ADCNet: Learning from Raw Radar Data via Distillation

Bo Yang[1] Ishan Khatri[1] Michael Happold[2] Chulong Chen[1]
[1]{bo.yang, ishan.khatri, chulong.chen}@motional.com [2]mhappold@startmail.com

Abstract

As autonomous vehicles and advanced driving assistance systems have entered wider deployment, there is an increased interest in building robust perception systems using radars. Radar-based systems are lower cost and more robust to adverse weather conditions than their LiDAR-based counterparts; however the point clouds produced are typically noisy and sparse by comparison. In order to combat these challenges, recent research has focused on consuming the raw radar data, instead of the final radar point cloud. We build on this line of work and demonstrate that by bringing elements of the signal processing pipeline into our network and then pre-training on the signal processing task, we are able to achieve state of the art detection performance on the RADIal dataset. Our method uses expensive offline signal processing algorithms to pseudo-label data and trains a network to distill this information into a fast convolutional backbone, which can then be fine-tuned for perception tasks. Extensive experiment results corroborate the effectiveness of the proposed techniques.

1. Introduction

Perception systems for autonomous vehicles have seen significant advancements over the past decade. This progression is of considerable importance, given that these systems serve as the primary source of information for downstream tasks, including those responsible for motion prediction and planning. Consequently, the ability to perceive the environment accurately is of utmost importance for the effective functioning of an autonomous vehicle.

Radars have long been applied in automotive applications such as blind spot monitoring and adaptive cruise control, however, their applications to semantic scene understanding have been limited. This is due to the fact that traditional automotive radars' resolution is usually not enough to produce high quality sensing output, rendering them insufficient for scene understanding tasks such as object detection.

In the past few years, imaging radars for autonomous driving have emerged. Compared to traditional automotive radars, imaging radars usually operate at around 77GHz, with roughly 4GHz available bandwidth – a much larger band than that of traditional automotive radars. In addition, these imaging radars are often equipped with multiple transmit and receiving antenna. Utilizing Multiple-Input-Multiple-Output (MIMO) technology [26], these new radars

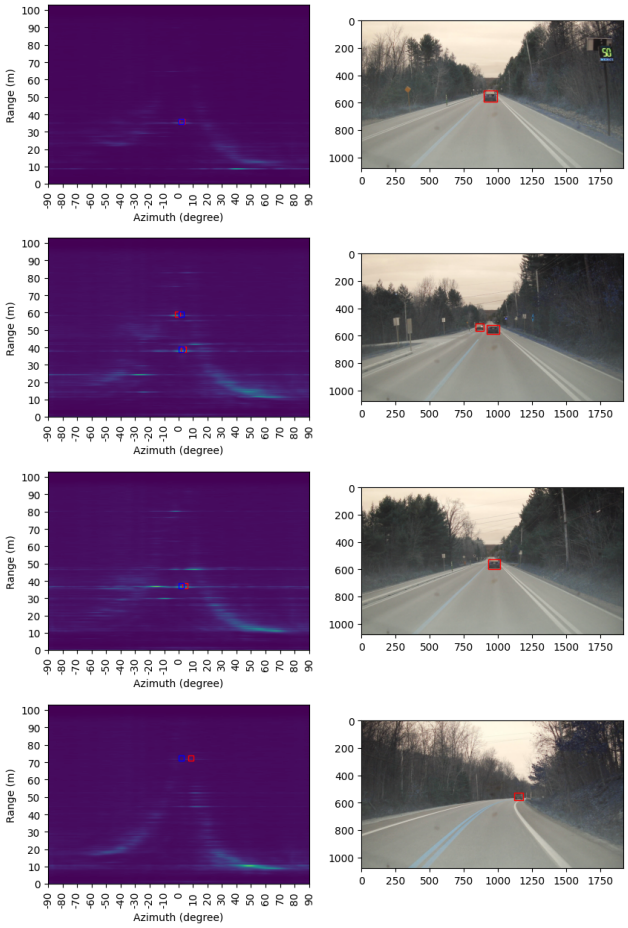

Figure 1. ADCNet prediction samples. The camera images are provided for reference only. The Range-Azimuth map is generated by the RADIal SDK. The small red squares in the RA image denote ground truth, while the blue squares denote network predictions

achieve much improved range and angle resolution.

Capitalizing on this improved sensing capability of imaging radar, several works [11, 15, 16, 19, 24, 25, 36] have attempted building perception models on these radars. One common theme of these works is that perception is done with low-level radar data, instead of the radar detection data as in traditional automotive radars. Low-level radar data refers to any radar sensor data that have not gone through the peak detection & thresholding step of the signal processing chain (see Figure 2). There are several low-level radar data representations: range-doppler (RD) radar cube, range-azimuth-doppler (RAD) cube, etc., that have gone through various stages of signal processing, but not the final peak detection step that leads to very sparse radar detections.

However, as described in [27], using the RAD tensor can be costly for imaging radars, as advanced angle finding operations are difficult to implement in real-time systems. As a result, FFTRadNet [27] uses the RD cubes, which contains no explicit angular information - a necessary component for 3D perception. This limitation motivates our pre-training via distillation pipeline. To equip the neural network with angle finding capability, we pre-train the network on the task of mimicking the full signal processing chain (from ADC to RAD) via a distillation pipeline. Our method employs offline signal processing algorithms to generate the RAD cubes for each ADC sample as pseudo-labels. Importantly, using these pseudo-labels enables training the model on a large set of data as no human annotation is needed. Training a neural network with these pseudo-labels distills these complex algorithms into the weights of the neural network, which can be run efficiently on GPU hardware.

In order to facilitate using the ADC to RAD conversion task as a pre-training step, we introduce a learnable signal processing module to the network. This module aims to mimic the first two steps of classic signal-processing pipelines. Our design jump-starts the training: since the raw ADC data is inherently intricate, learning from a generic model architecture can be immensely difficult, as shown in Sec. 6.5.

In summary, we propose a framework which features 1) a pre-training pipeline that infuses angle finding capability into neural networks and 2) a learnable signal processing module that is tailored to radar signal. The proposed method is performant as it achieves SOTA performance on the RADIal dataset, while also being flexible in accommodating different backbones (e.g. to achieve better latency) or different offline signal processing algorithms as teachers.

2. Background on radar signal processing

This work is mainly concerned with a type of automotive radar termed Frequency Modulated Continuous Wave (FMCW) radar, see *e.g*. [2]. At each measuring cycle, these radars send out a series of rapid "chirps" – short waves with increasing frequency. At the receiving antenna, the reflections are captured, and sampled by an ADC device. This digitized signal is then passed to other software modules in the radar sensor for signal processing. A typical radar signal processing chain is shown in Figure 2.

