ARC: ANCHORED REPRESENTATION CLOUDS FOR HIGH-RESOLUTION INR CLASSIFICATION

Joost Luijmes, Alexander Gielisse, Roman Knyazhitskiy, Jan van Gemert*
Computer Vision Lab
Delft University of Technology
The Netherlands

ABSTRACT

Implicit neural representations (INRs) encode signals in neural network weights as a memory-efficient representation, decoupling sampling resolution from the associated resource costs. Current INR image classification methods are demonstrated on low-resolution data and are sensitive to image-space transformations. We attribute these issues to the global, fully-connected MLP neural network architecture encoding of current INRs, which lack mechanisms for local representation: MLPs are sensitive to absolute image location and struggle with high-frequency details. We propose ARC: Anchored Representation Clouds, a novel INR architecture that explicitly anchors latent vectors locally in image-space. By introducing spatial structure to the latent vectors, ARC captures local image data which in our testing leads to state-of-the-art implicit image classification of both low- and high-resolution images and increased robustness against image-space translation. Code can be found at https://github.com/JLuij/anchored_representation_clouds.

1 INTRODUCTION

From novel view synthesis to inverse problems, implicit neural representations (INRs) have enabled leaps in accuracy across a variety of problems and domains due to their unique data compression and generalisation capabilities (Mildenhall et al., 2021; Essakine et al., 2024). As such, interest has grown in whether INRs can similarly enrich conventional computer vision tasks like image classification; the focus of this research.

An INR is a neural network (NN) that learns a mapping from coordinates in the signal's domain to the signal's values, *e.g.* from 2D pixel coordinates to RGB colours. After training, the INR can reconstruct an approximation of the original signal when queried on all signal coordinates, implying that the signal is encoded inside the INR weights. INRs are able to compress signals in a variety of domains (Dupont et al., 2021; Strümpler et al., 2022; Schwarz et al., 2023; Fons et al., 2022; Huang & Hoefler, 2022), and exhibit excellent generalisation capabilities outside of the signal's domain (Mildenhall et al., 2021; Yu et al., 2021; Chen et al., 2021).

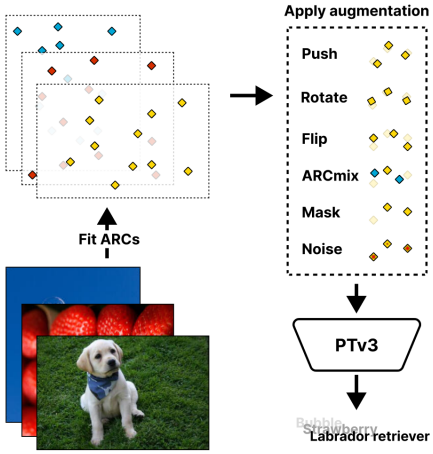

Figure 1: ARC anchors latent vectors directly in image coordinate space, preserving the local spatial image structure within the INR weight-space. Once trained, ARC can be processed by a point cloud classifier.

Images are formed by sensor elements placed on a grid. In a grid, elements of equal size are positioned equidistantly over the domain, irrespective of the image content. This uniform data representation results in memory costs that scale exponentially with signal dimensionality and resolution. To make training on image datasets feasible, images are typically downsampled, which may eliminate relevant features (Hou et al., 2016; Kong & Henao, 2022) or lead to decreased performance (Jiang et al., 2020b; Tan & Le, 2019). While these issues can be mitigated (Kong & Henao, 2022; Hou et al., 2016), INRs offer a more efficient, content-based, image representation.

*Correspondence to joostluijmes.academia@gmail.com

Current INR classification methods (Kofinas et al., 2024; Kalogeropoulos et al., 2024; Zhou et al., 2024a) face several shortcomings. First, these methods are only demonstrated on low-resolution image datasets such as MNIST (LeCun, 1998), Fashion-MNIST (Xiao et al., 2017), and CIFAR10 (Krizhevsky et al., 2009). Second, the employed INR architecture learns entirely different representations under camera transformations such as translation. Translation equivariance is a fundamental property in traditional image classifiers (Cohen & Welling, 2016; Zeiler & Fergus, 2014), especially because higher-resolution image datasets allow for less restricted object positions and present challenges in terms of background noise (Szabó & Horváth, 2022). Third, INR classifiers typically suffer from overfitting (Kofinas et al., 2024; Shamsian et al., 2024; Kalogeropoulos et al., 2024), with restricted data augmentation methods to combat this (Navon et al., 2023; Shamsian et al., 2024), resorting to the resource-intensive process of fitting several redundant INRs per image to improve generalisation (Navon et al., 2023; Bauer et al., 2023; Shamsian et al., 2024; Zhou et al., 2024a).

In this paper, we introduce a novel type of INR named ARC: Anchored Representation Clouds, along with a flexible classification pipeline and data augmentation methods (Figure 1). An ARC consists of 1) an specific encoder which anchors a cloud of latent vectors in image coordinate space, and 2) an MLP decoder that is shared among ARC instances. In querying a coordinate, ARC finds the nearest latent vectors and decodes their content. This way, ARC retains local image features, making it more robust against image translation and more performant in higher-complexity image classification. Furthermore, as the latent vectors can be positioned freely, they can be anchored more densely in high-frequency image regions, biasing model capacity towards complex regions rather than encoding the image globally. By increasing the number of anchored latents, ARC can trivially scale to larger image complexity. By converting images to a set of ARCs, each image is effectively represented by a cloud of latent vectors, allowing point-cloud architectures for downstream use, along with intuitive and effective point-cloud data augmentation methods which eliminate the need for expensive redundant INR fitting. To our knowledge, this is the first work to utilise an INR's entire weight-space on a high-resolution dataset like Imagenette (Howard, 2019), achieving a classification accuracy of 75.92%. We further demonstrate ARC's robustness to image-space transformations and its ability to capture high-resolution images.

Our contributions include the following. (1) A novel INR architecture that anchors latent vectors in the image coordinate space, preserving spatial locality. (2) An accompanying classification pipeline that enables effective and intuitive weight-space augmentation methods. (3) Enhanced robustness to image-space translations in INR classification.

2 Related Work

Implicit neural representations. An implicit neural representation (INR) (Sitzmann et al., 2020) is a neural network trained to represent a signal. In the image domain, an INR aims to learn a mapping between pixel coordinates and pixel values. After training, a forward pass of an INR produces a *reconstruction* of the original image, which is now captured inside the INR's weights. Alternative names for implicit neural representations are neural fields (Xie et al., 2022; Papa et al., 2024; Wessels et al., 2024), coordinate networks (Martel et al., 2021; Lindell et al., 2022; Zheng et al., 2022) and coordinate-based neural representation (Tancik et al., 2021).

The premise of INRs does not prescribe any particular architecture, prompting early work to assess the suitability of the simple MLP (Mildenhall et al., 2021; Park et al., 2019; Mescheder et al., 2019). Such models struggle to capture the high-frequency components of the signal, due to the *spectral bias*, which in INR-context results in blurry image reconstructions (Rahaman et al., 2019; Tancik et al., 2020; Sitzmann et al., 2020). This issue inspired a large set of diverse solutions, such as transforming the input using sinusoids (Tancik et al., 2020; Mildenhall et al., 2021; Zheng et al., 2022), modifying the activation function (Sitzmann et al., 2020; Ramasinghe & Lucey, 2022; Chng et al., 2022; Saragadam et al., 2023; Liu et al., 2024; Gao & Jaiman, 2024), or positioning learnable elements in the signal coordinate space to represent local image regions (Liu et al., 2020; Peng et al., 2020; Jiang et al., 2020a; Chen et al., 2023; Li et al., 2022; Giebenhain & Goldlücke, 2021; Martel et al., 2021; Müller et al., 2022; Chabra et al., 2020).

We posit that positioning learnable elements in signal coordinate space can aid INR classification, as by coupling latent information to local image regions, we retain image structure in the latent space and the image features encoded therein. Other methods which position latents freely in image

coordinate space focus on improving the signal reconstruction quality (Chen et al., 2023; Giebenhain & Goldlücke, 2021). These methods hybridise INR methods (Chen et al., 2023), and require intricate initialisation and latent decoding schemes (Chen et al., 2023; Giebenhain & Goldlücke, 2021). In contrast, simplicity is a core design principle in ARC; reducing computational complexity ensures that fitting an entire image dataset remains efficient.

