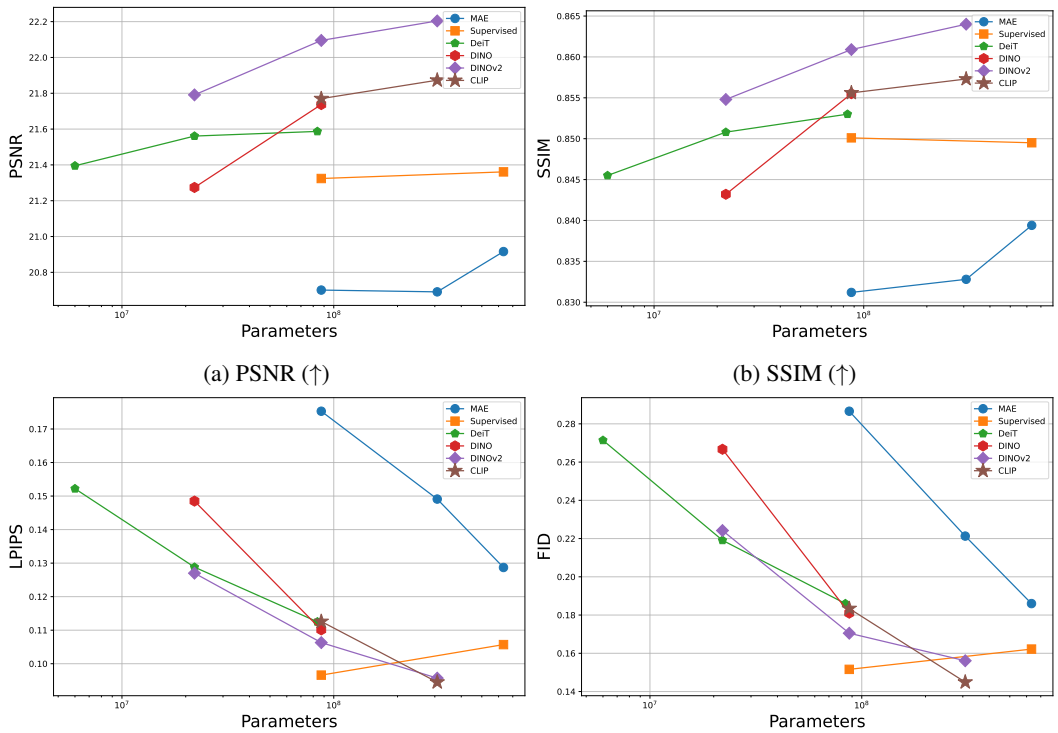
In Figure 2, we compare the random initialization and foundation model strategies when trained on 1%, 10%, 20%, and 50% of the data. We find that for PSNR, SSIM, and LPIPS, the foundation model approach significantly outperforms random initialization on every category, and that these metrics are closely correlated. We observe that, unlike the other metrics, FID seems to plateau quickly and may even increase slightly with more data. One possible explanation is that FID may not detect gradual improvements in image quality and may instead incorrectly indicate quality degradation (Jayasumana et al., 2024), which may be happening here as the image quality gradually improves due to the increasing amount of training data. This indicates that foundation model-based hypernetworks will be better able to leverage the increasingly larger datasets being published, such as Objaverse (Deitke et al., 2023).

3.7 Frozen foundation models enable parameter efficient hypernetworks

In Table 3, we find that even if the Transformer encoder weights are frozen, the model's performance can perform on par or even exceed that of the same model randomly initialized, despite using only a fraction of the learnable parameters (100K vs 87M). This means that even if there are no computational considerations, training a hypernetwork using the simple formula of extra input tokens, a frozen pre-trained backbone, and MLP heads is a promising approach. Surprisingly, despite the modality difference between image features and the weights of an INR, prompt tuning can succeed with only a linear head producing the weights of each layer.

