Towards Generalising Neural Implicit Representations

Theo W. Costain[0000−0002−7803−6965]
Victor A. Prisacariu[0000−0002−0630−6129]

Active Vision Lab, University of Oxford
{costain,victor}@robots.ox.ac.uk

Abstract. Neural implicit representations have shown substantial improvements in efficiently storing 3D data, when compared to conventional formats. However, the focus of existing work has mainly been on storage and subsequent reconstruction. In this work, we show that training neural representations for reconstruction tasks alongside conventional tasks can produce more general encodings that admit equal quality reconstructions to single task training, whilst improving results on conventional tasks when compared to single task encodings. We reformulate the semantic segmentation task, creating a more representative task for implicit representation contexts, and through multi-task experiments on reconstruction, classification, and segmentation, show our approach learns feature rich encodings that admit equal performance for each task. Further, through hold-out experiments, we show that adding semantic supervision when training implicit encoders can significantly improve performance on later unseen tasks, without requiring encoder retraining.

1 Introduction

Implicit neural representations have garnered significant interest recently for their ability to reconstruct complex 3D structures and shapes. The appeal of these methods stems from a number of useful properties they possess for both reconstructing 3D shapes, as well as storing them efficiently. By learning to reconstruct the shapes, networks are able to encode and use a rich set of priors over the 3D domain to improve the quality of the reconstructions over what can be achieved with classical methods[24, 27]. Efficiency in storage is achieved by decoupling the encoding from the input and output modality so, unlike voxel based representation, the

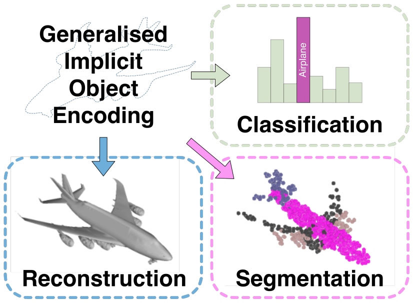

Fig. 1. Through multi-task training, implicit representations can be enriched creating a more general representation of a shape or object, and allowing for their use in a number of tasks rather than reconstruction alone.

storage requirements do not grow cubically with the output resolution. Further, implicit representations do not suffer from the limitations of mesh and point-cloud based representations, where the quality of the reconstruction is typically limited by the output size constraints of a single feed forward pass[24].

Conditioning a "decoder" network on an encoded representation of the input data, neural representations query the network at sample point locations for occupancy or distance function information. This approach allows for reconstructions to be generated with arbitrary resolutions at run-time[24].

Whilst these properties are impressive, we argue that further useful properties have been left on the table. In many of the works making use of implicit methods, training is performed with the loss function targeted only at reconstruction accuracy. This approach, whilst clearly effective, misses a significant potential benefit. We argue that using a multi task loss, including loss terms related to common tasks such as classification, produces encodings that are equally effective for reconstruction, but that still provide a richer set of features for use in other downstream tasks. We suggest that in applications such as augmented reality, where efficient representations are very useful, the ability to encode more than just shape information into the representation is likely to be useful.

Whilst a number of works have produced impressive and high quality reconstructions using a number of different approaches, there is a general aim in the design neural network encoders to produce features that have meaningful uses beyond a single task. However, we observe that this aim has not yet translated to implicit representation works. Many practical applications of implicit representations to real world problems would benefit from the ability to perform multiple tasks using the same stored data, rather than having to render and re-encode before performing other downstream tasks.

In this paper, we examine the generation of more descriptive neural representation encodings. Through experiments, we show that encodings generated purely for reconstruction can produce poor results on other tasks. We demonstrate the expected result that the encodings used in neural representations can be trained to develop properties useful for other tasks common in computer vision, without any appreciable reduction in reconstruction performance. We then detail surprising results in our hold-out experiments, showing that training encodings on even a single semantic task significantly improves performance on later held out tasks. We also argue that the conventional 3D semantic segmentation task does not translate well to implicit representations, where the object being reconstructed is not or cannot be operated on directly. To address this, we propose, among other experiments, a re-formulation of the semantic segmentation task that is a more representative formulation when applied to implicit representations.

In summary, the key contributions of this paper are

- Investigation of the use of implicit encodings on a number of common computer vision tasks (in a multi-task setting), showing improved performance in these tasks when compared to reconstruction-only encodings, without compromising reconstruction accuracy.

- Showing that the addition of a simple semantic task alongside reconstruction is able to substantially improve performance on new tasks, *without* requiring retraining of the encoder. For example, we show training an encoder for both reconstruction and classification significantly improves later segmentation results.
- A re-formulation of the semantic segmentation task, that is more representative of a real world task in the context of implicit representations.

2 Related Work

Implicit (or Neural) representations have been the subject of much recent work. Early works focused on single objects[24, 35, 32, 27, 25, 11, 2, 12, 30, 41, 5], encoding either image or point-cloud input into a feature vector, which is used to condition a decoder network. These decoder networks typically come in one of two forms: concatenation/biasing or hyper-networks. In concatenation based conditioning ([10] argue that biasing and concatenation are analogous), the encoding is concatenated with the point being queried and then passed through the network. In hyper-networks, the encodings are passed through a small network, whose outputs are the weights used in the network that predicts the value for a given query point. The outputs of these decoders can typically be divided into two categories, namely occupancy generating or signed distance function (SDF) generating[1].