INR classification. INR literature typically emphasises training and parameter efficiency whilst optimising reconstruction quality (Chen et al., 2023; Müller et al., 2022; Dupont et al., 2021; Huang & Hoefler, 2022; Essakine et al., 2024), or utilises INRs as a component in a broader method for their compression and generalisation abilities (Cole et al., 2023; Dollinger et al., 2024). Here, we investigate INRs in a classification setting.

Early works on INR classification studied flattened weight matrices (Unterthiner et al., 2020) or their weights' statistics (Eilertsen et al., 2020). Such flattened representations remain in use in INR-context as low-dimensional embeddings that are trained along with (Dupont et al., 2022; Bauer et al., 2023), or after (De Luigi et al., 2023), INR fitting. These methods do not generalise well to image classification (De Luigi et al., 2023; Kalogeropoulos et al., 2024) or larger-scale classification tasks (Bauer et al., 2023) however. Instead, a key insight to process *whole* INR weight-spaces was to consider a neural network's symmetries; transformations which alter the weights but preserve the INR's function (Godfrey et al., 2022). Architectures which incorporate equivariances to such symmetries obtain a significant increase in INR classification accuracy (Navon et al., 2023; Zhou et al., 2024b; Kofinas et al., 2024; Kalogeropoulos et al., 2024). With ARC, we propose an architecture that anchors low-dimensional latent embeddings in image-space. This ties the learnt encodings directly to local image content, effectively compressing an image into a latent point cloud. A similar observation is made and implemented in a concurrent work with an attention-based architecture (Wessels et al., 2024). Their method is demonstrated across various domains but does not address the shortcomings of INR classification concerning image classification, such as confinement to low-resolution image datasets and sensitivity to image-space translations.

Flexibility of INRs. The flexibility of NNs allows for wildly varying weight-spaces that faithfully capture a signal. This variability makes it difficult for downstream models to capture consistent image features across INR instances (Kofinas et al., 2024; Shamsian et al., 2024; Kalogeropoulos et al., 2024). Previous work has shown that establishing a form of alignment or 'common ground' among INRs improves classification accuracy. Such methods include sharing the INR weight initialisation (Navon et al., 2023; Papa et al., 2024), introducing shared learnable elements to the fitting process (Giebenhain & Goldlücke, 2021; Chen et al., 2023; Wessels et al., 2024), learning low-dimensional shifts to an established INR base network (Dupont et al., 2022; Bauer et al., 2023) or sharing a part of the INR over all instances (Vyas et al., 2024). In this spirit, ARC shares a decoder over the whole dataset, which is jointly pretrained on a subset of the data and then frozen.

Even if INRs share a form of alignment, overfitting remains a persistent issue in INR classification (Kofinas et al., 2024; Shamsian et al., 2024; Kalogeropoulos et al., 2024). A limited set of weight-space augmentations are available to combat this (Navon et al., 2023; Shamsian et al., 2024). Hence, INR classification methods fit redundant INRs for each image in the dataset, which is a resource-intensive task (Zhou et al., 2024a; Kalogeropoulos et al., 2024; Navon et al., 2023; Bauer et al., 2023; Zhao et al., 2024). Instead, we can leverage the unique weight-space of ARC to apply intuitive data augmentation methods on-the-fly which we demonstrate to be competitive in regularisation effectiveness to redundant INR fitting, at a fraction of the computational cost.

3 METHOD: ARC

ARC consists of an encoder and a decoder. The encoder is mainly a cloud of latent vectors and retrieval logic to obtain the n nearest latent vectors for a given input coordinate. These vectors are then concatenated and passed to the decoder, which maps the vector to a colour. More specifically, given a coordinate $\boldsymbol{x} \in \mathbb{R}^{d_{\text{in}}}$, an indexing function U_n retrieves the n nearest anchored latents and their relative positions to $\boldsymbol{x}$ from latent cloud $\mathcal{P} = \{(\boldsymbol{p}_i, \boldsymbol{w}_i)\}_{i=1}^{k}$. These latents and their relative positions are concatenated. The concatenated vector is then passed through the MLP decoder g, mapping it to an RGB colour. Further details are in the Appendix.

Encoder. The encoder consists of a cloud of learnable latent vectors and aggregation logic. The latent vectors are anchored in the image coordinate space. The latent dimension z and the number of latents k anchored in the image are hyperparameters.

Latent vector positions. We follow Chen et al. (2023); Li et al. (2022), whereby learnable elements are positioned in signal-space near high-frequency content so as to bias the model capacity to more difficult to encode content. To this end, the latents' positions are determined by sampling the image gradient norm. The latents' positions remain fixed. The indexing function U_n can therefore *cache* the index of nearest latents upon ARC initialisation, significantly decreasing training latency.

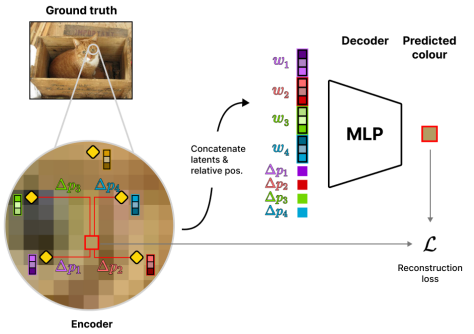

Figure 2: Given an image coordinate $\boldsymbol{x}$, ARC finds the 4 nearest latent vectors, along with their relative position to $\boldsymbol{x}$. These are concatenated into a long descriptive vector which the decoder maps to an RGB colour. The reconstruction loss is computed between the ground-truth and predicted colour.

Indexing and aggregation. Given an input coordinate $\boldsymbol{x}$, an indexing function U_n retrieves the n nearest latent vectors along with their relative position to $\boldsymbol{x}$. n can be any value but we found 4 nearest neighbours to be sufficiently expressive. In contrast to other methods which predefine an interpolation function to aggregate the latent vectors (Chen et al., 2023; Giebenhain & Goldlücke, 2021), ARC defers to the decoder to learn this from the latent vectors and relative positions, similar to (Chen et al., 2021). The latent vectors and their relative positions are thus simply concatenated and fed to the decoder. We found Fourier features (Tancik et al., 2020; Chen et al., 2023) to not improve results much so these were omitted in favour of simplicity.

Decoder. The decoder is a simple MLP with ReLU activations, as opposed to more elaborate activation functions (Chen et al., 2023). To align latent vectors across different ARC instances, we let instances share the decoder. This shared decoder is pre-trained along with several ARCs on a subset of the data and then frozen for the complete dataset. Consequently, the memory cost of the decoder can be amortised across the whole dataset as only the anchored representation cloud is required for classification.

Downstream processing. Due to its unique weight-space, ARC transforms the problem of INR classification into point cloud classification. This allows us to leverage well-studied downstream architectures for ARC classification. Contrary to SIREN classifiers, no further mechanisms against weight-space equivariances are needed (Navon et al., 2023; Kofinas et al., 2024; Kalogeropoulos et al., 2024).

Point Transformer v3. Any point cloud architecture that supports arbitrary point feature dimensions can naturally process ARCs. However, since the anchored latent vectors encode local information, an architecture that emphasises local interaction is preferred. To this end, we select Point Transformer v3 (PTv3) (Wu et al., 2024), a state-of-the-art method that performs local attention. When using PTv3, we provide it only the learnt latent vectors. The latent positions are used only for the relative positional encoding and pooling operations. PTv3 allows for a varying number of points within a batch, enabling ARCs to adjust the number of latents to the image complexity. Note however that in this work we follow (Chen et al., 2023) by letting the number of latent vectors be proportional to image size. Modifications made to PTv3 for ARC compatibility are discussed in Appendix A.4.

Data augmentation. We can leverage the unique weight space of ARC to apply intuitive data augmentation methods which we show to be almost as effective as using redundant ARC instances in our experiments. These data augmentations are applied 'on-the-fly' on the anchored representation cloud. The augmentation methods are named Noise, Mask, Push, Rotate, Flip, and ARCmix (Figure 8). Noise applies random Gaussian noise to the latent content, Masking omits a specified fraction of the points, Push, Rotate and Flip only augment the latent vector coordinates within a single ARC instance. Furthermore, ARCmix, a method inspired by CutMix (Yun et al., 2019; Zhang et al., 2022), mixes two ARC by combining their latent clouds.