3.8 Scaling laws for hypernetworks

As shown in Figure 3, we find that increasing the number of parameters of the foundation model generally increases performance on all metrics. This suggests that being able to scale foundation models to more parameters would directly lead to an increase hypernetwork performance. All foundation models investigated showed improved performance with more data with the exception of the supervised ViT (Dosovitskiy et al., 2020), where PSNR increased but all other metrics decreased, with the caveat that many models, including the supervised ViT, only had two model sizes tested. It has been observed before that for ViTs trained on image classification, better upstream performance does not necessarily result in better performance on downstream tasks (Zhai et al., 2022).



(a) PSNR (↑) (b) SSIM (↑)

(c) LPIPS (↓) (d) FID (↓)

Figure 3: Plots of NVS performance vs number of Transformer encoder parameters, as measured by the four metrics, on the NVS task using the Trans-INR algorithm. We find that increasing model size generally leads to increased performance, with supervised ViTs (Dosovitskiy et al., 2020) being a clear outlier.

3.9 Robustness between algorithms and modalities

Since our hypernetwork framework is based on Trans-INR (Chen & Wang, 2022), follow-up improvements (see Sec. 3.1) to this framework can also be enhanced with foundation models. In Table 4, we find that the improvement provided by using foundation model backbones persists regardless of the type of algorithm used (e.g. probabilistic (Gu et al., 2023) or weight-sharing (Kim et al., 2023)). We note that while past work showed that weight-sharing approaches (Kim et al., 2023; Lee et al., 2024) were state-of-the-art when training a separate model per category, they perform much worse than competing algorithms when training is done across categories. We hypothesize that this is due to these methods sharing too many of the INR parameters among all data instances, limiting the expressivity of the model and resulting in underfitting. This drop in performance holds with the addition of foundation models. In contrast, PONP continues to perform slightly better than Trans-INR in this setting, but with the addition of foundation models, fine-tuned Trans-INR performs slightly better than PONP.

Table 5 shows that this effect extends across different modalities to audio, indicating the robustness of the benefits of foundation models to hypernetworks. Notably, we see that parameter-efficient prompt tuning performs slightly better than a model with random initialization.

4 Related Works

Implicit Neural Representations (INRs) INRs represent complex data such as 3D objects, scenes, and audio by parameterizing them using a neural network. Architectures for INRs include using Fourier features (Mildenhall et al., 2021; Tancik et al., 2020) and sinusoidal activation functions (Sitzmann et al., 2020b). The flexibility of the INR framework has led to applications in a wide variety of domains, including 3D shape and scene reconstruction (Mildenhall et al., 2021; Sitzmann

Table 5: Audio reconstruction on the LibriSpeech dataset using the weight-sharing hypernetwork approach of Instance Pattern Composers (Kim et al., 2023). The best performing model is **bolded**.

Method	Params	PSNR (↑)
Randomly initialized	72.7M (100%)	24.431
Prompt-tuned Whisper-base (Radford et al., 2023)	100K (0.1%)	24.432
Fine-tuned Whisper-tiny (Radford et al., 2021)	37.9M (52.1%)	24.431
Fine-tuned Whisper-base (Radford et al., 2023)	72.7M (100%)	**24.434**

