TINC: Tree-structured Implicit Neural Compression

Runzhao Yang

Department of Automation, Tsinghua University

Beijing 100084, China

yangrz20@mails.tsinghua.edu.cn

Abstract

Implicit neural representation (INR) can describe the target scenes with high fidelity using a small number of parameters, and is emerging as a promising data compression technique. However, limited spectrum coverage is intrinsic to INR, and it is non-trivial to remove redundancy in diverse complex data effectively. Preliminary studies can only exploit either global or local correlation in the target data and thus of limited performance. In this paper, we propose a Tree-structured Implicit Neural Compression (TINC) to conduct compact representation for local regions and extract the shared features of these local representations in a hierarchical manner. Specifically, we use Multi-Layer Perceptrons (MLPs) to fit the partitioned local regions, and these MLPs are organized in tree structure to share parameters according to the spatial distance. The parameter sharing scheme not only ensures the continuity between adjacent regions, but also jointly removes the local and non-local redundancy. Extensive experiments show that TINC improves the compression fidelity of INR, and has shown impressive compression capabilities over commercial tools and other deep learning based methods. Besides, the approach is of high flexibility and can be tailored for different data and parameter settings. The source code can be found at https://github.com/RichealYoung/TINC.

1. Introduction

In the big data era, there exist massive and continuously growing amount of visual data from different fields, from surveillance, entertainment, to biology and medical diagnosis. The urgent need for efficient data storage, sharing and transmission all requires effective compression techniques. Although image and video compression have been studied for decades and there are a number of widely used commercial tools, the compression of medical and biological data

---

This work was done in collaboration with Tingxiong Xiao and Yuxiao Cheng, under supervision of Prof. Jinli Suo and Prof. Qionghai Dai affiliated with Department of Automation, Tsinghua Univ.

are not readily available.

Implicit neural representation (INR) is becoming widely-used for scene rendering [19, 21, 23, 31, 44], shape estimation [8, 11, 25, 29], and dynamics modeling [15, 26, 28]. Due to its powerful representation capability, INR can describe nature scenes at high fidelity with a much smaller number of parameters than raw discrete grid representation, and thus serves as a promising data compression technique.

In spite of the big potential, INR based compression is quite limited confronted with large sized data [20], since INR is intrinsically of limited spectrum coverage and cannot envelop the spectrum of the target data, as analyzed in [41]. Two pioneering works using INR for data compression, including NeRV [7] and SCI [41], have attempted to handle this issue in their respective ways. Specifically, NeRV introduces the convolution operation into INR, which can reduce the required number of parameters using the weight sharing mechanism. However, convolution is spatially invariant and thus limits NeRV's representation accuracy on complex data with spatial varying feature distribution. Differently, to obtain high fidelity on complex data, SCI adopts divide-and-conquer strategy and partitions the data into blocks within INR's concentrated spectrum envelop. This improves the local fidelity, but cannot remove non-local redundancies for higher compression ratio and tend to cause blocking artifacts.

To compress large and complex data with INR, in this paper we propose to build a tree-structured Multi-Layer Perceptrons (MLPs), which consists of a set of INRs to represent local regions in a compact manner and organizes them under a hierarchical architecture for parameter sharing and higher compression ratio. Specifically, we first draw on the idea of ensemble learning [12] and use a divide-and-conquer strategy [20, 41] to compress different regions with multiple MLPs separately. Then we incorporate these MLPs with a tree structure to extract their shared parameters hierarchically, following the rule that spatially closer regions are of higher similarity and share more parameters. Such a joint representation strategy can remove the redundancy both within each local region and among non-local

regions, and also ensures the continuity between adjacent regions. The scheme of the proposed network is illustrated in Fig. 1, and we name our approach TINC (Tree-structured Implicit Neural Compression).

Using the massive and diverse biomedical data, we conduct extensive experiments to validate that TINC greatly improves the capability of INR and even outperforms the commercial compression tools (H.264 and HEVC) under high compression ratios. The proposed TINC is also a general framework, which can be flexibly adapted to diverse data and varying settings.

2. Related Work

Data Compression. Data compression is a key technique for efficient data storage, transmission and sharing, especially in the big data era. In recent decades, many commercial compression tools for massive visual data have become increasingly mature, like JPEG [37], JPEG2000 [32], H.264 [39], and HEVC [33], while emerging deep learning-based approaches are also gaining momentum and demonstrating continuously improving compression capabilities [2, 3, 16, 22, 40, 42, 43]. Compression for nature images and videos have been widely and successfully used, however, there still lack effective compression techniques for the huge amount of three or higher dimensional data such as biomedical volume. Image/Video compressors cannot be applied directly, because these high dimensional data is a discrete sampling of 3D organism, instead of a video describing temporal evolution of 2D slices. Hence, the optical flow algorithms that are widely applicable in video compression do not work well on such inherently spatially structured data [41], and tend to cause discontinuity and degenerate the downstream tasks. Our approach differs completely from existing image/video compression methodologies in that we encode the intrinsic data structure with INR. Besides, our approach searches for the best network structure and parameters representing the target data and is free of generalization issues.

Implicit Neural Compression. Recent works [7, 17, 41] have introduced INR to the field of data compression with exciting results. They fit the original discrete grid-based data with continuous implicit functions in a transform-coding-like manner, which takes advantage of the powerful representation capabilities of neural networks while avoids the generalization problem. Obviously, the compression capability of INR is closely related to its representation capability, which can be improved from proper design of network structure [7, 10, 17], selection of activation function [14, 20, 30], embedding [4, 21, 23, 36, 45], and learning strategies such as manifold learning [9, 20, 30, 31], meta learning [5, 9, 35] and ensemble learning [1, 12, 24].