4 EXPERIMENTS

In INR classification literature, the seminal SIREN architecture remains the most prominent and has seen incremental classification improvements over recent years (Navon et al., 2023; Kofinas et al., 2024; Zhou et al., 2024a;b; Kalogeropoulos et al., 2024). We focus on two representative baselines: the foundational DWSnets (Navon et al., 2023) and the state-of-the-art ScaleGMN (Kalogeropoulos et al., 2024). Additional experiments are presented in Appendix A.2, along with more details about the experiments and implementation.

Experiment 1. High-resolution image classification. How well do ARCs and existing INR classification pipelines perform as image resolution increases? To answer this question, we analyse INR classification accuracy on Fashion-MNIST (FMNIST) (Xiao et al., 2017) whereby we pad the images with zeroes to a 100×100 and 1024×1024 resolution (Figure 9). These datasets are then converted into SIRENs and ARCs and classified by their respective methods. The test accuracy is reported in Table 1.

While both SIREN and ARC classification pipelines show a degradation in classification accuracy, ARC is demonstrably stronger. This difference can be attributed to how each method handles increasing image resolution. A SIREN's input domain is fixed to $[-1, 1]^{d_{\text{in}}}$, regardless of image size. On higher resolutions, the number of pixels mapped within $[-1, 1]^{d_{\text{in}}}$ grows, requiring SIRENs to learn higher frequency mappings. Subsequently, spectral bias effects re-emerge, resulting in overly smooth reconstructions. To mitigate this, the SIREN architecture is increased in width and depth, yielding a $\times 33$ increase in the parameters between 100×100 and 1024×1024. This substantial increase significantly increased fitting time, requiring us to limit the SIREN dataset size to approximately a third of that used for ARC. For ARC, the image resolution is decoupled from the latent representation. Consequently, as image size grows but image complexity remains low, the number of latent vectors does not have to be adapted.

We continue our investigation of high-resolution image INR classification with Imagenette (Howard, 2019), a subset of Imagenet (Krizhevsky et al., 2012). Imagenette is a dataset of natural images with a median resolution of 375×500 and a maximum of 4268×2912. As image complexity increases, INR capacity must increase proportionally. For ARC, we can increase the number of anchored latent vectors. For SIRENs however, the scaling is performed in either the number of hidden layers or the hidden dimension. Neither DWSnets nor ScaleGMN support variable SIREN architectures, so a fixed architecture size must be picked. This invariably leads to undercapacity or overcapacity on a subset of the image data. To decrease discrepancies that may arise from this, we use the prescaled dataset variant Imagenette320 (Howard, 2019), and apply a centre-crop to standardise all images to 320×320. These images are converted into SIRENs and ARCs and subsequently classified. This procedure is additionally performed on the original, full-resolution Imagenette dataset with ARCs. As no test set is provided, we report validation accuracy in Table 2. We were not able to train ScaleGMN on this dataset due to instability issues, which we discuss further in Appendix A.5.

ARC demonstrates high classification accuracy on Imagenette. We hypothesise that this performance is due to the higher resolution of Imagenette images, where relevant features are distributed across larger regions of the image. With latent vectors anchored in these regions, ARC can represent the

Side Length	**28**	**100**	**1024**	**INR #param increase**
DWSNets	67.06$^{\diamond}$	67.60	53.28	$\times 33$
ScaleGMN	**80.78**$^{\diamond}$	74.50	48.77	$\times 33$
Ours	80.42	**79.36**	**73.57**	$\times$**1**

Table 1: **Exp. 1:** Test accuracy (%↑) on the padded FMNIST datasets and the required increase in INR parameters (↓) to produce recognisable reconstructions. The SIREN 1024×1024 dataset is a third of the size of the corresponding ARC dataset due to steep fitting costs as a consequence of the increased number of parameters. Entries marked with $^{\diamond}$ are taken from their original publication (Navon et al., 2023; Kalogeropoulos et al., 2024). Next to being a more parameter efficient representation, our method is more resilient against increasing image size.

	Imagenette 320x CenterCrop	Imagenette full resolution
DWSnets	41.05	-
Ours	**71.71**	**75.92**

Table 2: **Exp. 1:** Validation accuracy (%↑) on Imagenette. ARC sets a new watermark in classifying full-resolution image data through their INR representation.

	MNIST	FMNIST	CIFAR10
DWSnets (Navon et al., 2023)	85.71 ± 0.6	67.06 ± 0.3	-
NG-GNN (Kofinas et al., 2024)	91.40 ± 0.6	68.00 ± 0.2	$36.04^{\diamond} \pm 0.44$
NG-T (Kofinas et al., 2024)	92.40 ± 0.3	72.70 ± 0.6	-
ScaleGMN (Kalogeropoulos et al., 2024)	$\mathbf{96.59} \pm 0.2$	$\mathbf{80.78} \pm 0.2$	38.82 ± 0.1
Ours	92.69 ± 1.2	$\mathbf{80.42} \pm 0.4$	$\mathbf{58.47 \pm 0.4}$

Table 3: **Exp. 2:** Test classification accuracy (%↑) on various image classification datasets. Entries marked with $^{\diamond}$ are taken from reproductions by (Kalogeropoulos et al., 2024). ARC classification accuracy is similar to baselines on low-complexity datasets and outperforms them on the more complex CIFAR10 dataset.

features with greater precision and redundancy, all while respecting the spatial integrity of these features. This in turn enables PTv3 to infer meaningful image features. The additional accuracy between the centre-crop and full-resolution sets can be attributed to the resolution bias present in Imagenette, which may be exploited by the relative positional encoding and pooling in PTv3; in a brief test we found that a simple ReLU-MLP of dimensions $[2, 64, 64, 64, 10]$ can obtain a validation accuracy of up to 23.46% on Imagenette when trained on image dimensions alone.

Experiment 2. Image classification benchmarks. How does ARC compare to established INR classification benchmarks? We follow other INR literature in using MNIST (LeCun, 1998), Fashion-MNIST (Xiao et al., 2017), and CIFAR10 (Krizhevsky et al., 2009). The complete datasets are used, along with data augmentation methods that are expanded upon in Experiment 3. Results are gathered on 3 different seeds and summarised in Table 3. On low-complexity grey-scale datasets, ARC performs similarly to current state-of-the-art methods. On the more complex CIFAR10 dataset, ARC obtains state-of-the-art accuracy. CIFAR10 contains natural RGB images which inherently contain background noise. It is therefore a more challenging benchmark than the gray-scale MNIST and FMNIST datasets. We expect that image features are better represented among ARC instances on these more complex images, leading to superior accuracy compared to SIRENs.

Experiment 3. Data augmentation. In SIREN classification methods, an increasingly used technique to reduce overfitting is to generate redundant INRs for each image in the dataset (Zhou et al., 2024a; Kalogeropoulos et al., 2024; Navon et al., 2023; Bauer et al., 2023; Zhao et al., 2024), demanding significantly more resources and time for the INR fitting process. To establish a baseline, we fit 20 ARCs per image on a 10k subset of CIFAR10. Without any additional augmentations, PTv3 is trained on two conditions: using a single ARC per image and using all 20 ARCs. We report the test accuracy per image in Table 4. ARC classification accuracy is on par with the state-of-the-art which has the advantage of being trained on the full CIFAR10 dataset.

We now ask, are data augmentation methods that operate on the ARC weight-space as effective as using redundant INRs during training? We convert the entire CIFAR10 dataset to ARC instances and evaluate the different data augmentation methods in Table 5. When comparing the test accuracies obtained under weight-space data augmentations to those obtained with redundant INR fitting Table 4, we observe that our data augmentations do not fully close the gap. Moreover, the gap may be slightly larger as we leverage the entire CIFAR10 in the weight-space augmentation experiments and just a 10k subset in the redundant INR experiment. Regardless, our data augmentation methods offer a compelling alternative that requires significantly less computational resources and time to execute.

#INRs per image	1	20
NG-GNN	36.04$^\diamond$	45.70$^\diamond$
ScaleGMN	**38.82**	**56.95**
Ours	38.12	55.87

Table 4: **Exp. 3:** CIFAR10 test accuracy (%↑) after training on either 1 or 20 INRs per CIFAR10 image. No further data augmentation is employed. Similar to the baselines, ARC classification accuracy improves significantly when trained on redundant INRs. Baseline data is taken from Kalogeropoulos et al. (2024), where data marked with $^\diamond$ denote their reproduction of (Kofinas et al., 2024).