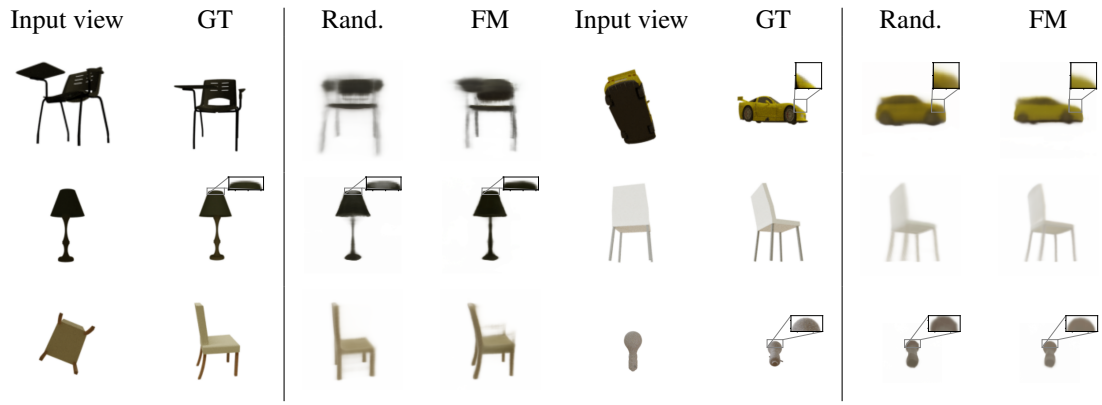


Figure 4: Comparison of qualitative results between the best foundation model-based hypernetwork and hypernetworks trained from scratch. Novel views generated with the hypernetwork approach (FM) are more faithful to the groundtruth than the baseline (Random). For example, the lamp in the middle row is better reconstructed at both the top of the lamp and on its stem, while for the two chairs the FM approach better captures their curved backs. You may need to zoom in to see the differences.

et al., 2019; 2020b;a), generative models (Poole et al., 2022; Liu et al., 2023), robotics (Li et al., 2022b), and more.

Generalizable INR The problem of learning a generalizable INR is usually formulated as a meta-learning task, where learning an INR for each signal is a separate task (Tancik et al., 2021; Chen & Wang, 2022; Gu et al., 2023). Early methods used auto-decoding (Mescheder et al., 2019; Park et al., 2019), where a latent vector is optimized per-instance and concatenated with the input to the INR. The current major approaches to this problem are gradient-based meta-learning, hypernetworks, and neural processes. The gradient-based meta-learning approach (Sitzmann et al., 2020a; Tancik et al., 2021; Lee et al., 2021) uses algorithms such as MAML (Finn et al., 2017) or Reptile (Nichol et al., 2018) to learn a good INR initialization that can be quickly finetuned, but has the disadvantage of requiring additional test-time optimization. Hypernetwork approaches (Mescheder et al., 2019; Sitzmann et al., 2020b; 2019; 2021; Chen & Wang, 2022) use a separate shared encoder that generates the weights of an INR, and have fast inference, as an INR can be generated in one forward pass of the encoder. Neural process approaches (Guo et al., 2023; Gu et al., 2023) use the neural process meta-learning framework (Garnelo et al., 2018b;a) which use neural networks to parameterize a stochastic process. This approach may be combined with hypernetwork approaches (Gu et al., 2023). Other approaches to generalizable INR follow the strategy of improving INRs by distillation from foundation models (Wang et al., 2022; Ye et al., 2023; Liao et al., 2024). In FeatureNeRF (Ye et al., 2023), foundation model features are distilled by training (non-hypernetwork) generalizable INRs to jointly predict foundation model features along with the reconstruction. Unlike these works, our model focuses only on improving hypernetwork architectures with foundation models.

Hypernetworks Hypernetworks are neural networks that produce or modify the parameters of another network. In this paper, we focus on hypernetworks that generate implicit neural representations. Hypernetworks are used as a means of conditioning implicit neural representation generation and also to create generalizable implicit neural representations. Early works using hypernetworks

to generate implicit neural representations (Mescheder et al., 2019; Sitzmann et al., 2020b; 2019; 2021). Early hypernetworks methods for generating INRs used simpler MLP (Sitzmann et al., 2019; 2021) or convolutional (Mescheder et al., 2019; Sitzmann et al., 2020b) architectures for the hypernetwork. Trans-INR (Chen & Wang, 2022) proposed using the more powerful vision transformer (ViT) (Dosovitskiy et al., 2020) as the base architecture for hypernetworks, and this method has been improved upon by incorporating neural processes (Gu et al., 2023) or using weight modulations to learn only some of the layers of the INR while sharing the rest of the parameters (Kim et al., 2023; Lee et al., 2024). Our work examines the impact of foundation models on hypernetworks using Transformer-based architectures, which to the best of our knowledge has not been examined before.