However, improving these capability does not necessarily head for a ready compressor for general scenes or ob-

jects, since there exist large diversity in the target data and pursuing a general network with high fidelity for all the data under given parameter budget. In order to pursue a more compact and efficient representation, we draw on the idea of ensemble learning [12] and use a divide-and-conquer strategy [20, 41] to compress the data in each region separately. On top of this, we improve INR by introducing a hierarchical parameter sharing mechanism in terms of data compression.

Parameter Sharing in INR. To improve the representation accuracy of INR for large and complex scenes, many algorithms use the parameter sharing strategy. One implementation is introducing convolution operations [6, 7, 27], which allows hierarchical extraction of local features, reduces the required parameters, and enables describing high resolution data. However, for those complex data with uneven distribution of features, the spatial invariance of convolution would lead to lower local representation accuracy. Another parameter sharing mechanism is positional encoding [21, 36], in which nearby spatial positions share partial mapping features hierarchically. However, this can only achieve higher representation accuracy when equipped with random or learnable mappings [4, 30, 36], which can be considered as an embedding that encodes only spatial distance. In addition, using embedding [18, 23, 34] intrinsically reduces the computational complexity at expenses of increased parameters, and thus not optimal for compression.

To make use of advantageous and bypass the above issues, we borrow the idea of hierarchical parameter sharing into local INRs, and build our compressor on a widely used tree structured data partitioning [18, 34, 41]. On the one hand, the parameter sharing is applied among block-specific INRs to cope with the large data with high local fidelity. On the other hand, we share parameters on INR rather than the data itself or its embedding, to encode richer information other than spatial distance.

3. Tree-structured Implicit Neural Compression

3.1. Ensemble of Implicit Neural Compressors

Implicit neural compression uses a compact neural network to parameterize a function describing the target data defined over a spatial coordinates system. More specifically, for an N dimensional data, let $\mathbb{V} := [-1, 1]^N$ denote its grid coordinates and $\mathbf{d}(\mathbf{v})$ with $\mathbf{v} \in \mathbb{V}$ denote the intensity value at coordinate $\mathbf{v}$, one can learn a compact implicit function f with parameter $\boldsymbol{\Theta}$ to estimate $\mathbf{d}(\mathbf{v})$ by $f(\mathbf{v}; \boldsymbol{\Theta})$ and finally serialize them to achieve a compressed representation. Mathematically, we can formulate the compression
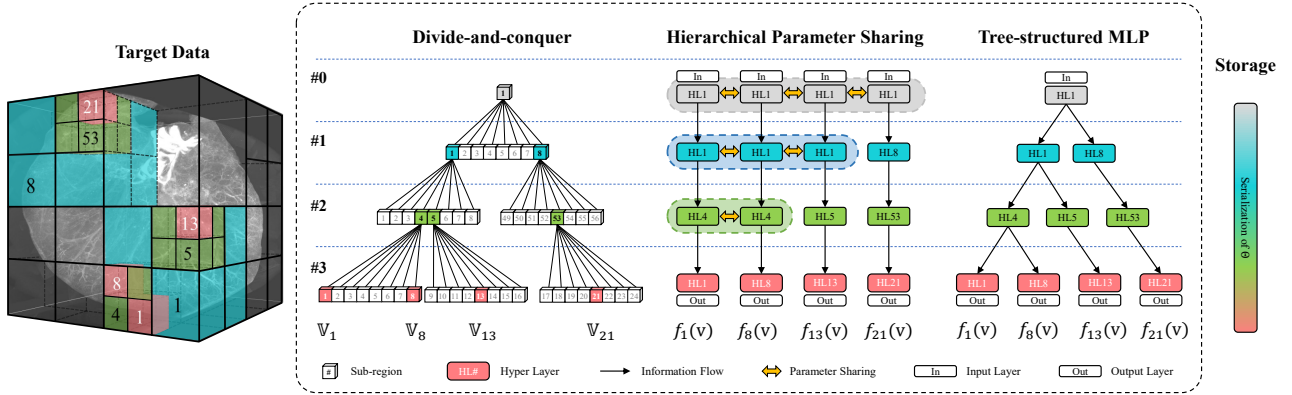
Figure 1. The scheme of the proposed approach TINC. For a target data, we divide the volume into equal-size blocks via octree partitioning, with some neighboring and far-apart blocks of similar appearances. Then each block can be representation with an implicit neural function, implemented as an MLP. After sharing parameters among similar blocks, we can achieve a more compact neural network with a tree shaped structure. Here we highlight the similar blocks sharing network parameters with the same color.

task as an optimization problem

$$\min_{\boldsymbol{\Theta}} \int_{\mathbf{v} \in \mathbb{V}} \mathcal{L}(f(\mathbf{v}; \boldsymbol{\Theta}), \mathbf{d}(\mathbf{v})), \tag{1}$$

where $\mathcal{L}$ measures the difference between the decompressed and target data. For decompression, one can retrieve f and $\boldsymbol{\Theta}$ from the compressed file and then calculat function values $f(\mathbf{v}; \boldsymbol{\Theta})$ over grid $\mathbb{V}$ via simple forward propagation.

Due to limited spectrum coverage, INR is incapable of compactly describe large and complex data [20,41], we borrow the idea of ensemble learning to partition the target volume into blocks and use multiple less expressive $f_k(\cdot, \boldsymbol{\Theta}_k)$ to achieve a powerful representation. Specifically, we adopt the divide-and-conquer strategy as in [12,20,41] to ensemble all implicit functions $\{f_k, k = 1, \cdots, K\}$ that represents data at its corresponding coordinate regions $\{\mathbb{V}_k, k = 1, \cdots, K\}$ and compose an f, which is equivalent to approximating a complex function by a piece-wise function composed of K simple functions

$$f(\mathbf{v}; \boldsymbol{\Theta}) := \sum_{k=1}^{K} \mathbf{1}_{\mathbb{V}_k}(\mathbf{v}, f_k(\mathbf{v}, \boldsymbol{\Theta}_k)), \tag{2}$$

$$\mathbf{1}_{\mathbb{V}_k}(\mathbf{v}, x) = \begin{cases} x, \mathbf{v} \in \mathbb{V}_k \\ 0, else \end{cases}. \tag{3}$$

In addition, we must constrain the total number of parameters to be consistent with the parameter budget, i.e., $|\boldsymbol{\Theta}| = \sum_{k=1}^{K} |\boldsymbol{\Theta}_k|$, where $|\cdot|$ is the support operation. Here we omit the parameters for serializing f_k which consists of only a few structural hyperparameters of neural network.