Early works such as [35, 27, 24, 5] showed that simple MLP networks were capable of representing complex distance functions and occupancy functions. [27] (DeepSDF) also detailed the use of auto-decoders to estimate optimal encodings for a given input, using a fixed decoder and simple backpropagation. [24] (Occupancy Networks) demonstrated the alternative occupancy paradigm for implicit representations, as well as proposing a procedure to extract high quality meshes in an efficient manner from the implicit representation, using an octree like approach. Concurrently, [5] (IM-Net), also using the occupancy paradigm, showed impressive results for single view reconstruction, as well as both 2D and 3D shape generation. [25] learn level sets to represent shapes. Similarly, [2] (SAL), used unsigned function priors to train a signed distance function. [30] combine both explicit atlas based reconstruction and implicit neural reconstruction, enforcing consistency between the two methods.

Later works investigated representing larger scenes[3, 28]. Many of these methods did not expand the size of the area described by a given embedding, instead proposing methods to recover encodings for a small local region where

[1] Arguably occupancy networks are simply SDF networks with the sign function applied to their output, however this ignores the increased complexity in regressing SDF values rather than simply their sign. We discuss this point in more detail in Sec. 3.1

an implicit representation can then extract local shape information. [28] (Convolutional Occupancy Networks) interpolated between encoded points in a volume or plane, to generate the conditioning vector for a occupancy network in a region around the encoded point. [6] took a similar approach, adding also multiple resolutions of encoded volumes. A separate group of works[17, 3, 38] all took a slightly different approach from above and divided the scene into regions, generating a small encoding for each region. Both [17] and [3] made use of a grid of small local encodings, whereas [38] made use of a number of oriented spherical patches of differing radii each with an encoding.

Further improvements to implicit representations in general were proposed by [33] and [36] showing that adding higher frequency information to simple networks drastically improved their ability to generate high quality reconstructions. [9] proposed a curriculum based learning approach for implicit representations, improving reconstruction quality of complex local details.

Other works have used implicit representations for a number of other tasks, most notably novel view synthesis[21, 26, 34, 43].

There have also been a number of works considering the application of multi-task approaches to 3D problems[29, 19, 14, 20]. Multi-task learning has enabled improvements where tasks are related or closely coupled, such as semantic and instance segmentation[19, 29]. As well, [14] made use of a multi-task setup in unsupervised training to learn a useful embedding space for 3D point-cloud inputs.

Latent representations have been used to bridge the gap between point-cloud and volumetric representations[23], whilst others have sought to learn directly on meshes rather than point clouds or voxel grids[13, 16].

The concurrent work of [43] examines a similar use of implicit representations in a multi-view synthesis setting (based on [21]). Their work showed that the addition of geometric information can improve robustness in the semantic task, achieving strong performance in noisy environments while being able to achieve remarkable accuracy with little semantic supervision. [18] also examine semantics in a multi-view setting, using [35] for the internal representation. Their method also demonstrates the close relationship between appearance, geometry, and semantics, being able to synthesise novel appearance views from semantic images and vice versa.

3 Tasks

In this section, we first cover the principles of implicit representations. We then describe the other tasks we consider, including our variation to the normal task of segmentation that we use in our experiments, for a fairer representation of the segmentation task in implicit contexts.

3.1 Implicit Representation

Neural implicit representations attempt to estimate the function describing the surface of a given object. A common formulation is to map from a point in

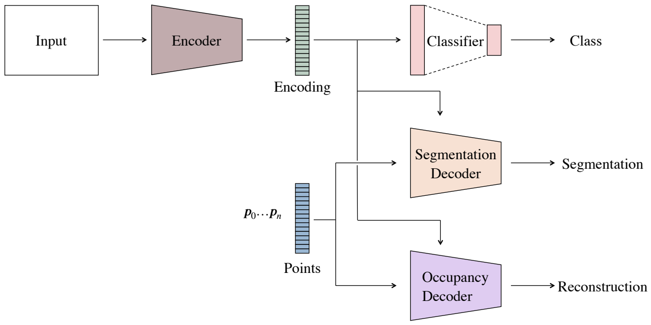

Fig. 2. An overview of our network architecture. The network takes as input either images or point-clouds, generating an encoding from them. This encoding can then be used in a number of ways. For classification, the encoding is passed directly into a simple classifier. For segmentation and reconstruction, the encoding is used to condition the decoder networks. The decoder networks take a number of points as input and returns for each point either, the probability that that point lies inside the encoded shape, or semantic label probabilities.