5 Conclusion

We present a rigorous investigation of using foundation models to improve hypernetworks for generalizable INR tasks, providing key insights for designing future hypernetwork models. We demonstrate that foundation models improve hypernetwork performance on both seen and unseen classes, and show that this effect is robust. We also provide a parameter-efficient way to create hypernetwork models based on prompt tuning. We also further analyze the effect of using foundation models, looking at the choice of foundation as well as scaling with data and parameters. We hope that our investigation serves as a starting point for investigating foundation models for hypernetwork architectures.

References

Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando De Freitas. Learning to learn by gradient descent by gradient descent. *Advances in neural information processing systems*, 29, 2016.

Jimmy Ba, Geoffrey E Hinton, Volodymyr Mnih, Joel Z Leibo, and Catalin Ionescu. Using fast weights to attend to the recent past. *Advances in neural information processing systems*, 29, 2016.

Rishi Bommasani, Drew A Hudson, Ehsan Adeli, Russ Altman, Simran Arora, Sydney von Arx, Michael S Bernstein, Jeannette Bohg, Antoine Bosselut, Emma Brunskill, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021.

Andrew Brock, Theodore Lim, James M Ritchie, and Nick Weston. Smash: one-shot model architecture search through hypernetworks. *arXiv preprint arXiv:1708.05344*, 2017.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9650–9660, 2021.

Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

Yinbo Chen and Xiaolong Wang. Transformers as meta-learners for implicit neural representations. In *European Conference on Computer Vision*, pp. 170–187. Springer, 2022.

Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13142–13153, 2023.

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*, pp. 248–255. Ieee, 2009.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pp. 1126–1135. PMLR, 2017.

Marta Garnelo, Dan Rosenbaum, Christopher Maddison, Tiago Ramalho, David Saxton, Murray Shanahan, Yee Whye Teh, Danilo Rezende, and SM Ali Eslami. Conditional neural processes. In *International conference on machine learning*, pp. 1704–1713. PMLR, 2018a.

Marta Garnelo, Jonathan Schwarz, Dan Rosenbaum, Fabio Viola, Danilo J Rezende, SM Eslami, and Yee Whye Teh. Neural processes. *arXiv preprint arXiv:1807.01622*, 2018b.

Anirudh Goyal, Alex Lamb, Jordan Hoffmann, Shagun Sodhani, Sergey Levine, Yoshua Bengio, and Bernhard Schölkopf. Recurrent independent mechanisms. *arXiv preprint arXiv:1909.10893*, 2019.

Jeffrey Gu, Kuan-Chieh Wang, and Serena Yeung. Generalizable neural fields as partially observed neural processes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5330–5339, 2023.

Zongyu Guo, Cuiling Lan, Zhizheng Zhang, Yan Lu, and Zhibo Chen. Versatile neural processes for learning implicit neural representations. *arXiv preprint arXiv:2301.08883*, 2023.

David Ha, Andrew Dai, and Quoc V Le. Hypernetworks. *arXiv preprint arXiv:1609.09106*, 2016.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16000–16009, 2022.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Yicong Hong, Kai Zhang, Jiuxiang Gu, Sai Bi, Yang Zhou, Difan Liu, Feng Liu, Kalyan Sunkavalli, Trung Bui, and Hao Tan. Lrm: Large reconstruction model for single image to 3d. *arXiv preprint arXiv:2311.04400*, 2023.

Sadeep Jayasumana, Srikumar Ramalingam, Andreas Veit, Daniel Glasner, Ayan Chakrabarti, and Sanjiv Kumar. Rethinking fid: Towards a better evaluation metric for image generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9307–9315, 2024.