Latent noise	Point space	ARC masking	ARCmix	Acc.
-	-	-	-	38.12
✓	-	-	-	39.08
-	✓	-	-	51.69
-	-	✓	-	50.25
-	-	-	✓	54.55
-	-	✓	✓	54.56
-	✓	✓	✓	50.34
✓	✓	✓	✓	51.03

Table 5: **Exp. 3:** Test accuracy (%↑) on different ARC augmentation methods. The 'Point space' column applies the push, flip and rotation augmentation methods. Compared to fitting and training on redundant INRs, these more efficient weight-space augmentations yield competitive test accuracy.

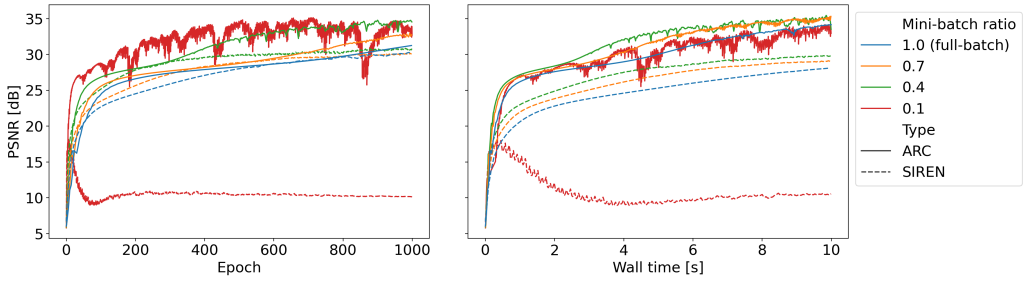

Figure 3: **Exp. 4:** Image reconstruction quality when trained with various mini-batch ratios. Interestingly, both SIREN and ARC show strong fitting performance when only using a subset of the pixels at each optimization step. This allows for achieving a higher PSNR in less wall time.

Experiment 4. Mini-batch training. In INR context, training for an epoch entails having supervised all image coordinates. We ask whether mini-batches can be an alternative, whereby optimisation steps are performed on a random subset of the image coordinates. To answer this question, we define *mini-batch ratio* as the fraction of the full-batch size used in a single step. For several mini-batch ratios, we train a SIREN and ARC instance three times on skimage's astronaut image and average them together. In Figure 3 we observe that, for both SIRENs and ARCs, mini-batch training is stable for high enough ratios, and yields similar reconstruction quality as full-batch training in fewer epochs and wall time. Encouraged by this finding, we leverage mini-batch training in all our ARC experiments with a mini-batch ratio of 0.25 unless otherwise noted.

Experiment 5: Image translation robustness. For regular image classifiers, translation invariance means that the classifier's prediction is unaffected by a shift in pixels. This runs counter to INRs where any change in pixels should be accurately reflected in the reconstruction, and consequently, in the learnt INR weights. For an INR classifier to be robust against benign image shifts, it must distinguish changes in INR weights that capture pixel shifts from changes that capture more meaningful semantic changes.

As ARC captures local image features, we hypothesise that the weight-space remains largely intact when fitting to a shifted image, making the ARC classification pipeline more robust to such transformations. To test this, we return to the 100×100 padded Fashion-MNIST objects, where, alongside centred FMNIST objects (named `Centered`), we fit INRs to randomly displaced FMNIST images (named `Displaced`). A sample of these data can be found in Figure 11. SIRENs share the same initialisation, while ARCs use the same decoder over all datasets. Both ScaleGMN and PTv3 are trained on their respective INRs of the `Centered` dataset and subsequently evaluated on `Displaced` INRs. Test accuracies are listed in Table 6.

Method	Test on *Centered*	Test on *Displaced*
ScaleGMN	74.50	13.00
Ours, PTv3	79.36	47.61

Table 6: **Exp. 5:** Test accuracy (%↑) when trained on INRs of *Centered* and evaluated on INRs of *Displaced*. SIREN-based methods (ScaleGMN) experience a complete collapse in classification accuracy due to significant differences in weight-space among the datasets. ARC paired with PTv3 demonstrates improved robustness but warrants further experimentation.

	No shift	Shift
PTv3	79.32	57.87
PTv1	76.37	76.46

Table 7: **Exp. 5:** Impact of absolute latent position shifts on PTv3 and PTv1 test accuracy (↑%). PTv3 shows a substantial drop in accuracy when the latent cloud is shifted, demonstrating its sensitivity to absolute latent positions. PTv1 shows no noticeable accuracy degradation which is in line with its relative position mechanisms.

Method	Test on *Centered*	Test on *Displaced*
PTv3	78.58	62.78
PTv1	76.37	49.36

Table 8: **Exp. 5:** Test accuracy (↑%) when training PTv3 on *Centered* with random latent cloud shifts. Naturally, training with latent cloud shifts improves PTv3's robustness to translation and allows us to analyse the accuracy gap due to latent content differences between *Centered* and *Displaced*. PTv1 has this property built in but shows significantly weaker generalisation across the datasets.

In SIREN-based methods, evaluating on *Displaced* is almost equivalent to random guessing, as the displaced FMNIST images have induced a drastic change to the SIREN weight-space. For ARC, we observe a much smaller, but still significant drop in generalising to *Displaced* ARCs. Like with SIRENs, the shifted image content may have induced significant differences in the latent content. However, an additional cause may be at play, where PTv3 is overfitting to the absolute positions of the ARC latent vectors. Although PTv3 does not explicitly use absolute latent positions, its pooling and local attention mechanisms *do* rely on them. We test the trained PTv3 model on *Centered* ARCs where the entire latent cloud is shifted. In Table 7, this shift causes a significant drop in accuracy degrades, confirming that PTv3 is not translation invariant. We therefore also consider Point Transformer v1 (PTv1) which uses a simpler nearest neighbour mechanism relying on relative position (Zhao et al., 2021). PTv1 shows no significant changes in accuracy drop under the same conditions as PTv3.

To address the issue of translation sensitivity, we retrain PTv3 but randomly shift the ARCs during training. This forces PTv3 to become more robust to absolute position differences. When evaluated on *Displaced*, the accuracy drop is significantly reduced compared to the original setup, as shown in Table 8. With the increased robustness against absolute position differences, the observed performance gap can be attributed to latent content differences between *Centered* and *Displaced*. PTv1 is also tested but shows significantly weaker generalisation properties.

5 Conclusion

In this paper, we introduced ARC: Anchored Representation Clouds, a novel type of INR that retains image structure in its weight-space. ARC can leverage point cloud classification architectures to obtain state-of-the-art classification results and process image datasets that were previously unattainable for INR-based classification methods. Additionally, the unique weight-space of ARC provides efficient data augmentation techniques. For future work, making ARC adapt its capacity dynamically to the image's complexity would be an interesting avenue. Additionally, a key advancement in INR classification would entail the unification of the INR fitting and classification processes, which are currently treated as distinct stages. End-to-end coupling could lead to more competitive classification performance compared to image-space classification whilst being more memory-efficient.

REFERENCES

Matthias Bauer, Emilien Dupont, Andy Brock, Dan Rosenbaum, Jonathan Richard Schwarz, and Hyunjik Kim. Spatial functa: Scaling functa to imagenet classification and generation. *arXiv preprint arXiv:2302.03130*, 2023.

Rohan Chabra, Jan E Lenssen, Eddy Ilg, Tanner Schmidt, Julian Straub, Steven Lovegrove, and Richard Newcombe. Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIX 16*, pp. 608–625. Springer, 2020.

Yinbo Chen, Sifei Liu, and Xiaolong Wang. Learning continuous image representation with local implicit image function. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8628–8638, 2021.

Zhang Chen, Zhong Li, Liangchen Song, Lele Chen, Jingyi Yu, Junsong Yuan, and Yi Xu. Neurbf: A neural fields representation with adaptive radial basis functions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4182–4194, October 2023.

Shin-Fang Chng, Sameera Ramasinghe, Jamie Sherrah, and Simon Lucey. Gaussian activated neural radiance fields for high fidelity reconstruction and pose estimation. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner (eds.), *Computer Vision – ECCV 2022*, pp. 264–280, Cham, 2022. Springer Nature Switzerland. ISBN 978-3-031-19827-4.

