ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) | 979-8-3503-4485-1/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICASSP48485.2024.10446136

TITAN: BRINGING THE DEEP IMAGE PRIOR TO IMPLICIT REPRESENTATIONS

Lorenzo Luzi[1], *Daniel LeJeune*[1], *Ali Siahkoohi*[1], *Sina Alemohammad*[1], *Vishwanath Saragadam*[1], *Hossein Babaei*[1], *Naiming Liu*[1], *Zichao Wang*[2], *Richard G. Baraniuk*[1]

[1]Rice University [2]Adobe Research

ABSTRACT

We study the interpolation capabilities of implicit neural representations (INRs) of images. In principle, INRs promise a number of advantages, such as continuous derivatives and arbitrary sampling, being freed from the restrictions of a raster grid. However, empirically, INRs have been observed to poorly interpolate between the pixels of the fit image; in other words, they do not inherently possess a suitable prior for natural images. In this paper, we propose to address and improve INRs' interpolation capabilities by explicitly integrating image prior information into the INR architecture via deep decoder, a specific implementation of the deep image prior (DIP). Our method, which we call TITAN, leverages a residual connection from the input which enables integrating the principles of the grid-based DIP into the grid-free INR. Through super-resolution and computed tomography experiments, we demonstrate that our method significantly improves upon classic INRs, thanks to the induced natural image bias. We also find that by constraining the weights to be sparse, image quality and sharpness are enhanced, increasing the Lipschitz constant.

***Index Terms*—** Implicit neural representations, deep image prior, sparsity

1. INTRODUCTION

Implicit neural representations (INRs) aim to represent images with a differentiable (neural network) function instead of the traditional discrete raster image of pixel point values in a 2-dimensional grid. Concretely, an INR might learn a function mapping from an arbitrary location in 2D, represented by (x, y), to the (r, g, b) values in the image:

$$\boldsymbol{I} : \mathbb{R}^2 \to \mathbb{R}^3, \quad \boldsymbol{I}(x, y) = (r, g, b)\,. \tag{1}$$

Thanks to their desirable characteristics including being grid-free, continuous, and differentiable, INRs have been an increasingly popular and integral component for a wide range of computer graphics and vision applications such as super-resolution [1, 2], 3D scene rendering [3–6], and image generation [7–10]. Among the most famous INRs is SIREN [11], which consists of a feed-forward neural network with sinusoidal activation functions. Others have used wavelet activations instead of sinusoidal ones with significant success [12].

The most common class of INR models only use a single datum and do not require a training data corpus. Models trained with a collection of images may suffer from generalization problems due to under-specialization. Interpolating a single image without training makes it more suitable and safer in high-stakes applications such as computed tomography (CT) imaging, where images may vary significantly from patient to patient due to their different anatomical structures [13]. Moreover, data-free INRs can be deployed to challenging imaging tasks in data-starved environments.

However, a notable drawback of INRs is that they are often poor interpolators. Prior work [11, 14] has empirically demonstrated that INRs often fail to represent images in finer scales; see Figure 1 for an illustration of the unsatisfactory image representation performance of SIREN in a super-resolution case study. This practical deficiency is concerning because the continuous, grid-free nature of INRs should enable image representation at arbitrary scale and resolution.

In this paper, we consider INRs' grid-free interpolation capabilities without relying on other (neural) image models which use discrete pixel representations. Instead, we take inspiration from the deep decoder [15] and design an INR architecture which is both grid-free and leverages the deep image prior [16].

1.1. Contributions

We propose a new method to significantly improve INR interpolation capabilities. Our method is inspired by the deep image prior (DIP) [16], an observation that certain architectural choices in generative networks bias them towards natural images. We postulate that INRs suffer from poor interpolation capabilities because they lack such architectural inductive biases.

To this end, we integrate DIP information into an INR using deep decoder, a simplified implementation of DIP,

This work was supported by NSF grants CCF-1911094, IIS-1838177, and IIS-1730574; ONR grants N00014-18-12571, N00014-20-1-2534, and MURI N00014-20-1-2787; AFOSR grant FA9550-22- 1-0060; and a Vannevar Bush Faculty Fellowship, ONR grant N00014-18-1-2047. Work done while ZW was at Rice University.