Menglin Jia, Luming Tang, Bor-Chun Chen, Claire Cardie, Serge Belongie, Bharath Hariharan, and Ser-Nam Lim. Visual prompt tuning. In *European Conference on Computer Vision*, pp. 709–727. Springer, 2022.

Chiheon Kim, Doyup Lee, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Generalizable implicit neural representations via instance pattern composers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11808–11817, 2023.

Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4015–4026, 2023.

Doyup Lee, Chiheon Kim, Minsu Cho, and WOOK SHIN HAN. Locality-aware generalizable implicit neural representation. *Advances in Neural Information Processing Systems*, 36, 2024.

Jaeho Lee, Jihoon Tack, Namhoon Lee, and Jinwoo Shin. Meta-learning sparse implicit neural representations. *Advances in Neural Information Processing Systems*, 34:11769–11780, 2021.

Chunyuan Li, Zhe Gan, Zhengyuan Yang, Jianwei Yang, Linjie Li, Lijuan Wang, Jianfeng Gao, et al. Multimodal foundation models: From specialists to general-purpose assistants. *Foundations and Trends® in Computer Graphics and Vision*, 16(1-2):1–214, 2024.

Siyuan Li, Di Wu, Fang Wu, Zelin Zang, Stan Li, et al. Architecture-agnostic masked image modeling–from vit back to cnn. *arXiv preprint arXiv:2205.13943*, 2022a.

Yunzhu Li, Shuang Li, Vincent Sitzmann, Pulkit Agrawal, and Antonio Torralba. 3d neural scene representations for visuomotor control. In *Conference on Robot Learning*, pp. 112–123. PMLR, 2022b.

Feng Liang, Yangguang Li, and Diana Marculescu. Supmae: Supervised masked autoencoders are efficient vision learners. *arXiv preprint arXiv:2205.14540*, 2022.

Guibiao Liao, Kaichen Zhou, Zhenyu Bao, Kanglin Liu, and Qing Li. Ov-nerf: Open-vocabulary neural radiance fields with vision and language foundation models for 3d semantic understanding. *arXiv preprint arXiv:2402.04648*, 2024.

Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9298–9309, 2023.

Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4460–4470, 2019.

Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*, 2018.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Jose Javier Gonzalez Ortiz, John Guttag, and Adrian Dalca. Magnitude invariant parametrizations improve hypernetwork learning. *arXiv preprint arXiv:2304.07645*, 2023.

Vassil Panayotov, Guoguo Chen, Daniel Povey, and Sanjeev Khudanpur. Librispeech: an asr corpus based on public domain audio books. In *2015 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pp. 5206–5210. IEEE, 2015.

Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 165–174, 2019.

Ben Poole, Ajay Jain, Jonathan T Barron, and Ben Mildenhall. Dreamfusion: Text-to-3d using 2d diffusion. *arXiv preprint arXiv:2209.14988*, 2022.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *International Conference on Machine Learning*, pp. 28492–28518. PMLR, 2023.

Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3d-structure-aware neural scene representations. *Advances in Neural Information Processing Systems*, 32, 2019.

Vincent Sitzmann, Eric Chan, Richard Tucker, Noah Snavely, and Gordon Wetzstein. Metasdf: Meta-learning signed distance functions. *Advances in Neural Information Processing Systems*, 33: 10136–10147, 2020a.

Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in neural information processing systems*, 33:7462–7473, 2020b.

Vincent Sitzmann, Semon Rezchikov, Bill Freeman, Josh Tenenbaum, and Fredo Durand. Light field networks: Neural scene representations with single-evaluation rendering. *Advances in Neural Information Processing Systems*, 34:19313–19325, 2021.

Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in neural information processing systems*, 33:7537–7547, 2020.

Matthew Tancik, Ben Mildenhall, Terrance Wang, Divi Schmidt, Pratul P Srinivasan, Jonathan T Barron, and Ren Ng. Learned initializations for optimizing coordinate-based neural representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2846–2855, 2021.

