From data to functa: Your data point is a function and you can treat it like one

Emilien Dupont [*1] **Hyunjik Kim** [*2] **S. M. Ali Eslami** [2] **Danilo Rezende** [2] **Dan Rosenbaum** [3 2]

Abstract

It is common practice in deep learning to represent a measurement of the world on a discrete grid, e.g. a 2D grid of pixels. However, the underlying signal represented by these measurements is often continuous, e.g. the scene depicted in an image. A powerful continuous alternative is then to represent these measurements using an *implicit neural representation*, a neural function trained to output the appropriate measurement value for any input spatial location. In this paper, we take this idea to its next level: what would it take to perform deep learning on these functions instead, treating them as data? In this context we refer to the data as

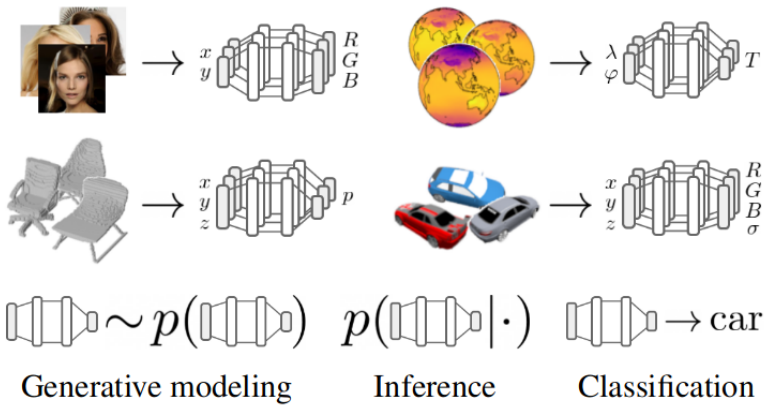

Figure 1. We convert array data into functional data parameterized by neural networks, termed *functa*, and treat these as data points for various downstream machine learning tasks.

Summary

Instead of using discrete grids of measurements, use a continuous alternative called **Implicit Neural Representation (INR)**, a neural function trained to output the appropriate measurement value for any input spatial location.

$$\min_{\theta} \mathcal{L}(f_\theta, \{\mathbf{x}_i, \mathbf{f}_i\}_{i\in\mathcal{I}}) = \min_{\theta} \sum_{i\in\mathcal{I}} \underbrace{\|f_\theta(\mathbf{x}_i) - \mathbf{f}_i\|_2^2}_{\text{INR}}.$$

Advantages:

- allows arbitrary resolutions
- memory-efficiency
- Multimodality
- Easing downstream task

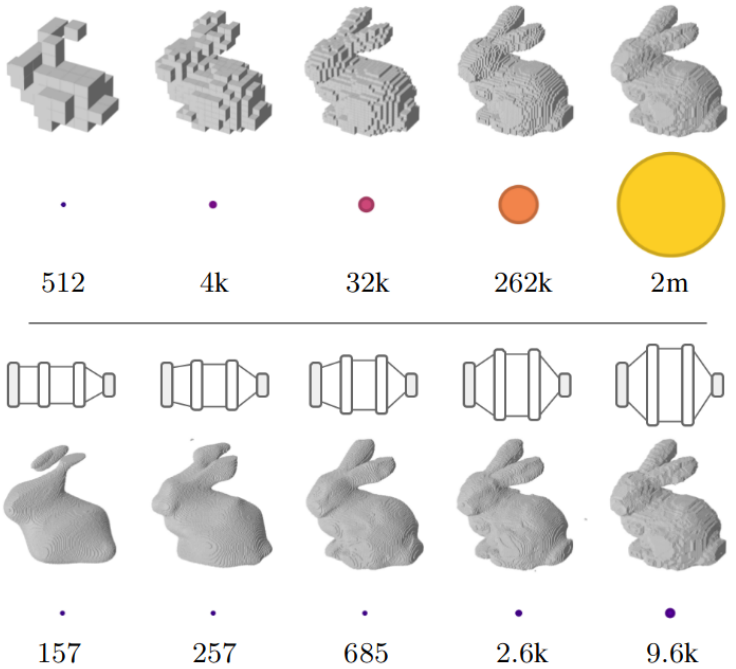

Figure 2. Functa scale much more gracefully with resolution than array representations. Circle area reflects the numerical size of the array (top) / function (bottom). See Appendix A.9 for details.