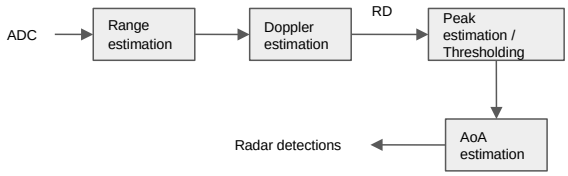

Figure 2. A simplified radar signal processing chain

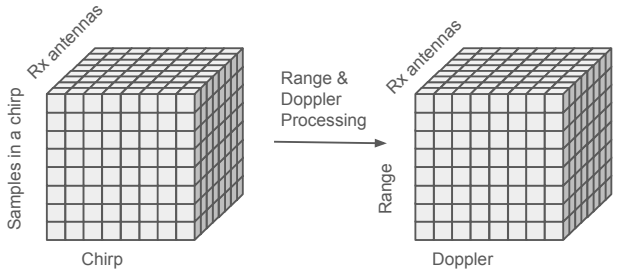

Figure 3. Illustration of ADC and RD arrays. Left: ADC, Right: RD. Notice the different dimensions between the two.

The ADC data is usually organized as arrays as shown in Figure 3. Each frontal slab has dimension (number of samples per chirp, number of chirps), and holds all the samples for one receiving antenna. Signal processing operations (mainly DFTs) transform these two dimensions into range and Doppler dimensions, forming the RD array, as shown in the right part of Figure 3.

The next step of the SP chain is typically to threshold the RD cube in order to reduce the number of range-doppler bins that must be processed and then to run an angle-of-arrival (AOA) estimation algorithm. A rudimentary estimate of AOA can simply be found by adding a 3rd DFT along the final axis of the tensor, however in order to achieve higher quality estimates, iterative optimization algorithms such as IAA [28], MUSIC [33] and ESPRIT [29] can be used. While these algorithms have many advantages for automotive use (such as strong robustness [32]), they are difficult to run in real-time and require knowledge of the radar's parameters in order to implement [13].

3. Related works

Radars for AV perception. There has been a considerable amount of work utilizing radar detections for AV perception [20, 30, 37]. In these papers, the radar detections are

usually fused with another sensor modality, such as camera and LiDAR. The main reason is that radar detections are very sparse, and reliable semantic understanding of the driving scene is hard to achieve with radar alone. In [37], the authors propose a fusion scheme that combines radar detections and LiDAR point clouds, and obtain improved performance for object detection and velocity estimation. In [30], the authors combine radar detections and LiDAR point clouds and achieve improved performance for trajectory prediction, compared to using LiDAR alone.

There is a growing body of work attacking the AV perception problem by means of low-level radar data [11, 15, 16, 19, 24, 25, 36]. CramNet [11] proposes a method for fusing radar Range-Azimuth (RA) images with camera images using the attention mechanism. In [19], a graph convolution network is developed to work with radar RA data. In [25], a radar-LiDAR fusion method is developed to improve perception robustness against adverse weather. In [16], the authors propose a method to cope with the missing modality problem for a radar-LiDAR fusion system. EchoFusion [18] proposes a low-level radar and camera fusion scheme, where an attention mechanism using the polar coordinates is designed.

It is also worth highlighting the low-level radar datasets that have been facilitating the research in this direction. These include RADIATE [31], RADIal [27], ORR [1], CRUW [35], while new ones (*e.g.* K-Radar[23]) continue to emerge. Different datasets often use different radar sensors, so their radar data representations tend to be different. For example, RADIATE provides radar radio-frequency (RF) images, ORR provides radar intensity maps, CRUW provies RA maps, while RADIal provides both ADC and Range-Doppler (RD) data. Since we are interested in exploring ADC data, we adopt the RADIal dataset to conduct our experiments, as among the currently ready-to-use radar datasets, RADIal is the only one that provides radar ADC data.

Pre-training Pre-training has been proven quite successful in several important domains, such as computer vision [4, 9, 22], natural language understanding [3, 6], speech and audio processing [5, 14, 39], etc. These methods usually entail designing a surrogate learning task, such as predicting masked part of the input signal, so that the neural network learns intrinsic structure of the data. While deceptively simplistic, many works in this genre have shown that powerful representations, that support impressive performance on downstream tasks (such as image classification, question answering with natural language, automatic speech recognition), can be learned from a large collection of data.

Distillation Distillation and pseudo-labeling techniques are an important way to scale machine learning methods in regimes where human labels are too costly or too difficult to obtain. In MODEST [38] a pipeline was created where LiDAR detection models were successively trained first with heuristic generated seed labels, and then on pseudo-labels generated by the previous iteration of the model. Distil-whisper [7] creates a high quality pseudo-labeled dataset for speech recognition. Finally, in ZeroFlow [34] the authors propose a similar offline optimization based pseudo-labeling technique for scene flow estimation, demonstrating the power of distilling offline techniques into small models by using larger unlabeled datasets.

Learnable signal processing (SP) module The learnable signal processing module idea for radar has been pursued in [40] for the hand gesture recognition task. Compared to [40], our method uses *perturbed* discrete Fourier transform (DFT), while [40] uses only exact DFT for initialization. Perturbation is an important novelty of this work, see discussions in Section 5.2 and experiment results in Table 5.

4. The RADIal dataset

To assist discussion, we provide a brief introduction to the RADIal [27] dataset, and readers are referred to the original publication for more details.

The RADIal dataset provides synchronized radar, LiDAR and camera images for about 2 hours of driving. Labels for vehicle detection and freespace segmentation are also provided. In this work, we are interested in the radar data. For this, the dataset provides ADC data and RD data. The RADIal SDK [1] also enables generating RAD maps from the ADC data.

The radar sensor used in the RADIal dataset is an imaging radar with 12 transmit antennas and 16 receiving antennas. The radar is mounted at the front of the vehicle, and has an approximate field-of-view (FOV) of 180 degrees, and a range of 103 meters.

The RADIal dataset contains a relatively balanced distribution of driving scenes in city, country side, and highway. For the object detection task and the freespace segmentation task, the labels are generated by models running on images and LiDAR, followed by human verification. Only vehicles are labeled for object detection.

Of the 25,000 time synchronized data frames, only about 8300 frames have object and segmentation labels. In Section 5.1 we will propose a method to utilize the unlabeled dataset for pre-training.