For a target data volume shown in the first column of Fig. 1, we use an L level tree structure containing K leaf nodes to organize f_k and $\mathbb{V}_k$, as illustrated in the 2nd column of Fig. 1. The root node represents the whole coordinate region $\mathbb{V}$ and each child node represents $\frac{1}{2^N}$ region of its parent node. We use f_k to compress the data in coordinate region $\mathbb{V}_k$ represented by the kth leaf node. This structure allows us to organize ensemble learning in high degree of flexibility.

3.2. Hierarchical Parameter Sharing Mechanism

We let these $\{f_k\}$ share their neural network parameters hierarchically with each other according to the spatial distance between corresponding regions $\mathbb{V}_k$. The sharing mechanism is defined on the octree structure. Specifically, for a leaf node at level l, its corresponding MLP-implemented f_k's hidden layers can be divided into l segments, i.e. $f_k = f_k^{out} \circ f_k^l \circ f_k^{l-1} \circ \cdots \circ f_k^1 \circ f_k^{in}$. For a set of leaf nodes, we determine the number of shared segments based on the number of their common ancestor nodes. For example, if f_i and f_j share the same ancestor nodes at $1 \sim 3$ levels, three pairs of hidden layer segments $(f_i^1, f_j^1), (f_i^2, f_j^2), (f_i^3, f_j^3)$ will share the same parameters.

The sharing mechanism is illustrated in the 3rd column in Fig. 1. Since the distance between two leaf nodes can also be inferred from the number of their shared ancestor nodes, above hierarchical parameter sharing mechanism implies that closer blocks share more parameters and are of higher similarity in the implicit representation space.

3.3. Tree-structured Network Architecture

We propose a tree-structured MLP based on the L level octree partitioning, as illustrated in the 4th column in Fig. 1. Each node contains a hyper layer consisting of some fully connected layers and takes the output of its parent node's hyper layer as input. Root node and leaf nodes additionally contain the input and output layers respectively.

Following the depth direction, from leaf nodes upward to the root of the tree, the output information of the leaf node is processed by the hyper layers in its ancestor nodes. If we consider the hyper layer as a part of hidden layers, then any leaf node can be considered as an MLP, whose hidden layers are explicitly divided into multiple segments contained in its ancestor nodes across different levels. In this way we can ensemble all $\{f_k\}$ into a tree-structured MLP.

At the same level, all sibling nodes share the same parent node and thus take the same information as input. Therefore, for any two leaf nodes sharing the same ancestor nodes, their output information have flown through the same hyper layers contained in the ancestor nodes. In other words, the MLPs of these two leaf nodes will share a part of hidden layers, which implements the hierarchical parameter sharing mechanism described in Sec. 3.2.

4. Cross Level Parameter Allocation

After the network design, we are left with some key implementation issues: (i) how to set a proper number of levels to finalize the tree structured network, (ii) how to allocate available parameters across MLPs at the tree nodes.

The Number of Tree Levels L. The number of tree levels L determines the number of divided segments of f_k's hidden layers, i.e., a larger L implies the parameter sharing mechanism proceeds to higher spatial resolution. Setting a large L has both advantageous and disadvantages. On the one hand, a larger L means finer partitioning with a larger number of blocks sharing parameters (with the same hyper layers as in the ancestor nodes), which is beneficial for describing the regions with rich details and thus improve the compression fidelity. On the other hand, with the same total number of parameters, a larger L will lead to more leaf nodes K and less parameters $|\Theta_k|$ each f_k, which might reduce f_k's representation capability and might harm the compression fidelity in turn.

Therefore, we should set the tree level L by trading off between increased fineness in partitioning and parameter allocation, and f_k decreased representation capability. In Sec. 5.3, we experimentally prove that it is favourable using large tree levels for the data with rich details or under low compression ratios.

Inter-level Parameter Allocation. The parameters at the shallow level describe the shared features at large scale, i.e., the non-local redundancy distributed over distant regions.

Similarly, from the perspective of data features, if some far apart sub-regions at a certain level are highly similar, one can allocate more parameters to their common ancestor nodes, and vice versa.

Therefore, for data with high global redundancy or repetitiveness, i.e., high non-local similarity among sub-regions at large scales, allocating more parameters to the shallow level will be more beneficial to improve its compression fidelity. Conversely, for data with weak global similarity, it is more efficient to allocate more parameters to the nodes at deep levels for better representation of the unique local features.

Intra-level Parameter Allocation. In addition to allocating parameters across levels, we can also allocate more parameters within a level to some nodes according to certain criteria instead of even allocation. For example, the most valuable information in neuronal data is often distributed in sparser regions. If we allocate more parameters to the more important regions, then we can sacrifice the unimportant regions in exchange for the improvement of compression fidelity of the important ones.

A simple way to take advantage of the TINC's hierarchy and flexibility is to allocate the number of parameters per node at each level according to the importance of the corresponding regions. Following this allocation, at each spatial resolution, the nodes corresponding to more important regions will have more parameters compared to other nodes in the same level. We demonstrate in Sec. 5.3 that this simple approach can significantly improve the compression fidelity of biological data, especially in regions with very low compression fidelity in the even allocation case, if an appropriate measure of the importance of the data region is adopted. Obviously there are more complex and efficient allocation methods, but this preliminary exploration is sufficient to demonstrate the advantages of TINC.