Yi Tay, Zhe Zhao, Dara Bahri, Donald Metzler, and Da-Cheng Juan. Hypergrid transformers: Towards a single model for multiple tasks. In *International conference on learning representations*, 2020.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pp. 10347–10357. PMLR, 2021.

Johannes Von Oswald, Christian Henning, Benjamin F Grewe, and João Sacramento. Continual learning with hypernetworks. *arXiv preprint arXiv:1906.00695*, 2019.

Can Wang, Menglei Chai, Mingming He, Dongdong Chen, and Jing Liao. Clip-nerf: Text-and-image driven manipulation of neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3835–3844, 2022.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

Jianglong Ye, Naiyan Wang, and Xiaolong Wang. Featurenerf: Learning generalizable nerfs by distilling foundation models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8962–8973, 2023.

Xiaohua Zhai, Alexander Kolesnikov, Neil Houlsby, and Lucas Beyer. Scaling vision transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12104–12113, 2022.

Chris Zhang, Mengye Ren, and Raquel Urtasun. Graph hypernetworks for neural architecture search. *arXiv preprint arXiv:1810.05749*, 2018a.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595, 2018b.

Dominic Zhao, Seijin Kobayashi, João Sacramento, and Johannes von Oswald. Meta-learning via hypernetworks. In *4th Workshop on Meta-Learning at NeurIPS 2020 (MetaLearn 2020)*. NeurIPS, 2020.

A TRAINING DETAILS

In this section, we provide the training details of our models.

A.1 TRAINING HYPERPARAMETERS

Table 6: Training hyperparameters for the models in Table 1. Step refers to a learning rate schedule where the initial learning rate is divided by 10 after 80% of the epochs have finished, and cos refers to a cosine learning rate schedule with a warmup of 10% of the total epochs.

Model	Epochs	Batch size	Learning rate	Scheduler
Randomly initialized	1000	32	1e-4	step
MAE	1000	128	1e-4	step
Supervised	1000	32	1e-4	step
DeiT	1000	128	1e-4	step
DINO	1000	128	1e-4	cos
CLIP	1000	128	1e-4	cos
DINOv2	1000	128	1e-4	cos

The training hyperparameters for the models in Table 1 can be found in Table 6. Models in Table 2 and Figure 2 were trained with 1000 epochs, batch size 128, learning rate 1e-4, and using the cos scheduler described above. The prompt-tuned model in Table 3 was trained with batch size 32, 1000 epochs, learning rate 1e-3, and the step scheduler; the hyperparameters for the other two can be found in Table 6. The IPC and PONP baselines in Table 4 were trained for 1000 epochs with batch size 128, learning rate 1e-4, and 1000 epochs. Additionally, PONP used the cos learning rate scheduler. The methods in Table 5 were trained for 100 epochs with batch size 64, but all other hyperparameters were the default hyperparameters from Kim et al. (2023).

A.2 INR ARCHITECTURE

Following previous work (Chen & Wang, 2022; Kim et al., 2023; Gu et al., 2023; Lee et al., 2024), our INR architecture is an MLP with 6 layers of hidden dimension 256, positional encoding with dimension 40, and ReLU activations.

B METRICS

In this section, we discuss the metrics used in our paper. Our main task of novel view synthesis from a single view of an object is a task where both image similarity metrics (such as PSNR, SSIM, LPIPS) and image generation metrics (FID) can provide complementary assessments of novel view quality. This is because the generated view may be partially determined by shared structures present in both views, while the other parts are under-determined and need to be generated. Besides PSNR, all other metrics were implemented using the `torchmetrics` library with their default parameters.

