EchMo: End-to-End Radar Echo Segmentation

Liwen Zhang[1,2†*] **Xinying Fu**[1,2†] **Youcheng Zhang**[1,2†] **Zijun Hu**[1,2] **Wen Chen**[1,2]

Shi Peng[1,2] **Jie Zhou**[1] **Zhe Ma**[1*]

[1]Intelligent Sci. and Tech. Academy (ISTA), CASIC, Beijing 100041, China
[2]Perception & Cognition Laboratory (PeCoLab@ISTA), Beijing, China
[*]Corresponding authors. Email: lwzhang9161@126.com, zhema_thu@163.com
[†]These authors contributed equally to this work.

Abstract

This paper presents **EchMo** (**ech**o-in-**m**ask-**o**ut), an end-to-end learnable radar semantic segmentation network that directly processes in-phase/quadrature (I/Q) echoes from pulse-Doppler radar (PDR). EchMo reinterprets conventional radar signal processing (RSP) as a deeply learnable pipeline, encompassing pulse compression (*i.e.*, matched filtering), moving target indication (MTI), coherent integration (CI), and target detection. The resulting compact model maps I/Q echoes to target semantic masks in one step, with a streamlined differentiable computational graph and substantial efficiency gains. Experimental results show that EchMo achieves better overall results than conventional pipelines and deep models based on intermediate radar representations (*i.e.*, range-Doppler maps), while providing a more interpretable foundation. To our knowledge, EchMo is the first end-to-end deep architecture for radar semantic segmentation directly from pulse-Doppler I/Q echoes; a prior-relaxation ablation study further clarifies how much of the classical radar prior should be retained. EchMo's codebase has been integrated into `https://github.com/zlw9161/PeCoRSS`, PeCoLab's RSS library.

1 Introduction

Radar detects targets through electromagnetic waves and remains operational in adverse weather and lighting conditions, making it indispensable for autonomous driving, aerospace, and security systems [5, 33]. This work aims at a deep integration of modern intelligent processing technologies with typical pulse-Doppler radar (PDR) systems, *enabling intelligent perception to reach further into the physical world*. The core challenge is to learn a mapping $\mathcal{F} : \mathbf{S} \to \mathbf{M}$ that converts raw I/Q echo signals into structured semantic outputs of targets of interest, *i.e.*, location, type, and motion, a problem addressed by conventional radar signal processing (RSP)[1] and deep learning through fundamentally different but incomplete approaches.

Hand-engineered expert pipeline. Conventional RSP decomposes the signal-to-semantic mapping $\mathcal{F} : \mathbf{S} \to \mathbf{M}$ into a handcrafted cascade $\mathcal{F}_{\text{RSP}} = f_{\text{CFAR}} \circ f_{\text{CI}} \circ f_{\text{MTI}} \circ f_{\text{MF}}$, extracting physical features through sequential stages, *i.e.*, matched filtering (MF), moving target indication (MTI) and coherent integration (CI), followed by constant false alarm rate (CFAR)-based target detection [19]. Despite strong theoretical foundations [36, 26], this modular design degrades in real-world conditions due to two limitations: i) *dependence on idealized assumptions* (*e.g.*, white Gaussian noise models in MF),

[1]All abbreviations are expanded at first occurrence; a complete glossary is provided in Table 4.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

which limits the environmental adaptability of $\mathcal{F}_{\text{RSP}}$; and ii) *shallow isolated module optimization* without end-to-end coordination, which inherently caps $\mathcal{F}_{\text{RSP}}$'s performance [11, 13].

Partially learnable model based on intermediate representations. Inspired by computer vision breakthroughs in object detection and semantic segmentation, deep learning has been integrated into RSP, forming two core perception pathways: Radar Object Detection (ROD) [2, 21, 4] and Radar Semantic Segmentation (RSS) [14, 24, 42, 43, 41, 16, 17]. These approaches learn a partial mapping $\mathcal{G} : \mathbf{X}_k \rightarrow \mathbf{M}$, where $\mathbf{X}_k$ denotes intermediate representations, *e.g.*, range-Doppler (RD) maps or range-azimuth-Doppler (RAD) tensors, produced by conventional RSP pipelines. Across both convolutional neural network (CNN)-based and Transformer-based frameworks, a consistent finding emerges: models tailored to the task characteristics or the statistical priors of these representations always outperform native convolution or attention mechanisms. Nevertheless, all these methods perform task learning on intermediate representations, a design that inherently *lacks end-to-end optimization* and suffers from *partial information loss* during the conversion from raw I/Q echoes, thereby limiting further performance improvements.

End-to-end perception from raw radar signals. A complementary line of research bypasses intermediate radar representations and learns directly from raw radar signals: ADCNet [40] distills conventional signal processing (SP) outputs into a backbone trained from raw ADC samples with a perturbed discrete Fourier transform (DFT) initialization, CubeLearn [44] replaces the fixed DFT preprocessing with a learnable module for human motion recognition, FFT-RadNet [29] learns angle recovery from RD spectra for detection and free-space segmentation, and T-FFTRadNet [9] couples complex transformation layers with Swin Transformers for object detection from raw ADC signals. These works share EchMo's spirit of rethinking SP through neural networks; however, they all operate on frequency-modulated continuous-wave (FMCW) radar, which differs fundamentally from PDR in waveform structure, signal processing pipeline, detection range, and application scenario. EchMo's learnable modules are inherently tied to the pulse-train signal structure, while these models are built around chirp-based FMCW processing; the modality gap therefore precludes direct comparison without a complete redesign on either side, making EchMo complementary to this line of work.

Following this line of thinking, a more general methodology can be derived: **prior-guided model design philosophy** is always conducive to obtaining better optimized structures. This is consistent with findings on other physics-related learning tasks [37, 6, 28], where domain knowledge (*e.g.*, physical laws, logical rules, expert experience) is encoded as constraints on network initialization, architecture, operators or criteria, improving generalization, interpretability, and robustness, a paradigm also proven successful in radar microwave imaging [39, 18] and radar detection [41, 16, 17]. Given this, we cannot help but wonder: *can raw echo processing also benefit from deep learning frameworks?*

We believe that **the answer lies at the intersection**. Conventional RSP leverages rigorous physical priors for reliable signal processing, while deep learning excels in adaptive large-scale data learning; yet their complementary strengths remain under-exploited: the fixed, hand-engineered RSP cascade incurs inevitable information loss, while purely data-driven black-box models suffer from optimization inefficiency and opaque computation paths. The path toward more flexible and general radar perception is thus to unify them within a single end-to-end learnable framework, *i.e.*, to construct *a differentiable computational architecture that intrinsically co-optimizes physical priors and data-driven representations* directly from raw I/Q signals.

To address the above challenge, we introduce EchMo (**Ech**o-in-**M**ask-**o**ut), an end-to-end RSS network that operates directly on base-band I/Q echoes and outputs range-Doppler-structured target masks. Our contributions span methodological design and engineering implementation:

† **At the methodological level.** Guided by the "First Principle" to fully excavate radar signal information, we present a *careful end-to-end learnable reformulation of the pulse-Doppler processing chain* by reconfiguring the key conventional RSP modules (*e.g.*, MF, MTI, CI) into parametric tensor-operable blocks, so that physical priors and data-driven parameters are co-optimized within a single differentiable architecture with seamless gradient flow from raw signals to semantic outputs. To scientifically validate the framework, we introduce the ReCK ablation strategy (**Re**laxation of **C**onstraint from Prior **K**nowledge), which goes beyond standard module ablations to *study how much of the classical radar prior should be retained*, systematically quantifying the contribution and constraint effects of prior-augmented modules in EchMo.

† **At the engineering level.** We construct Syn-RED and HiL-RED for controlled simulation and hardware-in-the-loop jamming scenarios, respectively, and establish raw-I/Q and radar-specific evaluation protocols. We further introduce module-wise acceleration and memory optimizations based on equivalent computation graph substitution, achieving over $5\times$ forward speedup and approximately 62% lower backward peak memory usage, thereby enabling practical depth, width, and temporal scaling variants of EchMo, *i.e.*, EchMoDeep, EchMo4MoE, and EchMoMemo.

2 Notation and Preliminaries

We begin with a unified notation to review the conventional RSP pipeline[2], facilitating the introduction to EchMo. Taking a typical pulse-Doppler radar (PDR) as an example, the key parameters include pulse width T, bandwidth B, pulse repetition time T_{PRT}, number of pulses P, number of range bins R, sampling frequency F_s, and medium frequency F_m. Generally, the linear frequency-modulated transmit signal (in the medium-frequency band) can be written as:

$$\mathbf{S}[t] = \exp\left\{2\pi j\left[\left(F_m - \frac{B}{2}\right)\cdot t + 0.5kt^2\right]\right\}, \tag{1}$$

where t enumerates the timestamps of $[0, \tau]$ with the interval $\frac{1}{F_s}$, and $k = \frac{B}{\tau}$ is a constant. The I/Q transmit signal serves as the reference signal $\mathbf{S} \in \mathbb{C}^T$, where $\tau = \lfloor T \cdot F_s \rfloor$ is the number of sampling points of $\mathbf{S}$.

The echo received in one PRT can be viewed as a mixture of multiple components, including the pulse scattered back from the target with a time delay, environmental clutter, and noise. We use $\mathbf{X}[i] \in \mathbb{C}^R$ to denote the discrete I/Q echo in the i-th PRT, where $R = F_s \cdot T_{\text{PRT}}$ is the number of range units. With these definitions, the classic RSP pipeline can be described as follows.

Matched Filter. MF is a core building block in PDR: it pinpoints the target echo in range, compresses the chirp into a sharp peak, and boosts the signal-to-clutter-noise ratio (SCNR). For a linear frequency-modulated (LFM) waveform, the optimal MF is the time-reversed conjugation of the reference signal $\mathbf{S}$, denoted as $\mathbf{S}^*_{\text{rev}}$. Classically, both $\mathbf{S}^*_{\text{rev}}$ and the echo $\mathbf{X}[i]$ are transformed into the frequency domain and multiplied, involving Fast Fourier Transform (FFT) and inverse FFT (IFFT) operations. Through *time-domain reversal-conjugation* $\Leftrightarrow$ *frequency-domain conjugation* and *frequency-domain multiplication* $\Leftrightarrow$ *time-domain convolution*, a more concise time-domain form of MF can be derived:

$$\mathbf{X}_{\text{mf}}[i] = \mathbf{X}[i] \otimes \mathbf{S}^*_{\text{rev}}, \tag{2}$$

where $\otimes$ denotes convolution. Eq. 2 shows that the time-domain MF reduces to a simple multiply-accumulate calculation, naturally fitting modern deep models. In practice, a window $\mathbf{W}$ (*e.g.*, Hamming/Hann/Taylor) is applied to curb spectral leakage and sidelobes,

$$\mathbf{X}_{\text{mf}}[i] = \mathbf{X}[i] \otimes \tilde{\mathbf{S}}^*_{\text{rev}}, \text{ where } \tilde{\mathbf{S}} = \mathbf{W} \odot \mathbf{S}. \tag{3}$$

Moving Target Indication. MTI suppresses clutter by canceling returns from stationary scatterers (*e.g.*, ground, vegetation). In PDR-based RSP, this is typically a delay canceller, *i.e.*, a temporal high-pass filter whose behavior is set by the delay order. For strong clutter and weak (low RCS) slow targets, double/triple-delay cancellers are preferred; Eq. 4 gives the double-delay case.

$$\begin{aligned}
&\mathbf{X}_{\text{mti}}[i] = \mathbf{X}_{\text{mf}}[i], && \text{for } i = 1;\\
&\mathbf{X}_{\text{mti}}[i] = \mathbf{X}_{\text{mf}}[i] - 2\mathbf{X}_{\text{mf}}[i-1], && \text{for } i = 2;\\
&\mathbf{X}_{\text{mti}}[i] = \mathbf{X}_{\text{mf}}[i] - 2\mathbf{X}_{\text{mf}}[i-1] + \mathbf{X}_{\text{mf}}[i-2], && \text{for } i = 3, \cdots, P.
\end{aligned} \tag{4}$$

Ignoring the boundary case $i = 1$, the delay canceller is a weighted sum of pulses. A higher order typically suppresses clutter better and exhibits band-pass characteristics, but may also amplify sidelobes and increase computation, reflecting the engineering limitation of classical RSP: manually preset filters can hardly adapt to complex real-world scenarios.

Coherent Integration/Moving Target Detection. MTI is a time-domain operation and leaves a slow-time sequence. To expose Doppler structure and exploit inter-pulse phase coherence, we move to the frequency domain via CI/MTD, which *coherently* accumulates moving-target energy along

[2]A more detailed review of conventional RSP can be found in Sec. A.1.

slow time and produces the range-Doppler (RD) matrix for CFAR detection. CI is thus also known as moving target detection (MTD) filtering, and in practice reduces to a DFT: for the r-th range bin $\mathbf{X}_{\text{mti}}[:][r] = \{\mathbf{X}_{\text{mti}}[p][r]\}_{p=1}^{P} \in \mathbb{C}^{P\times 1}$, a D-bin Fourier filter bank maps it to Doppler bins as:

$$\mathbf{X}_{\text{ci}}[d][r] = \sum_{p=1}^{P} \mathbf{X}_{\text{mti}}[p][r] \cdot e^{-\mathrm{j}\cdot 2\pi d \frac{p}{P}}, \text{ where } d = 1, \cdots, D. \tag{5}$$

Here $\{e^{-\mathrm{j}\cdot 2\pi d \frac{p}{P}}\}_{p=1}^{P}$ is the d-th Fourier basis vector of the Doppler filter bank. CI outputs the RD matrix $\mathbf{X}_{\text{ci}} \in \mathbb{C}^{D\times R}$, on whose magnitude $|\mathbf{X}_{\text{ci}}|$ CFAR operates. Although many deep RD-based detectors exist (*e.g.*, PKC [41], AdaPKC [16]), here we focus on the conventional pre-CFAR RSP stages and highlight our new contributions.

3 EchMo: End-to-End Echo-Based Radar Semantic Segmentation

For nearly half a century, the aforementioned RSP pipeline has been the "ironclad rule" of PDR systems. However, in increasingly complex electromagnetic environments and with the rapid emergence of unmanned aerial vehicles (UAVs), these methods have revealed considerable limitations, the essence of which lies in excessive reliance on a priori design while *lacking inherent data-driven capability*. Given this, we present **EchMo**, an end-to-end echo-to-mask radar semantic segmentation model: starting from raw echoes, EchMo directly outputs segmentation masks, which consists of:

† **Learnable Matched Filter (LMF)**. We start from the analytic MF (Eq. 2) and learn a lightweight, data-driven perturbation to compensate for waveform distortions and practical mismatches.

† **Multi-order Learnable Delay Canceller (MLDC)**. A differentiable MTI-style delay subtraction (Eq. 4) with multiple learnable orders, suppressing stationary clutter while keeping training and inference consistent.

† **Group-perturbable Doppler Filter (GDF)**. Instead of fixed, uniformly spaced DFT bins (Eq. 5), we learn non-uniform, group-wise frequency offsets/perturbations to better accumulate Doppler energy and target velocity cues.

† **PeakConv-based Segmenter (PKC-Seg)**. A segmentation head built upon PeakConv [41] (PKC), a CFAR-inspired learnable operator, maps the filtered features to semantic masks, validated on measured radar datasets [16, 25].