5. Experiments

5.1. Implementation Details

All the hyper layers in each node are implemented using MLP, where the activation functions are all sine function and the parameters are initialized following the scheme proposed in [30]. All MLPs are optimized by Adamax [13] with initiate learning rate of 0.001 and decreased by a factor of 0.2 at the 2000th and 5000th iterations, respectively. We adopted a complete octtree, which means $K = 8^{(L-1)}$. Before optimization, all the coordinates and the intensities are normalized to $[-1, 1]$ and $[0, 100]$ respectively. After optimization, all the parameters of each hyper layers are serialized and stored in binary. There are also some information that need to be saved for data decompression, such as MLP structure, original data bit depth, image size, inverse normalization basic information. Because all of the items are

Table 1. The scores on each dataset as compression ratio ranges from 64 to 1024, with the #best and the ●second best marked by color and symbols. The metrics include PSNR(dB), SSIM, and Acc.τ with τ acts as a threshold for binarization. The "All" or "High" suffixes after the metrics name indicate that it is averaged for all compression rations or only high compression ratios (around 1024×).

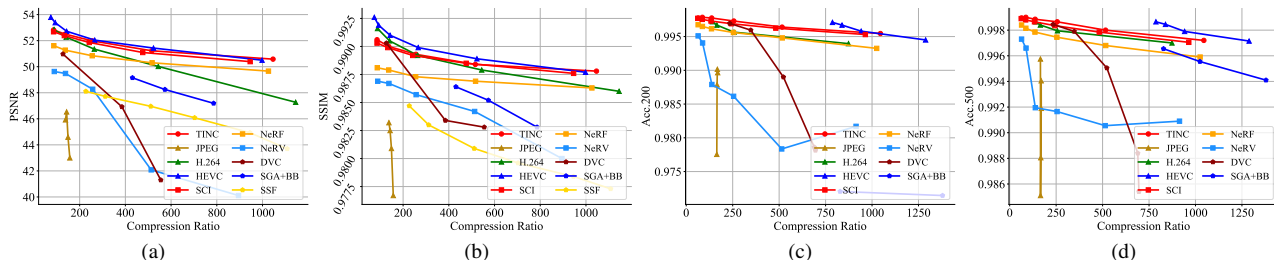| Method	Medical data				Biological data			
	PSNR All. (dB)	SSIM All.	PSNR High. (dB)	SSIM High.	Acc.200 All.	Acc.500 All.	Acc.200 High.	Acc.500 High.
TINC (ours)	●52.02	●0.9897	#50.59	#0.9878	●0.9945	●0.9970	#0.9934	#0.9958
JPEG	41.41	0.9722	30.49	0.9374	0.6612	0.9834	0.0197	0.9882
H.264	51.18	0.9896	47.28	0.9860	0.9919	0.9959	0.9860	0.9926
HEVC	#52.31	#0.9903	●50.51	●0.9877	#0.9955	#0.9975	0.9917	0.9930
SCI	51.90	0.9894	50.39	0.9876	0.9943	0.9965	●0.9921	●0.9951
NeRF	50.93	0.9875	49.66	0.9863	0.9935	0.9962	0.9903	0.9940
NeRV	47.11	0.9859	40.11	0.9800	0.9815	0.9901	0.9732	0.9867
DVC	47.39	0.9865	45.74	0.9840	0.9827	0.9900	0.9692	0.9789
SGA+BB	46.56	0.9836	43.02	0.9808	0.8038	0.9883	0.4817	0.9798
SSF	46.25	0.9807	43.70	0.9773	0.7221	0.9603	0.7790	0.9542

Figure 2. Comparison of different compression methods on medical (a,b) and biological data (c,d) at different compression ratios.

discrete, countable, and have a limited set of possible values, we can easily create a table listing all potential values and store their indices, to reduce the file size.

We evaluated our approach on both massive medical and biological data. For the former we used HiP-CT dataset [38], which provides cellular level imaging of several organisms at multiple anatomical levels. We used four human organs (*Lung*, *Heart*, *Kidney* and *Brain*) from this dataset. For the latter we used the sub-micrometer resolution mouse whole-brain microscopic data (*Neurons* and *Vessels*) captured by our self-developed imaging system (see supplementary note 1 for details), which has varying brightness and structures across different regions. The access to these data are listed in supplementary note 2.

For convenient and fair comparison, we cropped the data into the same size ($256 \times 256 \times 256$ voxels for medical data and $64 \times 512 \times 512$ voxels for biological data).

5.2. Performance Comparison with State-of-the-arts

Benchmark Methods and Evaluation Metrics. The proposed method is comprehensively compared with state-of-the-art methods, which can be classified into three groups: (i) commercial compression tools, including JPEG [37], H.264 [39], and HEVC [33]; (ii) recently proposed data-driven deep learning based ones, including DVC [16], SGA+BB [43], and SSF [2]; (iii) rcently proposed INR based ones, including SCI [41], NeRV [7], and NeRF [21]. We evaluated all the methods at 6 different compression ratios ranging from 64× to 1024×. It is worth noting that only INR based methods can precisely regulate the compression ratio. We adopted PSNR and SSIM for medical data including rich details, while Accuracy with threshold for binarization as evaluation metrics for biological data with sparse structures. For convenience we abbreviate Accuracy with threshold as "Acc.threshold" in the following descriptions.

JPEG was tested on OpenCV. H.264 and HEVC were tested with their FFmpeg implementations. For SGA+BB, we adopted the settings with the best performance mentioned in the original paper. For DVC, we used its Pytorch implementation, and compressed the "I Frames" using JPEG method with quality parameter set to 50. For SSF, we used its CompressAI's implementation. In addition, for the data-driven methods, i.e. DVC, SGA+BB, and SSF, we fine-tuned the provided pre-trained models on 300 volumes randomly selected from the dataset. For SCI, we set the hyper parameter $a_{max} = 32$. For NeRF, the hyper parameter "frequency" was set to 10. For NeRV, we control the compression ratio by adjusting the number and width of the fully connected layers. For 2D images compression methods, i.e. JPEG and SGA+BB, we firstly segmented the 3D data int 2D images before compression.