5. The ADCNet Framework

In the traditional radar signal processing chain, raw ADC data is converted into a Range Doppler cube via two DFT operations along two dimensions. The RD cube is then thresholded and run through an angle-of-arrival layer to estimate the azimuth of the targets. As explained in [27], this

[1] https://github.com/valeoai/RADIal

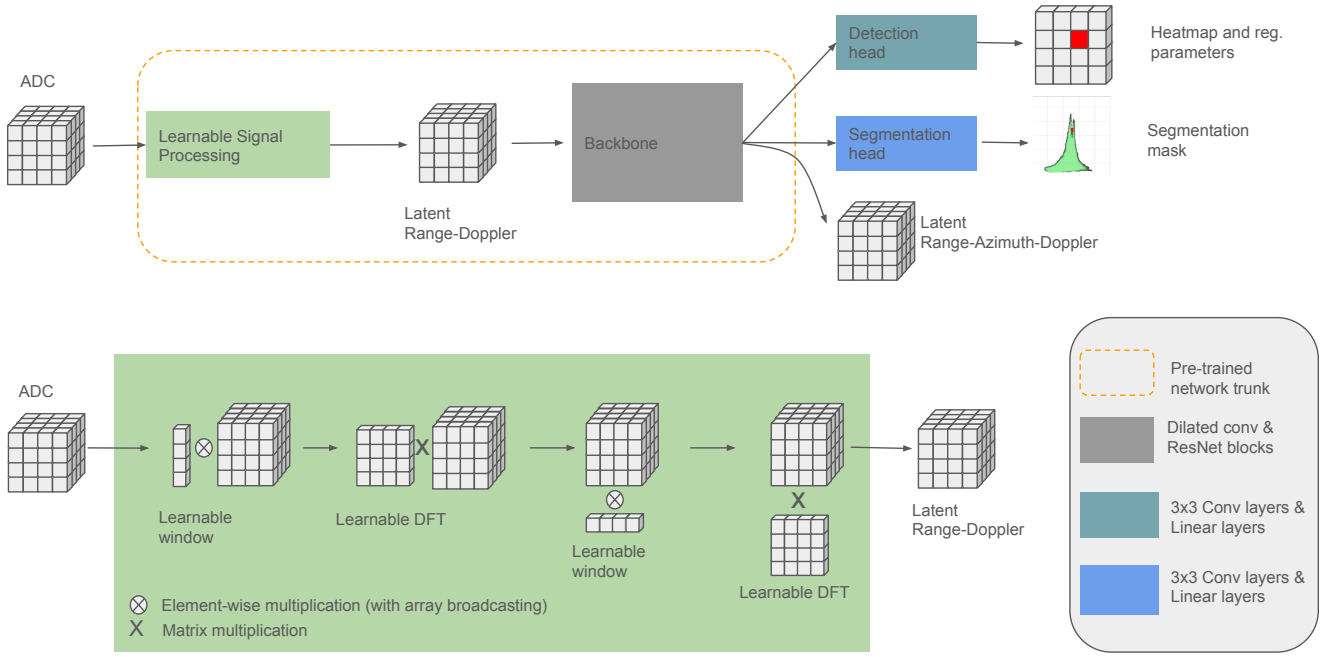

Figure 4. The ADCNet framework. A learnable signal processing module, as shown in the dashed box, is introduced at the start the network to enable end-to-end learning. The dashed yellow box represents the portion of the network that is pre-trained using our Distillation method described in Section 5.1

angle-finding step is typically the most computationally expensive process of the signal processing chain. The network used in [27], FFTRadNet uses RD cubes as input and then implicitly estimates the azimuth of the targets using the supervisory signal from the detection and segmenatation heads. In our work, we aim to bring the full signal-processing chain into the network and start from the raw ADC data. The goal is to design a framework for using raw ADC data and pre-training a network backbone that could be used for any number of downstream tasks (such as detection or segmentation).

Our framework is illustrated in Figure 4. Starting with the raw ADC data we run a learnable signal processing layer and then a backbone network. This network is trained to reconstruct the full RAD tensor that the classical signal processing pipeline would produce, using the SP pipeline as a teacher which provides ground-truth. In effect, we distil the full SP pipeline (which includes DFT, thresholding and AOA estimation) into the learnable SP module and the backbone. Once our backbone has been pre-trained we can fine tune it for downstream tasks such as detection and segmentation. Importantly, our framework is generalizable and can be applied to any sensor, signal processing chain, and backbone network architecture.

5.1. Pre-training ADC to RAD via Distillation

The pre-training via distilation step of our framework is our most important contribution in this work. This pre-training can be roughly broken down into two steps.

1. Using an offline SP algorithm, generate RAD from all the ADC data;
2. Using the collected (ADC, RAD) pairs as input and (pseudo) labels to train the learnable SP and backbone network in a supervised learning fashion.

Using the ADC as input, our learnable SP module + backbone network predicts RAD tensors which are supervised using the loss function described below:

$$\mathcal{L} = \text{smooth-L1}(\mathbf{Y}_{\text{RAD}} - \hat{\mathbf{Y}}_{\text{RAD}}), \tag{1}$$

where $\mathbf{Y}_{\text{RAD}}$ denotes the generated RAD tensor and $\hat{\mathbf{Y}}_{\text{RAD}}$ denotes the prediction of the neural network. From the RADIal SDK, the generated RAD tensor is of shape $512 \times 751 \times 256$. We downsample the tensor to have shape $128 \times 248 \times 256$ for easier training. After pre-training the trunk of the ADCNet, we proceed to fine-tune the network with the multi-task setting as described in Section 5.3.

While conceptually simple, this step has a number of key advantages. The first of which is that it explicitly trains the network on the task of azimuth estimation which directly leads to an improvement in detection and segmentation performance (see Section 6.4).

The second important advantage is that this step is a low cost addition to any existing pipeline. The data required does not need any human annotation which is a key advantage for raw radar datasets. This is because raw radar data is not human interpretable and thus requires sensor fusion techniques for accurate 3D labels, as shown in [23].

In this work we use the signal processing pipeline that is provided by the RADIal SDK to work with the specific radar sensor used to collect the RADIal dataset. The SDK provides a matched-filer type of algorithm for angle estimation. Clearly, more advanced algorithms (such as IAA, MUSIC etc.) could be used to generate the RAD with greater accuracy. However we are unable to try these alternative teacher algorithms for comparison due to lack of key radar configuration parameters (e.g. antenna array placement, multiplexing schemes, etc.). Investigation for better teacher algorithms is thus left as a future work.

5.2. Learnable Signal Processing Layers

Since the DFT operations and windowing operations are relatively cheap, we can easily incorporate them into our network. This is similar to the method used in [40], where the DFT is incorporated as a learnable module in the network, however unlike [40], we also add the windowing function and perform a perturbed initialization described below. Both the DFT and windowing operations can be represented by matrix multiplications and we can simply set up linear layers which are initialized to the correct parameters in the network (see the Appendix for additional implementation details). However, we find that in practice initializing the SP module to use the exact DFT parameters can hurt performance, since the training process may fail to drive the weights away from the local minima to which they are initialized. To alleviate the issue, we developed a novel perturbed-DFT initialization strategy:

$$\mathbf{M} = \mathbf{M}_{\text{DFT}} + \mathcal{N}(0, \gamma) \tag{2}$$

In Eq. 2, $\mathbf{M}_{\text{DFT}}$ denotes the DFT matrix, and $\mathcal{N}(0, \gamma)$ denotes a random Gaussion matrix (with the same shape as the DFT matrix), with i.i.d. elements each has mean 0, and variance γ. Intuitively, this γ parameter should be small, such that the final matrix $\mathbf{M}$ used for initializing the SP module is perturbed, but still resemble the DFT matrix. In our experiments, the γ is set to 0.1. The effect of this perturbation is examined in Section 6.5

5.3. Fine Tuning for Downstream Tasks

To facilitate comparison with FFT-RadNet, we adopt the same multi-task learning setup. Specifically, our ADCNet backbone is fine-tuned to perform both object detection and freespace segmentation at the same time by adding detection and segmentation heads as shown in Figure 4.