3.1 Learnable Matched Filter

The matched filter (MF) in Eq. 2 aims to find the part of the echo $\mathbf{X}$ most correlated with the reference signal $\mathbf{S}$. However, environmental clutter and device noise cause signal distortion, leading to a mismatch between theoretically optimal MF coefficients and practical requirements. To address this, we propose **LMF**, which naturally introduces data-driven perturbations to the classical MF.

An obvious approach to implementing LMF is to employ a learnable weight matrix to modulate the input echoes, using ideal filtering results as supervision and thereby transforming LMF into a reconstruction task preceding segmentation, resembling the self-supervised pretraining of large language models (LLMs) [23, 1, 7]. However, this risks rendering EchMo a stagewise-trained model, hindering its evolution into *a truly end-to-end, stand-alone foundational architecture*. A more critical issue is the *"chicken-and-egg"* dilemma: since authentic detection environments are extremely difficult to construct, ideal pulse compression results are scarcely available, precluding LMF optimization through self-supervised reconstruction. We therefore reformulate LMF optimization as reducing the optimization difficulty, or equivalently, identifying *a suitable starting point for optimization*. To this end, we propose the idea of reference signal kernelization.

Reference Signal Kernelization. Our key idea is to re-implement time-domain MF as a learnable non-causal convolution layer while initializing its kernel with the reference signal. Given the input echo $\mathbf{X} \in \mathbb{C}^{P\times R}$, we define a complex non-causal convolution layer $\mathcal{K}_{\text{nc}}^{\dagger}\{\cdot;\ \mathbf{K}^{\dagger}\} : \mathbb{C}^{1\times P\times R} \to \mathbb{C}^{1\times P\times R}$ with kernel weights $\mathbf{K}^{\dagger} \in \mathbb{R}^{2\times 1\times \tau}$ as follows:

$$\begin{aligned}\mathcal{K}_{\text{nc}}^{\dagger}\left\{\mathbf{X};\ \mathbf{K}^{\dagger}\right\} =& \mathcal{K}_{\text{nc}}\left\{\Re(\mathbf{X});\ \mathbf{K}^{\dagger}[1]\right\} - \mathcal{K}_{\text{nc}}\left\{\Im(\mathbf{X});\ \mathbf{K}^{\dagger}[2]\right\} \\ &+ \mathrm{j}\cdot\left(\mathcal{K}_{\text{nc}}\left\{\Re(\mathbf{X});\ \mathbf{K}^{\dagger}[2]\right\} + \mathcal{K}_{\text{nc}}\left\{\Im(\mathbf{X});\ \mathbf{K}^{\dagger}[1]\right\}\right).\end{aligned} \tag{6}$$

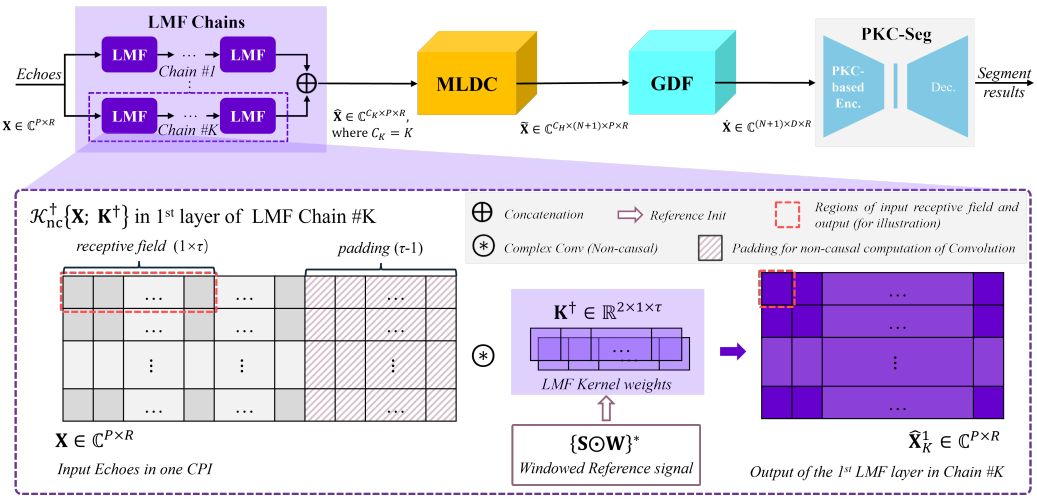


Figure 1: Schematic of a multi-layer LMF module with K chains and its position in EchMo. Each chain re-implements the time-domain matched filter (Eq. 2) as a learnable non-causal complex convolution initialized from the (windowed) reference signal (Reference Init, Eq. 8).

We adopt complex-domain convolution [34] to keep the LMF natively compatible with I/Q processing, preserving amplitude and phase jointly [33, 5]. Non-causality is enforced by a range-wise zero-padding operation, $\texttt{Pad}_{\text{nc}}(\cdot) : \mathbb{R}^{P\times R} \to \mathbb{R}^{P\times(R+\tau-1)}$, which is applied before each of the four internal real-valued convolutions; for the first convolution:

$$\mathcal{K}_{\text{nc}}\{\Re(\mathbf{X});\ \mathbf{K}^{\dagger}[1]\}[i][j] = \sum_{t=1}^{\tau} \mathbf{K}^{\dagger}[1][t] \cdot \texttt{Pad}_{\text{nc}}\left(\Re\left(\mathbf{X}\right)\right)[i][j+t-1], \quad \text{for} \quad i = 1, \cdots, P \quad \text{and} \quad j = 1, \cdots, R. \tag{7}$$

The kernel is initialized with the reference signal to incorporate expert domain knowledge:

$$\text{Reference Init}^{\dagger}: \quad \mathbf{K}^{\dagger}[1] \xleftarrow{\text{Ini.}} \Re(\mathbf{S}^{*}) \text{ or } \Re(\tilde{\mathbf{S}}^{*}); \quad \mathbf{K}^{\dagger}[2] \xleftarrow{\text{Ini.}} \Im(\mathbf{S}^{*}) \text{ or } \Im(\tilde{\mathbf{S}}^{*}). \tag{8}$$

Fig. 1 summarizes LMF's forward pass (detailed derivations in Appendix A.2.2). To boost capacity while keeping parameter growth scalable, a single-layer LMF is extensible in depth (Appendix A.3) and width, *i.e.*, stacking LMF layers within a chain and running multiple chains in parallel.

3.2 Multi-order Learnable Delay Canceller

MTI suppresses stationary clutter through delay cancellation along slow time, where the delay order δ determines the frequency response, yet an appropriate order is difficult to determine from prior knowledge alone. We propose **a multi-order learnable delay canceller (MLDC)** combining multiple learnable filters with different frequency responses.

Given the LMF output $\hat{\mathbf{X}} \in \mathbb{C}^{C_{\text{K}}\times P\times R}$, each learnable delay canceller (LDC) with order n is implemented as a causal convolution as follows [35]:

$$\mathcal{H}^{n}_{\text{caus}}\left\{\hat{\mathbf{X}};\ \mathbf{H}^{n}\right\} = \left\{\sum_{c=1}^{C_{\text{K}}}\sum_{i=0}^{n} \mathbf{H}^{n}[c][i] \cdot \hat{\mathbf{X}}[c][j-i][r]\right\}_{j=n+1,r=1}^{P,R}. \tag{9}$$

The kernel $\mathbf{H}^{n} \in \mathbb{R}^{C_{\text{K}}\times(n+1)\times 1}$ is initialized by the general solution of an n-order delay canceller:

$$\text{LDC Init:} \quad \mathbf{H}^{n}[:][i] \xleftarrow{\text{Ini.}} h_i^{n} \cdot \mathbf{1}, \quad \text{where} \quad h_i^{n} = \begin{cases} 1 & \text{if } i = 0; \\ (-1)^{i}\binom{n}{i} & \text{if } i = 1, \cdots, n, \ \&\ n \geq 1. \end{cases} \tag{10}$$

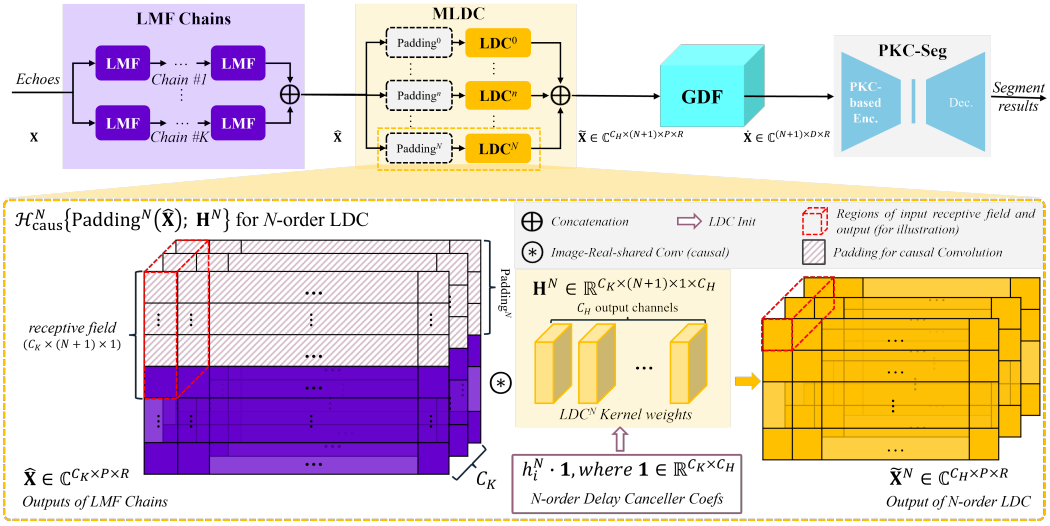


Figure 2: Schematic of an MLDC with N orders and its position in EchMo. The module concatenates $N+1$ parallel LDC branches, each branch has a causal convolution along slow time, which is initialized from the general solution of an n-order delay canceller (LDC Init), thus jointly learning complementary frequency responses for clutter suppression.

Here, h_i^n denotes the general delay-canceller solution used as prior knowledge; $\binom{n}{i} = \frac{n!}{i!(n-i)!}$ and $\mathbf{1} \in \mathbb{R}^{C_K}$ is the all-ones vector. This general solution kernelization improves interpretability and scalability while preserving the crucial identity-mapping property; in particular, for $n = 0$, the LDC reduces to a learnable gate on the identity path. As in LMF, we prepend each LDC with causal padding, $\mathtt{Pad}^n_{\text{caus}}(\cdot) : \mathbb{R}^{P\times R} \to \mathbb{R}^{(P+n)\times R}$, so that the pulse dimension matches the input without breaking causality. The complete MLDC concatenates $N+1$ parallel LDC branches (see Appendix A.2.3 for detailed derivations and Fig. 2 for the architecture):

$$\begin{aligned} &\mathcal{H}^*_{\text{caus}}\left\{\{\mathtt{Pad}^n_{\text{caus}}(\hat{\mathbf{X}})\}_{n=0}^N;\ \{\mathbf{H}^n\}_{n=0}^N\right\} = \tilde{\mathbf{X}}^0 \oplus \tilde{\mathbf{X}}^1 \oplus \cdots \oplus \tilde{\mathbf{X}}^N, \\ &\text{where}\quad \tilde{\mathbf{X}}^n = \mathcal{H}^n_{\text{caus}}\left\{\mathtt{Pad}^n_{\text{caus}}(\hat{\mathbf{X}});\ \mathbf{H}^n\right\},\ n = 0, 1, \cdots, N. \end{aligned} \tag{11}$$

3.3 Group-perturbable Doppler Filters

After range compression and clutter suppression, radar systems require precise velocity information through CI. Conventional CI uses fixed, uniformly spaced DFT bins, *i.e.*, the Fourier basis $\{e^{-\mathrm{j}\cdot 2\pi d\frac{p}{P}}\}_{p=1}^P$ in Eq. 5, which lacks flexibility. We propose the **Group-perturbable Doppler Filter (GDF)**, which learns non-uniform frequency perturbations.

Borrowing the concept "group" from group operations [15, 38], each group handles the output from an n-order LDC, and each Doppler filter (DF) is defined as:

$$\mathcal{D}^n\left\{\tilde{\mathbf{X}}^n;\ \mathbf{D}^n\right\} = \left\{\sum_{c=1}^{C_{\text{H}}}\sum_{p=1}^{P}\tilde{\mathbf{X}}^n[c][p][r]\cdot\mathbf{D}^n[c][p][d]\right\}_{d=1,r=1}^{D,R},\ \text{where } n = 0,\cdots,N. \tag{12}$$

To provide a hardware- and software-friendly equivalent computational flow, we introduce Euler's formula to decompose the Fourier basis under the complex-domain representation, allowing the kernel $\mathbf{D}^n \in \mathbb{C}^{C_{\text{H}}\times P\times 1\times D}$ of the GDF to be initialized as:

$$\begin{aligned} \text{DF Init: } &\mathbf{D}^{n\dagger}[1][p][d] \xleftarrow{\text{Ini.}} \mathbf{1}\cdot\cos(2\pi dp/P); \\ &\mathbf{D}^{n\dagger}[2][p][d] \xleftarrow{\text{Ini.}} \mathbf{1}\cdot\sin(2\pi dp/P), \\ &\text{for } n = 0,\cdots,N, \quad \text{and where } \mathbf{1}\in\mathbb{R}^{C_{\text{H}}}. \end{aligned} \tag{13}$$

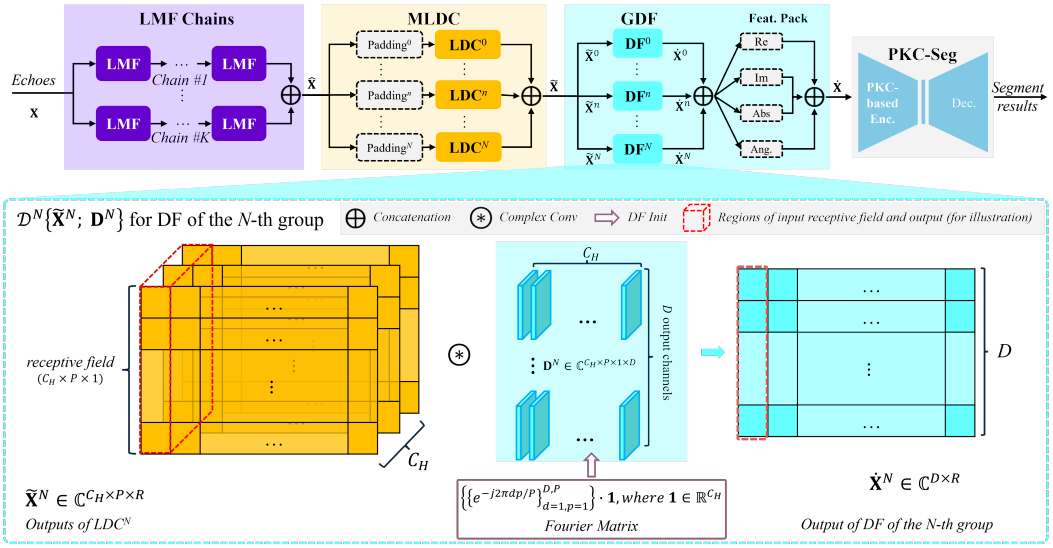


Figure 3: Schematic of a GDF with N groups and D channels, and its position in EchMo. Each group coherently integrates one LDC-order output through a learnable Doppler filter bank initialized from the cosine and sine Fourier bases (DF Init, Eq. 13).