space, $\mathbf{p} \in \mathbb{R}^3$, to the smallest signed distance between the point and the outer face of a surface, *i.e.* a SDF. This gives rise to an expression[27, 41] of the form $s : \mathbb{R}^3 \rightarrow \mathbb{R}$. Another common formulation is to estimate the probability that a given point lies within the object (*i.e.* probability of occupancy), rather than regressing the SDF directly. This gives a function[24] of the form $o : \mathbb{R}^3 \rightarrow \{0, 1\}$. However, we note the following relationship

$$o = \sigma(-s) > \tau \tag{1}$$

where τ is a threshold parameter, and σ is a sigmoid function. This relationship suggests that the occupancy function is a simplified version of the SDF, moving the problem from a regression context and into a classification context. Further, this reformulation suggests that where the extra information provided by the SDF (but not the occupancy function) such as the surface normal (given as, $\frac{\partial s}{\partial \mathbf{p}}$, the spatial gradient of the SDF[27]) is not needed, the occupancy function is likely to be easier to learn. If this is true, we suggest this is the result of the occupancy network only needing to learn a decision boundary over $\mathbb{R}^3$ rather than having to learn both the boundary and then regress a points distance from it. In addition, we note that DeepSDF applies clamping to its loss function, such that points away from the surface do not impact the loss. Further, DeepSDF does not enforce the Eikonal constraint[2] like other works[25, 12]. Accordingly the network is mainly learning the zero crossing point of the SDF rather than a metric SDF.

This then implies that differences between Occupancy Networks and DeepSDF lie mostly in the training and design of the two networks, and not in the functions they embed. For this reason, we believe that although the evaluation in this

[2] A requirement for a "true" SDF

paper is conducted soley on Occupancy Networks, the results should also apply to DeepSDF based encodings. Given this, and the simplicity of the method, we chose to use the formulation of Occupancy Networks[24] for our experiments.

3.2 Other Tasks

The conventional 3D segmentation task as explored in a number of papers[31, 40] typically involves predicting a semantic label for each point in an given input point-cloud. However, in the context of implicit representations, this task loses much of its meaning, as we want to learn semantic information at locations not in the input pointcloud. This of particular concern when the input is a degraded and noisy point-cloud (Sec. 4). As we are considering the occupancy of a given spatial location, it makes more sense to consider the task as determining the semantic label of regions within the shape.

Hence, given a mesh $\mathcal{X}$, for each vertex $\mathbf{x} \in \mathcal{X}$ with a semantic label $c_{\mathbf{x}}$, the semantic class of any location $\mathbf{p} \in \mathbb{R}^3 \cap \mathcal{X}$ lying inside the mesh, has the semantic class of the nearest vertex of $\mathcal{X}$.

$$c_{\mathbf{p}} = c_{\mathbf{z}} \qquad \text{where } \mathbf{z} = \arg\min_{\mathbf{x}} ||\mathbf{p} - \mathbf{x}||_2$$

This scheme has the same effect as producing a Voronoi partitioning of the space inside the mesh. Points that lie outside the mesh, are considered to be background and therefore have no valid semantic class. The segmentation task then becomes predicting the label of a point inside the shape according to the nearest neighbour assignment. During both training and inference, we evaluate the semantic label task at the same locations as the reconstruction task.

As well as segmentation, we also investigate the performance of our approach in the task of classification. Unlike the segmentation task which requires the implicit code to encode information about the properties of spatial regions (similarly also with the reconstruction task), classification requires that the encodings allow simple classification networks to discriminate between them. Later experiments (Sec. 5.2 & Sec. 5.3), show that the requirements classification has for the encodings are noticeably different to reconstruction.

Our results show that implicit representations can be encouraged to be more representative of objects, rather than merely encoding their shape. We focus on two particular tasks that are common tasks, but expect that generalising the encodings over further tasks is likely to also be possible.

4 Experiments

The experiments are divided into three parts. First we consider the original dataset from [24], establishing a baseline and some preliminary experiments involving reconstruction and classification. Next we examine a more challenging classification task, before turning finally to a semantic segmentation task.

An overview of our network architecture is shown in Fig. 2. Specific details of the architecture for each experiment are outlined in Sec. 4.2. The loss functions used depends on the task. For the reconstruction loss, $\mathcal{L}_{\text{rec}}$, we use binary cross entropy as in[24]. Both classification, $\mathcal{L}_{\text{cls}}$, and segmentation, $\mathcal{L}_{\text{seg}}$, use the cross entropy loss. In the multi task settings, the losses are combined in a weighted linear fashion as $\mathcal{L}_{\text{tot}} = \mathcal{L}_{\text{rec}} + \lambda_{\text{cls}} \times \mathcal{L}_{\text{cls}} + \lambda_{\text{seg}} \times \mathcal{L}_{\text{seg}}$. For all experiments $\lambda_{\text{cls}} = \lambda_{\text{seg}} = 1.0$. We use an ADAM optimiser with learning rate of 10^{-4}. Training takes approximately 4 days on a NVIDIA GeForce GTX 1080Ti.

4.1 Datasets

We perform our experiments on a number of datasets. The original dataset (hereafter Choy) from [24] is the subset of ShapeNetCore[4] from [7]. We also make use of ModelNet40[39] for further classification experiments and ShapeNetPart[42] for our segmentation experiments. Data pre-processing was accelerated with GNU Parallel[37].

We limit our experiments to datasets with similar properties to those used in [24], as we are not seeking to validate the specific implicit representation format we are using, rather the benefits of more feature rich encodings. This means that we do not consider larger scale datasets such as Stanford3D[1] that our chosen method might struggle with. We leave this to future experiments with other methods such as [28] or [3] that are better able to reconstruct larger scenes.