For model training, the object detection part of the loss is

$$\mathcal{L}_{\text{det}} = \text{focal}(\mathbf{y}_{\text{cls}}, \hat{\mathbf{y}}_{\text{cls}}) + \alpha \text{smooth-L1}(\mathbf{y}_{\text{reg}}, \hat{\mathbf{y}}_{\text{reg}}), \tag{3}$$

where $\mathbf{y}_{\text{cls}}$ denotes the ground-truth classification labels on the feature map, with 1 indicates presence of object while 0 otherwise. The shape of $\mathbf{y}_{\text{cls}}$ is $N_{\text{range-bins}} \times N_{\text{azimuth-bins}}$.The $\mathbf{y}_{\text{reg}}$ denotes the regression targets. Following [27], the network predicts a range and an azimuth value. The regression targets are thus the remainders of range and azimuth modulo range and azimuth bin sizes. The shape of $\mathbf{y}_{\text{reg}}$ is $N_{\text{range-bins}} \times N_{\text{azimuth-bins}} \times 2$, where the last dimension corresponds to range and azimuth. The focal() function is from [17], and the smooth-L1() function is also known as Huber loss [10]. The α is a hyper-parameter balancing the two parts.

The freespace segmentation part of the loss is

$$\mathcal{L}_{\text{seg}} = \sum_{r,a} \text{BCE}(\mathbf{y}_{\text{seg}}(r, a), \hat{\mathbf{y}}_{\text{seg}}(r, a)), \tag{4}$$

where $\mathbf{y}_{\text{seg}}(r, a)$ denotes the ground truth at location (r, a), with 1 denotes object presence and 0 otherwise. The BCE() term denotes the binary cross entropy loss.

Combining the two parts, the loss function for training is thus

$$\mathcal{L} = \mathcal{L}_{\text{det}} + \beta \mathcal{L}_{\text{seg}}, \tag{5}$$

where β is a hyper-parameter controlling the weights on the freespace segmentation task.

6. Experiment results

In this section, we detail experiment results to verify the effectiveness of the proposed techniques. We present an overall comparison in Section 6.2, with ablation studies in the following subsections.

6.1. Experiment setup

We use the same data partitioning as that of [27]. The whole labelled dataset is split into train, val, and test set by *sequence* (driving session). That is, data samples from a sequence can only appear in one of the splits. This is to avoid allocating closely-resembling data samples (*e.g.* two consecutive data samples in the same driving session) to the same split, which can lead to inflated performance metrics.

For evaluation, we also follow the setup as in [27]. Object detection performance is measured by average-precision, average-recall and average-f1 scores, where average is done across a range of detection thresholds. The freespace segmentation results are evaluated with mIOU metric: the average IOU (intersection over union) score across test samples.

Table 1. Comparing ADCNet with baselines. AP: Average Precision, AR: Average Recall, RE: Range Error on detected objects, AE: Azimuth Error on detected objects. The full testset is broken down into an Easy and a Hard subset for detailed comparison. *: results cited from [27].

		All					Easy					Hard				
	Radar Input	**F1**	**AP**	**AR**	**RE (m)**	**AE (°)**	**F1**	**AP**	**AR**	**RE (m)**	**AE (°)**	**F1**	**AP**	**AR**	**RE (m)**	**AE (°)**
FFT-RadNet *	RD	0.89	0.97	0.82	0.11	0.17	0.95	0.98	0.92	0.10	0.13	0.76	0.93	0.65	0.13	0.26
Pixor - PC *	Point cloud	0.48	0.96	0.32	0.17	0.25	0.45	0.99	0.29	0.15	0.19	0.55	0.93	0.39	0.19	0.33
Pixor - RA *	RA	0.89	0.97	0.82	0.10	0.20	0.92	0.97	0.88	0.09	0.16	**0.81**	0.96	0.70	0.12	0.27
Cross Modal DNN [12]	RD	0.90	0.97	0.84	-	-	0.95	0.98	0.92	-	-	0.77	0.93	0.66	-	-
T-FFTRadNet [8]	RD	0.90	0.90	0.90	0.15	0.12	-	-	-	-	-	-	-	-	-	-
T-FFTRadNet [8]	ADC	0.87	0.88	0.87	0.16	0.13	-	-	-	-	-	-	-	-	-	-
ADCNet (Ours)	ADC	**0.92**	0.95	0.89	0.13	0.11	**0.97**	0.96	0.98	0.12	0.11	**0.81**	0.91	0.73	0.16	0.12

Table 2. Freespace segmentation comparison. The performance is measured by mIOU, the higher the value the better. *: results cited from [27].

	All	Easy	Hard
FFT-RadNet *	74.00%	74.60%	72.30%
PolarNet [21] *	60.6%	61.9%	57.4%
Cross Modal DNN [12]	80.40%	81.60%	76.70%
T-FFTRadNet - RD [8]	80.20%	-%	-%
T-FFTRadNet - ADC [8]	79.60%	-%	-%
ADCNet	78.59%	79.63%	75.90%

For the pre-training experiments in Section 6.4, it is important to note that we make sure any samples from the labeled test set sequences are not included in the pre-training stage. This avoids giving unfair advantages to the ADCNet, so that it is not allowed to see test samples even in the pre-training stage.

6.2. Comparing ADCNet to baselines

The object detection performance results are shown in Table 1, and the freespace segmentation results are presented in Table 2. Overall, we can see that ADCNet is the best method on object detection, capturing state of the art results even when compared to other methods which use more complex architectures such as T-FFTRadNet [8] which uses a Swin Transformer based backbone. Looking at the breakdown of results we can see that ADCNet provides a 5% F1 improvement over FFTRadNet on hard samples and is able to increase recall by 8%. Additionally, when comparing just the hard samples, ADCNet is tied for the top score with Cross Modal DNN [12] which also uses additional supervision (from camera information) during training.

On the semantic segmentation task we achieve competitive results, falling only 2% short of Cross Modal DNN [12] which captures the best performance. This suggests that the visual cues from camera signal may be providing important value for segmentation. Fortunately, this camera supervision technique can be combined with our proposed, radar-only framework. Exploring these fusion techniques is left for future work.

We present several model prediction samples in Figure 1. It can be seen from the samples that the network is able to correctly recognize the cars while rejecting other bright spots in the RA images. In the second sample, the network misses one car, possibly due to the road sign on the right, which causes a bright spot in the RA map, at a similar range of the missed car.

6.3. Network architecture ablation studies

As described in section 5.1 our framework for pre-training via distillation can be used with other network architectures. In order to validate this and to demonstrate the effectiveness of our learnable SP modules we perform two experiments; first we replace the FFTRadNet backbone with a simple UNet to create ADC UNet, and second we remove the learnable SP module and replace it with 3D convolutions to create Conv3D + FFTRadNet. Both variations are trained with and without pre-training in order to understand the effects of our pre-training step on various networks. Please see the appendix for more details on the architecture of the ADC Unet and Conv3D + FFTRadNet model. Additionally we train an ADCNet without pre-training for comparison.