Quantitative Performance Evaluation and Analysis.
The average performance of each method under different compression ratios is plotted as a rate-distortion curve in Fig. 2. We can see that our algorithm outperforms almost all methods at high compression ratios and is comparable to HEVC at other compression ratios. For more detailed analysis, we summarize the scores on each dataset in Table 1. It should be noted that even if the PSNR of the medical data are higher than 40dB, it does not mean that there are no distortion in the decompressed data, which is different from the natural scene.

Some visual comparisons of decompressed organ slices are presented in Fig. 3 (see supplementary figure 1 for more results). For better visual comparison, we applied the same post-processing to the decompressed data. Under compression ratio around $87\times$(since HEVC and SGA+BB cannot precisely regulate compression ratio), we consistently outperform existing INR based (SCI and NeRV), data-driven (SGA+BB) and commercial (HEVC) compression methods. Although SCI achieved comparable results in the regions away from boundary, as shown in Fig. 3(a), it produced severe blocking artifacts at the boundary, as shown by the arrows in (b), which will affect the medical research as well as other downstream tasks. NeRV retained the general structures but suffered from the checkerboard artifacts caused by the aliasing in convolution. HEVC captured high fidelity details in "I Frames" but suffered from the artifacts in "P Frames", which distorted the edges in the volumes, such as the white matters in *Brain* data and muscles in *Heart* data, which would be unusable for medical research and other downstream tasks. SGA+BB struggled with retaining the rich details, since it cannot represent the similarity within the 3D volume as a 2D image compression algorithm. In addition, HEVC produced ghosting artifacts and position offset in neighbor slices, since the optical flow algorithms that are widely applicable in video compression do not work well on such inherently spatially structured biomedical data. In contrast, our approach use INR to encode the intrinsic data structure.

Comparison of Running Time. We summarized the time required for compression and decompression for each algorithm in our experiments, as shown in the supplementary tables 1 and 2. Comparatively, INR based compressor (TINC, SCI, NeRV, and NeRF) is slower in compression stage but can decompress faster. It is worth noting that biomedical data need to be compressed only once before saving or transmission, but decompressed frequently for processing and analysis. Therefore, TINC is advantageous in biomedical fields.

5.3. Flexibility Settings for Different Data

Setting of Tree Levels L. We evaluated TINC at different tree levels L on medical data under different com-

(a) Three ROIs from *Brain* data; compression ratio: ~87×

(b) Three ROIs from *Heart* data; compression ratio: ~87×

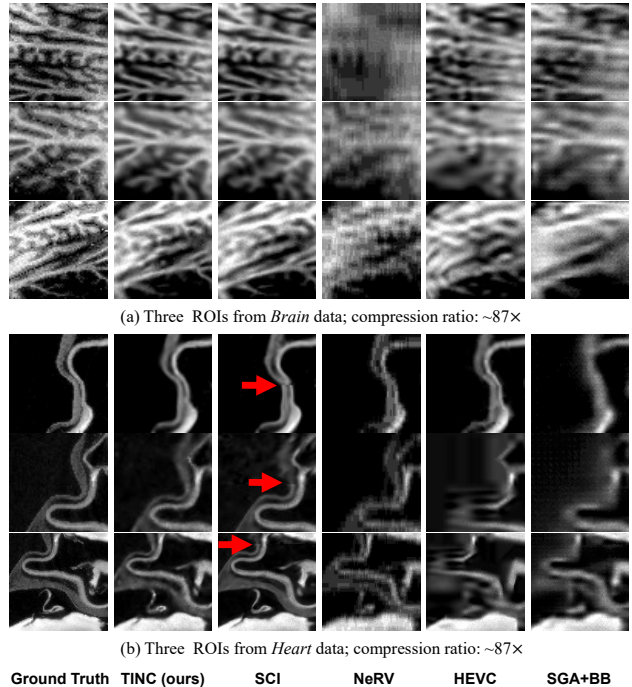Ground Truth   TINC (ours)   SCI   NeRV   HEVC   SGA+BB

Figure 3. Visual comparisons of decompressed organs slices from *Brain* and *Heart* data between TINC and other benchmark methods under similar compression ratio around $87\times$.

pression ratios. As shown in Fig. 4(a), the performance of TINC was evaluated in terms of SSIM on 8 medical data, under $256\times$ compression ratio. To better demonstrate the effect of increasing tree levels, we used the growth rate of SSIM, calculated by normalizing all SSIM values by the one when L=1. The SSIM scores on all the test data have a steadily increasing growth rate (dashed lines) as the number of tree depth increases from 1 to 3. Experimental results on other compression ratios show consistent trend (supplementary figure 2).

We further investigated the relationship between the data complexity and the performance gain when increasing tree levels. For data complexity, we adopt the spectrum concentration defined in [41], which measures the percentage of high frequency information contained in the data. The results are plotted in Fig. 4(b), with the solid line being a quadratic fitting of the growth rate for the 8 medical data using least squares. From the plot one can see that the performance gain increases monotonously with data complexity. In other words, the benefit of using a large tree is more significant when the target data contains more high-frequency information. This trend is consistent with our reasoning in Sec. 4 that a large L favours parameter sharing at coarser levels and leave more parameters for representing the details. Experimental results on other compression ratios and other data are consistent with this (supplementary figure 3).
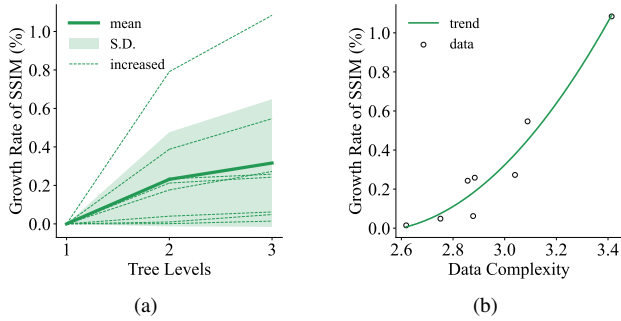
Figure 4. Effect of tree levels L on TINC's compression fidelity, in terms of SSIM on 8 medical data under $256\times$ compression ratio. (a) The growth rate of SSIM for each data when increasing the tree levels. The dashed lines represent the change in each data. The solid line represents the mean of changes, and the filled area represents the standard deviation. (b) The scatter plot of each data's complexity and the growth rate of SSIM when increasing tree levels from 1 to 3. The solid line represents the trend of change, estimated by a quadratic fit.