Finally, we have the complete GDF formulation for all groups (see Appendix A.2.4 for detailed derivations and Fig. 3 for the architecture):

$$\begin{aligned} &\mathcal{D}^{*\dagger}\left\{\{\tilde{\mathbf{X}}^n\}_{n=0}^N;\ \{\mathbf{D}^{n\dagger}\}_{n=0}^N\right\} = \dot{\mathbf{X}}^0 \oplus \dot{\mathbf{X}}^1 \oplus \cdots \oplus \dot{\mathbf{X}}^N, \\ &\text{where}\quad \dot{\mathbf{X}}^n = \mathcal{D}^{n\dagger}\left\{\tilde{\mathbf{X}}^n;\ \mathbf{D}^{n\dagger}\right\},\ n = 0, 1, \cdots, N. \end{aligned} \tag{14}$$

With these components defined ($\mathcal{K}_{\text{nc}}^{\dagger}$, $\mathcal{H}_{\text{caus}}^{*}$ and $\mathcal{D}^{*\dagger}$), the classical RSP is transformed into a fully learnable architecture, upon which downstream tasks such as target detection or segmentation can be performed, yielding the complete EchMo architecture for end-to-end echo processing.

4 Experiments

4.1 Datasets and Training Setups

We construct two complementary **R**adar **E**cho **D**atasets: **Syn-RED**, a **Syn**thetic dataset for low-slow-small target detection with controlled signal-to-noise ratios (SNRs) via electromagnetic simulation; and **HiL-RED**, a **H**ardware-**i**n-the-**L**oop dataset for target-jammer recognition under active electronic countermeasures with varying jamming-to-signal ratios (JSRs). HiL-RED30K, a uniform 20% subset, is used for ablation studies and comparisons with milestone segmentation architectures; the full HiL-RED150K ($150,000$ coherent processing intervals, CPIs) is used for scaling experiments. All models use AdamW [20] (weight decay and initial learning rate $1\text{e}-4$) with cosine annealing and weighted cross-entropy loss. Each dataset uses within-sequence[3] train/val/test splits. We monitor precision, recall, IoU, and Dice during training; evaluation reports mIoU and mDice on the test set.

4.2 Relaxation of Constraint from Prior Knowledge

EchMo injects expert knowledge from classical RSP into learnable modules through differentiable computational flows and warm (initialization with expert knowledge) for optimization. To test when this prior helps or limits, we introduce **ReCK**, a unified ablation protocol defining four levels of progressively weaker prior constraints for each EchMo module (see details in Appendix A.7):

[3]A sequence is a collection of consecutive CPIs generated under fixed scenario settings, *i.e.*, trajectory, range, and SNR for Syn-RED; jamming type and JSR for HiL-RED. Contiguous CPI ranges within each sequence are assigned to the training, validation, and test partitions, which therefore share the same scenario conditions. See Appendix A.6 for full specifications.

† **ReCK-0 (frozen prior)**: initialize from the prior, then freeze the module parameters (Fig. 4-(a)), the tightest constraint, isolating the prior's effect under maximal trust.

† **ReCK-1 (cautious fine-tune)**: initialize from the prior and fine-tune with a reduced learning rate $\eta^* = \alpha\eta$ ($0 < \alpha < 1$; Fig. 4-(b)); the prior is treated as a good-but-not-perfect solution.

† **ReCK-2 (full fine-tune)**: the default EchMo setting, *i.e.*, initialize from the expert priors and train with the global learning rate (Fig. 4-(c)).

† **ReCK-3 (flow-only prior)**: keep only the prior-inspired computational flow but randomly initialize parameters (Fig. 4-(d)), the loosest constraint, serving as the control for prior initialization.

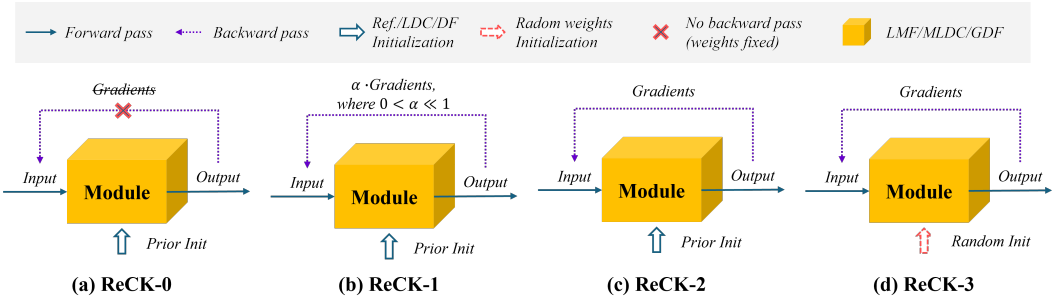


Figure 4: The four ReCK constraint levels for each EchMo module: (a) ReCK-0 freezes the prior-initialized parameters; (b) ReCK-1 fine-tunes them with a reduced learning rate; (c) ReCK-2 trains them with the global learning rate (default); (d) ReCK-3 keeps only the prior computational flow and re-initializes parameters randomly.

We evaluate ReCK under two configurations (Fig. 5): **Global ReCK** applies identical constraint levels across LMF, MLDC, and GDF to assess holistic behavior under uniform relaxation, while **Module-wise ReCK** sets a single module to frozen prior (ReCK-0) with the rest at full fine-tune (ReCK-2), identifying the modules most sensitive to prior constraints.

Global ReCK experiments (Fig. 5-(a)) demonstrate that EchMo's computational flow achieves strong performance across both data domains. Further analysis reveals divergent requirements for prior constraint intensity: on Syn-RED, cautious fine-tuning (ReCK-1) is optimal and further relaxation (ReCK-3) causes substantial degradation; HiL-RED30K exhibits the opposite pattern, where the frozen prior performs poorly and full relaxation (ReCK-3) is optimal. This indicates that pure digital synthetic data benefit from moderate inductive biases, whereas hardware-in-the-loop data generated under complex radio-frequency excitation benefit from relaxed constraints.

Module-wise ReCK experiments (Fig. 5-(b)) validate this principle: on Syn-RED, freezing any single module outperforms the global ReCK-2 baseline, indicating that physical priors are effective under idealized conditions; on HiL-RED30K, freezing any module severely degrades performance, indicating that priors should be fully relaxed in complex scenarios. A consistent finding across both datasets is the efficacy ranking of the priors: LMF > MLDC > GDF, *i.e.*, the closer a module is to the input, the more stable its theoretical prior appears. For visualizations supporting these findings, see Appendices A.8.1 and A.8.2.

4.3 Comparison with Milestone Architectures

EchMo represents the first *foundational architecture* for I/Q echo segmentation; here "foundational" is used not in the sense of pre-trained foundation models (*e.g.*, GPT [23]), but to denote a reusable end-to-end paradigm and a practical baseline for PDR echo perception. As comparative baselines, we select framework-level landmark models in semantic segmentation: TMVA-Net [24], U-Net [30], DeepLabv3+ [3], FCN-8s [32], and ViT [8][4]. Baselines operate on RSP-processed RD maps, while EchMo learns from raw echoes end-to-end. For the jamming-free small target detection scenario, *i.e.*, Syn-RED, conventional RSP effectively extracts features, enabling strong RD-based segmentation, and EchMo achieves comparable performance with minimal parameters and the fastest convergence (only 20 epochs). Under the complex jamming scenario, *i.e.*, HiL-RED, RSP struggles to extract

[4]All these baselines are adapted for our task (*e.g.*, TMVA-Net to single-view, ViT with a segmentation head).

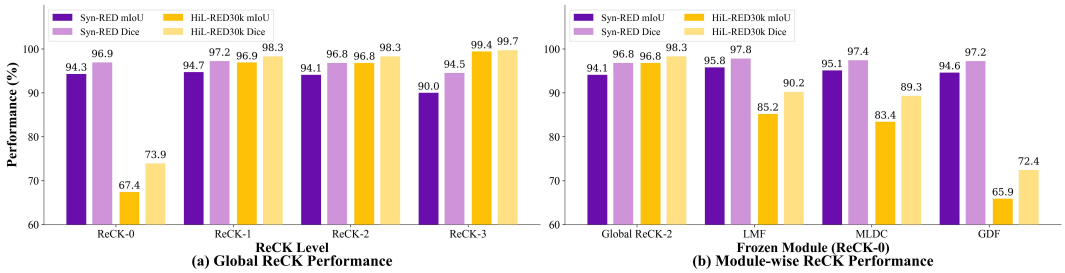


Figure 5: ReCK ablation results on Syn-RED and HiL-RED30K. (a) Global ReCK applies a uniform constraint level to all modules; (b) module-wise ReCK freezes one module at ReCK-0 with the others at ReCK-2. Moderate constraint (ReCK-1) is optimal on Syn-RED, whereas full relaxation (ReCK-3) performs best on HiL-RED30K.

Table 1: Comparison with milestone segmentation architectures. The w/ I/Q columns report training the architectures directly on raw I/Q echoes (real/imaginary parts as two channels) without RSP preprocessing; all degrade severely, confirming that the performance gap stems primarily from the input representation rather than baseline capacity.

Architecture	Params.	Syn-RED					HiL-RED30K			Avg. Performance	
		mIoU	mDice	Epochs	mIoU w/ I/Q	mDice w/ I/Q	mIoU	mDice	Epochs	mIoU	mDice
TMVA-Net	∼1.15M	93.6%	96.6%	98	68.2%	76.7%	51.0%	64.5%	163	72.3%	80.6%
U-Net	∼17.2M	85.1%	91.2%	284	77.4%	85.4%	73.5%	81.9%	228	79.3%	86.6%
DeepLabv3+	∼58.7M	**96.8%**	**98.3%**	124	71.2%	79.7%	62.3%	71.9%	158	79.6%	85.1%
FCN-8s	∼134.3M	95.3%	97.6%	206	69.2%	77.8%	87.3%	92.1%	271	91.3%	94.8%
ViT	∼1.40M	94.1%	96.9%	211	67.3%	75.8%	44.2%	56.2%	226	69.2%	83.1%
EchMo	∼1.29M	95.8%	97.8%	**20**	**95.8%**	**97.8%**	**99.4%**	**99.7%**	211	**97.6%**	**98.8%**

target features, as barrage jamming floods RD maps globally while deceptive jamming introduces ambiguous replicas. RD-based models thus degrade significantly, except for FCN-8s with the largest parameter scale. EchMo processes raw echoes directly and maintains stable performance where conventional feature extraction fails, outperforming all landmark architectures with $104\times$ fewer parameters than FCN-8s.

To confirm that this gap stems from the input representation rather than insufficient baseline capacity, we further trained all milestone architectures directly on raw I/Q echoes, feeding the real and imaginary parts as two input channels (w/ I/Q columns in Table 1, see details in Appendix A.8.4). Without RSP preprocessing, every architecture degrades severely, *e.g.*, DeepLabv3+ drops from 96.8% to 71.2% mIoU, because virtually no target structure is discernible in raw echoes before matched filtering and coherent integration (especially from a visual perspective). This directly leads to performance degradation for model architectures designed based on visual priors; and convergence behavior varies across these architectures. Note that all compared methods use exactly the same raw I/Q inputs, ensuring a fundamentally fair comparison.

Overall, *both learnable model capacity and prior utilization serve as prerequisites for stable performance guarantees*: DeepLabv3+ and FCN-8s benefit from larger capacity (average mIoU of 79.6% and 91.3% with 58.7M and 134.3M parameters), while EchMo, organically integrating physical priors, achieves 97.6% with merely 1.29M parameters. See Appendix A.8.3 for qualitative comparison and performance-cost bubble plots. We thus adopt EchMo as our new baseline for scalability studies.

4.4 Scalability Analysis

We build upon the accelerated backbone EchMo† (configuration in Table 23; acceleration details in Appendix A.4), which achieves $> 5\times$ forward speedup and 62% backward memory reduction, and scale it along three orthogonal dimensions, *i.e.*, **depth**, **width**, and **temporal**: EchMoDeep introduces a reduced-resolution refine stage alongside the full-resolution anchor, EchMo4MoE employs 4 parallel backbones routed by a lightweight gate with top-2 sparse fusion [31], and EchMoMemo augments predictions via an exponential moving average (EMA) memory state [22] across 4 frames/CPIs (detailed designs in Appendix A.5). Table 2 reports the results on the full-scale HiL-RED150K. Due

Table 2: Scalability analysis on HiL-RED150K

Model	Computing Platform	Params.	GFLOPs	Latency w/ BN sync. (ms/iter)	Mem. per GPU (MB)	mIoU (%)	mDice (%)	Epochs @Max Eps.
EchMo†	4×RTX4090	∼9.3M	∼81.2	1325.9	21978.6	93.4%	96.3%	48@50
EchMoDeep	2×RTX6000$^{\text{Ada}}$	∼16.0M	∼121.8	2410.6	37222.2	99.5%	99.7%	50@50
EchMo4MoE	3×A100	∼32.0M	∼324.8	4288.6	71175.8	98.6%	99.3%	41@50
EchMoMemo	8×L40S	∼9.3M	∼81.2	2494.0	42169.6	97.1%	98.5%	50@50

to limited computational resources, experiments were conducted on different platforms with varying batch sizes, and two checkpoints per epoch were recorded for fairness. The conclusions are clear: *EchMoDeep and EchMo4MoE both show better scaling*, surpassing the EchMo† backbone within 50 epochs, which supports further study of EchMo generalization with larger-scale datasets. Due to space limitations, additional experimental analyzes including radar-specific evaluation, per-class performance and sensitivity can be found in the Appendices A.8.5, A.8.6, and A.8.7.

5 Conclusion

Summary. EchMo is the first end-to-end deep RSS architecture that processes raw pulse-Doppler I/Q echoes, jointly learning echo representation and semantic segmentation in a single differentiable pipeline. For validation, we constructed two radar echo datasets covering two typical challenging scenarios, *i.e.*, Syn-RED for low-slow-small UAV detection and HiL-RED for complex electromagnetic environments. Through equivalent computation graph substitution, the accelerated EchMo† achieves major speed and memory gains, enabling scalable expansion. As a foundational architecture for remote sensing, EchMo lays the foundation for future advances in radar echo perception, with broader validation on operational platforms as the next step.

Limitations. EchMo currently has three main limitations:

i. **generalization validation**: effectiveness on other radar modalities such as continuous-wave or synthetic aperture radar, and on *independently collected real-world data* from different radar platforms and acquisition conditions;

ii. **lower-precision training**: whether similarly stable performance can be achieved with lower-precision computation, *e.g.*, bfloat16 (BF16);

iii. **dataset split design**: our within-sequence split evaluates EchMo on disjoint CPIs from shared conditions, faithfully testing within-condition generalization (echoes are highly stochastic, and JSR/SNR are statistical notions controlled only at the median level), but does not evaluate transfer to unseen conditions.

Future Work. Condition-level hold-out, *e.g.*, excluding entire jamming types from training, requires few-shot learning; our roadmap proceeds from Sim2Real transfer to few/zero-shot evaluation with condition-held-out splits, and ultimately to on-hardware deployment.

Code and Resources. The source code of EchMo is integrated into PeCoRSS, which is a general library of RSS developed and maintained by PeCoLab. The interactive visualization app and model checkpoints are publicly available at PeCoLab/EchMo-Vis and PeCoLab/EchMo-ckpts, respectively.

Acknowledgments and Disclosure of Funding

This work was supported by research funding from the Intelligent Science and Technology Academy, CASIC. We thank the group led by Wen Hu at Nanjing University of Aeronautics and Astronautics for their technical assistance with the construction of the Syn-RED dataset, and IntelliCube for their technical assistance with the hardware-in-the-loop design used to construct the HiL-RED dataset. We also thank all the reviewers for their valuable feedback and suggestions.