From Table 3, when removing pre-training, we see a reduction of detection F1 score of 1%, 5% and 10% for the ADCNet, the ADC UNet, and the Conv3d+FFTRadNet, respectively. There is an additional 2% to 4% drop in mIOU for the freespace segmentation performance for these methods. We suspect that the improved effect of pre-training on the UNet backbone is because it lacks any of the specific radar modules such as MIMO Pre-encoder that are found in FFTRadNet [8]. These modules are helpful due to the Doppler division multiplexing used in the specific radar for this dataset. Nonetheless, owing to its simplicity, the UNet backbone runs in half the time as the FFTRadNet backbone and also consumes half the VRAM (see Table 4). This demonstrates the power of our pre-training framework: it allows simpler network architectures to punch far above their weight class by taking advantage of unlabeled data. Lastly, we find that the Conv3d+FFTRadNet architecture performs the worst overall, demonstrating the importance of using the learnable SP module that introduces radar inductive bias.

Table 3. Comparing different backbones using our framework. The numbers represent the performance on the full test set. Models denoted with NPT (no pre-training) are ones that do not use pre-training via distillation.

	F1	AP	AR	RE(m)	AE (°)	mIOU
ADCNet	0.92	0.95	0.89	0.13	0.11	78.59%
ADCNet - NPT	0.91	0.96	0.87	0.12	0.10	74.45%
ADC UNet	0.85	0.88	0.82	0.18	0.11	77.16%
ADC UNet - NPT	0.80	0.83	0.77	0.19	0.10	73.04%
Conv3d + FFTRadNet	0.47	0.58	0.39	0.19	0.33	74.69%
Conv3d + FFTRadNet - NPT	0.37	0.58	0.27	0.16	0.46	72.53%

Table 4. Efficiency comparison. Batch size is 20 for throughput and memory measurements. *All measurements taken on an RTX 3090 except for [12] which was measured with unknown hardware.

	Latency (ms)	Throughput (samples/sec)	Memory (GB)
FFTRadNet [27]	15.93	68	12.33
Cross Modal DNN [12]	68.00*	-	-
T-FFTRadNet [8]	20.00	-	-
ADCNet	18.13	58	14.89
ADC U-Net	8.18	137	7.45

6.4. Effectiveness of distillation

In this section, we are interested in examining the prediction accuracy of RAD tensor in the pre-training stage. A good RAD prediction accuracy would signal a successful distillation. This is a valid concern, as both the ADC tensor and the RAD are large: ADC is of shape $512 \times 256 \times 8$ while RAD is $128 \times 248 \times 256$ for the RADIal dataset. It is unclear whether learning a mapping between these two high-dimensional vectors is feasible.

We visualize several randomly selected RAD prediction samples in Figure 5. To visualize this 3D RAD tensor, we summed over the third dimension, and get a 2D image of dimension range and azimuth. To get a quantitative error measure, we compute the Relative Absolute Error (RAE) for each RAD entry as

$$\text{RAE}(i,j,k) = \frac{|\mathbf{Y}_{\text{RAD}}(i,j,k) - \hat{\mathbf{Y}}_{\text{RAD}}(i,j,k)|}{|\mathbf{Y}_{\text{RAD}}(i,j,k)|}. \quad (6)$$

As such, we get a relative error measure for each entry, and we report the maximum and mean of these errors in Figure 5.

Table 5. Comparing different initialization methods for ADCNet. The numbers represent the performance on the full test set.

	F1	AP	AR	RE(m)	AE (°)	mIOU
Exact-DFT	0.87	0.91	0.84	**0.12**	0.10	**75.60%**
Random-Doppler	0.82	0.91	0.74	0.16	0.10	74.43%
Perturbed-DFT	**0.91**	0.96	0.87	**0.12**	0.10	74.50%
Random	Failed to converge					

It can be seen from Figure 5 that the network can predict the RAD tensor to a very high accuracy: the ground truth and predicted images are visually close, and the RAE measures confirm this observation. This confirms that distillation of the expensive offline SP algorithm is indeed successful.

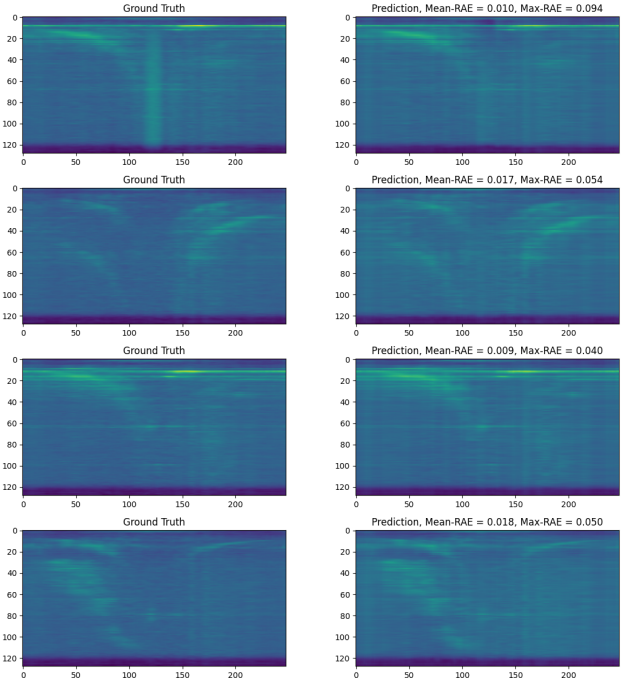

Figure 5. RAD prediction examples. The mean and max of Relative Absolute Error (RAE) for each sample are reported. As can be seen these examples, after pre-training, the network is able to predict RAD tensors to a very high accuracy.

6.5. Initialization for the learnable SP module

In this section, we compare different ways to initialize the learnable SP module. We compare these variations of the proposed ADCNet:

- Perturb-DFT initialization: perturbed DFT matrices as shown in Equation 2 are used to initialize both the range and Doppler DFT module in Figure 4
- Exact-DFT initialization: the exact DFT matrices are used to initialize both range and Doppler DFT modules
- Random-Doppler initialization: the exact DFT matrix is used to initialize the range DFT, while a randomly generated matrix is used for Doppler
- Random: randomly generated matrices are used for both range and Doppler modules

These methods employ a various amount of SP knowledge: the Exact-DFT method relies fully on SP, and the Random method almost completely disregard SP, while Perturbed-DFT and Random-Doppler lie in between.

The results are shown in Table 5. As can be seen, the Perturbed-DFT yields the best results. As we suspected, using the Exact-DFT initialization method can lead to sub-

Table 6. Comparing Exact-DFT and Perturbed-DFT initialization method for the object detection task. Again NPT refers to the models that have not used pre-training via distillation.