Figure 5. Effect of allocating more parameters to the shallow level on TINC's compression fidelity for each sub-regions in a *Brain* data under $512\times$ compression ratio. All sub-figures share the same y-axis label. (a) The heatmap of similarities between 8 equally spaced distant sub-regions. The serial numbers of the regions represent their z-curve order. (b) The non-local similarity of each region. (c) The growth rate of SSIM for each region when allocating more parameters to the shallow level.

In addition, we studied the performance gain of using a deep tree structure under limited bandwidth budget $|\Theta|$, e.g., at high compression ratio. Supplementary figure 4 plots the performance improvement at increasing compression ratios, calculated from 8 medical data volumes. Almost all data have a smaller growth rate of SSIM or PSNR when compression ratio increases, i.e. reducing the number of available parameters. At high compression ratio ($512\times$), some data are even of inferior fidelity using a deep tree, which is in line with our analysis in Sec. 4 that one should set a proper L for choosing the right partitioning granularity under the constraint from available budget.

Inter-level Parameter Allocation. We tested the performance at different parameter allocation proportions across levels on medical data under different compression ratios. Assuming there are K^l nodes at lth level, we denote the number of allocated parameters at this level as $|\Theta^l|$ and ratio with respect to that of its adjacent upper level as $r^l = \frac{|\Theta^l|/K^l}{|\Theta^{l-1}|/K^{l-1}}$. Simply speaking, $r^l=1$ means even inter-level allocation, a r^l larger than 1 means imposing higher priority to the shallow levels and vice versa. We let r^l of all levels take 0.8,1 and 1.2 to represent the three typical cases of parameter allocation.

We first investigated the effect of allocating more parameters to the shallow level, intending to share parameters among far apart similar blocks. As shown in Fig. 5(a), we divided a *Brain* data into 64 sub-regions, laid out in z-curve order, and selected 8 equally spaced distant regions. Then, we calculated the SSIM of these regions with respect to each other and used min-max normalization to obtain their non-local similarities. It can be seen that #1 and #33 regions are of high similarity. For convenient display, we also draw the
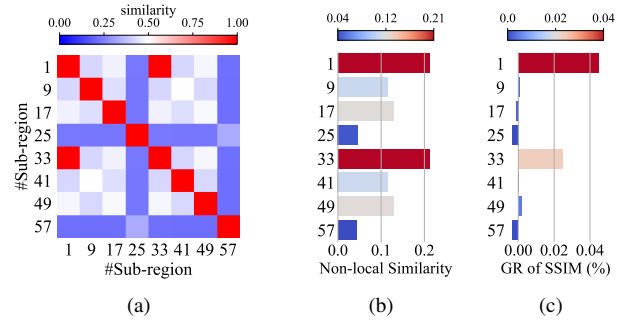
bar plot of the normalized marginal sums of the similarity matrix in Fig. 5(b), which tells that #1 and #33 are both of high similarity to other blocks, while #25 and #57 regions are highly distinctive from others.

To quantitatively measure the performance change at different blocks when allocating more parameters to the shallow level, we plot he growth rate of SSIM under $512\times$ ratio, as shown in Fig. 5(c). Analyzing jointly with (b), we can find that the two regions with high non-local similarity, #1 and #33, gain a large boost, while those with the lowest similarity, #25 and #57, suffer a loss of fidelity. This is consistent with the intuition and analysis in Sec. 4 that allocating more parameters to the nodes at shallow levels would benefit representing distant but similar regions, but at expense of slight degeneration at very individual regions. Experimental results on other compression ratios and other data are shown in supplementary figures 5 and 6.

We then investigated how to allocate parameters across levels according to the global consistency of the medical data. We divided the target data into 64 sub-regions, calculated the marginal sums of the similarity matrix. We change the setting of r^l and the results are shown in the supplementary figure 7. The results show allocating more parameters to the shallow level is more likely to improve fidelity when the global consistency of the data is greater than 0.7, and allocating more to the deep level is better when the global consistency lower than 0.6. Otherwise, an even allocation may be a better choice.

Intra-level Parameter Allocation. We tested the approach of allocating parameters to the nodes at the same level according to their importance proposed in Sec. 4, on biological data and under different compression ratios. Specifically, we let the number of allocated parameters be proportional to their importance. For example, with regard

to the biological data such as *Neurons* the importance can be measured by the percentage of the number of valid signals, where the valid signal can be determined either by the result of traced neurons or by the intensity value. Here we chose the latter for simplicity.

We first compare the compression fidelity of 6 *Neurons* data using the above two parameter allocation approaches under different compression ratios, as shown in supplementary figure 8. It can be seen that important allocation significantly improves the fidelity compared to even allocation.

We then look further into the performance at all the sub-regions in the same target data. As shown in Fig. 6(a), we divided a *Neurons* data into 64 sub-regions and compute their growth rate of Acc.2000 when switching from even allocation to important allocation. It should be noted that we omitted those regions that have reached 1 (the beast value) under both allocation approaches in the subsequent experiments. From the scatter diagram we can see three groups of data: (i) Some regions show non-significant change (less than 10% of the maximum change of the others, denoted as cross markers) using important allocation under $64\times$ compression ratio, because they are already with sufficient parameters. (iii) Other regions show significant improvement after introducing important allocation, denoted with circles. We performed a linear fit using least squares, as shown by the solid line. Overall, allocating parameters according to importance is helpful for the compression fidelity of important regions while at expenses of little fidelity degeneration in other less important ones.