For all experiments, the properties of the inputs remain constant. For point-clouds we sample 300 points from the ground truth point-cloud, and apply noise using a Gaussian distribution with zero mean and standard deviation 0.05 to the sampled point clouds, identically to [24]. For images we crop and resize the images identically to [24].

Choy / ShapeNetCore The dataset used in [24] from which our work builds on, uses the renderings and voxelisations[7] of a subset of the ShapeNetCore[4] dataset. We use the rendered images to train the image based encoder in later experiments. The fully processed dataset was provided by [24] as part of their publication. Briefly, meshes are loaded and a large number of depth images are rendered. These depth images are fused to form a watertight mesh from which points and their corresponding occupancy value can be sampled. Although the occupancy samples are not provided as part of the dataset in[7], to reduce ambiguity we will refer to the dataset from [24] as the Choy dataset throughout this paper. The dataset consists of 30,648 training meshes, 4,358 validation meshes and 3,738 test meshes across 13 object categories.

We use the Choy dataset both for our baseline experiments, as well as some preliminary classification experiments. Our experiments with this dataset are outlined in Sec. 5.1.

ModelNet40 For further classification experiments, we make use of the ModelNet40[39] dataset. As rendered images were not readily available, we rendered images using Pyrender[22] in the same fashion as [7], choosing 24 viewpoints with constant radius and altitude, but random azimuth. The occupancy samples are generated with the code provided by [24]. The dataset consists of 9,843 training meshes and 2,468 testing meshes across 40 object categories. Our experiments with this dataset are outlined in Sec. 5.2.

ShapeNetPart For our semantic segmentation experiments, we make use of the dataset from [42], which we refer to as ShapeNetPart. Again the occupancy samples were generated using the code from [24]. Semantic labels were assigned to the occupancy samples using the procedure outlined in Sec. 3.2 from the ground truth semantic labels in[42]. The dataset consists of 12,121 training, 1,854 validation, and 2,858 testing meshes following the corresponding splits from ShapeNetCore. Our experiments with this dataset are outlined in Sec. 5.3.

4.2 Architecture

Our network takes either point-clouds or images as input to the encoder. In all the experiments, we use the same two encoders: one for point-cloud input, and another for image input.

Point-cloud input We use the same variation on the original network from [31] as [24]. In this formulation, the fully connected (FC) layers normally present in the original network are replaced by residual FC blocks[15]. During training the network samples 300 points from the input point cloud and applies Gaussian noise ($\sigma = 0.005$) before passing these into the encoder (identically to [24]).

Image input We use a pre-trained ResNet-18[15], followed by a linear layer to reduce the output dimension following [24].

The encoded features are then passed to a decoder. For decoding point locations into either occupancy values or semantic labels we use one or more of the following, depending on the task(s). For classification, the encoding is passed directly to the classifier.

Task Decoders Occupancy Decoder This is the same decoder used in [24]. The network takes a number of points $\mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_n$ as input and uses conditional batchnorms[8], which take the encoding as their input, to condition the network.
Classifier A simple 2 layer MLP, that takes the encoding directly as input and returns class probabilities.
Segmentation Decoder The same network as the occupancy decoder but with a larger output channel dimension.
Parallel Segmentation and Occupancy Decoders This is the configuration shown in Fig. 2, in which the segmentation and occupancy decoders operate in parallel.
Joint Segmentation and Occupancy Decoder Also the same network as the occupancy decoder, however rather than two separate networks for each task,

	Recon.		Class.
	IOU ↑	Chamfer L1 ↓	Accuracy ↑
ONet baseline	0.78	0.0081	–
Classification baseline	–	–	0.92
Classification w/ ONet encoder	–	–	0.80
Joint Classification & ONet	0.77	0.0084	0.92

Table 2. Experiments on the Choy dataset with point-cloud input, showing shape IOU, Chamfer L1, and classification accuracy. For "Classification w/ ONet encoder" the encoder is pre-trained on the ONet baseline, fixed, and only the classification decoder trained.

the same network performs both tasks simultaneously. The output is then sliced along the channel dimension to yield two tensors, one containing the occupancy probability, the other containing the semantic label probabilities.

4.3 Encoder Baselines

To fairly compare later experiments to Occupancy Networks, we use the encoders from the original paper throughout. To provide comparisons between our tasks and conventional tasks on point clouds, we run two small experiments to show how the encoder (a modified PointNet) from Occupancy Networks performs compared to a baseline PointNet[31]. These experiments are shown in Table 1, using results directly from [31]. For classification on ModelNet40, the network receives 1024 input points perturbed by Gaussian noise with zero mean and 0.02 standard deviation, as in [31]. For segmentation on ShapeNetPart, the network receives 2048 points as input to the encoder, and the input is also used as the query points ($p_0 \dots p_n$ in Fig. 2). We note that for these experiments the network receives at least 3x more input points, with less additive noise, than in later experiments.

(a) Classification on ModelNet40

	Accuracy
PointNet[31]	**87.1**
ONet Encoder + Classifier	85.9

(b) Segmentation on ShapeNetPart

	mIOU
PointNet[31]	**83.7**
ONet Encoder + Segmentation Decoder	83.0

Table 1. Encoder comparison baselines.