Taco Cohen and Max Welling. Group equivariant convolutional networks. In Maria Florina Balcan and Kilian Q. Weinberger (eds.), *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pp. 2990–2999, New York, New York, USA, 20–22 Jun 2016. PMLR. URL https://proceedings.mlr.press/v48/cohenc16.html.

Elijah Cole, Grant Van Horn, Christian Lange, Alexander Shepard, Patrick Leary, Pietro Perona, Scott Loarie, and Oisin Mac Aodha. Spatial implicit neural representations for global-scale species mapping. In *International Conference on Machine Learning*, pp. 6320–6342. PMLR, 2023.

Luca De Luigi, Adriano Cardace, Riccardo Spezialetti, Pierluigi Zama Ramirez, Samuele Salti, and Luigi Di Stefano. Deep learning on implicit neural representations of shapes. In *International Conference on Learning Representations (ICLR)*, 2023.

Johannes Dollinger, Philipp Brun, Vivien Sainte Fare Garnot, and Jan Dirk Wegner. Sat-sinr: High-resolution species distribution models through satellite imagery. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 10:41–48, 2024.

Emilien Dupont, Adam Golinski, Milad Alizadeh, Yee Whye Teh, and Arnaud Doucet. COIN: COmpression with implicit neural representations. In *Neural Compression: From Information Theory to Applications – Workshop @ ICLR 2021*, 2021. URL https://openreview.net/forum?id=yekxhcsVi4.

Emilien Dupont, Hyunjik Kim, S. M. Ali Eslami, Danilo Jimenez Rezende, and Dan Rosenbaum. From data to functa: Your data point is a function and you can treat it like one. In *39th International Conference on Machine Learning (ICML)*, 2022.

Gabriel Eilertsen, Daniel Jönsson, Timo Ropinski, Jonas Unger, and Anders Ynnerman. Classifying the classifier: dissecting the weight space of neural networks. *ArXiv*, abs/2002.05688, 2020. URL https://api.semanticscholar.org/CorpusID:211096977.

Amer Essakine, Yanqi Cheng, Chun-Wun Cheng, Lipei Zhang, Zhongying Deng, Lei Zhu, Carola-Bibiane Schönlieb, and Angelica I Aviles-Rivero. Where do we stand with implicit neural representations? a technical and performance survey, 2024. URL https://arxiv.org/abs/2411.03688.

Elizabeth Fons, Alejandro Sztrajman, Yousef El-Laham, Alexandros Iosifidis, and Svitlana Vyetrenko. Hypertime: Implicit neural representations for time series. In *NeurIPS 2022 Workshop on Synthetic Data for Empowering ML Research*, 2022. URL https://openreview.net/forum?id=DZ2FaoMhWRb.

Rich Franzen. Kodak lossless true color image suite. 1999. URL https://r0k.us/graphics/kodak/.

Rui Gao and Rajeev K. Jaiman. H-siren: Improving implicit neural representations with hyperbolic periodic functions, 2024. URL https://arxiv.org/abs/2410.04716.

Simon Giebenhain and Bastian Goldlücke. Air-nets : An attention-based framework for locally conditioned implicit representations. In *2021 International Conference on 3D Vision, 3DV 2021 : , virtual conference ; 1-3 December 2021 : proceedings*, pp. 1054–1064, Piscataway, 2021. IEEE. ISBN 978-1-66542-688-6. doi: 10.1109/3DV53792.2021.00113.

Charles Godfrey, Davis Brown, Tegan Emerson, and Henry Kvinge. On the symmetries of deep learning models and their internal representations. *Advances in Neural Information Processing Systems*, 35:11893–11905, 2022.

Le Hou, Dimitris Samaras, Tahsin M Kurc, Yi Gao, James E Davis, and Joel H Saltz. Patch-based convolutional neural network for whole slide tissue image classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2424–2433, 2016.

Jeremy Howard. Imagenette, 2019. URL https://github.com/fastai/imagenette/.

Langwen Huang and Torsten Hoefler. Compressing multidimensional weather and climate data into neural networks. *arXiv preprint arXiv:2210.12538*, 2022.

Chiyu Jiang, Avneesh Sud, Ameesh Makadia, Jingwei Huang, Matthias Nießner, Thomas Funkhouser, et al. Local implicit grid representations for 3d scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6001–6010, 2020a.

Huaizu Jiang, Ishan Misra, Marcus Rohrbach, Erik Learned-Miller, and Xinlei Chen. In defense of grid features for visual question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 10267–10276, 2020b.

Ioannis Kalogeropoulos, Giorgos Bouritsas, and Yannis Panagakis. Scale equivariant graph metanetworks. *arXiv preprint arXiv:2406.10685*, 2024.

Miltiadis Kofinas, Boris Knyazev, Yan Zhang, Yunlu Chen, Gertjan J Burghouts, Efstratios Gavves, Cees GM Snoek, and David W Zhang. Graph neural networks for learning equivariant representations of neural networks. *arXiv preprint arXiv:2403.12143*, 2024.

Fanjie Kong and Ricardo Henao. Efficient Classification of Very Large Images with Tiny Objects. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2374–2384, New Orleans, LA, USA, June 2022. IEEE. ISBN 978-1-66546-946-3. doi: 10.1109/CVPR52688.2022.00242. URL https://ieeexplore.ieee.org/document/9879892/.

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

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

Tianyang Li, Xin Wen, Yu-Shen Liu, Hua Su, and Zhizhong Han. Learning deep implicit functions for 3d shapes with dynamic code clouds. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

David B Lindell, Dave Van Veen, Jeong Joon Park, and Gordon Wetzstein. Bacon: Band-limited coordinate networks for multiscale scene representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16252–16262, 2022.

Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *Advances in Neural Information Processing Systems*, 33:15651–15663, 2020.

Zhen Liu, Hao Zhu, Qi Zhang, Jingde Fu, Weibing Deng, Zhan Ma, Yanwen Guo, and Xun Cao. Finer: Flexible spectral-bias tuning in implicit neural representation by variable-periodic activation functions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2713–2722, 2024.

Julien N. P. Martel, David B. Lindell, Connor Z. Lin, Eric R. Chan, Marco Monteiro, and Gordon Wetzstein. Acorn: adaptive coordinate networks for neural scene representation. *ACM Trans. Graph.*, 40(4), July 2021. ISSN 0730-0301. doi: 10.1145/3450626.3459785. URL https://doi.org/10.1145/3450626.3459785.

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 (CVPR)*, June 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. *Commun. ACM*, 65(1):99–106, December 2021. ISSN 0001-0782. doi: 10.1145/3503250. URL https://doi.org/10.1145/3503250.

Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans. Graph.*, 41(4):102:1–102:15, July 2022. doi: 10.1145/3528223.3530127. URL https://doi.org/10.1145/3528223.3530127.

Aviv Navon, Aviv Shamsian, Idan Achituve, Ethan Fetaya, Gal Chechik, and Haggai Maron. Equivariant architectures for learning in deep weight spaces. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 25790–25816. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/navon23a.html.

Samuele Papa, Riccardo Valperga, David Knigge, Miltiadis Kofinas, Phillip Lippe, Jan-Jakob Sonke, and Efstratios Gavves. How to train neural field representations: A comprehensive study and benchmark. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22616–22625, 2024.

Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional occupancy networks. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*, pp. 523–540. Springer, 2020.

Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5301–5310. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/rahaman19a.html.

Sameera Ramasinghe and Simon Lucey. Beyond periodicity: Towards a unifying framework for activations in coordinate-mlps. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner (eds.), *Computer Vision – ECCV 2022*, pp. 142–158, Cham, 2022. Springer Nature Switzerland. ISBN 978-3-031-19827-4.

Vishwanath Saragadam, Daniel LeJeune, Jasper Tan, Guha Balakrishnan, Ashok Veeraraghavan, and Richard G Baraniuk. Wire: Wavelet implicit neural representations. In *Conf. Computer Vision and Pattern Recognition*, 2023.

Jonathan Richard Schwarz, Jihoon Tack, Yee Whye Teh, Jaeho Lee, and Jinwoo Shin. Modality-agnostic variational compression of implicit neural representations. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

Aviv Shamsian, Aviv Navon, David W. Zhang, Yan Zhang, Ethan Fetaya, Gal Chechik, and Haggai Maron. Improved generalization of weight space networks via augmentations. In *Forty-first International Conference on Machine Learning*, 2024. URL https://openreview.net/forum?id=3o7G6tIo4X.