For a statistical analysis, we plotted the Acc.2000 of all 64 sub-regions under even and important allocation, as shown in Fig. 6(b). We found that importance based allocation can significantly improve those regions that have very poor compression fidelity under even allocation. This is mainly because that those regions with poor fidelity tend to contain more complex structures (with more neural signals), and thus are of higher importance scores and thus allocated more parameters accordingly. Experimental results on other compression ratios are detailed in supplementary figure 9.

6. Conclusions

Summary and Discussions. In this paper we propose a hierarchical parameter sharing mechanism for INR and implement it based on a tree-structured MLP for large and complex data compression tasks. The proposed approach TINC makes use of the local redundancy via block wise compression with separate MLPs and non-local redundancy via organizing these MLPs in tree structure to share parameters hierarchically. Such implementation forms a single network for the complex target data, not only incorporate the similarity distributed far apart and nonuniformly, and avoid discontinuity between adjacent regions as well.
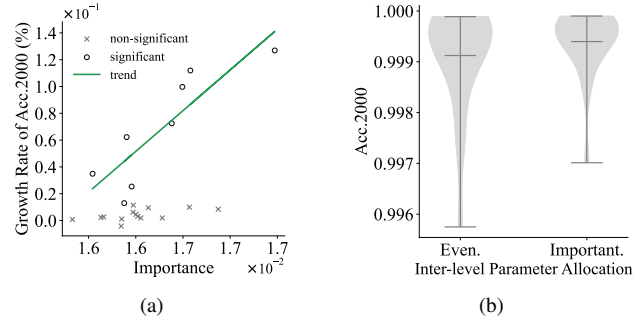

(a)   (b)

Figure 6. Effect of important allocation on TINC's compression fidelity, in terms of Acc.2000 on a *Neurons* data under $64\times$ compression ratio. (a) The growth rate of Acc.2000 for each region. The regions with and without significant growth rate are represented as cross and square respectively. The solid line represents the trend of change, estimated by a linear fit. (b) The violin plots of Acc.2000 for all these regions. The shaded area represent the distribution of Acc.2000 across all regions. The three horizontal lines from top to bottom represent the maximum, average and minimum values respectively.

Extensive experiments show that this hierarchical parameter sharing mechanism introduced in TINC greatly improves the compression capability and wide applicability of INR. Our method outperforms state-of-the-art compressors including commercial tools (JPEG, H.264, HEVC), data-driven methods (DVC, SSF, SGA+BB), and existing INR based ones (SCI, NeRV, and NeRF) on diverse medical and biological data.

We also analyze TINC's flexibility to different cases via experimentally studying the effect of three key settings—number of tree levels, intra-level and inter-level parameter allocation methods, and provide highlights on proper parameter setting and strategy selection according to the features of target data.

Limitations and Future Extensions. Similar to all current INR based compression methods, TINC is of high decompression speed but slow in compression, since it takes time to pursue the MLPs matching the target data. We plan to combine meta-learning to find the best initialization parameters for each organ to speed up TINC.

Acknowledgements

References

[1] Arya Aftab, Alireza Morsali, and Shahrokh Ghaemmaghami. Multi-Head Relu Implicit Neural Representation Networks. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 2510–2514, 2022. 2

[2] Eirikur Agustsson, David Minnen, Nick Johnston, Johannes Balle, Sung Jin Hwang, and George Toderici. Scale-space flow for end-to-end optimized video compression. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8503–8512, 2020. 2, 5

[3] Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational Image Compression with a Scale Hyperprior. In *International Conference on Learning Representations*, 2018. 2

[4] Nuri Benbarka, Timon Höfer, Hamd ul-moqeet Riaz, and Andreas Zell. Seeing Implicit Neural Representations as Fourier Series. In *2022 IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2283–2292, 2022. 2

[5] Alexander William Bergman, Petr Kellnhofer, and Gordon Wetzstein. Fast Training of Neural Lumigraph Representations using Meta Learning. In *Advances in Neural Information Processing Systems*, 2021. 2

[6] Alexandre Boulch and Renaud Marlet. Poco: Point Convolution for Surface Reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6302–6314, 2022. 2

[7] Hao Chen, Bo He, Hanyu Wang, Yixuan Ren, Ser Nam Lim, and Abhinav Shrivastava. NeRV: Neural Representations for Videos. *Advances in Neural Information Processing Systems*, 2021. 1, 2, 5

[8] Zhiqin Chen and Hao Zhang. Learning Implicit Fields for Generative Shape Modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5939–5948, 2019. 1

[9] Yilun Du, Katherine Collins, Josh Tenenbaum, and Vincent Sitzmann. Learning Signal-agnostic Manifolds of Neural Fields. *Advances in Neural Information Processing Systems*, 2021. 2

[10] Rizal Fathony, Anit Kumar Sahu, Devin Willmott, and J. Zico Kolter. Multiplicative Filter Networks. In *International Conference on Learning Representations*, 2021. 2

[11] Kyle Genova, Forrester Cole, Daniel Vlasic, Aaron Sarna, William T Freeman, and Thomas Funkhouser. Learning Shape Templates with Structured Implicit Functions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7154–7164, 2019. 1

[12] Milad Soltany Kadarvish, Hesam Mojtahedi, Hossein Entezari Zarch, Amirhossein Kazerouni, Alireza Morsali, Azra Abtahi, and Farokh Marvasti. Ensemble Neural Representation Networks. *arXiv preprint arXiv:2110.04124*, 2021. 1, 2, 3

[13] Diederik P Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. *arXiv preprint arXiv:1412.6980*, 2014. 4