5 Results

5.1 Choy Experiments

We begin with the dataset from [24]. Our experiments with point-cloud input are shown in Table 2. Given the small number of classes and fairly distinct visual properties of the classes in this dataset, the high accuracy in classification is not unexpected, even with the reduced quality of the input point-clouds. To evaluate the classification performance of the baseline encoder, we fix its parameters

	Recon.		Class.
	IOU ↑	Chamfer L1 ↓	Accuracy ↑
ONet baseline	(0.58 / 0.57)	(0.021 / 0.021)	–
Classification baseline	–	–	(0.92 / –)
Classification w/ ONet encoder	–	–	(– / 0.63)
Joint Classification & ONet	(0.59 / 0.57)	(0.020 / 0.023)	(0.92 / 0.91)

Table 3. Experiments on the Choy dataset with image input, showing shape IOU, Chamfer L1, and classification accuracy. Values in brackets: left with ResNet pre-training, right without.

	Recon.		Class.
	IOU ↑	Chamfer L1 ↓	Accuracy ↑
ONet baseline	0.73	0.011	–
Classification baseline	–	–	0.82
Classification w/ ONet encoder	–	–	0.57
Joint Classification & ONet	0.70	0.012	0.82

Table 4. Experiments on the ModelNet40 dataset with point-cloud input, showing shape IOU, Chamfer L1, and classification accuracy.

and train a simple 2 layer MLP classifier to operate on its output encodings. Classification on this fixed encoder shows a substantial reduction in accuracy compared to the jointly trained case, *i.e.* where the encoder is *not* fixed. Notably in this joint training scenario, full accuracy in both tasks is recovered.

Our experiments with image input are shown in Table 3. The results are similar to the point-cloud experiments. As discussed in [24], the lower performance in reconstruction for the ONet can potentially be attributed to occlusion. We also train the baseline from scratch without ResNet pre-training. Without pre-training the same "Classification w/ONet encoder" experiment shows a similar result to the point-cloud experiment where the network performs poorly on the classification task, as the network is not encouraged to generate features meaningful to other tasks. Further, without pre-training, the network was significantly less stable, and convergence was slower. The joint training result shows that the encoding is capable of performing both tasks without loss of accuracy.

5.2 ModelNet40 Experiments

To better evaluate the classification performance, as well as the shortcomings of the reconstruction encodings in classification, we run the same experiments as in Sec. 5.1 on ModelNet40, a more conventional 3D classification benchmark.

Our experiments with point-cloud input are shown in Table 4. The results follow a similar pattern to the point-cloud results from the Choy dataset. As we expected, when we train the classifier using the fixed encoder from the reconstruction task, the classification performance is poor. This reduction in performance is much more severe than on the Choy dataset, but is consistent with the

	Recon.		Class.
	IOU ↑	Chamfer L1 ↓	Accuracy ↑
ONet baseline	(0.54 / 0.49)	(0.034 / 0.038)	–
Classification baseline	–	–	(0.85 / –)
Classification w/ ONet encoder	–	–	(– / 0.51)
Joint Classification & ONet	(0.51 / 0.48)	(0.036 / 0.042)	(0.84 / 0.81)

Table 5. Experiments on the ModelNet40 dataset with image input, showing shape IOU, Chamfer L1, and classification accuracy. Values in brackets: left with ResNet pre-training, right without.

increased difficulty shown by the lower accuracy figure on the classification baseline. However, this performance loss is completely recovered in the joint training, with only a minor decrease in reconstruction performance.

Our experiments with image input are shown in in Table 5. Here we see that the joint training is able to recover much of the performance on either of the single tasks. Again, the experiments without pre-training show similar trends to the fixed encoder point-cloud experiments, where the classifier struggles with the ONet encoder, but full accuracy is recovered in joint training. We suspect the lower performance of the non-pre-trained networks can be attributed to the more varied nature of ModelNet40.

We also suggest that the reduction in performance for the joint task IOU in Tables 4 & 5 is more a result of specifics of the IOU metric than a meaningful reduction in performance. This can be seen from Table 3 where the Chamfer L1 loss is reduced by the same amount, but the IOU performance is less affected than in Tables 4 & 5.

5.3 ShapeNetPart Experiments

Our metric for the segmentation task is mean average Intersection over Union (mIOU). Points are sampled within the shape and assigned semantic labels by the decoder. The same sample points are used for both segmentation and reconstruction. Whilst in a real world scenario points would be sampled both inside and outside the shape, we wish to assess the performance of the segmentation decoder independently of the reconstruction performance, and so only consider points inside the shape. The IOU is computed for each part of the shape, and averaged to give a shape IOU. If there are no ground truth points for a given part (*e.g.* ‘armrest’ is a part of the chair class, but several chair instances do not have armrests), the part is automatically assigned an IOU of 1. We can then compute mIOU as the average of the shape IOUs. At inference time, points are sampled randomly from a padded bounding box of the ground truth object, as in [24].