		All					Easy					Hard				
	Initialization	**F1**	**AP**	**AR**	**RE (m)**	**AE (°)**	**F1**	**AP**	**AR**	**RE (m)**	**AE (°)**	**F1**	**AP**	**AR**	**RE (m)**	**AE (°)**
ADCNet	Exact-DFT	0.89	0.93	0.86	0.13	**0.10**	0.96	0.95	0.97	0.12	0.10	0.75	0.86	0.67	**0.15**	0.12
ADCNet	Perturbed-DFT	**0.92**	0.95	0.89	0.13	0.11	**0.97**	0.96	0.98	0.12	0.11	**0.81**	0.91	0.73	0.16	0.12
ADCNet - NPT	Exact-DFT	0.90	0.96	0.84	0.13	**0.10**	0.95	0.98	0.93	0.12	**0.09**	0.77	0.91	0.67	**0.15**	0.12
ADCNet - NPT	Perturbed-DFT	0.91	0.96	0.87	**0.12**	**0.10**	**0.97**	0.98	0.97	**0.11**	0.10	0.78	0.91	0.68	0.16	0.12

Table 7. Comparing Exact-DFT and Perturbed-DFT initialization method for the freespace segmentation task

	Initialization	**All**	**Easy**	**Hard**
ADCNet	Exact-DFT	77.28%	78.37%	74.50%
ADCNet	Perturbed-DFT	78.59%	79.63%	75.90%
ADCNet - NPT	Exact-DFT	70.39%	70.81%	69.31%
ADCNet - NPT	Perburbed-DFT	74.45%	75.85%	70.89%

optimal performance. We hypothesize that this is because the network gets stuck in a local minima when initialized exactly. The Random-Doppler method yields considerably worse performance, while the Random method failed to converge to any meaningful model.

Since the Exact-DFT and Perburbed-DFT initialization methods get better performance than other choices, we provide a more in-depth comparison between the two. Moreover, we implement these initialization methods in the pre-training stage, then fine-tune to get the final detection and segmentation results. For this experiment, the multi-task learning setup as described in Section 5.3 is performed, with and without pre-training. All the hyper-parameters are kept the same, and only the initialization method is varied.

The results are reported in Table 6 and Table 7. As can be seen from these tables, with and without pre-training, the Perturbed-DFT initialization method always yields better performance than Exact-DFT. The improvement by using Perturbed-DFT is especially visible for ADCNet, confirming our intuition that a larger dataset is needed for end-to-end learning, as limited training data may not be able to nudge the network too far from the SP-based initialization.

The γ parameter in Equation 2 should be chosen with care: a large γ can destroy the DFT structure, while a too small γ can leave the SP module in a non-optimal local minimum. Here we present an experiment, where different γ is applied in the Perturb-DFT initialization scheme, and an ADCNet is trained (without pre-training) on the labeled RADIal dataset using the supervised multi-task learning setup in Section 5.3. Table 8 shows the results: a large γ (*e.g.* $\gamma = 2$) brings significant degradation to both object detection and freespace segmentation, while a small γ like 0.1 achieves better performance than if no perturbation ($\gamma = 0$) is applied.

Table 8. The effect of the γ parameter in Perturb-DFT initialization

γ	**F1**	**AP**	**AR**	**RE(m)**	**AE (°)**	**mIOU**
0	0.90	0.96	0.84	0.13	**0.1**	70.39%
0.1	**0.91**	0.96	0.87	**0.12**	**0.1**	**74.45%**
0.5	0.84	0.88	0.81	0.16	0.13	70.03%
2	0.65	0.75	0.57	0.18	0.12	69.47%

7. Conclusion

We present a pre-training via distillation pipeline that takes advantage of raw rdar data. By utilizing the task of ADC to RAD conversion as a pre-training step, we effectively distill important AOA estimation capabilities into our backbone network. Additionally, we propose a novel perturbed initialization technique for our improved learnable signal processing module. The combination of these techniques results in state of the art object detection and improved freespace segmentation results on the RADIal dataset within a unified multi-task model.

The designed pre-training via distillation process can be used with other SP algorithms as the teacher or other backbones as the student. We provide a simple example of replacing the backbone, and perform ablations to demonstrate the effectiveness of our pre-training pipeline as well as our perturbation initialization technique. We expect these techniques to be more powerful in more challenging environments (e.g. dense urban roads), or when scaling to larger unlabeled datasets for pre-training. We leave these explorations for future works.

References

[1] Dan Barnes, Matthew Gadd, Paul Murcutt, Paul Newman, and Ingmar Posner. The Oxford radar robotcar dataset: A radar extension to the oxford robotcar dataset. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, Paris, 2020.

[2] Graham M Brooker. Understanding millimetre wave FMCW radars. In *1st International Conference on Sensing Technology*, 2005.

[3] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020.

[4] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *Proceedings of the 37th International Conference on Machine Learning*, pages 1597–1607. PMLR, 2020.

[5] Yu-An Chung and James Glass. Generative pre-training for speech with autoregressive predictive coding. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3497–3501. IEEE, 2020.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[7] Sanchit Gandhi, Patrick von Platen, and Alexander M. Rush. Distil-whisper: Robust knowledge distillation via large-scale pseudo labelling, 2023.

[8] James Giroux, Martin Bouchard, and Robert Laganiere. T-fftradnet: Object detection with swin vision transformers from raw adc radar signals, 2023.

[9] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020.

[10] Peter J Huber. Robust estimation of a location parameter. *Breakthroughs in statistics: Methodology and distribution*, pages 492–518, 1992.

[11] Jyh-Jing Hwang, Henrik Kretzschmar, Joshua Manela, Sean Rafferty, Nicholas Armstrong-Crews, Tiffany Chen, and Dragomir Anguelov. CramNet: Camera-radar fusion with ray-constrained cross-attention for robust 3d object detection. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXVIII*, pages 388–405. Springer, 2022.

[12] Yi Jin, Anastasios Deligiannis, Juan Fuentes, and Martin Vossiek. Cross-modal supervision-based multitask learning with automotive radar raw data. *IEEE Transactions on Intelligent Vehicles*, PP:1–15, 2023.

[13] Jian Li. Over a century of array signal processing. `https://ieee-aess.org/files/ieeeaess/slides/2023AESSPowerPointDLJianLi.pdf`. Accessed 2023-11-17.

[14] Mao Li, Bo Yang, Joshua Levy, Andreas Stolcke, Viktor Rozgic, Spyros Matsoukas, Constantinos Papayiannis, Daniel Bone, and Chao Wang. Contrastive unsupervised learning for speech emotion recognition. In *2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6329–6333. IEEE, 2021.

[15] Peizhao Li, Pu Wang, Karl Berntorp, and Hongfu Liu. Exploiting temporal relations on radar perception for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17071–17080, 2022.

[16] Yu-Jhe Li, Jinhyung Park, Matthew O'Toole, and Kris Kitani. Modality-agnostic learning for radar-lidar fusion in vehicle detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 918–927, 2022.

[17] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2980–2988, 2017.