[14] Sylwester Klocek, Łukasz Maziarka, Maciej Wołczyk, Jacek Tabor, Jakub Nowak, and Marek Śmieja. Hypernetwork Functional Image Representation. In *International Conference on Artificial Neural Networks*, pages 496–510, 2019. 2

[15] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural Scene Flow Fields for Space-time View Synthesis of Dynamic Scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6498–6508, 2021. 1

[16] Guo Lu, Wanli Ouyang, Dong Xu, Xiaoyun Zhang, Chunlei Cai, and Zhiyong Gao. DVC: An End-to-end Deep Video Compression Framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11006–11015, 2019. 2, 5

[17] Y. Lu, K. Jiang, J. A. Levine, and M. Berger. Compressive Neural Representations of Volumetric Scalar Fields. *Computer Graphics Forum*, 40(3):135–146, 2021. 2

[18] 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), jul 2021. 2

[19] Ricardo Martin-Brualla, Noha Radwan, Mehdi SM Sajjadi, Jonathan T Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7210–7219, 2021. 1

[20] Ishit Mehta, Michaël Gharbi, Connelly Barnes, Eli Shechtman, Ravi Ramamoorthi, and Manmohan Chandraker. Modulated Periodic Activations for Generalizable Local Functional Representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14214–14223, October 2021. 1, 2, 3

[21] 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. In *European Conference on Computer Vision*, pages 405–421, 2020. 1, 2, 5

[22] David Minnen, Johannes Ballé, and George D Toderici. Joint Autoregressive and Hierarchical Priors for Learned Image Compression. In *Advances in Neural Information Processing Systems*, volume 31, 2018. 2

[23] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant Neural Graphics Primitives with a Multiresolution Hash Encoding. *ACM Trans. Graph.*, 41(4), jul 2022. 1, 2

[24] Michael Niemeyer and Andreas Geiger. Giraffe: Representing Scenes as Compositional Generative Neural Feature Fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11453–11464, 2021. 2

[25] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning Continuous Signed Distance Functions for Shape Representation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 165–174, 2019. 1

[26] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo

Martin-Brualla. Nerfies: Deformable Neural Radiance Fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5865–5874, 2021. 1

[27] Songyou Peng, Michael Niemeyer, Lars Mescheder, Marc Pollefeys, and Andreas Geiger. Convolutional Occupancy Networks. In *European Conference on Computer Vision*, pages 523–540. Springer, 2020. 2

[28] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural Radiance Fields for Dynamic Scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10318–10327, 2021. 1

[29] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Angjoo Kanazawa, and Hao Li. Pifu: Pixel-aligned Implicit Function for High-resolution Clothed Human Digitization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2304–2314, 2019. 1

[30] Vincent Sitzmann, Julien N. P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein. Implicit Neural Representations with Periodic Activation Functions. In *Advances in Neural Information Processing Systems*, 2020. 2, 4

[31] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene Representation Networks: Continuous 3d-structure-aware Neural Scene Representations. *Advances in Neural Information Processing Systems*, 32, 2019. 1, 2

[32] Athanassios Skodras, Charilaos Christopoulos, and Touradj Ebrahimi. The JPEG 2000 Still Image Compression Standard. *IEEE Signal Processing Magazine*, 18(5):36–58, 2001. 2

[33] Gary J Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand. Overview of the High Efficiency Video Coding (HEVC) Standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(12):1649–1668, 2012. 2, 5

[34] Towaki Takikawa, Joey Litalien, Kangxue Yin, Karsten Kreis, Charles Loop, Derek Nowrouzezahrai, Alec Jacobson, Morgan McGuire, and Sanja Fidler. Neural Geometric Level of Detail: Real-time Rendering with Implicit 3D Shapes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11358–11367, 2021. 2

[35] 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*, pages 2846–2855, 2021. 2

[36] 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. In *Advances in Neural Information Processing Systems*, 2020. 2

[37] Gregory K Wallace. The JPEG Still Picture Compression Standard. *IEEE Transactions on Consumer Electronics*, 38(1):xviii–xxxiv, 1992. 2, 5

[38] CL Walsh, P Tafforeau, WL Wagner, DJ Jafree, A Bellier, C Werlein, MP Kühnel, E Boller, S Walker-Samuel, JL Rober-

tus, et al. Imaging Intact Human Organs with Local Resolution of Cellular Structures using Hierarchical Phase-Contrast Tomography. *Nature Methods*, 18(12):1532–1541, 2021. 5

[39] Thomas Wiegand, Gary J Sullivan, Gisle Bjontegaard, and Ajay Luthra. Overview of the H.264/AVC Video Coding Standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):560–576, 2003. 2, 5

[40] Ren Yang, Fabian Mentzer, Luc Van Gool, and Radu Timofte. Learning for Video Compression with Hierarchical Quality and Recurrent Enhancement. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6628–6637, 2020. 2

[41] Runzhao Yang, Tingxiong Xiao, Yuxiao Cheng, Qianni Cao, Jinyuan Qu, Jinli Suo, and Qionghai Dai. SCI: A Spectrum Concentrated Implicit Neural Compression for Biomedical Data. *arXiv preprint arXiv:2209.15180*, 2022. 1, 2, 3, 5, 6

[42] Ruihan Yang, Yibo Yang, Joseph Marino, and Stephan Mandt. Hierarchical Autoregressive Modeling for Neural Video Compression. In *ICLR*, 2021. 2

[43] Yibo Yang, Robert Bamler, and Stephan Mandt. Improving Inference for Neural Image Compression. In *Advances in Neural Information Processing Systems*, volume 33, pages 573–584, 2020. 2, 5

[44] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. Pixelnerf: Neural Radiance Fields from One or Few Images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4578–4587, 2021. 1

[45] Ellen D Zhong, Tristan Bepler, Joseph H Davis, and Bonnie Berger. Reconstructing Continuous Distributions of 3D Protein Structure from Cryo-EM Images. *arXiv preprint arXiv:1909.05215*, 2019. 2