We evaluate two different part segmentation decoders: the joint decoder where one decoder produces both segmentation and reconstruction at the same time, and the parallel decoder (the configuration shown in Fig. 2) where two

	Recon.		Seg.	Class.
	IOU ↑	Chamfer L1 ↓	mIOU ↑	Accuracy ↑
ONet baseline	0.69	0.010	–	–
Classification baseline	–	–	–	0.95
Segmentation baseline	–	–	0.53	–
Joint Segmentation & ONet	0.70	0.0098	0.50	–
Joint Segmentation & Classification & ONet	0.72	0.0086	0.50	0.95
Parallel Segmentation & ONet	0.68	0.011	0.53	–
Parallel Segmentation & Classification & ONet	0.70	0.0095	0.53	0.95

Table 6. Experiments on the ShapeNetPart dataset with pointcloud input, showing shape IOU, Chamfer L1, segmentation mIOU, and classification accuracy.

similar decoders handle segmentation and reconstruction respectively. A more detailed explanation is presented in Sec 4.2.

Table 6 shows the reconstruction accuracy, mIOU, and classification accuracy of our different experiments on the ShapeNetPart dataset. The results show little to no accuracy being lost in any of the tasks for the jointly trained settings. In the reconstruction task, the network is attempting to learn an encoding that represents the shape properties of a given region of space, such as the curvature and boundaries. These properties are likely also useful for the task of segmentation, *i.e.* the semantic class probabilities are potentially somewhat dependant on properties like local curvature.

Table 7 shows the per-class segmentation results for the baseline, joint training, and the reconstruction IOU for the baseline. These results show that the parallel architecture is superior to the joint architecture, and suggest that, although these particular tasks are well correlated at the encoding level, they may be less well corre-

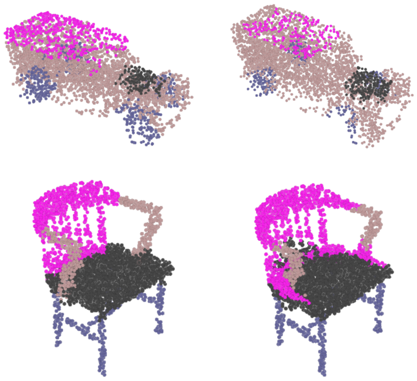

Fig. 3. Qualitative joint reconstruction and segmentation results on the ShapeNetPart[42] dataset, showing and example from the car (top) and chair (bottom) classes. Right shows a Joint segmentation & classification & ONet reconstructed mesh coloured with semantic class labels. Left shows the same reconstruction with ground truth labels.

Decoder Type	ONet	Seg.	Class.	Airplane	Bag	Cap	Car	Chair	Earphone	Guitar	Knife	Lamp	Laptop	Motorbike	Mug	Pistol	Rocket	Skateboard	Table	Mean
Reconstruction IOU	✓			*0.75*	*0.71*	*0.56*	*0.8*	*0.70*	*0.56*	*0.75*	*0.7*	*0.54*	*0.81*	*0.53*	*0.76*	*0.75*	*0.73*	*0.68*	*0.70*	*0.69*
Seg. Only		✓		0.59	0.46	0.45	0.52	0.63	0.31	0.68	0.55	0.52	0.59	0.53	0.58	0.59	0.39	0.51	0.57	0.53
Joint Decoder	✓	✓		0.55	0.45	0.40	0.51	0.62	0.30	0.66	0.53	0.49	0.59	0.47	0.53	0.56	0.30	0.47	0.56	0.50
	✓	✓	✓	0.55	0.42	0.38	0.49	0.62	0.32	0.66	0.54	0.49	0.59	0.45	0.55	0.56	0.3	0.46	0.56	0.50
Parallel Decoders	✓	✓		0.59	0.50	0.45	0.53	0.63	0.33	0.68	0.56	0.52	0.59	0.53	0.58	0.60	0.39	0.50	0.57	0.53
	✓	✓	✓	0.59	0.51	0.42	0.53	0.63	0.34	0.68	0.55	0.53	0.60	0.55	0.57	0.58	0.38	0.50	0.57	0.53

Table 7. Experiments on the ShapeNetPart dataset with point-cloud input, detailing per class results, showing segmentation mIOU. Note the first row is reconstruction IOU.

lated within the decoder. The poor performance on some of the classes such as rocket and headphones may be explained by the thin sections in parts of those objects. Because the network samples points within the shapes randomly, thin sections like the fins(rocket), cable(earphones), or handlebar(motorbike) are likely to be undersampled and therefore have poor performance at inference time. As well as this imbalance, there is also significant imbalance in the number of models in certain categories, which can negatively affect accuracy at inference time. This is reflected in the higher mIOU scores (across all the experiments), for the classes with more shapes.

Fig. 3 shows selected qualitative joint reconstruction & segmentation results.

5.4 Hold-out Experiments

Table 8 outlines our hold-out experiments, in which we aim to show the task generalisation of the network. In these experiments, we train the encoder and reconstruction decoder alongside either the segmentation decoder or the classifier. These weights are then frozen and the remaining decoder is trained.