Functa as MLP modulations

- Use **SIREN** as base architectures because they are known to efficiently resent a wide range of data modalities

$$\mathbf{x} \mapsto \sin(\omega_0(\mathbf{W}\mathbf{x} + \mathbf{b}))$$

- The naive approach for representing functa is to take the parameter vector of the SIREN.
- Many suggest modulations as an alternative that uses a shared base network across data points to model common structure, with modulations modeling the variation specific to each data point

$$\mathbf{x} \mapsto \sin(\omega_0(\mathbf{W}^{(i)}\mathbf{x} + \mathbf{b}^{(i)} + \mathbf{s}^{(i)}))$$

s is shift modulation

- We use **SIREN with latent modulation**s:

$$\mathbf{s} = \mathbf{W}'\phi + \mathbf{b}'$$

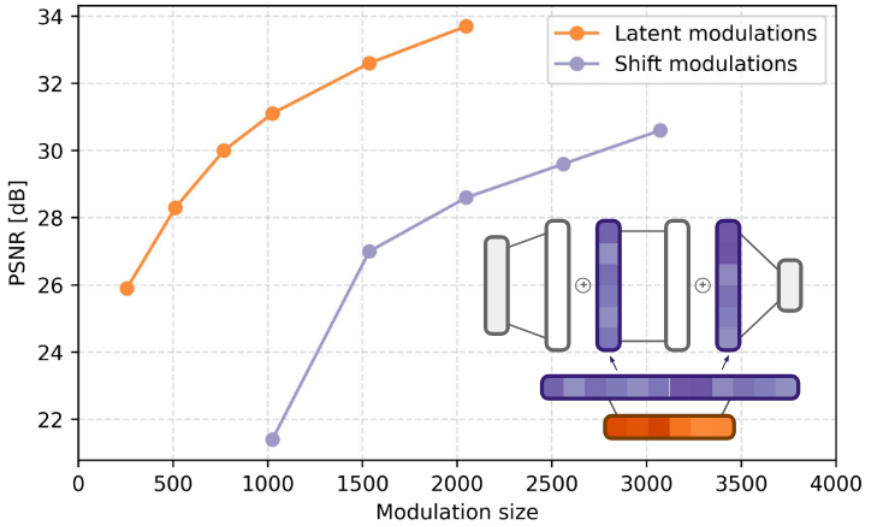

Figure 3. Reconstruction accuracy (in PSNR) vs modulation dimensionality on CelebA-HQ 64×64. Reconstruction accuracy is computed from the MSE between the image array and its INR evaluated at each pixel location. The model architecture is shown on the bottom right, with purple vectors corresponding to shift modulations and orange vectors to latent modulations.

W' & b' used for mapping latents to shift modulations are part of the base network shared across data points)

Meta-learning functa

1.Meta-learn the base network

2.For each data sample, run the inner loop with a few gradient steps

Algorithm 1 Meta-learning functa

Randomly initialize shared base network θ
while not done **do**
  Sample batch $\mathcal{B}$ of data $\{\{\mathbf{x}_i^{(j)}, \mathbf{f}_i^{(j)}\}_{i\in\mathcal{I}}\}_{j\in\mathcal{B}}$
  Set batch modulations to zero $\phi_j \leftarrow 0\ \forall j \in \mathcal{B}$
  for all step $\in \{1, \ldots, N_{inner}\}$ and $j \in \mathcal{B}$ **do**
    $\phi_j \leftarrow \phi_j - \epsilon \nabla_\phi \mathcal{L}(f_{\theta,\phi}, \{\mathbf{x}_i^{(j)}, \mathbf{f}_i^{(j)}\}_{i\in\mathcal{I}})|_{\phi=\phi_j}$
  end for
  $\theta \leftarrow \theta - \epsilon' \frac{1}{|\mathcal{B}|} \sum_{j\in\mathcal{B}} \nabla_\theta \mathcal{L}(f_{\theta,\phi}, \{\mathbf{x}_i^{(j)}, \mathbf{f}_i^{(j)}\}_{i\in\mathcal{I}})|_{\phi=\phi_j}$
end while

"In the inner loop we therefore only update the modulations, whereas in the outer loop we only update the base network weights."