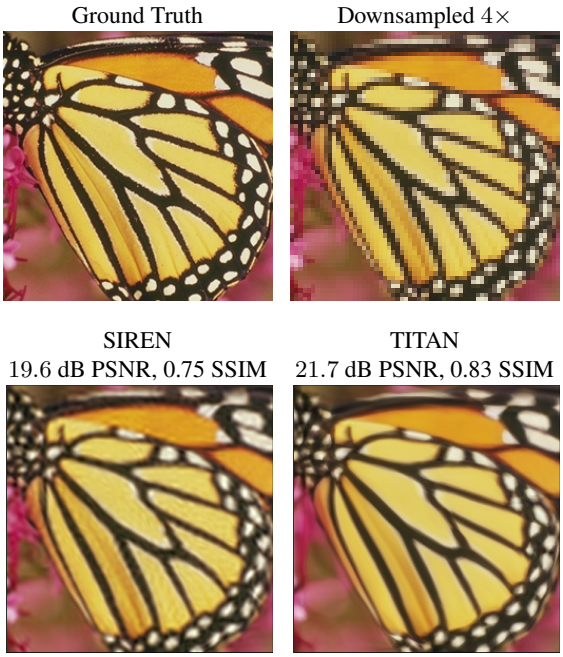

Fig. 1. Using TITAN we outperform SIREN on a $4\times$ super-resolution task. TITAN here has approximately the same parameters ($\approx 112\,\mathrm{k}$) as SIREN ($\approx 133\,\mathrm{k}$) due to the weights being sparse.

which automatically enforces strong prior information for a given image with a powerful but simple neural network. We add residual connections from the input that enable us to integrate the principles of deep decoder, which normally operates in the pixel representation of images, into an INR, which operates in the grid-free functional representation of images. This yields an image representation with powerful inductive biases. Through a series of experiments on image super-resolution and CT recovery, we show that our method, TITAN, outperforms the INR methods without image prior by a large margin, demonstrating the importance and the promise in leveraging image prior information in INRs to turn them into powerful interpolators.

2. BACKGROUND

Our method fuses DIP techniques, specifically deep decoder, with INRs that allow grid-free image inference.

2.1. Deep image priors and deep decoder

Deep image prior [16] proposes an untrained network to capture image statistics prior for inverse problems, such as denoising, image inpainting, and super-resolution. DIP takes a random vector as input and outputs the image prior using a UNet-like [17] network such as hourglass network or encoder–decoder with skip connections. It surprisingly shows that the simple structure of a deep convolution network is sufficient to enforce reasonable image priors without the need for extensive training on large datasets. Inspired by the work of DIP, we propose our method TITAN to learn a high-quality implicit representation from a single image.

Deep decoder. The deep decoder [15] is a raster-based (i.e., defined on a grid) implicit image representation that distills the ideas behind the DIP to its essential components. It differs from the DIP in that it has no skip connections and no convolutional operations, but rather limits inter-pixel interactions to only upsampling operations. Starting with a fixed $n_0 \times n_0 \times k_0$ random "image" $\boldsymbol{B}_0$, the architecture layers proceed via the following recursive relation:

$$\boldsymbol{B}_{i+1} = \mathrm{cn}(\mathrm{relu}(\boldsymbol{U}_i \boldsymbol{B}_i \boldsymbol{C}_i)). \tag{2}$$

The nonlinearity is $\mathrm{relu}(t) = \max\{0, t\}$, and cn is a channel-wise normalization, also called a one-dimensional batch normalization [18], with optional learned parameters. For $\boldsymbol{B}_{i+1}$ of shape $n_i \times n_i \times k_i$, $\boldsymbol{U}_i$ is a fixed bilinear upsampling operator that lifts each channel from $n_i \times n_i$ to $n_{i+1} \times n_{i+1}$, where $n_{i+1} = 2n_i$. The weights $\boldsymbol{C}_i \in \mathbb{R}^{k_i \times k_{i+1}}$ are learned parameters which mix the k_i channels to k_{i+1} channels. The final result is an $n_d \times n_d \times k_d$ raster image $\boldsymbol{I}$ formed by

$$\boldsymbol{I} = \mathrm{sigmoid}(\boldsymbol{B}_d \boldsymbol{C}_d), \tag{3}$$

where $\mathrm{sigmoid}(t) = 1/(1 + e^{-t})$. Training the weights $\boldsymbol{C}_i$ via gradient descent results in a robust image prior.