References

[1] Anthropic. Claude. `https://claude.ai`, 2026. Large language model, accessed March 10, 2026.

[2] Daniel Brodeski, Igal Bilik, and Raja Giryes. Deep radar detector. In *2019 IEEE Radar Conference (RadarConf)*, pages 1–6, 2019. doi: 10.1109/RADAR.2019.8835792.

[3] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 833–851, 2018.

[4] Lei Cheng and Siyang Cao. Transrad: Retentive vision transformer for enhanced radar object detection. *IEEE Transactions on Radar Systems*, 3:303–317, 2025. doi: 10.1109/TRS.2025.3537604.

[5] G. Richard Curry. *Radar System Performance Modeling*. Artech House, 2nd edition, 2004. ISBN 978-1-58053-816-9.

[6] Artur S. d'Avila Garcez, Marco Gori, L. Lamb, Luciano Serafini, Michael Spranger, and S. Tran. Neural-symbolic computing: An effective methodology for principled integration of machine learning and reasoning. *FLAP*, 6:611–632, 2019. URL `https://api.semanticscholar.org/CorpusID:155092677`.

[7] DeepSeek-AI. DeepSeek. `https://chat.deepseek.com`, 2026. Large language model, accessed March 10, 2026.

[8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *Proceedings of the 9th International Conference on Learning Representations (ICLR)*, 2021. URL `http://proceedings.mlr.press/v139/dosovitskiy21a.html`.

[9] James Giroux, Martin Bouchard, and Robert Laganière. T-fftradnet: Object detection with swin vision transformers from raw adc radar signals. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pages 4030–4039, 2023.

[10] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, 9:249–256, 2010.

[11] Gor Hakobyan and Bin Yang. High-performance automotive radar: A review of signal processing algorithms and modulation schemes. *IEEE Signal Processing Magazine*, 36(5):32–44, 2019.

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1026–1034, 2015.

[13] Wen Jiang, Yihui Ren, Ying Liu, and Jiaxu Leng. Artificial neural networks and deep learning techniques applied to radar target detection: A review. *Electronics*, 11(1):156, 2022.

[14] Prannay Kaul, Daniele de Martini, Matthew Gadd, and Paul Newman. Rss-net: Weakly-supervised multi-class semantic segmentation with fmcw radar. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 431–436, 2020. doi: 10.1109/IV47402.2020.9304674.

[15] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[16] Teng Li, Liwen Zhang, Youcheng Zhang, ZijunHu, Pengcheng Pi, Zongqing Lu, Qingmin Liao, and Zhe Ma. AdaPKC: Peakconv with adaptive peak receptive field for radar semantic segmentation. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL `https://openreview.net/forum?id=oLcPadFrY3`.

[17] Teng Li, Liwen Zhang, Youcheng Zhang, and Qingmin Liao. Axfl: Axial prior-guided cross-view fusion learning for radar semantic segmentation. *Expert Systems with Applications*, 303:130552, 2026. ISSN 0957-4174. doi: https://doi.org/10.1016/j.eswa.2025.130552. URL `https://www.sciencedirect.com/science/article/pii/S0957417425041673`.

[18] Leiyao Liao, Zishuo Hong, Ziwei Liu, and Gengxin Zhang. Integrated physically interpretable model for sar target recognition: Unified fusion of electromagnetic and deep features. *IEEE Geoscience and Remote Sensing Letters*, 22:1–5, 2025. doi: 10.1109/LGRS.2025.3548166.

[19] Salman Liaquat, Nor Muzlifah Mahyuddin, and Ijaz Haider Naqvi. An end-to-end modular framework for radar signal processing: A simulation-based tutorial. *IEEE Aerospace and Electronic Systems Magazine*, 39(9):98–118, 2024. doi: 10.1109/MAES.2023.3334689.

[20] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

[21] Bence Major, Daniel Fontijne, Amin Ansari, Ravi Teja Sukhavasi, Radhika Gowaikar, Michael Hamilton, Sean Lee, Slawomir Grzechnik, and Sundar Subramanian. Vehicle detection with automotive radar using deep learning on range-azimuth-doppler tensors. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 924–932, 2019. doi: 10.1109/ICCVW.2019.00121.

[22] Daniel Morales-Brotons et al. Exponential moving average of weights in deep learning. *arXiv preprint arXiv:2411.18704*, 2024.

[23] OpenAI. ChatGPT. https://chatgpt.com, 2026. Large language model, accessed March 10, 2026.

[24] Arthur Ouaknine, Alasdair Newson, Patrick Pérez, Florence Tupin, and Julien Rebut. Multi-view radar semantic segmentation. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 15651–15660, 2021. doi: 10.1109/ICCV48922.2021.01538.

[25] Arthur Ouaknine, Alasdair Newson, Julien Rebut, Florence Tupin, and Patrick Pérez. Carrada dataset: Camera and automotive radar with range- angle- doppler annotations. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 5068–5075, 2021. doi: 10.1109/ICPR48806.2021.9413181.

[26] H Vincent Poor. *An introduction to signal detection and estimation*. Springer Science & Business Media, 2013.

[27] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

[28] M. Raissi, P. Perdikaris, and G.E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2018.10.045. URL https://www.sciencedirect.com/science/article/pii/S0021999118307125.

[29] 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 (CVPR)*, pages 17021–17030, 2022.

[30] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241. Springer International Publishing, 2015.

[31] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. In *International Conference on Learning Representations*, 2017.

[32] Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):640–651, 2017. doi: 10.1109/TPAMI.2016.2572683.

[33] Merrill I. Skolnik. *Radar Handbook*. McGraw-Hill Education, 3rd edition, 2008.

[34] Chiheb Trabelsi, Olexa Bilaniuk, Ying Zhang, Dmitriy Serdyuk, Sandeep Subramanian, João Felipe Santos, Soroush Mehri, Negar Rostamzadeh, Yoshua Bengio, and Christopher J Pal. Deep complex networks. *arXiv preprint arXiv:1705.09792*, 2017.

[35] Aäron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew W. Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio, 2016. URL https://api.semanticscholar.org/CorpusID:6254678.

[36] Harry L. Van Trees, Kristine L. Bell, and Zhi Tian. *Detection, estimation, and modulation theory, Part I: Detection, estimation, and filtering theory*. John Wiley & Sons, 2nd edition, 2013.

[37] Laura von Rueden, Sebastian Mayer, Katharina Beckh, Bogdan Georgiev, Sven Giesselbach, Raoul Heese, Birgit Kirsch, Julius Pfrommer, Annika Pick, Rajkumar Ramamurthy, Michal Walczak, Jochen Garcke, Christian Bauckhage, and Jannis Schuecker. Informed machine learning – a taxonomy and survey of integrating prior knowledge into learning systems. *IEEE Transactions on Knowledge and Data Engineering*, 35(1):614–633, 2023. doi: 10.1109/TKDE.2021.3079836.

[38] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 3–19, 2018.

[39] Álvaro Yago Ruiz, Marta Cavagnaro, and Lorenzo Crocco. A physics-assisted deep learning microwave imaging framework for real-time shape reconstruction of unknown targets. *IEEE Transactions on Antennas and Propagation*, 70(8):6184–6194, 2022. doi: 10.1109/TAP.2022.3162320.

[40] Bo Zhang, Ishan Khatri, Michael Happold, and Chulong Chen. Adcnet: Learning from raw radar data via distillation, 2023.

[41] Liwen Zhang, Xinyan Zhang, Youcheng Zhang, Yufei Guo, Yuanpei Chen, Xuhui Huang, and Zhe Ma. Peakconv: Learning peak receptive field for radar semantic segmentation. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17577–17586, 2023. doi: 10.1109/CVPR52729.2023.01686.

[42] Tongrui Zhang and Yunsheng Fan. A 3d u-net based on a vision transformer for radar semantic segmentation. *Sensors*, 23(24), 2023. ISSN 1424-8220. doi: 10.3390/s23249630. URL `https://www.mdpi.com/1424-8220/23/24/9630`.

[43] Youcheng Zhang, Liwen Zhang, Zijun Hu, Pengcheng Pi, Teng Li, Yuanpei Chen, Shi Peng, and Zhe Ma. Tarss-net: Temporal-aware radar semantic segmentation network. In A. Globerson, L. Mackey, D. Belgrave, A. Fan, U. Paquet, J. Tomczak, and C. Zhang, editors, *Advances in Neural Information Processing Systems*, volume 37, pages 4906–4933. Curran Associates, Inc., 2024. doi: 10.52202/079017-0159. URL `https://proceedings.neurips.cc/paper_files/paper/2024/file/08a362bd4ae1934e099ce025f06039fe-Paper-Conference.pdf`.

[44] 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*, 10(12):10236–10249, 2023. doi: 10.1109/JIOT.2023.3237494.

A Appendix for EchMo

Appendix Overview and Reading Instructions

This appendix provides supplementary technical details that complement the main manuscript. Appendix A.1 reviews the conventional RSP pipeline that underpins Sec. 2, covering matched filtering, moving target indication, and coherent integration. Appendix A.2 provides complete derivations of the three core EchMo modules, *i.e.*, LMF, MLDC, and GDF, introduced in Sec. 3, and Appendix A.3 further presents the deep LMF variants. Appendix A.4 details the profiling-guided acceleration of EchMo, and Appendix A.5 describes the depth, width, and temporal scaling designs evaluated in Sec. 4.4. Appendix A.6 specifies the Syn-RED and HiL-RED datasets (Sec. 4.1), Appendix A.7 supplements the ReCK ablation protocol (Sec. 4.2), and Appendix A.8 collects additional experimental results and visualizations. Table 3 gives a roadmap of this appendix with page pointers, and Table 4 summarizes the abbreviations used in the main text and in this appendix for quick reference.

A.1 Conventional Radar Signal Processing Pipeline Review

A.1.1 Matched filter

The matched filter (MF) is an important technique in PDR systems. It performs three functions: i) localizing the target return along the range dimension; ii) compressing a wide pulse into a narrow peak spanning one or a few range bins; and iii) suppressing noise to enhance the target signal. For a linear frequency-modulated waveform, the optimal MF can be formulated as follows.

$$\mathbf{X}_{\text{mf}}[i] = \text{IFFT}\left[\text{FFT}\left(\mathbf{S}^{*}_{\text{rev}}\right) \odot \text{FFT}\left(X[i]\right)\right] \tag{15}$$

Table 3: Roadmap of this appendix.

Sec.	Content	Page
A.1	**Conventional Radar Signal Processing Pipeline Review**	13
	Matched filter	13
	Moving Target Indication	14
	Coherent Integration (or Moving Target Detection)	16
A.2	**Detailed Methodology of EchMo**	17
	Extended Discussion on End-to-End Radar Perception	17
	Details of LMF	18
	Details of MLDC	20
	Details of GDF	21
A.3	**Going Deep for LMF**	22
	Prior-Residual Separation for Conv-based LMF Layers	22
	Residual Cross-Attention for LMF	24
A.4	**EchMo Acceleration for Scaling**	26
	Comprehensive Profiling and Bottleneck Analysis for Baseline	26
	Packed Convolution and Chain-Level Parallelism for LMF	26
	Kernel Packing for Parallel MLDC	29
	Einsum-Based Projection for GDF	30
	Comprehensive Profiling for Improved EchMo	31
A.5	**Scaling Up EchMo**	31
	Deep Scaling via Multi-Resolution Stages	33
	Wide Scaling via Mixture of Experts	34
	Temporal Scaling via Memory-Augmented Processing	36
A.6	**Dataset Details**	37
	Syn-RED	37
	HiL-RED	39
	Dataset Split Strategy	41
A.7	**Supplementary Description of ReCK**	43
A.8	**Additional Experimental Results**	43
	Intermediate Feature Visualizations	43
	Visualization and Analysis of Learned GDF Kernels	43
	Milestone Architectures Comparison Visualizations	45
	Raw-I/Q Baseline Comparisons	47
	ROC and P_d–FAR Analysis	50
	Per-Class Metrics	50
	Robustness and Sensitivity Analysis	51
A.9	**Broader Impacts and Responsible Use**	51

where $\mathbf{S}^*_{\text{rev}}$ denotes the time-reversed conjugate of $\mathbf{S}$, and $\odot$ denotes element-wise multiplication. Equation 15 shows the frequency-domain MF commonly used in conventional radar systems. Using the equivalences *time-domain reversal and conjugation* $\Leftrightarrow$ *frequency-domain conjugation*, we obtain

$$\mathbf{X}_{\text{mf}}[i] = \text{IFFT}\left[\text{FFT}(\mathbf{S})^* \odot \text{FFT}(X[i])\right]. \tag{16}$$

Then the time-domain MF can be written as follows.

$$\mathbf{X}_{\text{mf}}[i] = \mathbf{X}[i] \otimes \mathbf{S}^*_{\text{rev}}, \tag{17}$$

where $\otimes$ denotes the convolution operation. Equations 16 and 17 show that time-domain MF requires only convolution (multiplication and addition) for digital pulse compression, making it naturally compatible with modern deep learning models. In practice, a window is applied to reduce spectral leakage and sidelobes. The windowed time-domain MF, with coefficients $\mathbf{W}$ drawn from, for example, the Hamming, Hann, or Taylor windows, is

$$\mathbf{X}_{\text{mf}}[i] = \mathbf{X}[i] \otimes \tilde{\mathbf{S}}^*_{\text{rev}}, \text{ where } \tilde{\mathbf{S}} = \mathbf{W} \odot \mathbf{S}. \tag{18}$$

To illustrate this process and motivate our work, Fig. 6 provides a detailed overview of the complete time-domain MF process in the conventional RSP pipeline. The frequency-domain MF is omitted because it is less relevant to the method presented here.

A.1.2 Moving Target Indication

Moving Target Indication (MTI) is essentially a clutter-suppression technique used to suppress returns from stationary environmental scatterers, *e.g.*, ground surfaces or vegetation. In the RSP pipeline of

Table 4: Alphabetical list of all abbreviations.