Vincent Sitzmann, Julien N.P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. In *arXiv*, 2020.

Yannick Strümpler, Janis Postels, Ren Yang, Luc Van Gool, and Federico Tombari. Implicit neural representations for image compression. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner (eds.), *Computer Vision – ECCV 2022*, pp. 74–91, Cham, 2022. Springer Nature Switzerland. ISBN 978-3-031-19809-0.

Gergely Szabó and András Horváth. Mitigating the bias of centered objects in common datasets. In *2022 26th International Conference on Pattern Recognition (ICPR)*, pp. 4786–4792, 2022. doi: 10.1109/ICPR56361.2022.9956649.

Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pp. 6105–6114. PMLR, 2019.

Matthew Tancik, Pratul P. Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan T. Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *NeurIPS*, 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.

Thomas Unterthiner, Daniel Keysers, Sylvain Gelly, Olivier Bousquet, and Ilya Tolstikhin. Predicting neural network accuracy from weights. *arXiv preprint arXiv:2002.11448*, 2020.

Kushal Vyas, Ahmed Imtiaz Humayun, Aniket Dashpute, Richard G. Baraniuk, Ashok Veeraraghavan, and Guha Balakrishnan. Learning transferable features for implicit neural representations, 2024. URL https://arxiv.org/abs/2409.09566.

David R Wessels, David M Knigge, Samuele Papa, Riccardo Valperga, Sharvaree Vadgama, Efstratios Gavves, and Erik J Bekkers. Grounding continuous representations in geometry: Equivariant neural fields. *arXiv preprint arXiv:2406.05753*, 2024.

Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point transformer v3: Simpler faster stronger. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4840–4851, 2024.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms, 2017.

Yiheng Xie, Towaki Takikawa, Shunsuke Saito, Or Litany, Shiqin Yan, Numair Khan, Federico Tombari, James Tompkin, Vincent sitzmann, and Srinath Sridhar. Neural fields in visual computing and beyond. *Computer Graphics Forum*, 41(2):641–676, 2022. doi: https://doi.org/10.1111/cgf.14505. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.14505.

Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelNeRF: Neural radiance fields from one or few images. In *CVPR*, 2021.

Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 6023–6032, 2019.

Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars (eds.), *Computer Vision – ECCV 2014*, pp. 818–833, Cham, 2014. Springer International Publishing. ISBN 978-3-319-10590-1.

Jinlai Zhang, Lyujie Chen, Bo Ouyang, Binbin Liu, Jihong Zhu, Yujin Chen, Yanmei Meng, and Danfeng Wu. Pointcutmix: Regularization strategy for point cloud classification. *Neurocomputing*, 2022.

Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. Point transformer, 2021. URL https://arxiv.org/abs/2012.09164.

Zelin Zhao, Fenglei Fan, Wenlong Liao, and Junchi Yan. Grounding and Enhancing Grid-based Models for Neural Fields, April 2024. arXiv:2403.20002 [cs].

Jianqiao Zheng, Sameera Ramasinghe, Xueqian Li, and Simon Lucey. Trading positional complexity vs deepness in coordinate networks. In *European Conference on Computer Vision*, pp. 144–160. Springer, 2022.

Allan Zhou, Kaien Yang, Kaylee Burns, Adriano Cardace, Yiding Jiang, Samuel Sokota, J Zico Kolter, and Chelsea Finn. Permutation equivariant neural functionals. *Advances in neural information processing systems*, 36, 2024a.

Allan Zhou, Kaien Yang, Yiding Jiang, Kaylee Burns, Winnie Xu, Samuel Sokota, J Zico Kolter, and Chelsea Finn. Neural functional transformers. *Advances in neural information processing systems*, 36, 2024b.

A APPENDIX

A.1 ARC DEFINITION

We interpret a sampled signal s as a set of equidistant discrete observations $\{(\boldsymbol{x}_i \in \mathbb{N}^{d_{\text{in}}}, s(\boldsymbol{x}_i) \in \mathbb{N}^{d_{\text{out}}})\}_{i=1}^{b}$. For instance, an RGB image would be a set of pixel locations ($d_{\text{in}} = 2$) with corresponding RGB colours ($d_{\text{out}} = 3$). An INR learns parameters θ by supervising the mapping between the signal's domain and codomain $f_\theta : \mathbb{R}^{d_{\text{in}}} \mapsto \mathbb{R}^{d_{\text{out}}}$, supervising on $s(\boldsymbol{x})$, with mean squared error (MSE) loss. ARC is defined as follows.

$$f_\theta(\boldsymbol{x}) = g_\psi(e(\boldsymbol{x})) \quad \text{ARC} \quad (1)$$

$$\mathbb{R}^{d_{\text{in}}} \mapsto \mathbb{R}^{d_{\text{out}}} \quad (2)$$

$$e(\boldsymbol{x}) = \text{Concat}(U_n(\boldsymbol{x})) \quad \text{Encoder} \quad (3)$$

$$(\mathbb{R}^z \times \mathbb{R}^{d_{\text{in}}})^n \mapsto \mathbb{R}^{n\cdot(z+d_{\text{in}})} \quad (4)$$

$$g_\psi(\boldsymbol{v}_{\boldsymbol{x}}) = \text{MLP}_\psi(\boldsymbol{v}_{\boldsymbol{x}}) \quad \text{Decoder} \quad (5)$$

$$\mathbb{R}^{n\cdot(z+d_{\text{in}})} \mapsto \mathbb{R}^{d_{\text{out}}} \quad (6)$$

$$U_n(\boldsymbol{x}) = \{(\Delta \boldsymbol{p}_i, \boldsymbol{w}_i)\}_{i=1}^{n} \quad \text{Indexing function} \quad (7)$$

$$\mathbb{R}^{d_{\text{in}}} \mapsto (\mathbb{R}^{d_{\text{in}}} \times \mathbb{R}^z)^n \quad (8)$$

$$\mathcal{P} = \{(\boldsymbol{p}_i, \boldsymbol{w}_i)\}_{i=1}^{k} \quad \text{Latent cloud} \quad (9)$$

$$\text{where } \boldsymbol{p}_i \in \mathbb{R}^{d_{\text{in}}},\ \boldsymbol{w}_i \in \mathbb{R}^z \quad (10)$$

$$\theta = \{\psi, \{\boldsymbol{w}_i\}_{i=1}^{k}\} \quad \text{Learnable parameters} \quad (11)$$

A.2 FURTHER EXPERIMENTS.

ARC feature locality. In this experiment, we test our claim that the anchored latent vectors of ARC represent local image features. If so, applying a transformation to the ARC coordinates should yield a similarly transformed reconstruction. Furthermore, in classifying ARCs, the latent vector positions should prove to be relevant. To remind the reader, the absolute position of the points is not given as a feature to PTv3 to learn from.

We first consider transformations on the ARC coordinates. Given a trained ARC, we manipulate only the latent vector positions. The index function cache is refreshed and a forward pass is performed. In Figure 4, we perform several such transformations and show the resulting reconstructions. We can indeed verify that transforming latent vector positions yields correspondingly transformed reconstructions. If the anchored latents represent local image content, it is possible to mix ARCs. We can *e.g.* select parts of different ARCs or simply stack them. This is shown on the right in Figure 4, where two jointly trained ARCs produce interesting reconstructions when mixed. In areas where the image gradient norm of either image is high, we retain the original image features. As such, mixing ARCs yields a reconstruction in which the original images are still relatively well represented. This finding inspired our ARCmix data augmentation method.

		Train	
		Intact	Pushed
Test	Intact	79.32	51.79
	Pushed	17.39	69.06

Table 9: **Feature locality exp.:** FMNIST ARC test accuracy (%↑) if the latent vectors are displaced at either train or test time. Keeping the latent vectors' positions intact yields the highest accuracy, demonstrating their significance.