PSNR PSNR stands for peak signal-to-noise ratio, and is computed with the formula

$$\mathrm{PSNR}(y, \hat{y}) = -10 \log_{10}(\mathrm{MSE}(y, \hat{y})) \tag{3}$$

where MSE is the mean squared error. PSNR is a measure of the absolute error between a reconstruction $\hat{y}$ and the ground truth y, which makes it less reliable for under- constrained reconstruction tasks such as novel view synthesis from one view of an object, where there may be many possible plausible reconstruction.

SSIM Structural similarity index (SSIM) (Wang et al., 2004) computes the similarity of two images in luminance, contrast, and structure. SSIM is designed to measure the perceived change in structural information rather than the absolute change measured by PSNR. Wang et al. (2004) shows that SSIM better correlates with human ratings than PSNR.

LPIPS LPIPS (Zhang et al., 2018b) measures the similarity between the activations of images computed by a pre-defined neural network. Zhang et al. (2018b) shows that deep similarities given by pre-trained neural networks correlate much better with human judgments than PSNR or SSIM.

FID FID (Heusel et al., 2017) measures the how similar the distribution of generated images is to the distribution of the ground truth images, and is more suited for generative tasks than tasks where there is a defined ground truth. However, it has drawbacks, as discussed in the main text as well as Jayasumana et al. (2024).

C Comparison to previous results

Table 7: Comparison of the results in Table 1 to previously published results. * indicates that the result was obtained from previous literature by averaging the performance of separate models for the three different classes. Previous results are shown in the first half of the table, while our results are shown in the second half of the table.

Model	PSNR (↑)	SSIM (↑)	LPIPS (↓)	FID (↓)
LearnIt (Tancik et al., 2020)	21.33*	-	-	-
Trans-INR (Chen & Wang, 2022)	22.07*	-	-	-
Trans-INR (repr. by Kim et al. (2023))	22.04*	-	-	-
PONP (Gu et al., 2023)	22.14*	-	-	-
IPC (Kim et al., 2023)	22.30*	-	-	-
Randomly initialized	20.862	0.8357	0.1511	0.2751
MAE (He et al., 2022)	20.701	0.8312	0.1753	0.2866
Supervised (Dosovitskiy et al., 2020)	21.324	0.8501	0.0966	0.1516
DeiT (Touvron et al., 2021)	21.587	0.8530	0.1125	0.1860
DINO (Caron et al., 2021)	21.737	0.8555	0.1101	0.1810
CLIP (Radford et al., 2021)	21.770	0.8556	0.1126	0.1834
DINOv2 (Oquab et al., 2023)	22.095	0.8609	0.1063	0.1705

In Table 7, we compare our results for single-view novel view synthesis (Tab. 1) on the LearnIt ShapeNet dataset (Tancik et al., 2021) to previously published results from LearnIt (Tancik et al., 2021), Trans-INR (Chen & Wang, 2022), PONP (Gu et al., 2023), and IPC (Kim et al., 2023) which all use the same INR architecture. We note that these numbers are not directly comparable, as our numbers are obtained on the harder task of learning all three categories simultaneously and without being able to tokenize NVS-specific auxiliary information such as poses. Compared to previous Transformer-based methods (Chen & Wang, 2022; Gu et al., 2023; Kim et al., 2023), our method uses a ViT/B-16 while previous methods use a smaller 6 layer Transformer architecture. We also note that the performance of the Transformer hypernetwork baselines Chen & Wang (2022); Gu et al. (2023); Kim et al. (2023) is significantly degraded in the combined class setting, especially IPC (see Tab. 4).

D Limitations

One limitation of our method is we do not tokenize task-specific information such as pose and camera parameters for novel view synthesis. Previous results suggest that this may further improve performance. Another limitation is that we have only used the simple volume renderer and simple NeRF Mildenhall et al. (2021) of Tancik et al. (2020), but better results could be obtained by using a more sophisticated volume renderer and INR. Another limitation is that we only investigate fine-tuning and freezing the foundation model backbone, but other approaches may perform better. We also were not able to investigate using larger datasets such as Objaverse (Deitke et al., 2023).