Abbr.	Full name	Abbr.	Full name
BF16	brain floating-point 16-bit	LMF	learnable matched filter
BN	batch normalization	LSS	low-slow-small
CA-CFAR	cell-averaging CFAR	mDice	mean Dice coefficient
CFAR	constant false alarm rate	MF	matched filtering
CI	coherent integration	mIoU	mean intersection over union
CNN	convolutional neural network	MLDC	multi-order learnable delay canceller
CPI	coherent processing interval	MLP	multi-layer perceptron
DDP	distributed data parallel	MoE	mixture of experts
DDC	differentiable delay canceller	MTD	moving target detection
DF	Doppler filter	MTI	moving target indication
DFT	discrete Fourier transform	NBJ	narrowband barrage jamming
DFTJ	dense false-target jamming	PDR	pulse-Doppler radar
DRFM	digital radio frequency memory	PKC	PeakConv
EchMo	echo-in-mask-out	PKC-Seg	PeakConv-based segmenter
EMA	exponential moving average	PRI	pulse repetition interval
FCN	fully convolutional network	PRT	pulse repetition time
FFT	fast Fourier transform	RAD	range-azimuth-Doppler
GDF	group-perturbable Doppler filter	RD	range-Doppler
GFLOPs	giga floating-point operations	RCS	radar cross section
GMAC	giga multiply-accumulate operations	ReCK	relaxation of constraint from prior knowledge
GPU	graphics processing unit	RED	radar echo dataset
HiL-RED	hardware-in-the-loop radar echo dataset	ReXAttn	residual cross-attention
I/Q	in-phase/quadrature	RF	radio frequency
IF	intermediate frequency	ROD	radar object detection
IFFT	inverse fast Fourier transform	RSP	radar signal processing
IoU	intersection over union	RSS	radar semantic segmentation
ISRJ	interrupted-sampling repeater jamming	SCNR	signal-to-clutter-noise ratio
JSR	jamming-to-signal ratio	SIJ	slice jamming
LDC	learnable delay canceller	SNR	signal-to-noise ratio
LFM	linear frequency modulation	Syn-RED	synthetic radar echo dataset
LLM	large language model	UAV	unmanned aerial vehicle
		ViT	vision transformer
		WBJ	wideband barrage jamming

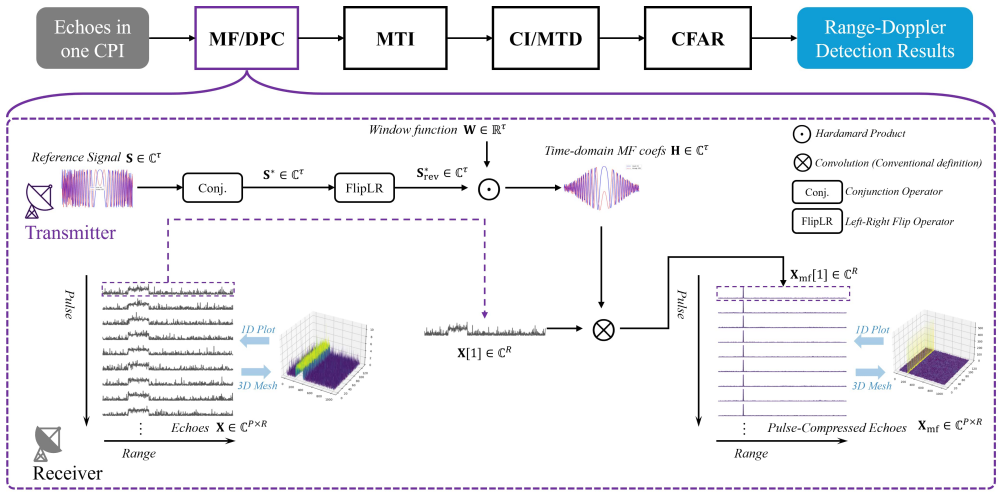


Figure 6: Illustration of the time-domain matched filtering process in the conventional radar signal processing pipeline.

Doppler radar systems, a delay canceller is commonly used. Figure 7 illustrates this process in the conventional RSP pipeline, which can be formally expressed as:

$$\begin{aligned} \mathbf{X}_{\text{mti}}[i] &= \mathbf{X}_{\text{mf}}[i], && \text{for } i = 1; \\ \mathbf{X}_{\text{mti}}[i] &= \mathbf{X}_{\text{mf}}[i] - \mathbf{X}_{\text{mf}}[i-1], && \text{for } i = 2, \cdots, P. \end{aligned} \tag{19}$$

Equation 19 shows the calculation of a single-delay canceller (or two-pulse cancellation). It has a high-pass frequency response and is suitable when clutter is relatively weak and slow-target detection requirements are modest. For scenarios with strong clutter and slow targets of interest, double- or triple-delay cancellers are more common. Equation 20 shows the calculation of a double-delay canceller.

$$\begin{aligned} \mathbf{X}_{\text{mti}}[i] &= \mathbf{X}_{\text{mf}}[i], && \text{for } i = 1; \\ \mathbf{X}_{\text{mti}}[i] &= \mathbf{X}_{\text{mf}}[i] - 2\mathbf{X}_{\text{mf}}[i-1], && \text{for } i = 2; \\ \mathbf{X}_{\text{mti}}[i] &= \mathbf{X}_{\text{mf}}[i] - 2\mathbf{X}_{\text{mf}}[i-1] + \mathbf{X}_{\text{mf}}[i-2], && \text{for } i = 3, \cdots, P. \end{aligned} \tag{20}$$

Ignoring the boundary cases, the double-delay canceller computes a weighted sum of three adjacent pulses. It generally provides better clutter suppression than the single-delay canceller and exhibits band-pass characteristics in the frequency domain. However, a higher delay order does not necessarily improve target detection: an excessively high order can increase sidelobe levels and degrade the response to moving targets, while also introducing additional computational overhead.

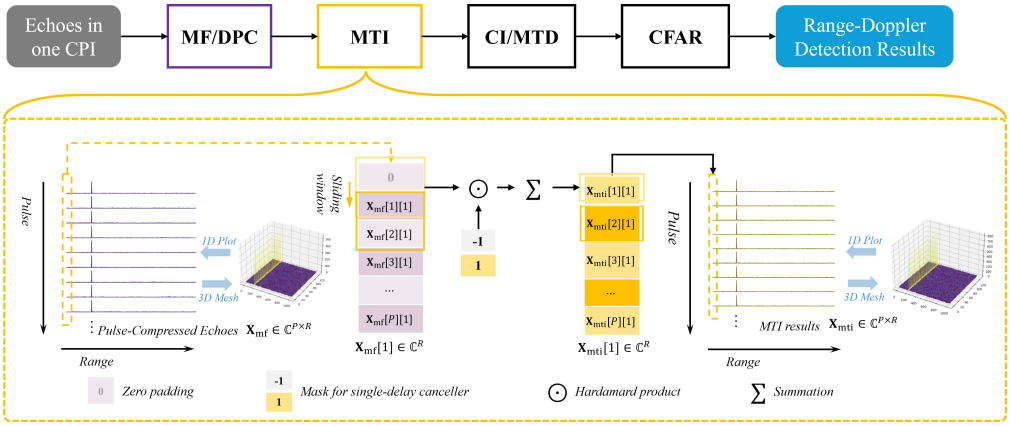


Figure 7: Illustration of delay-canceller-based moving target indication in the conventional radar signal processing pipeline. The single-delay canceller is shown for simplicity; double-delay cancellers are more common in practice.

A.1.3 Coherent Integration (or Moving Target Detection)

MTI is a purely time-domain operation, and its output remains a slow-time sequence. Time-domain analysis alone is therefore often insufficient to exploit all echo information, motivating explicit frequency-domain analysis through coherent integration (CI).

Like MTI, CI is applied along the pulse (slow-time) dimension. Unlike MTI, it exploits phase correlation across pulses to accumulate moving-target returns according to their Doppler shifts. This frequency dimension is also called the Doppler or velocity dimension. CI is therefore commonly referred to as MTD filtering, and its output is the RD matrix used for CFAR detection. Mathematically, CI reduces to a discrete Fourier transform. Given the MTI output in the r-th range bin, $\mathbf{X}_{\text{mti}}[:][r] = \{\mathbf{X}_{\text{mti}}[p][r]\}_{p=1}^{P} \in \mathbb{C}^{P\times 1}$, a Doppler filter bank constructed from D Fourier basis vectors projects this sequence onto different Doppler bins:

$$\mathbf{X}_{\text{ci}}[d][r] = \sum_{p=1}^{P} \mathbf{X}_{\text{mti}}[p][r] \cdot e^{-\text{j}\cdot 2\pi d \frac{p}{P}}, \text{ where } d = 1, \cdots, D. \tag{21}$$

Here $\{e^{-\text{j}\cdot 2\pi d \frac{p}{P}}\}_{p=1}^{P}$ is the d-th Fourier basis vector of the Doppler filter bank. CI can also be implemented through matrix multiplication, as shown in Fig. 8.

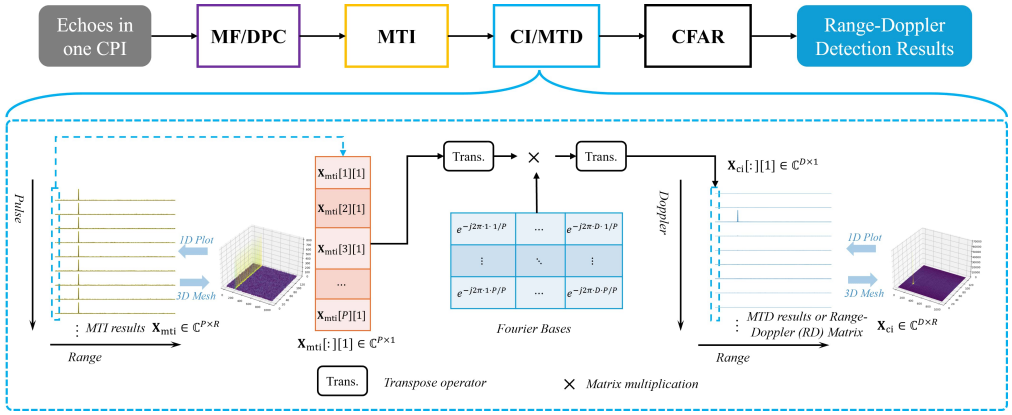


Figure 8: Illustration of coherent integration (MTD filtering) in the conventional radar signal processing pipeline. Matrix multiplication depicts the Fourier transform; this matrix form is equivalent to Eq. 21.

Table 5: Comparison between the FMCW and pulse-Doppler radar (PDR) regimes: existing end-to-end models learning from raw signals [40, 44, 29, 9] all belong to the FMCW regime, whereas EchMo operates on PDR echoes.

	FMCW	**PDR (this work)**
Waveform	chirp-based continuous wave	coherent pulse train
SP pipeline	range-FFT $\rightarrow$ Doppler-FFT $\rightarrow$ beamforming	MF $\rightarrow$ MTI $\rightarrow$ CI/MTD
Task type	detection, classification	dense semantic segmentation
Scenario	autonomous driving, indoor sensing	long-range surveillance, anti-jamming

The output of CI, $\mathbf{X}_{\text{ci}} \in \mathbb{C}^{D\times R}$, is known as the RD matrix. Finally, CFAR detection is performed on the real-valued RD magnitude matrix $|\mathbf{X}_{\text{ci}}|$. Deep RD-based radar detection models have been extensively studied, including PKC [41], which makes CA-CFAR learnable, and AdaPKC [16] for adaptive clutter estimation, among others. We therefore review only the conventional RSP stages preceding CFAR and emphasize the contributions of this work.

A.2 Detailed Methodology of EchMo

A.2.1 Extended Discussion on End-to-End Radar Perception

Before delving into the detailed methodology, for clarity, we further specify the positioning and applicable scope of EchMo, to avoid potential confusion with other raw-signal-based radar perception methods [40, 44, 29, 9]. As discussed in Sec. 1, existing raw-signal/ADC-based radar perception methods all operate on FMCW radar, which differs fundamentally from EchMo's PDR modality. Table 5 provides a detailed comparison between PDR and FMCW.

We summarize the following three distinctions to help readers better understand EchMo's positioning.

- **Radar modality.** All four baselines operate on FMCW (chirp-based continuous wave); EchMo operates on pulse-Doppler (pulse-train-based). The signal structures and processing pipelines are fundamentally different.

- **Task type.** ADCNet/FFT-RadNet [40, 29] perform object detection (bounding boxes); CubeLearn [44] performs classification (activity labels); only EchMo performs dense semantic segmentation (unit/cell-level masks).

- **Application scenario.** All four are designed for autonomous driving or indoor sensing; EchMo targets long-range surveillance and detection under active jamming with pulse-Doppler radar, a fundamentally different operational environment.

A.2.2 Details of LMF

It can be seen from the implementation of MF in the time domain in Eq. 2 that MF essentially aims to find the part of the given echo $\mathbf{X}$ that is most correlated with the transmitted reference signal $\mathbf{S}$. However, the received signal is inevitably affected by environmental clutter and device noise during propagation and reception. Consequently, theoretically optimal MF coefficients may be mismatched in practice. The fundamental reason is that the classical MF is designed from the transmitted reference signal alone and does not account for clutter and noise in the received echoes $\mathbf{X}$. Motivated by this limitation, we propose learning the MF from the echoes, *i.e.*, LMF.

An obvious approach is to use a learnable weight matrix to modulate the input echoes and employ ideal pulse-compression results as supervision. This formulation turns learning into a signal reconstruction task. However, just as environmental clutter and device noise are difficult to model in the real world, ideal pulse-compression results are difficult to obtain. Therefore, using this learning component as a pre-processing module for the detection task is a more feasible solution with a lower implementation cost. As the saying goes, there is *"no free"* lunch. Although such a choice reduces the cost of obtaining ground truth supervision signals, it increases the difficulty of optimization. As mentioned earlier, current deep models commonly take the RD matrix as input largely because raw echoes are substantially less structured than the RD matrix obtained after a series of entropy-reducing operations. The key question then becomes how to reduce the optimization difficulty, or equivalently, *where should we start the optimization?*

As with large language models (LLMs) [23, 1, 7], initializing from pre-trained parameters has become standard practice because it provides a better optimization starting point, often closer to a desirable solution. For radar-echo perception, where no such pre-trained models are available, alternative ways to obtain a good optimization starting point are needed. Fortunately, the conventional RSP integrates valuable domain-expert knowledge, which might be a good starting point for our LMF. Following this motivation, we propose the idea reference signal kernelization, which can naturally introduce expert domain knowledge into EchMo, while retaining the benefits of end-to-end deep learning.