3. METHOD

We would like to take advantage of the inductive biases of the deep image prior and deep decoder in an INR, but the convolutional and upsampling operations that incorporate inter-pixel interactions preclude a direct implementation. To address this, we propose TITAN, which stands for Deep Implicit Decoder Network,[1] which implements a deep decoder architecture as an INR via a careful replacement of the upsampling operator with spatial residual connections. A diagram of TITAN is shown in Figure 2.

We seek an implicit representation $\boldsymbol{I}_\theta \colon \mathbb{R}^2 \to \mathbb{R}^{k_d}$ with parameters $\theta \in \mathbb{R}^p$ of an image such that given a coordinate $\boldsymbol{x} \in \mathbb{R}^2$, $\boldsymbol{I}_\theta(\boldsymbol{x})$ returns the pixel values at that coordinate across the k_d channels. We start the same as the deep decoder with $n_0 = 1$: we let $\boldsymbol{B}_0 \in \mathbb{R}^{k_0}$ be a vector representing a constant image of k_0 channels. We wish to implement the next layer of the deep decoder, which according to equation 2 should look something like

$$\boldsymbol{B}_{i+1}(\boldsymbol{x}) = \mathrm{cn}(\mathrm{relu}(\boldsymbol{C}_i[(\boldsymbol{U}_i \boldsymbol{B}_i)(\boldsymbol{x})])). \tag{4}$$

[1] We prefer the nearly phonetically identical TITAN to DIDN because it is both beautiful and good.

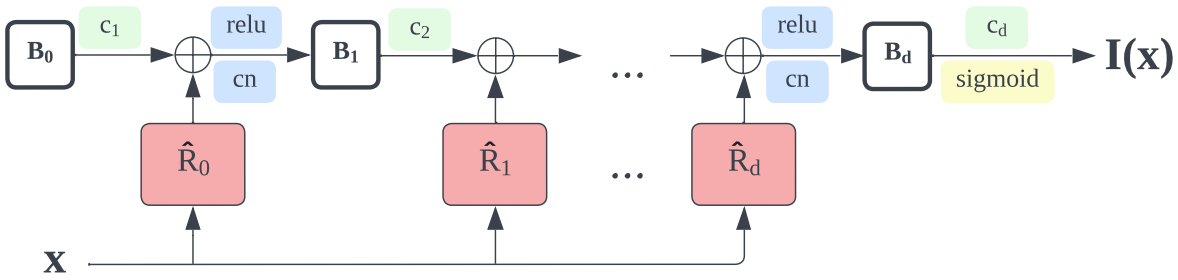

Fig. 2. A diagram of the TITAN architecture. The input coordinate $\boldsymbol{x}$ passes through the residual blocks $\widehat{\boldsymbol{R}}_i$ as a substitute for upsampling but otherwise the output pixel value follows the deep decoder architecture.

However, in order to have a one-to-one implementation of deep decoder as an INR, we would need $(\boldsymbol{U}_i\boldsymbol{B}_i)(\boldsymbol{x})$ to be the upsampled version of the image $\boldsymbol{B}_i\colon \mathbb{R}^2 \to \mathbb{R}^{k_i}$ evaluated at $\boldsymbol{x}$. Upsampling does not really have meaning for non-raster images; for raster images, however, it is the inverse operation of downsampling, which is analogous to blurring. Thus, $\boldsymbol{B}_i(\boldsymbol{x})$ should be similar to $(\boldsymbol{U}_i\boldsymbol{B}_i)(\boldsymbol{x})$, but with less detail. We can therefore define the upsampling *residual*

$$\boldsymbol{R}_i(\boldsymbol{x}) \triangleq (\boldsymbol{U}_i\boldsymbol{B}_i)(\boldsymbol{x}) - \boldsymbol{B}_i(\boldsymbol{x}). \tag{5}$$