[18] Yang Liu, Feng Wang, Naiyan Wang, and Zhaoxiang Zhang. Echoes beyond points: Unleashing the power of raw radar data in multi-modality fusion. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[19] Michael Meyer, Georg Kuschk, and Sven Tomforde. Graph convolutional networks for 3d object detection on radar data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3060–3069, 2021.

[20] Ramin Nabati and Hairong Qi. Centerfusion: Center-based radar and camera fusion for 3d object detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 1527–1536, 2021.

[21] Farzan Erlik Nowruzi, Dhanvin Kolhatkar, Prince Kapoor, Elnaz Jahani Heravi, Fahed Al Hassanat, Robert Laganière, Julien Rebut, and Waqas Malik. Polarnet: Accelerated deep open space segmentation using automotive radar in polar domain. *arXiv preprint arXiv:2103.03387*, 2021.

[22] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

[23] Dong-Hee Paek, Seung-Hyun Kong, and Kevin Tirta Wijaya. K-radar: 4d radar object detection for autonomous driving in various weather conditions. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.

[24] Andras Palffy, Jiaao Dong, Julian FP Kooij, and Dariu M Gavrila. CNN based road user detection using the 3d radar cube. *IEEE Robotics and Automation Letters*, 5(2):1263–1270, 2020.

[25] Kun Qian, Shilin Zhu, Xinyu Zhang, and Li Erran Li. Robust multimodal vehicle detection in foggy weather using complementary lidar and radar signals. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 444–453, 2021.

[26] Sandeep Rao. MIMO radar. `https://www.ti.com/lit/an/swra554a/swra554a.pdf?ts=1675435882950&ref_url=https%253A%252F%`

252Fwww.ti.com%252Fproduct%252FIWR6843. Accessed: 2023-02-19.

[27] Julien Rebut, Arthur Ouaknine, Waqas Malik, and Patrick Pérez. Raw high-definition radar for multi-task learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17021–17030, 2022.

[28] William Roberts, Petre Stoica, Jian Li, Tarik Yardibi, and Firooz A Sadjadi. Iterative adaptive approaches to mimo radar imaging. *IEEE Journal of Selected Topics in Signal Processing*, 4(1):5–20, 2010.

[29] Richard Roy and Thomas Kailath. ESPRIT-estimation of signal parameters via rotational invariance techniques. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37(7):984–995, 1989.

[30] Meet Shah, Zhiling Huang, Ankit Laddha, Matthew Langford, Blake Barber, Sidney Zhang, Carlos Vallespi-Gonzalez, and Raquel Urtasun. Liranet: End-to-end trajectory prediction using spatio-temporal radar fusion. *arXiv preprint arXiv:2010.00731*, 2020.

[31] Marcel Sheeny, Emanuele De Pellegrin, Saptarshi Mukherjee, Alireza Ahrabian, Sen Wang, and Andrew Wallace. RADIATE: A radar dataset for automotive perception in bad weather. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–7. IEEE, 2021.

[32] Shunqiao Sun, Athina P. Petropulu, and H. Vincent Poor. Mimo radar for advanced driver-assistance systems and autonomous driving: Advantages and challenges. *IEEE Signal Processing Magazine*, 37(4):98–117, 2020.

[33] Harry L Van Trees. *Optimum array processing: Part IV of detection, estimation, and modulation theory*. John Wiley & Sons, 2002.

[34] Kyle Vedder, Neehar Peri, Nathaniel Chodosh, Ishan Khatri, Eric Eaton, Dinesh Jayaraman, Yang Liu, Deva Ramanan, and James Hays. Zeroflow: Scalable scene flow via distillation, 2023.

[35] Yizhou Wang, Zhongyu Jiang, Xiangyu Gao, Jenq-Neng Hwang, Guanbin Xing, and Hui Liu. RODNet: Radar object detection using cross-modal supervision. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 504–513, 2021.

[36] Yizhou Wang, Gaoang Wang, Hung-Min Hsu, Hui Liu, and Jenq-Neng Hwang. Rethinking of radar's role: A camera-radar dataset and systematic annotator via coordinate alignment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2815–2824, 2021.

[37] Bin Yang, Runsheng Guo, Ming Liang, Sergio Casas, and Raquel Urtasun. Radarnet: Exploiting radar for robust perception of dynamic objects. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII 16*, pages 496–512. Springer, 2020.

[38] Y. You, K. Luo, C. Phoo, W. Chao, W. Sun, B. Hariharan, M. Campbell, and K. Q. Weinberger. Learning to detect mobile objects from lidar scans without labels. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1120–1130, Los Alamitos, CA, USA, 2022. IEEE Computer Society.

[39] Yu Zhang, Daniel S Park, Wei Han, James Qin, Anmol Gulati, Joel Shor, Aren Jansen, Yuanzhong Xu, Yanping Huang, Shibo Wang, et al. BigSSL: Exploring the frontier of large-scale semi-supervised learning for automatic speech recognition. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1519–1532, 2022.

[40] Peijun Zhao, Chris Xiaoxuan Lu, Bing Wang, Niki Trigoni, and Andrew Markham. Cubelearn: End-to-end learning for human motion recognition from raw mmwave radar signals. *IEEE Internet of Things Journal*, 2023.

Supplementary for "ADCNet: Learning from RAW Radar Data via Distillation"

1. More experiment details

For training the ADCNet (both training from scratch and fine-turing from a pre-trained checkpoint), we use a single machine with 4 NVIDIA A10G GPUs. We use the Adam optimizer [2], with initial learning rate of 4×10^{-4}, and a batch size of 4 per GPU. We use the sum of validation F1 score (for object detection) and validation mIOU score (for freespace segmentation) to pick the best checkpoint.

For pre-training, we use 4 nodes each equipped with 4 NVIDIA A10G GPUs. The learing rate 16×10^{-4}, and the batch size is 12. We train the model for a total of 60 epochs, and pick the model with smallest validation loss for the subsequent fine-tuning step.

2. Learnable DSP training

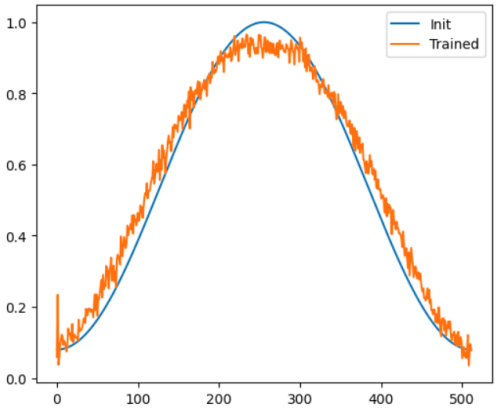

Figure 1. The first learnable window function of ADCNet before and after training

2.1. Implementation of the learnable SP module

One difficulty of implementing the learnable SP module is the fact that the DFT operation is in complex domain. To avoid using complex operations in the neural network, we split the complex tensors (ADC array and DFT matrix) into real and imaginary parts, and perform the multiplications separately, as shown in Figure 3. These are standard operations and can be easily implemented in a typical deep learning framework such as Pytorch. The learnable window module involves only one parameter vector, and it is multiplied to the inputs in the forward pass.