In particular, given the i-th echo of a CPI, $\mathbf{X}$, the design idea of "reference signal kernelization" can be implemented with the following steps:

i. **MF re-implementation with non-causal convolution**. Revisiting the time-domain MF in Eq. 17 shows that MF is a convolution with each $\mathbf{X}[i]$ as input and $\mathbf{S}^*_{\text{rev}}$ as the kernel. Considering only the real part, this operation can be rewritten as follows.

$$\Re\left(\mathbf{X}_{\text{mf}}[i][j-\tau+1]\right) = \sum_{t=1}^{\tau} \Re\left(\mathbf{S}^*_{\text{rev}}[t]\right) \cdot \Re\left(\mathbf{X}[i][j-t+1]\right). \tag{22}$$

Here, $j = \tau, \tau+1, \cdots, R$, and $\Re(\cdot)$ denotes the operation of taking the real part. It can equivalently be written in the following more concise form, which avoids reversing $\mathbf{S}^*$:

$$\Re\left(\mathbf{X}_{\text{mf}}[i][j]\right) = \sum_{t=1}^{\tau} \Re\left(\mathbf{S}^*[t]\right) \cdot \Re\left(\mathbf{X}[i][j+t-1]\right), \tag{23}$$

where, $j = 1, \cdots, R-\tau+1$. This shows that classical MF is a weighted sum whose output at each range bin depends on both future and present input values. The process can therefore be implemented by a non-causal convolution layer $\mathcal{K}_{\text{nc}}\{\cdot;\ \mathbf{K}\} : \mathbb{R}^{1\times P\times R} \to \mathbb{R}^{1\times P\times(R-\tau+1)}$ with unit stride in both dimensions and kernel weights $\mathbf{K} \in \mathbb{R}^{1\times\tau}$, which can be viewed as a learnable version of $\Re\{\mathbf{S}^*\}$. The forward pass of $\mathcal{K}_{\text{nc}}$ for the given input $\Re\{\mathbf{X}\}$ can be written as:

$$\begin{aligned} \mathcal{K}_{\text{nc}}\{\Re(\mathbf{X});\ \mathbf{K}\}[i][j] &= \sum_{t=1}^{\tau} \mathbf{K}[1][t] \cdot \Re\left(\mathbf{X}[i][j+t-1]\right), \\ &\text{where} \quad i = 1, \cdots, P \quad \text{and} \quad j = 1, \cdots, R-\tau+1. \end{aligned} \tag{24}$$

Here, to make the range dimension of the output result consistent with that of the input while preserving the non-causality of the calculation process, we use the $\texttt{Pad}_{\text{nc}}(\cdot) : \mathbb{R}^{P\times R} \to \mathbb{R}^{P\times(R+\tau-1)}$ operator to append an all-zero vector $\mathbf{0} \in \mathbb{R}^{\tau-1}$ to the end of each input pulse

$\Re\{\mathbf{X}\}$. Equation 25 then preserves the remaining output values:

$$\begin{aligned}\mathcal{K}_{\text{nc}}\{\Re(\mathbf{X});\ \mathbf{K}\}[i][j] = \sum_{t=1}^{\tau} \mathbf{K}[1][t] \cdot \texttt{Pad}_{\text{nc}}\left(\Re\left(\mathbf{X}\right)\right)[i][j+t-1], \\ \text{where} \quad i = 1, \cdots, P \quad \text{and} \quad j = 1, \cdots, R.\end{aligned} \tag{25}$$

So far, taking the real part as an example, we have introduced how to equate the calculation process of the conventional MF to a convolutional module in deep learning. This enables the coefficients of MF to be parameterized and learnable. In other words, we redefine MF as a parameterized function whose coefficients can be further optimized. However, it should be noted that there is a certain difference between the conventional convolution operation $\otimes$ mentioned in Eq. 17 and the convolution operation in deep learning. Specifically, the latter does not flip the convolution kernel. This distinction is important when deriving the above steps.

ii. **Kernel initialization with reference signal**. The preceding re-implementation ensures that LMF matches classical MF computationally. However, a good optimization starting point depends on the initialization of the convolutional kernel weights. Generally speaking, in the absence of a pre-trained network, the commonly used initialization methods for learnable parameters are typically based on certain standard distributions, *e.g.*, a uniform distribution for Xavier initialization [10] or a Gaussian distribution for He initialization [12]. Taking Eq. 23 as an example, the kernel weights $\mathbf{K}$ can be initialized as follows:

$$\begin{aligned}\text{Xavier:} \quad \mathbf{K}[1][i] &\sim \mathcal{U}\left(-\sqrt{\frac{6}{n_{\text{in}}+n_{\text{out}}}}, \sqrt{\frac{6}{n_{\text{in}}+n_{\text{out}}}}\right); \\ \text{He:} \quad \mathbf{K}[1][i] &\sim \mathcal{N}\left(0, \sqrt{\frac{2}{n_{\text{in}}+n_{\text{out}}}}\right).\end{aligned} \tag{26}$$

where i ranges from 1 to τ, $\mathcal{U}$ and $\mathcal{N}$ denote uniform and Gaussian distributions, respectively, and n_{in} and n_{out} are the numbers of input and output channels for $\mathcal{K}_{\text{nc}}$. However, these initialization methods are not inherently related to the radar itself, thus making this approach to initialization somewhat blind in nature. To achieve the goal of *giving a "good start"* for LMF, we directly assign the reference signal $\Re(\mathbf{S}^*)$ or the windowed reference signal $\Re(\tilde{\mathbf{S}}^*)$ (see Eq. 18) to the convolutional kernel weights $\mathbf{K}$ during the initialization phase. Then $\mathbf{K}$ can be initialized as:

$$\text{Reference Init:} \quad \mathbf{K} \xleftarrow{\text{Ini.}} \Re(\mathbf{S}^*) \text{ or } \Re(\tilde{\mathbf{S}}^*). \tag{27}$$

Thus, the expert knowledge embedded in the conventional MF is naturally incorporated into the convolutional kernels of LMF as initial conditions. Furthermore, employing the same methodology, we can transform the fixed coefficients, $\mathbf{W}$ in the windowed reference signal $\tilde{\mathbf{S}}^*$ into learnable masks. These masks, along with the initialized convolutional kernels, participate in the learning process to further enhance the nonlinear representation capability of LMF.

iii. **Complex convolution for handling I/Q echoes**. Using orthogonal I/Q signals in radar transmission and reception offers numerous advantages, including joint amplitude-and-phase transmission, improved spectrum utilization, and enhanced interference suppression [33, 5]. Thus, the ability to process I/Q dual-channel echoes is an essential consideration for designing LMF. An I/Q signal sample is represented by a complex number as follows[5]:

$$x = \Re(x) + \mathrm{j} \cdot \Im(x), \tag{28}$$

where, $\Im(\cdot)$ represents the operation of taking the imaginary part. The real-valued convolution in Eq. 25 is then extended to the complex-valued form derived from the definition in [34]:[6]:

$$\begin{aligned}\mathcal{K}_{\text{nc}}^{\dagger}\left\{\mathbf{X};\ \mathbf{K}^{\dagger}\right\} =& \mathcal{K}_{\text{nc}}\left\{\Re(\mathbf{X});\ \mathbf{K}^{\dagger}[1]\right\} - \mathcal{K}_{\text{nc}}\left\{\Im(\mathbf{X});\ \mathbf{K}^{\dagger}[2]\right\} \\ &+ \mathrm{j} \cdot \left(\mathcal{K}_{\text{nc}}\left\{\Re(\mathbf{X});\ \mathbf{K}^{\dagger}[2]\right\} + \mathcal{K}_{\text{nc}}\left\{\Im(\mathbf{X});\ \mathbf{K}^{\dagger}[1]\right\}\right).\end{aligned} \tag{29}$$

Here, $\mathcal{K}_{\text{nc}}^{\dagger}\left\{\cdot;\ \mathbf{K}^{\dagger}\right\} : \mathbb{C}^{1\times P\times R} \to \mathbb{C}^{1\times P\times R}$ is the complex-valued non-causal convolution layer with kernel weights $\mathbf{K}^{\dagger} \in \mathbb{R}^{2\times 1\times \tau}$, and all four internal real-valued non-causal convolutions

[5]Note that the upright j denotes the imaginary unit, whereas the italic j denotes an index.

[6]To facilitate algorithmic implementation, this paper presents the derivation results in a convolution-first manner.

follow the padded definition in Eq. 25. For the initialization of $\mathbf{K}^\dagger$, we extend the definition in Eq. 27 as follows:

$$\begin{aligned} \text{Reference Init}^\dagger: \quad & \mathbf{K}^\dagger[1] \xleftarrow{\text{Ini.}} \Re(\mathbf{S}^*) \text{ or } \Re(\tilde{\mathbf{S}}^*); \\ & \mathbf{K}^\dagger[2] \xleftarrow{\text{Ini.}} \Im(\mathbf{S}^*) \text{ or } \Im(\tilde{\mathbf{S}}^*). \end{aligned} \tag{30}$$

So far, we have provided the basic definition of forward computation (Eq. 29) and the initialization method for learnable parameters (Eq. 30) of the proposed LMF. Because this basic form has only one output channel (a single convolution kernel), its representational capacity is limited. In our implementation, we therefore increase the number of output channels using multiple parallel chains, as shown in Fig. 1, and initialize the additional kernels with reference signals processed by different window functions, further enhancing the learnability of LMF. Deeper LMF variants can likewise be obtained by stacking multiple LMF layers.

A.2.3 Details of MLDC

As mentioned above, the purpose of MTI is to suppress stationary clutter near zero frequency, and a common implementation is a delay canceller applied along the slow-time (pulse) dimension. In fact, stationary clutter suppression is essentially the suppression of low-frequency components in the echoes. Therefore, the characteristic exhibited by the delay canceller in the frequency domain is that of a high-pass filter. The delay order δ of a delay canceller (*e.g.*, $\delta = 1$ for the single-delay canceller in Eq. 19 and $\delta = 2$ for the double-delay canceller in Eq. 20) influences the filter's high-pass response and low-frequency suppression. Theoretically, a higher δ results in improved suppression of stationary clutter and exhibits certain band-pass characteristics. However, a higher δ does not necessarily lead to better target detection. When δ is too large, it can increase sidelobe levels and degrade the response to moving targets. Therefore, *an appropriate order is difficult to determine from prior knowledge alone*.

Therefore, in order to introduce a more advanced and robust MTI-like filtering mechanism into EchMo, this work proposes the idea of using a multi-order learnable delay cancellation approach (MLDC). On the one hand, the original single MTI filter is expanded into a multi-order parallel structure to increase filtering diversity. On the other hand, a learnable mechanism is introduced, allowing the frequency domain characteristics of each filter to be adaptively adjusted based on the task and data. Let $\hat{\mathbf{X}} \in \mathbb{C}^{C_\text{K} \times P \times R}$ denote the output of an LMF module, $\mathcal{K}^\dagger_\text{nc}\{\cdot;\ \mathbf{K} \in \mathbb{R}^{2\times\tau\times 1\times C_\text{K}}\}$, as defined in Sec. 3.1, where C_K is the number of output channels. Then our MLDC would aim to process $\hat{\mathbf{X}}$ with multiple learnable filters with different frequency response characteristics, *i.e.*, learnable delay cancellers (LDCs) with different orders. For each LDC, we have the following definition:

$$\mathcal{H}^n_\text{caus}\left\{\hat{\mathbf{X}};\ \mathbf{H}^n\right\} = \left\{\sum_{c=1}^{C_\text{K}}\sum_{i=0}^{n}\mathbf{H}^n[c][i]\cdot\hat{\mathbf{X}}[c][j-i][r]\right\}_{j=n+1,r=1}^{P,R}. \tag{31}$$

The above computation is equivalent to causal convolution in WaveNet [35]: the n-order LDC $\mathcal{H}^n_\text{caus}\{\cdot;\ \mathbf{H}^n\}$ is implemented as a causal convolution layer with C_K input channels and, for clarity, one output channel, with kernel weights $\mathbf{H}^n \in \mathbb{R}^{C_\text{K}\times(n+1)\times 1}$. To give each LDC distinct frequency-response characteristics from the start of training, $\mathbf{H}^n \in \mathbb{R}^{C_\text{K}\times(n+1)\times 1}$ is initialized from the general coefficient solution $\{h^n_i\}_{i=0}^n$ of an n-order delay canceller:

$$\begin{aligned} \text{LDC Init:} \quad & \mathbf{H}^n[:][i] \xleftarrow{\text{Ini.}} h^n_i \cdot \mathbf{1}, \\ & \text{where} \quad h^n_i = \begin{cases} 1 & \text{if } i = 0; \\ (-1)^i\binom{n}{i} & \text{if } i = 1, \cdots, n, \ \&\ n \geq 1. \end{cases} \end{aligned} \tag{32}$$

where $\binom{n}{i} = \frac{n!}{i!(n-i)!}$ and $\mathbf{1} \in \mathbb{R}^{C_\text{K}}$ is an all-ones vector. When $n = 0$, $\mathcal{H}^0_\text{caus}$ degenerates into an identity mapping. As with LMF, we prepend each LDC with a $\texttt{Pad}^n_\text{caus}(\cdot) : \mathbb{R}^{P\times R} \to \mathbb{R}^{(P+n)\times R}$ operator so that the output pulse dimension matches the input while preserving causality.

In our implementation, each LDC module processes the input tensor in parallel, as shown in Fig. 2. Their outputs are concatenated along the channel dimension to obtain a more expressive echo

representation:

$$\begin{aligned}&\mathcal{H}^{*}_{\text{caus}}\left\{\{\texttt{Pad}^{n}_{\text{caus}}(\hat{\mathbf{X}})\}_{n=0}^{N};\ \{\mathbf{H}^{n}\}_{n=0}^{N}\right\}=\tilde{\mathbf{X}}^{0}\oplus\tilde{\mathbf{X}}^{1}\oplus\cdots\oplus\tilde{\mathbf{X}}^{N},\\&\text{where}\quad\tilde{\mathbf{X}}^{n}=\mathcal{H}^{n}_{\text{caus}}\left\{\texttt{Pad}^{n}_{\text{caus}}(\hat{\mathbf{X}});\ \mathbf{H}^{n}\right\},\ n=0,1,\cdots,N.\end{aligned}\tag{33}$$

Here, $\mathcal{H}^{*}_{\text{caus}}\left\{\cdot;\ \{\mathbf{H}^{n}\}_{n=0}^{N}\right\}:\mathbb{C}^{C_{\text{K}}\times P\times R}\to\mathbb{C}^{(N+1)\times P\times R}$ is the MLDC module constructed by $N+1$ parallel LDC branches. Moreover, the representational capability of the MLDC with $N+1$ branches can be further enhanced by simply extending the output channel from 1 to C_{H}.

A.2.4 Details of GDF

After range-domain target enhancement and stationary-clutter suppression, the radar system requires precise target velocity information, obtained through coherent integration (CI) of multiple pulses (Doppler filtering or MTD; Eq. 21). We design EchMo's group-perturbable Doppler filter (GDF) around two principles: preserving the essential computation and providing a good optimization starting point.

First, we design a filter bank that processes the multi-order inputs produced by the MLDC with N orders (C_{H} channels per order), with each group adapted to its corresponding input. Therefore, we borrow the concept of "group" from group convolution [15] and group normalization [38] to design the GDF, where each group handles the output, $\tilde{\mathbf{X}}^{n}\in\mathbb{C}^{C_{\text{H}}\times P\times R}$ from an n-order LDC. Specifically, each DF uses information from the entire slow-time dimension to capture target Doppler shifts across frequency bins:

$$\mathcal{D}^{n}\left\{\tilde{\mathbf{X}}^{n};\ \mathbf{D}^{n}\right\}=\left\{\sum_{c=1}^{C_{\text{H}}}\sum_{p=1}^{P}\tilde{\mathbf{X}}^{n}[c][p][r]\cdot\mathbf{D}^{n}[c][p][d]\right\}_{d=1,r=1}^{D,R},\ \text{where } n=0,\cdots,N.\tag{34}$$

Here, each DF, $\mathcal{D}^{n}\{\cdot;\ \mathbf{D}^{n}\}:\mathbb{C}^{C_{\text{H}}\times P\times R}\to\mathbb{C}^{D\times 1\times R}$ is a convolution layer with kernel weights of $\mathbf{D}^{n}\in\mathbb{C}^{C_{\text{H}}\times P\times 1\times D}$, where the receptive field covers the entire slow-time dimension and the output results of the D channels, *i.e.*, the Doppler dimension. Equation 34 is a multichannel extension of Eq. 21, and both share the same processing mechanism.