What we do in TITAN is explicitly approximate $\boldsymbol{R}_i(\boldsymbol{x})$ via a simple nonlinear function, like a small SIREN network:

$$\widehat{\boldsymbol{R}}_i(\boldsymbol{x}) = g(\alpha_i(\boldsymbol{W}_i\boldsymbol{x} + \boldsymbol{v}_i))/d, \tag{6}$$

where g is an element-wise nonlinearity that captures spatial frequency information—we use $g(t) = \sin(t)$; α_i is a fixed frequency scaling parameter; and dividing by d ensures that the total contribution of the residuals is fixed. We then have the TITAN layer update function

$$\boldsymbol{B}_{i+1}(\boldsymbol{x}) = \mathrm{cn}(\mathrm{relu}(\boldsymbol{C}_i\boldsymbol{B}_i(\boldsymbol{x}) + \widehat{\boldsymbol{R}}_i(\boldsymbol{x}))), \tag{7}$$

with final output

$$\boldsymbol{I}_\theta(\boldsymbol{x}) = \mathrm{sigmoid}(\boldsymbol{C}_d\boldsymbol{B}_d(\boldsymbol{x})). \tag{8}$$

In our experiments, we are not concerned with ensuring the differentiability of our TITANs, so we use the relu nonlinearity even though it is non-differentiable. If differentiability is necessary, the nonlinearity can be replaced by a differentiable alternative such as the softplus function.

4. EXPERIMENTS

For all experiments, we set the frequency scaling parameters of TITAN to $\alpha_i = 4(i+1)$ so that higher frequencies are captured in deeper layers, and we let $k_0 = k_1 = \ldots k_{d-1} = 100$. Weight initializations for TITAN are the default PyTorch initialization. We use the Adam [19] optimizer with learning rate 10^{-3} for SIREN and TITAN and learning rate 10^{-2} for DIP unless otherwise specified. Code can be found at `https://github.com/dlej/titan-implicit-prior`

4.1. TITAN for super-resolution

We perform $4\times$ super-resolution of a 256×256 image downsampled to 64×64 and show the result in Figure 1. For TITAN, we use depth $d = 10$, and for SIREN we use $d = 2$. We use the AdaBreg [20] optimizer for sparsity.

As we can see, TITAN results in a much sharper image than SIREN, which suffers from Gibbs-like ringing artifacts. For this task, DIP achieves a PSNR of 23.9 and 0.89 SSIM, so TITAN is a solid step in the direction of inductive image biases for INRs.

4.2. TITAN for computed tomography

For the ground truth CT image $\boldsymbol{Y} \in \mathbb{R}^{W\times H}$ in Figure 3, we take noisy measurements of the following form:

$$\boldsymbol{B} = \mathrm{RT}(\boldsymbol{Y}, m) + \boldsymbol{S} \in \mathbb{R}^{m\times H} \tag{9}$$

Where $\mathrm{RT}(\boldsymbol{Y}, m)\colon \mathbb{R}^{W\times H} \to \mathbb{R}^{m\times H}$ is the Radon Transform with m uniformly spaced samples from 0 to π and $\boldsymbol{S}_{ij} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma^2)$ is random Gaussian noise with standard deviation $\sigma = 2$. We optimize the following cost function

$$\mathcal{L}(\theta) = \|\mathrm{RT}(\boldsymbol{I}_\theta(\boldsymbol{x}), m) - \boldsymbol{B}\|_F^2 \tag{10}$$

using Adam with cosine annealing rate. Results of all methods for several numbers of measurements m are shown in Table 1. TITAN provides a significantly better implicit representation for solving this inverse problem than SIREN, nearly matching or even surpassing the DIP at all measurement levels.