Deep Generative modelling on functa

- Train deep learning directly on the modulations
- Experiments on generativemodeling with Normalizing flows, to map a simple base distribution through a sequence of invertible layers parameterized by neural networks.

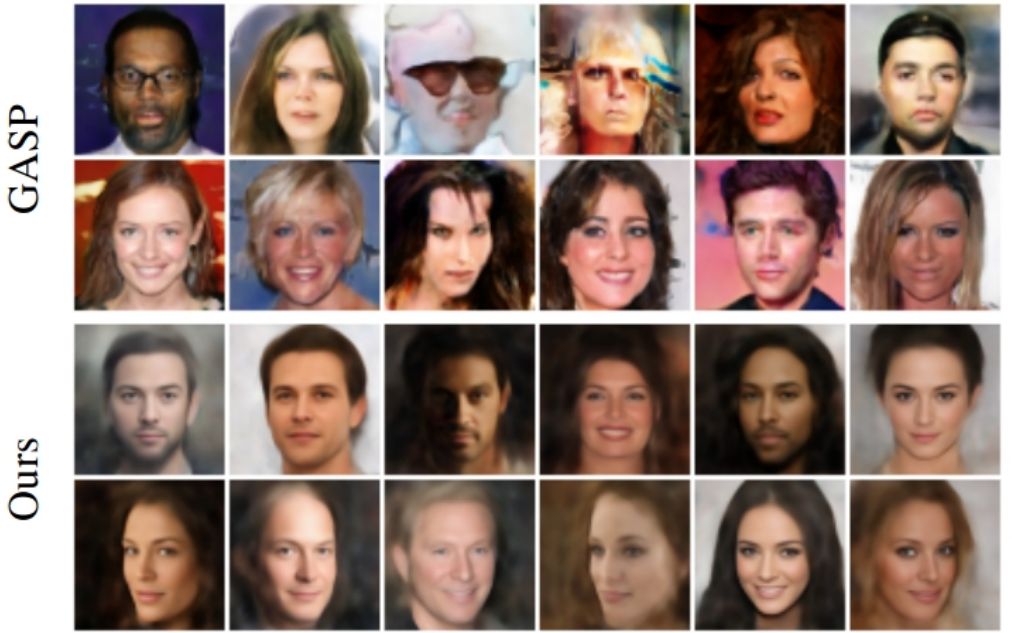

Figure 5. Uncurated samples from GASP and DDPM (diffusion) trained on 256-dim CelebA-HQ 64×64 modulations.

Classification Experiments

CLASSIFIER	TEST ACCURACY	n_{PARAMS}
MLP ON FUNCTA	$93.6 \pm 0.1\%$	83K
3D CNN	$93.3 \pm 0.3\%$	550K

Table 2. Classification accuracies and parameter count for MLP on functa vs 3D CNN on array data for ShapeNet 10 Classes, 64^3.

	MLP ON FUNCTA			3D CNN ON ARRAY		
TEST ACCURACY	$93.6 \pm 0.1\%$	$93.9 \pm 0.1\%$	$94.0 \pm 0.1\%$	$93.3 \pm 0.3\%$	$94.5 \pm 0.2\%$	$94.8 \pm 0.0\%$
n_{PARAMS}	83K	212K	924K	550K	2.0M	7.9M

Table 6. Same as Table 2 but for a wider range of model sizes

References

https://proceedings.mlr.press/v162/dupont22a/dupont22a.pdf

https://github.com/google-deepmind/functa