To provide a favorable optimization starting point, we initialize each $\mathbf{D}^{n}$ from an orthogonal Fourier basis:

$$\begin{aligned}\text{DF Init: }&\mathbf{D}^{n}[:][p][d]\xleftarrow{\text{Ini.}}e^{-\text{j}\cdot 2\pi dp/P}\cdot\mathbf{1},\\&\text{for } n=0,\cdots,N,\quad\text{and where }\mathbf{1}\in\mathbb{C}^{C_{\text{H}}}.\end{aligned}\tag{35}$$

Because each Fourier basis $e^{-\text{j}\cdot 2\pi dp/P}$ is complex-valued, whereas deep models implement complex-domain computation using real-valued operations (Eq. 6), we decompose it into real and imaginary parts using Euler's formula:

$$e^{-\text{j}\cdot 2\pi dp/P}=\cos(2\pi dp/P)-\text{j}\cdot\sin(2\pi dp/P).\tag{36}$$

Then, Eq. 35 can be rewritten as:

$$\begin{aligned}\text{DF Init: }&\mathbf{D}^{n\dagger}[1][p][d]\xleftarrow{\text{Ini.}}\mathbf{1}\cdot\cos(2\pi dp/P);\\&\mathbf{D}^{n\dagger}[2][p][d]\xleftarrow{\text{Ini.}}\mathbf{1}\cdot\sin(2\pi dp/P),\\&\text{for } n=0,\cdots,N,\quad\text{and where }\mathbf{1}\in\mathbb{R}^{C_{\text{H}}}.\end{aligned}\tag{37}$$

It can be seen that $\mathbf{D}^{n}\in\mathbb{C}^{C_{\text{H}}\times P\times 1\times D}$ is reorganized as $\mathbf{D}^{n\dagger}\in\mathbb{R}^{C_{\text{H}}\times 2\times P\times 1\times D}$. According to the complex-convolution rule in Eq. 29 and Euler's formula in Eq. 36, Eq. 34 is further extended as follows:

$$\begin{aligned}\mathcal{D}^{n\dagger}\left\{\tilde{\mathbf{X}}^{n};\ \mathbf{D}^{n\dagger}\right\}=&\mathcal{D}^{n}\left\{\Re(\tilde{\mathbf{X}}^{n});\ \mathbf{D}^{n\dagger}[:][1]\right\}+\mathcal{D}^{n}\left\{\Im(\tilde{\mathbf{X}}^{n});\ \mathbf{D}^{n\dagger}[:][2]\right\}\\&-\text{j}\cdot\mathcal{D}^{n}\left\{\Re(\tilde{\mathbf{X}}^{n});\ \mathbf{D}^{n\dagger}[:][1]\right\}+\text{j}\cdot\mathcal{D}^{n}\left\{\Im(\tilde{\mathbf{X}}^{n});\ \mathbf{D}^{n\dagger}[:][2]\right\}.\end{aligned}\tag{38}$$

Here, $\mathcal{D}^{n}\{\cdot;\ \mathbf{D}^{n}\}:\mathbb{R}^{C_{\text{H}}\times P\times R}\to\mathbb{R}^{D\times 1\times R}$ represents a standard real-valued convolution layer. Following the parallel multi-branch design of MLDC, we can further give the definition of our GDF

as follows:

$$\begin{aligned}&\mathcal{D}^{*\dagger}\left\{\{\tilde{\mathbf{X}}^n\}_{n=0}^N;\ \{\mathbf{D}^{n\dagger}\}_{n=0}^N\right\} = \dot{\mathbf{X}}^0 \oplus \dot{\mathbf{X}}^1 \oplus \cdots \oplus \dot{\mathbf{X}}^N,\\ &\text{where}\quad \dot{\mathbf{X}}^n = \mathcal{D}^{n\dagger}\left\{\tilde{\mathbf{X}}^n;\ \mathbf{D}^{n\dagger}\right\},\ n = 0, 1, \cdots, N.\end{aligned} \tag{39}$$

Fig. 3 gives the detailed computation principles of a GDF, where the application of Euler's formula in kernel initialization is omitted for clarity.

Relation to ADCNet's perturbed-DFT. The closest prior design to our GDF is the perturbed-DFT module of ADCNet [40], which also relaxes fixed Fourier bases for learning, yet the two designs differ in seven dimensions:

i. *radar modality*: chirp-based FMCW in ADCNet versus pulse-train PDR in EchMo;

ii. *perturbation purpose*: ADCNet perturbs the Fourier basis because an exact DFT is too rigid for distillation, while the GDF (Sec. 3.3) learns non-uniform frequency offsets for coherent integration;

iii. *initialization*: ADCNet adds Gaussian noise to the DFT matrix, while DF Init decomposes the Fourier basis via Euler's identity and initializes the real and imaginary kernel channels with cosine and sine components (Eq. 13);

iv. *module structure*: a single globally perturbed matrix with windowing versus group-perturbable filters, each processing one LDC-order output;

v. *control mechanism*: the noise variance at initialization versus the systematic ReCK protocol (Sec. 4.2);

vi. *training paradigm*: two-stage training (offline SP distillation plus task fine-tuning) versus a single-stage end-to-end redesign without offline pre-processing;

vii. *complex-valued handling*: separate real/imaginary processing versus complex convolution with joint I/Q processing.

In essence, ADCNet treats the classical pipeline as a *teacher* whose outputs bound the learnable ceiling, whereas EchMo treats it only as an *optimization starting point*, with performance bounded directly by the ground-truth supervision.

A.3 Going Deep for LMF

The Learnable Matched Filter (LMF) introduced in Sec. 3.1 provides a principled way to incorporate physical priors from classical matched filtering into a learnable framework. However, as we scale LMF to deeper architectures, two fundamental challenges emerge: i) maintaining the stability of prior-guided behavior in early layers while enabling adaptive capacity in deeper layers and ii) augmenting the convolutional inductive bias with complementary mechanisms for enhanced flexibility. This section presents two advanced LMF variants, LMF-V6 and ReXAttn-LMF-V6[7], which address these challenges through principled architectural innovations.

A.3.1 Prior-Residual Separation for Conv-based LMF Layers

Motivation. In complex electromagnetic environments, deep LMF architectures often exhibit two critical issues: i) the prior matched filter kernels in early layers may drift too rapidly during training, weakening the physical prior at the frontend; and ii) when deeper LMF blocks attempt to learn full mappings directly from initialization, early optimization can become unstable, manifesting as background collapse or insufficient foreground recall. To address these challenges, we propose a *layer-wise role separation* strategy that distinguishes between prior-preserving (in the first layer) and residual-learning layers (the remaining layers).

Architectural Design. The core innovation lies in assigning distinct roles to different LMF layers:

[7]It should be noted that both LMF variants presented in this section represent improvements upon LMF V5—the default version described in the main text of this paper. The version numbering does not start at V1 because EchMo has been under development for more than two years, during which we conducted multiple rounds of iterative development. We ultimately designated V5 as the default version of LMF. This nomenclature is intentionally aligned with the EchMo module in our PeCoRSS repository to facilitate usage and further development by other researchers.

† **Layer** 0 **(Prior Layer):** The first LMF layer operates in *prior mode*, where the kernel is initialized from the matched filter coefficients (Eq. 8) and undergoes direct training with a controlled learning rate. This layer serves as the stable frontend that preserves the theoretically optimal filtering behavior.

† **Layers** 1 **to** $L-1$ **(Residual Layers):** All subsequent LMF layers operate in *residual mode*, where kernels are initialized to zero and trained as *incremental compensation terms* rather than learning full mappings from scratch.

Mathematically, for a residual LMF layer with input echo $\mathbf{X}$ (here, we temporarily set aside the complex-domain processing components to focus exclusively on elucidating the underlying principles) and zero-initialized kernel $\Delta\mathbf{K}$, the forward pass is formulated as:

$$\mathbf{R} = \mathbf{X} + \alpha \cdot \phi\left(\mathcal{K}_{\text{nc}}^{\dagger}\{\mathbf{X};\ \Delta\mathbf{K}\}\right), \tag{40}$$

where $\phi(\cdot)$ denotes the complex activation function (*e.g.*, CReLU), and α is a tunable residual scaling factor (`residual_scale` in the implementation) that controls the contribution of the residual branch.

Prior Kernel Regularization. To prevent excessive deviation of prior-layer kernels from their initialized values, we introduce a prior regularization term that penalizes large deviations from the initial matched filter coefficients. Let $\mathcal{P}$ denote the set of prior-mode layers, and let $\mathbf{K}^{(m,0)}$ represent the initial kernel snapshot for the m-th prior layer. The prior regularization loss is defined as:

$$\mathcal{L}_{\text{prior}} = \sum_{m\in\mathcal{P}} \left(\|\mathbf{K}_{\text{real}}^{(m)} - \mathbf{K}_{\text{real}}^{(m,0)}\|_2^2 + \|\mathbf{K}_{\text{imag}}^{(m)} - \mathbf{K}_{\text{imag}}^{(m,0)}\|_2^2 \right). \tag{41}$$

The total training objective combines the segmentation loss with the prior regularization term:

$$\mathcal{L} = \mathcal{L}_{\text{seg}} + \lambda_{\text{prior}}(t) \cdot \mathcal{L}_{\text{prior}}, \tag{42}$$

where $\lambda_{\text{prior}}(t)$ is a time-varying regularization weight that adapts during training.

Three-Stage Curriculum Training. The optimization of the residual LMF follows a staged curriculum that progressively releases learning capacity. Let E denote the total training epochs, e the current epoch, and $p = e/(E-1)$ the normalized training progress. Define r_f as the freeze ratio and r_t as the transition ratio ($r_t \geq r_f$). The training proceeds through three stages as summarized in Table 6.

Table 6: Three-stage curriculum training strategy for residual Conv-based LMF.

Stage	**Progress**	**Prior LR**	**Residual LR**	$\boldsymbol{\lambda}_{\text{prior}}$	$\boldsymbol{\alpha}$
Stage 0: Freeze	$0 \leq p < r_f$	$s_{\text{prior}} = 0$	$s_{\text{res,start}}$	λ_{start}	α_{start}
Stage 1: Transition	$r_f \leq p < r_t$	s_{prior}	$s_{\text{res,start}} \to 0.7$	$\lambda_{\text{start}} \to \lambda_{\text{end}}$	$\alpha_{\text{start}} \to 0.5$
Stage 2: Fine-tune	$p \geq r_t$	s_{prior}	$0.7 \to s_{\text{res,end}}$	$\lambda_{\text{end}} \to 0$	$0.5 \to \alpha_{\text{end}}$

The stage-wise scheduling is formally defined as follows:

Stage 0 (Freeze Stage, $p < r_f$**):** The prior layer parameters are frozen ($s_{\text{prior}} = 0$) to protect the matched filter initialization from early gradient noise. The residual layers learn at a reduced rate $s_{\text{res,start}}$ with small gate value α_{start}.

Stage 1 (Transition Stage, $r_f \leq p < r_t$**):** Let $\tau_1 = (p - r_f)/(r_t - r_f)$. The prior layer is unfrozen with fixed learning rate scale s_{prior}. The residual parameters and gates are progressively increased:

$$\lambda_{\text{prior}} = \lambda_{\text{start}} + \tau_1(\lambda_{\text{end}} - \lambda_{\text{start}}), \quad \alpha = \alpha_{\text{start}} + \tau_1(0.5 - \alpha_{\text{start}}), \quad s_{\text{res}} = s_{\text{res,start}} + \tau_1(0.7 - s_{\text{res,start}}). \tag{43}$$

Stage 2 (Joint Fine-tuning Stage, $p \geq r_t$**):** Let $\tau_2 = (p - r_t)/(1 - r_t)$. The prior regularization decays to zero while residual capacity reaches its full potential:

$$\lambda_{\text{prior}} = \lambda_{\text{end}}(1 - \tau_2), \quad \alpha = 0.5 + \tau_2(\alpha_{\text{end}} - 0.5), \quad s_{\text{res}} = 0.7 + \tau_2(s_{\text{res,end}} - 0.7). \tag{44}$$

Key Differences from Standard LMF. Table 7 summarizes the fundamental distinctions between standard LMF (V5) and the residual variant (V6).

Practical Configuration. The default hyperparameter configuration for LMF-V6 is as follows: freeze ratio $r_f = 0.1$, transition ratio $r_t = 0.5$, prior regularization range $\lambda_{\text{start}} = 0.01$ to $\lambda_{\text{end}} = 0.0001$, prior learning rate scale $s_{\text{prior}} = 0.1$, residual learning rate scale range $s_{\text{res,start}} = 0.3$ to $s_{\text{res,end}} = 1.0$, and residual gate range $\alpha_{\text{start}} = 0.1$ to $\alpha_{\text{end}} = 1.0$.

Table 7: Comparison between standard version (LMF-V5) and residual variant (LMF-V6).

Aspect	LMF-V5	LMF-V6 (Residual)
Layer parameterization	Each layer learns full kernel	Role-separated: prior + residual
Kernel initialization	Unified MF initialization	Prior: MF; Residual: zero
Forward form	$\mathbf{y} = \phi(\mathcal{F}(\mathbf{x}; \mathbf{K}))$	$\mathbf{y} = \mathbf{x} + \alpha \cdot \phi(\mathcal{F}(\mathbf{x}; \Delta\mathbf{K}))$
Prior regularization	None	$\mathcal{L}_{\text{prior}}$ with staged scheduling
Training strategy	Standard optimization	Three-stage curriculum

A.3.2 Residual Cross-Attention for LMF

Motivation. While convolution-based LMF provides strong inductive bias from matched filtering theory, its local receptive field may limit adaptability to complex interference patterns. To augment the prior-guided convolution with enhanced flexibility while preserving its stability, we propose **ReXAttn-based LMF** (ReXAttn: **Re**sidual Cross-**Atten**tion), which introduces a lightweight cross-attention (as in CLIP [27]) residual branch alongside the main convolutional path.

Design Philosophy. The ReXAttn design follows two core principles: i) *prior-guided convolution as the stable backbone*: *i.e.*, the main convolutional path preserves the physical prior from matched filtering theory; and ii) *cross-attention residual for adaptive capacity*: *i.e.*, a lightweight attention branch provides complementary flexibility with bounded impact via controlled residual scaling.

Architecture Overview. The ReXAttn-based LMF maintains the same tensor interface as standard LMF-V5, ensuring compatibility with downstream MLDC, GDF, and PKC-based segmentation modules. The architecture consists of two parallel pathways:

† **Main Path (Unchanged):** The complex convolution from LMF-V6, including real-imaginary decomposition, convolutions, and recomposition into a complex output with complex activation.

† **Residual ReXAttn Path:** A lightweight cross-attention branch where queries are derived from prior kernel statistics, keys and values are projected from input echo features, and a gating-style attention map is generated and projected back to complex form.

Mathematical Formulation. Let $\mathbf{X} \in \mathbb{C}^{B \times C_{\text{in}} \times P \times R}$ denote the input echo tensor, where B is the batch size, C_{in} is the input channel dimension, P is the number of pulses, and R is the number of range bins. The baseline LMF output is:

$$\mathbf{Y}_{\text{lmf}} = \text{LMF-V6}(\mathbf{X}) \in \mathbb{C}^{B \times C_{\text{out}} \times P \times R}. \tag{45}$$

The ReXAttn branch computes cross-attention features as follows. First, the query is derived from prior kernel statistics $\mathbf{K}_{\text{prior}}$ (or current kernel parameters in residual mode):

$$\mathbf{Q} = \mathbf{W}_q \cdot \phi_q(\mathbf{K}_{\text{prior}}) \in \mathbb{R}^{B \times N_h \times d_k}, \tag{46}$$

where $\mathbf{W}_q$ is the query projection matrix, $\phi_q(\cdot)$ is a feature extraction function, N_h is the number of attention heads, and d_k is the key dimension per head.

The keys and values are projected from the input echo features:

$$\mathbf{K} = \mathbf{W}_k \cdot \psi_k(\mathbf{X}), \quad \mathbf{V} = \mathbf{W}_v \cdot \psi_v(\mathbf{X}), \tag{47}$$

where $\psi_k(\cdot)$ and $\psi_v(\cdot)$ are feature extraction functions that pool spatial information from the input, and $\mathbf{W}_k, \mathbf{W}_v$ are projection matrices.