Let us investigate the local feature representations' significance on ARC classification accuracy. We reuse the FMNIST dataset that was padded to 100×100 in Experiment 1. The latent vectors of this dataset are pushed into a random direction once, and then used to 1) evaluate a normally trained PTv3 instance, and 2) train a PTv3 instance from scratch. In Table 9 we list the resulting test accuracies. Unsurprisingly, keeping the latent vector positions intact leads to the highest test accuracy. The regularly trained PTv3 instance shows a large drop in test accuracy when evaluated on the pushed data, as the learned relative positional encoding within PTv3 becomes meaningless. Conversely, when we train on pushed ARC data, the classifier still works relatively well. We hypothesise that the latent features are descriptive enough that, through pooling and attention, PTv3 learns a relatively informative global context. Accuracy drops when evaluated on intact ARCs, which we attribute to PTv3 learning inconsistent or incorrect relative positional encodings from the displaced data.

ARC reconstruction quality. Obtaining a high parameter efficiency or high reconstruction quality is not the primary objective of ARC. However, as it is commonplace in INR research, we will evaluate ARC's reconstruction capabilities and compare it to several baselines. Since reconstruction quality scales with INR capacity, we investigate various configurations of each INR type and report the

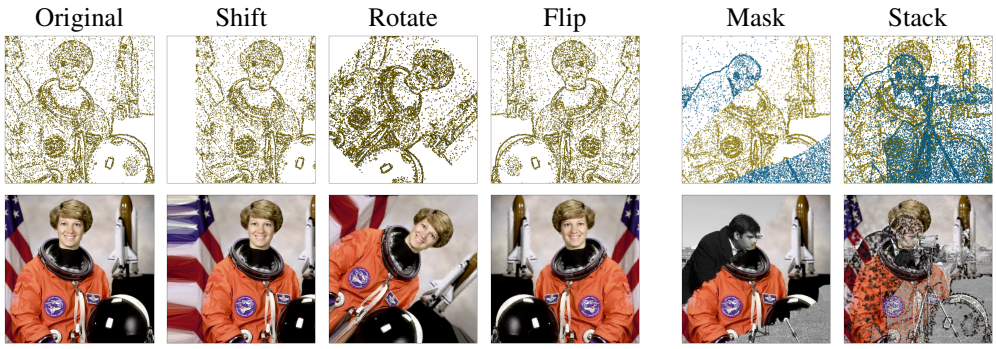

Figure 4: **Feature locality exp.:** Each column depicts a transformation on the latent vector positions, where the bottom row shows the resulting construction. No other changes are made to the ARC. The correspondence between the latent cloud transformation and the new reconstructions demonstrates how ARC encodes image features locally. In the rightmost two cases, two ARC, trained with a shared decoder, are mixed by masking specific latents or by simply stacking them.

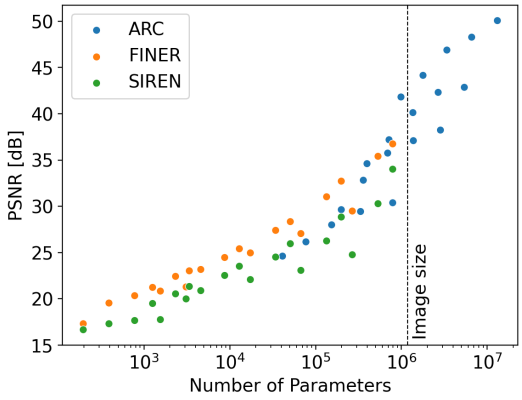

Figure 5: Image reconstruction results on the KODAK dataset. We compare ARC against SIREN and FINER, reporting PSNR as a function of model capacity. The vertical line represents the parameter count of a raw KODAK image, indicating the memory efficiency of different INR representations. SIREN and FINER experiments were bounded by computation costs, whereas ARC was not. ARC has a higher lower bound on memory than baselines but demonstrates superior scaling capabilities.

number of parameters alongside the obtained PSNR after 1000 training steps. Each configuration is used to fit the KODAK image dataset (Franzen, 1999) three times, taking the mean over all three runs and final reconstruction PSNRs. A mini-batch ratio of 1.0 is used except for ARC which uses a mini-batch ratio of 0.5. Results for SIREN, FINER (Liu et al., 2024), and ARC, are shown in Figure 5. For ARC, all combinations of $\{0.01, 0.05, 0.1, 0.25\} \cdot$ #pixels latent vectors and latent dimensions $\{8, 16, 32, 64, 128\}$ are used. For SIREN and FINER, combinations of $\{1, 2, 3, 4\}$ hidden layers and $\{32, 64, 128, 256, 512\}$ hidden dimension are used. SIREN and FINER dimensions could not be increased further due to the computation time increasing very steeply since all their weights must be updated for each supervised coordinate. Contrastingly, ARC uses 4 neighbouring latents for each coordinate, regardless of its capacity. ARC is therefore significantly faster to train and more scalable.

A vertical line denotes the number of values a KODAK image contains if it were loaded into memory at $768 \cdot 512 \cdot 3 = 1.1\text{e}6$ parameters. Any INR instance with fewer parameters is arguably more memory-efficient than the original image. Larger INR representations show how PSNR increases if more memory is available, though such representations would not directly yield memory benefits. For ARC, the decoder's parameters are not included as the decoder is shared across all instances of a given architecture and its cost is thus amortised. ARC shows a generally higher parameter cost than MLP-based INR methods.

A.3 Further ablations

Ablation vs. classification accuracy. In this ablation experiment, we ablate various aspects of ARC and PTv3 to observe their impact on classification accuracy. For each ablation, ARCs are fit on a subset of 10k CIFAR10 images.

First, we compare latent vector normalisation techniques and their impact on validation accuracy. Three normalisation strategies are compared. None: where no normalisation is applied, Normalise Whole: where the latent vectors are normalised using a scalar mean and standard deviation which are precomputed on a subset of the ARCs, and Normalise Per-dim: where the mean and standard deviation are precomputed per latent dimension on a subset of the ARCs. The effect of these techniques on test accuracy is listed in Table 10. Generally, applying normalisation to the latent vectors improves classification accuracy. This is in line with findings in other INR classification literature (Navon et al., 2023; Zhou et al., 2024b;a; Kofinas et al., 2024). Furthermore, applying normalisation across individual latent dimensions yields the highest improvement. This suggests that capturing variations specific to each latent dimension provides a more robust representation in classifying ARCs. We hypothesise that certain latent dimensions specialise in capturing distinct image features. This type of alignment would be induced by sharing the decoder. A similar property is introduced manually in Chen et al. (2023) using harmonics of different frequencies, which results in better INR reconstructions.

Next, we explore the trade-off between the number of latent vectors and the latent dimensionality of each latent vector in ARC, and aim to analyse their impact on classification accuracy. We fit a subset of 10k CIFAR10 images to each combination. In Table 11 the validation accuracies which PTv3 converges to are listed. Accuracy improves significantly as we increase the number of latent vectors in the image. Conversely, increasing the latent dimension yields diminishing returns on accuracy.

Latent normalisation	Val. Acc.
None	52.67
Normalise Whole	54.58
Normalise Per-dim	58.68

Table 10: Validation accuracy (%↑) under different latent normalisation strategies. We show how normalising each latent dimension independently yields the highest increase in performance.

		Latent dimension		
		8	**16**	**32**
Latents	**10**	30.95	32.91	29.01
	25	41.12	40.05	41.43
	50	49.14	48.62	48.02

Table 11: Validation accuracy (%↑) for differing number of latents and latent dimensions. Increasing the number of latents has a clear positive impact on classification accuracy, whereas high latent dimensions provide diminishing results, particularly when there are few latents.

Ablation vs. PSNR. In this ablation experiment, we investigate different ARC configurations and observe their effect on image reconstruction quality. Given skimage's astronaut image, we train an ARC instance with various combinations of latent dimensions and number of latents, the latter being expressed as a factor of the number of pixels in the image. Figure 6 shows the gradual increase in reconstruction quality as either the number of latents or the latent dimensions increase. This plot is generally in line with our expectations.