As the above results show, an encoder trained for reconstruction alone learns features that encode shape information well. However the results suggest that shape information alone is not sufficient information for either segmentation or classification where significant accuracy is recovered for both tasks in the joint setting. Table 8 shows a particularly interesting result in that the joint training obtains similar results regardless of whether classification or segmentation was used alongside reconstruction to train the encoder. We suggest the following explanation for this behaviour: the reconstruction task teaches geometric information to the network, but not necessarily any semantic information. Both segmentation and classification are able to provide sufficient semantic information that, when combined with the geometric information, give sufficient capability for new tasks that require semantic and/or geometric information. To support this we point out that training for classification and reconstruction,

	Recon. IOU ↑	Seg. mIOU ↑	Class. Accuracy ↑
Classification w/ ONet encoder	–	–	0.89
Segmentation w/ ONet encoder	–	0.48	–
Multi Task w/ Seg & ONet encoder	0.70	0.53	0.95
Multi Task w/ Classification & ONet encoder	0.70	0.53	0.96

Table 8. Hold-out experiments on the ShapeNetPart dataset with point-cloud input, showing shape IOU, segmentation mIOU, and classification accuracy.

despite providing no further point specific information than reconstruction only, still improves the point specific segmentation task by 5%. Furthermore, training the encoder for segmentation and reconstruction, despite providing no explicit class information, allows the network to recover the full accuracy on the later classification task, giving a performance improvement of 6%. We note that, given this dataset is a fairly simple and quite biased as a classification task, an improvement of this amount suggests noticeably degraded performance (for the reconstruction-only encoder) on this task.

6 Conclusion

In this paper we have discussed generalising the encodings used by implicit representations to a broader range of tasks. We discuss the current narrow focus of implicit representations, and the potential issues this raises for applications of implicit representations in the real world. We introduced a modified formulation of the conventional segmentation task that is more applicable to implicit contexts, and detail an appropriate network to use for this new formulation. We show that as currently used, implicit encodings struggle to match the performance of the original data they aim to replace on common computer vision tasks, such as classification or part segmentation. We choose two common tasks and demonstrate that through multi-task training, we can enrich the encodings, achieving strong performance across the tasks without any loss in reconstruction accuracy. Through hold-out experiments, we showed improved performance on unseen tasks, when compared to single task training, without needing to retrain the encoder.

Acknowledgements

This work was supported by the Engineering and Physical Sciences Research Council [EP/R513295/1]

Bibliography

[1] Armeni, I., Sener, O., Zamir, A.R., Jiang, H., Brilakis, I., Fischer, M., Savarese, S.: 3d semantic parsing of large-scale indoor spaces. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1534–1543 (2016)

[2] Atzmon, M., Lipman, Y.: Sal: Sign agnostic learning of shapes from raw data. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 2565–2574 (2020)

[3] Chabra, R., Lenssen, J.E., Ilg, E., Schmidt, T., Straub, J., Lovegrove, S., Newcombe, R.: Deep local shapes: Learning local sdf priors for detailed 3d reconstruction. arXiv preprint arXiv:2003.10983 (2020)

[4] Chang, A.X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., Xiao, J., Yi, L., Yu, F.: ShapeNet: An Information-Rich 3D Model Repository. Tech. Rep. arXiv:1512.03012 [cs.GR], Stanford University — Princeton University — Toyota Technological Institute at Chicago (2015)

[5] Chen, Z., Zhang, H.: Learning implicit fields for generative shape modeling. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5939–5948 (2019)

[6] Chibane, J., Alldieck, T., Pons-Moll, G.: Implicit functions in feature space for 3d shape reconstruction and completion. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6970–6981 (2020)

[7] Choy, C.B., Xu, D., Gwak, J., Chen, K., Savarese, S.: 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In: European conference on computer vision. pp. 628–644. Springer (2016)

[8] De Vries, H., Strub, F., Mary, J., Larochelle, H., Pietquin, O., Courville, A.C.: Modulating early visual processing by language. In: Advances in Neural Information Processing Systems. pp. 6594–6604 (2017)

[9] Duan, Y., Zhu, H., Wang, H., Yi, L., Nevatia, R., Guibas, L.J.: Curriculum deepsdf. arXiv preprint arXiv:2003.08593 (2020)

[10] Dumoulin, V., Perez, E., Schucher, N., Strub, F., Vries, H.d., Courville, A., Bengio, Y.: Feature-wise transformations. Distill (2018). https://doi.org/10.23915/distill.00011, https://distill.pub/2018/feature-wise-transformations

[11] Genova, K., Cole, F., Vlasic, D., Sarna, A., Freeman, W.T., Funkhouser, T.: Learning shape templates with structured implicit functions. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 7154–7164 (2019)

[12] Gropp, A., Yariv, L., Haim, N., Atzmon, M., Lipman, Y.: Implicit geometric regularization for learning shapes. arXiv preprint arXiv:2002.10099 (2020)

[13] Hanocka, R., Hertz, A., Fish, N., Giryes, R., Fleishman, S., Cohen-Or, D.: Meshcnn: a network with an edge. ACM Transactions on Graphics (TOG) **38**(4), 1–12 (2019)

[14] Hassani, K., Haley, M.: Unsupervised multi-task feature learning on point clouds. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 8160–8171 (2019)

[15] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)

[16] Hu, S.M., Liu, Z.N., Guo, M.H., Cai, J.X., Huang, J., Mu, T.J., Martin, R.R.: Subdivision-based mesh convolution networks. arXiv preprint arXiv:2106.02285 (2021)

[17] Jiang, C., Sud, A., Makadia, A., Huang, J., Nießner, M., Funkhouser, T.: Local implicit grid representations for 3d scenes. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6001–6010 (2020)