4.3. The effect of sparsity

Partly motivated by the success of INRs in data compression [21], we propose to compensate for the larger parameter count of TITAN compared to SIREN via sparsity-promoting INF optimization (c.f. equation 10). We achieve this via an optimization approach [20] based on linearized Bregman iterations [22]. Unlike pruning methods [23], this Bregman learning method initializes the INR with a

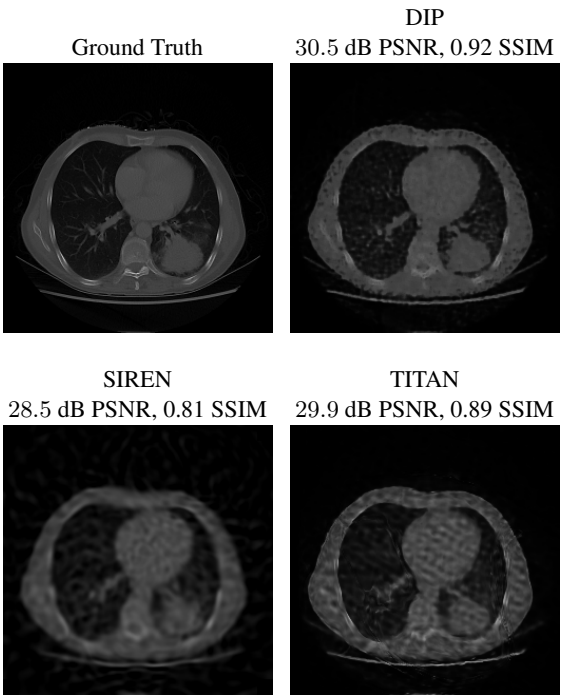

Fig. 3. For number of measurements $m = 30$, TITAN outperforms SIREN by leveraging the deep image prior's inductive bias. Thus, TITAN has all the benefits of being an implicit representation network with only a small ($\approx$ 0.6dB) cost to performance. Also TITAN begin to outperform SIREN and DIP for larger number of measurements.

few nonzero weights, successively adding limited nonzero weights throughout optimization. We tune the final sparsity factor of the weights via a hyperparameter, which controls the initialization sparsity factor. Our empirical results for image super-resolution and computed tomography demonstrate both quantitative, measured via PSNR, and perceptual improvement of the resulting images, specifically complementing the inductive bias of TITAN towards attenuating of Gibbs ringing artifacts typically observed when using SIREN. Surprisingly, the Bregman learning algorithm was not able to sparsify the weights of SIREN when initialized with the same sparsity factor as TITAN.

4.4. The effect of sparsity on Lipschitz constants

We investigate whether the smoothness induced by the sparsity yields an implicit model which has a lower Lipschitz constant than the non-sparse model. To do this, we focus on the super-resolution problem and generate a fine grid of 256×256 pixel locations and their corresponding model outputs. We follow this by calculating the largest singular value of the Jacobian of our implicit model at each pixel, computed via backpropagation. Finally, we take the largest of these values to obtain the Lipschitz constant of our INR.

Method	Metric	$m = 30$	40	50	100
SIREN	PSNR	28.5	29.9	31.3	33.2
	SSIM	0.81	0.86	0.90	0.95
TITAN	PSNR	29.9	31.5	**32.5**	**36.1**
	SSIM	0.89	0.91	0.94	0.97
DIP	PSNR	**30.5**	**31.8**	31.9	32.6
	SSIM	0.92	0.94	0.95	0.96

Table 1. Results showing that TITAN outperforms SIREN on computed tomography tasks with varying number of measurements m.

Our findings are somewhat counter-intuitive: the Lipschitz constant decreases as we increase the number of non-zero weights at the beginning of training as shown in Figure 4. This is especially counter-intuitive because lower Lipschitz constants are correlated with better generalization performance [24], and we see the opposite here. However, this happens because the largely smooth TITAN representation we see has sharp edges, which induces a large Lipschitz constant.