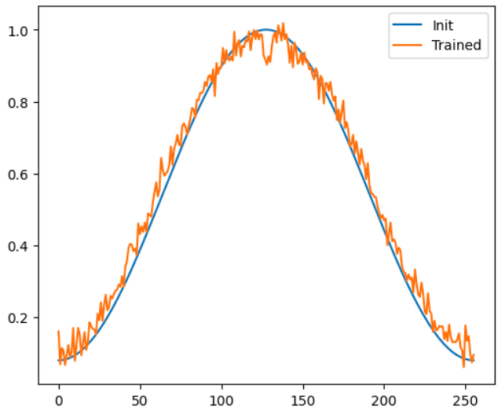

Figure 2. The second learnable window function of ADCNet before and after training

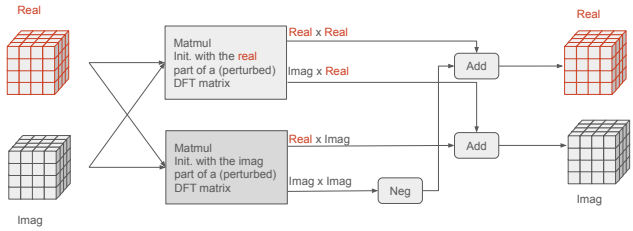

Figure 3. Implementing learnable DFT using real paramater matrices

2.2. Effects of training on the windowing functions

The effect of end-to-end training on window functions are shown in Figure 1 and Figure 2. It can be seen that in both the functions, the top is reduced and sides are raised. In signal processing, choice of window functions (see *e.g.* [1]) are usually made by a human expert, while the proposed approach amounts to data-driven window design – a process that could be incorporated in radar signal processing implementation.

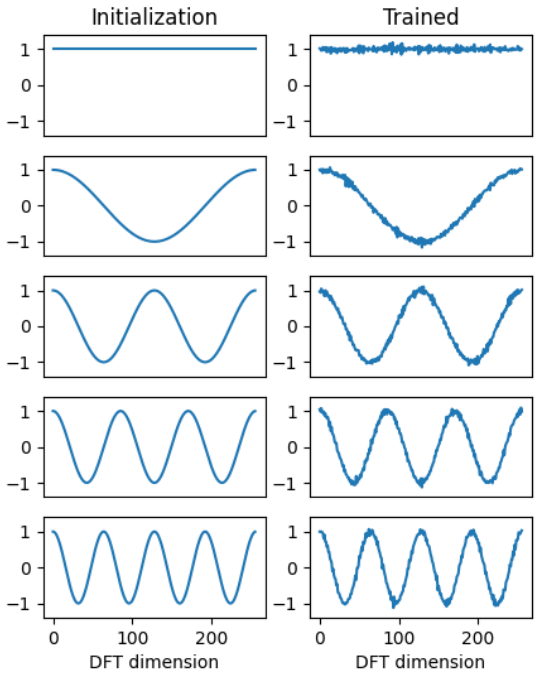

Figure 4. The real weights of the DFT matrix before and after training when *Exact-DFT* initialization is used. x-axis: DFT index; y-axis: real part of the DFT coefficient.

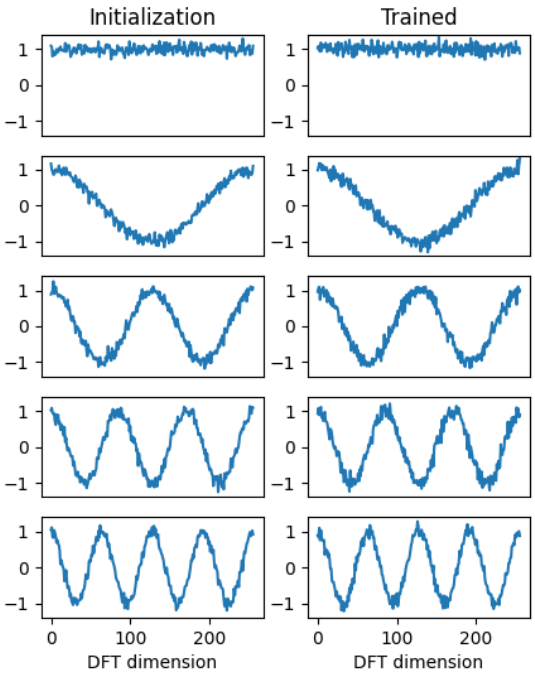

Figure 5. The real weights of the DFT matrix before and after training when *Perturbed-DFT* ($\gamma = 0.1$) initialization is used. x-axis: DFT index; y-axis: real part of the DFT coefficient.

2.3. Effects of perturbation on the learnable DFT

When training with the *Exact-DFT* initialization, we found that the DFT weights almost do not change before and after training, as shown in Figure 4. Figure 4 shows the real part of the first 5 rows of the second DFT matrix in ADCNet. This confirms our suspicion that without perturbation, the learnable DFT layer can get stuck in the local optimum of the DFT matrix. For a comparison, the same visualization for ADCNet with *Perturbed-DFT* initialization is shown in Figure 5.

Table 1. The difference between initialization and trained DFT weights, using either *Exact-DFT* or *Perturbed-DFT* initialization

	The first learnable DFT		The second learnable DFT	
	real	imaginary	real	imaginary
Exact-DFT	0.039	0.039	0.039	0.039
Perturbed-DFT	0.120	0.120	0.121	0.121

To get a quantitative measure of the difference between the initialization and trained DFT weights, we compute the average absolute difference between the two set of weights. The real and imaginary part is computed separately, and the results are shown in Table 1. As can be seen from Table 1, with Exact-DFT initialization, the DFT matrix indeed does not change much compared with the one with Perturbed-DFT, confirming the necessity to add perturbation to the DFT matrix for a strong end-to-end model.

3. Additional description on the ADC Unet and Conv3d+FFTRadNet baselines

The ADC Unet and Conv3d+ FFTRadNet are designed such that they have similar number of learnable parameters as the ADCNet.

For ADC Unet, we use a standard Unet backbone to replace the backbone the in the ADCNet. Importantly, we also removed the dilated convolution layer that is supposed to exploit the radar signal structure. Removal of the dilated convolution layer is motivated by the fact that these special convolution layer runs much slower than the standard 3x3 convoutional layers. As a result, we get a baseline model that, albeit yields worse accuracy, runs much faster. The proposed pre-training technique improves this baseline quite notably (5% absolute improvement on F1 score).

For the Conv3d+FFTRadNet baseline, we removed the learnable SP module from the ADCNet and replaced with a stack of Conv3d layers. The motivation is to test if such a generic model can handle the intricate radar signal purely

from end2end training: the results from our experiment (in the main paper) is negative, highlighting the necessity of incorporating signal processing knowledge and techniques into the end2end learning regime.

References

[1] Window function. https://en.wikipedia.org/wiki/Window_function. Accessed: 2023-03-03. 1

[2] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 1