[18] Kohli, A.P.S., Sitzmann, V., Wetzstein, G.: Semantic implicit neural scene representations with semi-supervised training. In: 2020 International Conference on 3D Vision (3DV). pp. 423–433. IEEE (2020)

[19] Lahoud, J., Ghanem, B., Pollefeys, M., Oswald, M.R.: 3d instance segmentation via multi-task metric learning. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 9256–9266 (2019)

[20] Liang, M., Yang, B., Chen, Y., Hu, R., Urtasun, R.: Multi-task multi-sensor fusion for 3d object detection. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 7345–7353 (2019)

[21] Martin-Brualla, R., Radwan, N., Sajjadi, M.S., Barron, J.T., Dosovitskiy, A., Duckworth, D.: Nerf in the wild: Neural radiance fields for unconstrained photo collections. arXiv preprint arXiv:2008.02268 (2020)

[22] Matl, M.: Pyrender (2020), https://github.com/mmatl/pyrender, (Version 0.1.43)

[23] Meng, H.Y., Gao, L., Lai, Y.K., Manocha, D.: Vv-net: Voxel vae net with group convolutions for point cloud segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 8500–8508 (2019)

[24] Mescheder, L., Oechsle, M., Niemeyer, M., Nowozin, S., Geiger, A.: Occupancy networks: Learning 3d reconstruction in function space. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4460–4470 (2019)

[25] Michalkiewicz, M., Pontes, J.K., Jack, D., Baktashmotlagh, M., Eriksson, A.: Deep level sets: Implicit surface representations for 3d shape inference. arXiv preprint arXiv:1901.06802 (2019)

[26] Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. arXiv preprint arXiv:2003.08934 (2020)

[27] Park, J.J., Florence, P., Straub, J., Newcombe, R., Lovegrove, S.: Deepsdf: Learning continuous signed distance functions for shape representation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 165–174 (2019)

[28] Peng, S., Niemeyer, M., Mescheder, L., Pollefeys, M., Geiger, A.: Convolutional occupancy networks. In: European Conference on Computer Vision (ECCV). Springer International Publishing, Cham (Aug 2020)

[29] Pham, Q.H., Nguyen, T., Hua, B.S., Roig, G., Yeung, S.K.: Jsis3d: joint semantic-instance segmentation of 3d point clouds with multi-task pointwise networks and multi-value conditional random fields. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 8827–8836 (2019)

[30] Poursaeed, O., Fisher, M., Aigerman, N., Kim, V.G.: Coupling explicit and implicit surface representations for generative 3d modeling. arXiv preprint arXiv:2007.10294 **2** (2020)

[31] Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 652–660 (2017)

[32] Sitzmann, V., Chan, E.R., Tucker, R., Snavely, N., Wetzstein, G.: Metasdf: Meta-learning signed distance functions (2020)

[33] Sitzmann, V., Martel, J.N.P., Bergman, A.W., Lindell, D.B., Wetzstein, G.: Implicit neural representations with periodic activation functions (2020)

[34] Sitzmann, V., Thies, J., Heide, F., Nießner, M., Wetzstein, G., Zollhofer, M.: Deepvoxels: Learning persistent 3d feature embeddings. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2437–2446 (2019)

[35] Sitzmann, V., Zollhöfer, M., Wetzstein, G.: Scene representation networks: Continuous 3d-structure-aware neural scene representations. In: Advances in Neural Information Processing Systems. pp. 1121–1132 (2019)

[36] Tancik, M., Srinivasan, P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J., Ng, R.: Fourier features let networks learn high frequency functions in low dimensional domains. Advances in Neural Information Processing Systems **33** (2020)

[37] Tange, O.: Gnu parallel - the command-line power tool. ;login: The USENIX Magazine **36**(1), 42–47 (Feb 2011). https://doi.org/http://dx.doi.org/10.5281/zenodo.16303, http://www.gnu.org/s/parallel

[38] Tretschk, E., Tewari, A., Golyanik, V., Zollhöfer, M., Stoll, C., Theobalt, C.: Patchnets: Patch-based generalizable deep implicit 3d shape representations. In: European Conference on Computer Vision. pp. 293–309. Springer (2020)

[39] Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., Xiao, J.: 3d shapenets: A deep representation for volumetric shapes. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1912–1920 (2015)

[40] Xie, Y., Tian, J., Zhu, X.: A review of point cloud semantic segmentation. IEEE Geoscience and Remote Sensing Magazine (GRSM) (2020)

[41] Xu, Q., Wang, W., Ceylan, D., Mech, R., Neumann, U.: Disn: Deep implicit surface network for high-quality single-view 3d reconstruction. In: Advances in Neural Information Processing Systems. pp. 492–502 (2019)

[42] Yi, L., Kim, V.G., Ceylan, D., Shen, I.C., Yan, M., Su, H., Lu, C., Huang, Q., Sheffer, A., Guibas, L.: A scalable active framework for region annotation in 3d shape collections. ACM Transactions on Graphics (ToG) **35**(6), 1–12 (2016)

[43] Zhi, S., Laidlow, T., Leutenegger, S., Davison, A.J.: In-place scene labelling and understanding with implicit scene representation. arXiv preprint arXiv:2103.15875 (2021)