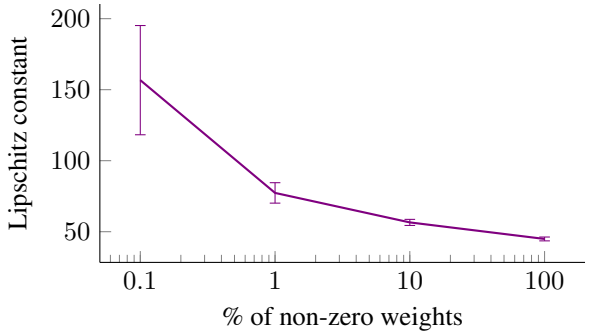

Fig. 4. As the number of non-zero weights of TITAN at initialization increases, the Lipschitz constant decreases; hence more sparse solutions tend to have larger Lipschitz constants since they are sharper. The sparsity of TITAN after training does not change much for a given initialization, and so we group them together and plot the average Lipschitz constant with its standard error for $n = 10$ random seeds.

5. CONCLUSION

We have demonstrated that it is possible to incorporate the inductive biases of the DIP into an INR, which we have implemented via residual connections in the place of the upsampling operator of a deep decoder. Complemented with sparsity-promoting optimization over weights, our proposed approach mitigates common perceptual artifacts when deploying INRs while maintaining a low parameter count. INRs with robust inductive biases will enable their deployment to solving hard imaging problems in resource-constrained settings.

6. REFERENCES

[1] Yinbo et al. Chen, "Learning continuous image representation with local implicit image function," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 8628–8638.

[2] Xingqian Xu et al., "UltraSR: Spatial encoding is a missing key for implicit image function-based arbitrary-scale super-resolution," *arXiv:2103.12716*, 2021.

[3] Zhiqin Chen and Hao Zhang, "Learning implicit fields for generative shape modeling," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[4] Lars et al. Mescheder, "Occupancy networks: Learning 3D reconstruction in function space," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[5] Jeong Joon et al. Park, "DeepSDF: Learning continuous signed distance functions for shape representation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019.

[6] Ben et al. Mildenhall, "NeRF: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[7] Ivan et al. Skorokhodov, "Adversarial generation of continuous images," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10753–10764.

[8] Tamar Rott et al. Shaham, "Spatially-adaptive pixelwise networks for fast image translation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 14882–14891.

[9] Emilien et al. Dupont, "Generative models as distributions of functions," in *International Conference on Artificial Intelligence and Statistics*. 2022, vol. 151, pp. 2989–3015, PMLR.

[10] Ivan et al. Anokhin, "Image generators with conditionally-independent pixel synthesis," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 14278–14287.

[11] Vincent et al. Sitzmann, "Implicit neural representations with periodic activation functions," in *Advances in Neural Information Processing Systems*, 2020, vol. 33, pp. 7462–7473.

[12] Vishwanath et al. Saragadam, "Wire: Wavelet implicit neural representations," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 18507–18516.

[13] K. Gong et al., "Learning personalized representation for inverse problems in medical imaging using deep neural network," *Physics in Medicine & Biology*, vol. 63, pp. 125011, 2018.

[14] Gizem et al. Yüce, "A structured dictionary perspective on implicit neural representations," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 19228–19238.

[15] Reinhard Heckel and Paul Hand, "Deep decoder: Concise image representations from untrained non-convolutional networks," in *International Conference on Learning Representations*, 2019.

[16] Victor et al. Lempitsky, "Deep image prior," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9446–9454.

[17] Olaf et al. Ronneberger, "U-Net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention*, 2015, pp. 234–241.

[18] Sergey Ioffe and Christian Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456.

[19] Diederik P. Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations*, 2015.

[20] Leon Bungert et al., "A Bregman learning framework for sparse neural networks," *Journal of Machine Learning Research*, vol. 23, no. 192, pp. 1–43, 2022.

[21] Vishwanath Saragadam et al., "MINER: Multiscale implicit neural representations," *arXiv:2202.03532*, 2020.

[22] Wotao et al. Yin, "Bregman iterative algorithms for ℓ_1-minimization with applications to compressed sensing," *SIAM Journal on Imaging Sciences*, vol. 1, no. 1, pp. 143–168, 2008.

[23] Yann et al. LeCun, "Optimal Brain Damage," in *Advances in Neural Information Processing Systems*, 1989, vol. 2.

[24] Sameera Ramasinghe and Simon Lucey, "Beyond periodicity: Towards a unifying framework for activations in coordinate-MLPs," *arXiv:2111.15135*, 2021.