A.4 Implementation details

ARC implementation details. Provided with an image, the latent positions are determined by sampling the image gradient norm. Like (Chen et al., 2023), the number of latent vectors is proportional to the number of pixels in the image. We experimented with different combinations of the number of latents and latent dimension and found that # latents $= 0.05 \cdot$ # image pixels, combined with a feature dimension of $z = 32$ works well for most cases. This configuration is used in all experiments, unless otherwise noted. Like (Chen et al., 2023), the feature vectors are drawn from $U(-1\text{e}{-4}, 1\text{e}{-4})$. Upon initialisation, the indexing function U_n caches the index and relative position to the n nearest latent vectors. In all our experiments, we use a decoder of size $[n \cdot (z+2), n \cdot (z+2), d_{\text{out}}]$. The decoder is pre-trained by jointly training 100 ARC instances, aggregating the loss over all instance

for each step. All our ARC instances, as well as the pretrained decoder, are trained for 500 steps. In fitting ARC we make use of mini-batching, whereby only 25% of the image coordinates are supervised each iteration. We found that this makes qualitatively little difference in reconstruction quality but makes fitting significantly faster. An ADAM optimiser is used with a learning rate of 0.005. We normalise images to [0,1] range.

Point Transformer v3 implementation details. Point Transformer v3 (PTv3) (Wu et al., 2024) is not intended to be used outside of point-cloud tasks, which typically contain a 3D coordinate with optional features such as colours or normals. Hence, PTv3 requires a few tweaks in order to be compatible with ARC. For instance, the latent coordinates are made three-dimensional by appending a 0 to them. To make PTv3 suitable for classification tasks, we replace the standard upsampling blocks by a global pooling layer, followed by a single linear layer that maps the feature dimension to the number of classes.

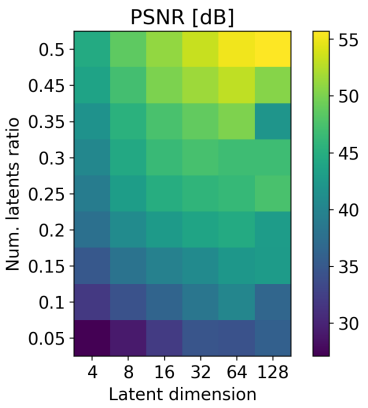

Figure 6: PSNR for different ARC configurations on skimage's astronaut image. The vertical axis denotes the ratio of the image's pixels that contain a latent vector.

SIREN implementation details. Several of our experiments required custom SIREN datasets. We aimed to follow DWSnets implementation but found it to be erroneous compared to the regular SIREN specification (Sitzmann et al., 2020). Specifically, a 0.5 offset is added to the network's output, the first layer does not follow the prescribed initialisation scheme, and the bias layer is initialised to zero rather than the prescribed weight initialisation. We found that these errors are not readily apparent in low-resolution images such as the ones commonly used in INR classification. In fitting our larger resolution images, we observe heavy blurring in the reconstruction Figure 7. We opted to use a correct SIREN implementation instead.

We use default SIREN hyperparameters; $\omega_0 = 30.0$, no final layer activation function, ADAM optimiser and learning rate of 0.0005. The SIRENs are trained for 1000 steps.

A.5 Baseline details

ScaleGMN. In utilising the ScaleGMN baseline (Kalogeropoulos et al., 2024), it proved to be rather unstable in training. The authors acknowledge this issue and take measures such as layer normalisation and skip connections to mitigate it. ScaleGMN is designed to work on SIREN datasets introduced by the DWSnets paper (Navon et al., 2023). We found that a faulty SIREN implementation was used in creating these datasets (section A.4). The SIREN datasets that we created therefore present an extra challenge as the provided ScaleGMN settings were created with DWSnet-SIRENs in mind. To quantify this error, we train ScaleGMN using the provided MNIST settings on 10k SIRENs that we fit ourselves. The resulting test accuracy is 88.89% after 71 epochs, which is 7.68% lower than the test accuracy which the authors obtained 96.57% on the whole MNIST dataset. We proceed to use ScaleGMN in our experiments where we use our own SIREN datasets, but are aware of the possibly suboptimal performance.

Toy datasets. In most experiments with toy datasets FMNIST was chosen as a basis. FMNIST poses a non-trivial classification challenge for current methods (Navon et al., 2023; Kofinas et al., 2024; Kalogeropoulos et al., 2024) and can easily be padded to increase the image size.

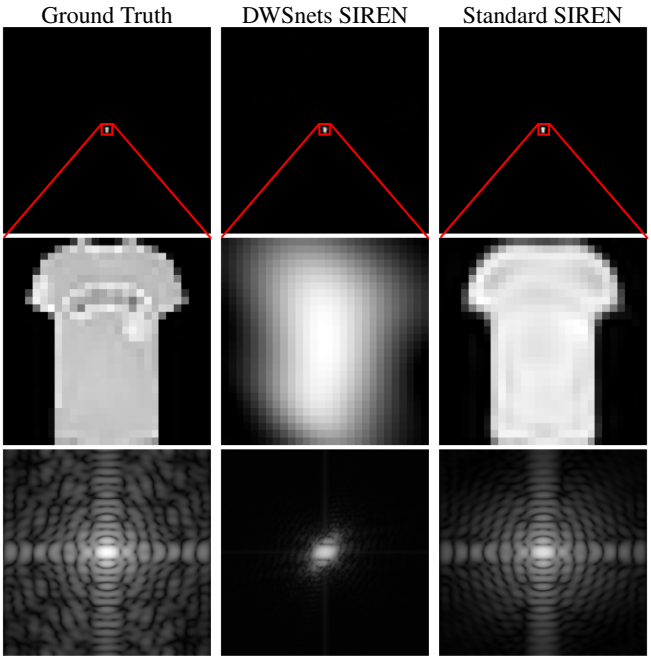

Figure 7: Comparison of 1024×1024 FMNIST objects reconstructed using DWSnets' SIREN implementation and a corrected SIREN implementation. Rows depict (from top to bottom) the full image, a zoomed-in view, and the FFT of the reconstruction. The left column shows the original image, highlighting its increased frequency content. From the FFTs, it is clear that spectral bias phenomena re-emerge in the SIREN INRs.

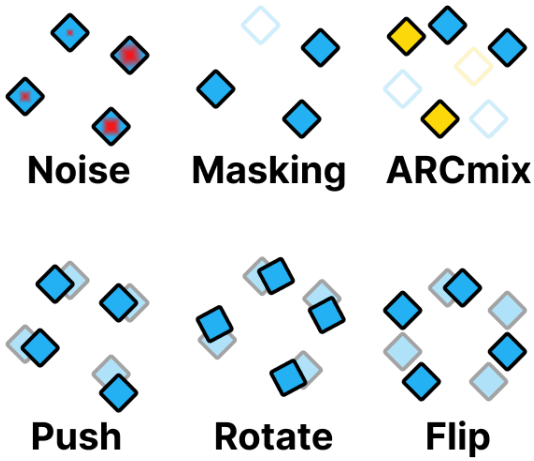

Figure 8: Data augmentation techniques for ARCs. Noise, Masking, and ARCmix manipulate the latent cloud to enhance data diversity, while Push, Rotate, and Flip change just the latent vector coordinates. These augmentations operate directly in the ARC weight-space.

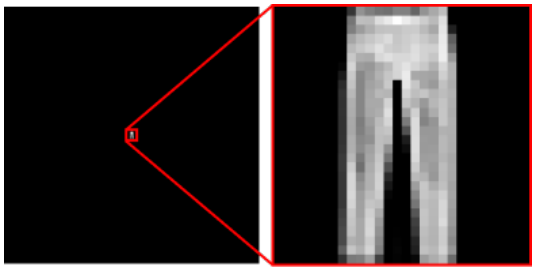

Figure 9: Left, a sample from the 1024×1024 padded FMNIST dataset. Right, a zoomed in view of the depicted object. The high-resolution of the image, transforms the relatively simple task of FMNIST classification into a challenge for baseline INR classification methods.

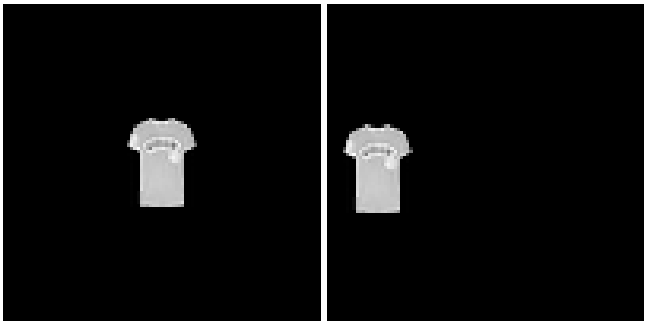

Figure 11: **Exp. 5:** Samples from the *Centered* and *Displaced* datasets.