The attention mechanism employs a lightweight sigmoid-gated interaction instead of heavy global softmax attention:

$$\mathbf{A} = \sigma\left(\frac{\mathbf{Q}\mathbf{K}^\top}{\sqrt{d_k}}\right) \in \mathbb{R}^{B \times N_h \times L_k}, \tag{48}$$

where $\sigma(\cdot)$ is the sigmoid activation and L_k is the key sequence length. The residual output is computed as:

$$\mathbf{R} = \mathbf{W}_o \cdot (\mathbf{A} \odot \mathbf{V}) \in \mathbb{C}^{B \times C_{\text{out}} \times P \times R}, \tag{49}$$

where $\mathbf{W}_o$ is the output projection matrix and $\odot$ denotes element-wise multiplication.

The final output combines both pathways with residual scaling:

$$\mathbf{Y} = \mathrm{CReLU}(\mathbf{Y}_{\mathrm{lmf}}) + \alpha_{\mathrm{rex}} \cdot \mathrm{CReLU}(\mathbf{R}), \tag{50}$$

where α_{rex} (`rexattn_scale` in the implementation) is the residual scaling factor that controls the contribution of the attention branch.

Shape Consistency and Safety. A critical requirement for the ReXAttn branch is strict shape alignment with the main LMF output. The implementation ensures:

i. Output shape of $\mathbf{R}$ exactly matches $\mathbf{Y}_{\mathrm{lmf}}$: $[B, C_{\mathrm{out}}, P, R]$.

ii. Channel consistency is verified through explicit shape checks in the ReXAttn branch.

iii. Query construction avoids mixing kernel-tap length with pulse length through careful dimension management.

Integration with EchMo. The ReXAttn-based LMF is integrated into EchMo through the `lmf_impl` configuration switch, which supports the following variants in the current PeCoRSS repository:

- `v5`: Standard LMF with unified MF initialization.
- `v6`: Residual Conv-based LMF with layer-wise role separation.
- `v6_rexattn`: ReXAttn-based LMF (serial chain only).
- `v7`: Advanced variant with extended capabilities [8].

Implementation Details. The ReXAttn module adds the following learnable projections:

- `q_proj`: Query projection from kernel statistics.
- `k_proj`: Key projection from input features.
- `v_proj`: Value projection from input features.
- `o_proj`: Output projection to residual feature space.

The ReXAttn hyperparameters include the attention dimension `rexattn_dim` (d_k) and the residual scale `rexattn_scale` (α_{rex}). These parameters are currently set as module defaults, with exposure in JSON configuration files planned for future iterations.

Expected Behavior and Summary. The ReXAttn-based LMF is designed to improve flexibility and robustness in challenging interference patterns while maintaining the stability of prior-guided convolution. The attention branch may introduce slight computational overhead compared to pure convolutional LMF due to additional projections and gating operations, but this trade-off is justified by enhanced adaptability in complex electromagnetic environments. Table 8 provides a comprehensive comparison of the LMF variants discussed in this section. In future work, we will discuss in greater detail the performance of pure convolutional implementations versus cross-attention implementations at the LMF stage, as well as their impact on EchMo as a whole.

Table 8: Summary of LMF variants for deep architectures.

Characteristic	**LMF-V5**	**LMF-V6**	**LMF-V6-ReXAttn**
Core operation	Complex convolution	Residual convolution	Conv + ReXAttn
Layer roles	Homogeneous	Prior + Residual	Prior + Residual
Initialization	All MF	MF + Zero	MF + Zero
Additional mechanism	None	Residual gating	Cross-attention
Training strategy	Standard	Curriculum	Curriculum
Key hyperparameters	LR, weight decay	$r_f, r_t, \lambda, \alpha$	$d_k, \alpha_{\mathrm{rex}}$

[8] See Appendix A.4.2 for details.

A.4 EchMo Acceleration for Scaling

To improve training and inference efficiency and enable larger-scale EchMo models on more extensive datasets with fewer computational resources, we accelerate each module's forward and backward computation. While the prior-guided design ensures strong inductive bias and optimization stability, the custom echo encoding frontend can introduce significant runtime overhead. This section presents a systematic acceleration strategy targeting each frontend component, achieving substantial speedups while preserving the learning semantics essential for training convergence.

A.4.1 Comprehensive Profiling and Bottleneck Analysis for Baseline

Baseline Setup. Before implementing the acceleration improvements, we systematically analyzed EchMo's computational cost. The hyperparameters and input settings of the baseline EchMo are listed in Table 9. The baseline is denoted as `EchMo-LMF-V6S-LDC-Ser-GDF-Cnv`, *i.e.*, EchMo with LMF-V6, serial MLDC, and convolution-based GDF.

Comprehensive Profiling. To analyze the computational bottlenecks of EchMo in detail, we systematically examined the baseline model's parameter scale, computational cost, data flow types, and peak GPU memory footprint as shown in Table 10. All experiments in this part are conducted on a single A100 (80GB) GPU. The conclusions under DDP (Distributed Data Parallel) are consistent with those from single-GPU runs.

Bottleneck Analysis. As shown in Table 11, considering both parameter count and computational cost, the latency bottlenecks for EchMo's forward computation and backward propagation are predominantly concentrated in the front-end LMF and MLDC modules. Consequently, these two modules warrant priority in acceleration optimization. Additionally, it is noteworthy that the backward propagation latency of the GDF module—relative to its actual computational cost—also exhibits substantial room for acceleration.

Acceleration Methodology. Generally, there are two primary approaches to accelerating modules or operators. The first involves optimizing and compiling operator computation paths at software layers closer to the hardware, such as our previous CUDA acceleration for the PKC operator. However, for EchMo's frontend components, the CUDA acceleration approach does not yield the greatest benefits. This is because, during the early design phase, we replaced the computations involved in each module with equivalent implementations *using mature, existing operators to maximize EchMo's generality and portability*—a key design highlight of EchMo. Consequently, we wish to avoid undermining this design philosophy during the acceleration process. In light of this, we turn to an alternative acceleration strategy: equivalent computation graph substitution. Through practical implementation, this approach can be distilled into two technical pathways: i) *more compact tensor multiply-add operations*, and ii) *more aggressive parallelization strategies*.

A.4.2 Packed Convolution and Chain-Level Parallelism for LMF

Motivation. As discussed in Sec. A.4.1, the LMF frontend is one of the dominant runtime hotspots in EchMo. The original LMF implementation suffered from two inefficiencies: i) complex convolution was emulated through four separate real-valued convolutions, incurring substantial kernel launch overhead; and ii) multiple LMF chains, designed as parallel feature heads semantically, were actually executed sequentially in loops. Therefore, building upon LMF-V6, we implemented the following modifications: packing convolutions to consolidate the four fragmented real-domain convolutions within complex-domain convolution operations (LMF-V7 serial, `LMF-V7S`), and employing a more explicit chain-level parallelism strategy to enhance the computational throughput of multi-chain LMF (LMF-V7 w/ Chain-Parallel, `LMF-V7P`).

Packed-Convolution Strategy. Recall from Eq. 29 that the original `LMF-V6` layer computes complex convolution through four real-valued convolutions:

$$\begin{aligned} &\mathcal{K}_{\text{nc}}\{\Re(\mathbf{X});\ \mathbf{K}^{\dagger}[1]\} - \mathcal{K}_{\text{nc}}\{\Im(\mathbf{X});\ \mathbf{K}^{\dagger}[2]\} \to \Re(\mathbf{Y}), \\ &\mathcal{K}_{\text{nc}}\{\Re(\mathbf{X});\ \mathbf{K}^{\dagger}[2]\} + \mathcal{K}_{\text{nc}}\{\Im(\mathbf{X});\ \mathbf{K}^{\dagger}[1]\} \to \Im(\mathbf{Y}), \end{aligned} \tag{51}$$

implemented as separate `RR`, `II`, `RI`, and `IR` convolution branches. `LMF-V7` reduces this to a single packed real convolution by concatenating real and imaginary inputs and constructing a packed kernel

Table 9: EchMo Baseline Configuration for Computation Profiling

Attribute	**Value**	**Tag in Repo.**
Model family	EchMo	`EchMo-LMF-V6S-LDC-Ser-GDF-Cnv`
Model scale	$\sim$15.7M	-
Input Specification		
Batch Size per GPU (B)	4	`batch_size`
Input channels (C)	1	`nb_input_channels`
Pulses (P)	128	`n_pulses`
Range bins (R)	1024	`n_range`
Input data type	`complex128`	`cmplx_flag=true`
Hyperparameters		
LMF chains (N_{chains})	8	`lmf_chains`
LMF layers/chain (N_{layers})	2	`lmf_layers`
LMF channels/chain (H_{lmf})	16	`lmf_channels`
LDC channels (C_{ldc})	128	`ldc_channels`
LDC orders (O_{ldc})	3	`ldc_orders`
GDF Doppler bins (N_{doppler})	128	`nb_doppler`
Baseline Implementation		
LMF	V6	`{lmf_impl:v6, lmf_chain_impl:serial}`
MLDC	Serial	`mldc_impl:serial`
GDF	Convolutional	`gdf_impl:conv`
FeatPack	Complex-to-real ops	`feat_pack:apiq`, `amp,angle,real,imag`

Table 10: Comprehensive Profiling Results for EchMo Baseline

Module	**#Params**	**Compute (GMAC)**	**Input DType**	**Compute DType**	**Output DType**	**Peak Mem FWD (MB)**	**Peak Mem BWD (MB)**
Norm.	0	N/A	`complex128`	`complex64`	`complex64`	24.52	32.00
LMF	1,745,664	457.481	`complex64`	`float32`	`complex64`	1061.92	304.94
MLDC	147,840	38.655	`complex64`	`float32`	`complex64`	7424.00	3072.00
GDF	12,583,680	25.770	`complex64`	`float32`	`complex64`	1590.00	5316.09
FeatPack	0	N/A	`complex64`	`complex64`	`float32`	36.00	74.25
PKC-Seg.	1,198,023	163.075	`float32`	`float32`	`float32`	1746.63	528.36
Loss	0	N/A	`float32`, `int64`	`float32`	`float32`	18.00	24.00
EchMo	**15,675,207**	684.981	`complex128`	`complex64`, `float32`	`float32`	**7940.50**	**1394.15**

Table 11: Per-Module Latency Profiling for EchMo Baseline

Module	**Forward Latency (ms/iter)**	**Backward Latency (ms/iter)**
Normalize	0.590	2.236
LMF	432.877	729.911
MLDC	770.636	101.250
GDF	18.842	45.802
FeatPack	0.205	1.006
PKC-Seg.	29.247	82.098
Loss	0.414	0.638
End-to-End	**1073.637**	**761.028**

bank. Let $\mathbf{X} \in \mathbb{C}^{B\times C_{\text{in}}\times P\times R}$ be the complex input and $\mathbf{K}^{\dagger} \in \mathbb{R}^{2\times C_{\text{in}}\times C_{\text{out}}\times \tau}$ be the kernel weights. The packed convolution computes:

$$\begin{bmatrix}\Re(\mathbf{Y})\\ \Im(\mathbf{Y})\end{bmatrix} = \mathcal{K}^{\dagger}_{\text{packed}}\left\{\begin{bmatrix}\Re(\mathbf{X})\\ \Im(\mathbf{X})\end{bmatrix}; \begin{bmatrix}\mathbf{K}^{\dagger}[1] & -\mathbf{K}^{\dagger}[2]\\ \mathbf{K}^{\dagger}[2] & \mathbf{K}^{\dagger}[1]\end{bmatrix}\right\}, \tag{52}$$

where $\mathcal{K}^{\dagger}_{\text{packed}}$ denotes the packed convolution operator that processes concatenated real-imaginary features through grouped convolution. This preserves the same complex-convolution mapping defined in Eq. 29 while significantly reducing kernel-launch overhead.

Chain-Level Parallelism. LMF Chain-Parallel extends the packed-conv strategy to all chains simultaneously using grouped convolution. Let K denote the number of LMF chains and C the channel count per chain. Each chain occupies one group, processing its own packed complex input slice. This transforms the execution model from:

- serial chains + serial over layers $\rightarrow$ parallel chains + serial only over depth.

Formally, given input $\mathbf{X} \in \mathbb{C}^{B\times 1\times P\times R}$, the tensor transformation proceeds as:

$$\mathbf{X} \rightarrow \mathbf{X}^{(K)} \in \mathbb{C}^{B\times K\times 1\times P\times R} \rightarrow \mathbf{Y}^{(K)} \in \mathbb{C}^{B\times K\times C\times P\times R} \rightarrow \mathbf{Y} \in \mathbb{C}^{B\times KC\times P\times R}, \tag{53}$$

where $\mathbf{X}^{(K)}$ denotes the input replicated across K chains, $\mathbf{Y}^{(K)}$ the per-chain outputs, and $\mathbf{Y}$ the concatenated final output.

Training Semantics Compatibility. The V7 refactor preserves all V6 learning semantics: layer 0 remains prior-guided with matched filter initialization, deeper layers remain residual-style correction blocks with zero initialization, and `residual_scale` remains available for controlling residual contribution. Consequently, `LMF-V7` maintains full curriculum compatibility with the `LMF-V6` training strategy described in Sec. A.3.1.

Configuration Options. Two configurable axes are introduced:

- **Layer Backend** (`lmf_impl`):
  - `v6`: Serial chain with four-conv execution as in Eq. 51.
  - `v7`: Serial chain with packed-conv execution as in Eq. 52.
- **Chain Execution Style** (`lmf_chain_impl`):
  - `serial`: Instantiate one `LMF_Chain` per chain.
  - `parallel`: Instantiate one `LMF_Chain_Parallel` with grouped convolution.

Recommended combinations include: `v6 + serial` (the original `LMF-V6S`), `v7 + serial` (layer-optimized version, `LMF-V7S`), and `v7 + parallel` (layer and chain optimized version, `LMF-V7P`).

Numerical Equivalence. Since radar echo signals are **physically sensitive to numerical precision**, computational numerical equivalence serves as a critical constraint in EchMo's acceleration modifications. Table 12 verifies numerical consistency across implementations. `LMF-V7S` is numerically almost identical to `LMF-V6S`, while `LMF-V7P` exhibits small differences consistent with operator reordering and fused convolution behavior.

Table 12: Numerical equivalence verification for LMF-V7 variants.

Comparison	Max Abs Err	Mean Abs Err	Rel. Mean Abs Err
`LMF-V6S` vs `LMF-V7S`	2.139e−4	1.390e−5	1.026e−6
`LMF-V6S` vs `LMF-V7P`	3.158e−2	3.955e−3	2.926e−4

Latency Impact. We evaluate the acceleration effects of the LMF-V7 series at both the module level (Table 13) and the end-to-end level (Table 14), while also monitoring their memory footprint (instantaneous throughput).

Summary for LMF Acceleration. The acceleration effects are evident: `LMF-V7P`, benefiting from convolution packing and enhanced parallel throughput, achieves a $4.499\times$ **speedup** in LMF module forward inference alongside modest backward propagation improvements. However, considering its

Table 13: LMF Acceleration Results: LMF-only Improvements

Variant	LMF Comp. (GMAC)	Latency (ms)		Peak Memory (MB)		Speedup vs Baseline	
		Forward	Backward	Forward	Backward	Forward	Backward
EchMo w/ `LMF-V6S`	457.481	432.877	729.911	1061.92	304.94	1.000x	1.000x
EchMo w/ `LMF-V7S`	457.481	128.560	525.574	1167.27	180.27	3.366x	1.389x
EchMo w/ `LMF-V7P`	457.481	96.200	463.049	3697.06	1850.53	**4.499x**	**1.576x**

Table 14: LMF Acceleration Results: End-to-End Improvements

Variant	End2End Comp. (GMAC)	Latency (ms)		Peak Memory (MB)		Speedup vs Baseline	
		Forward	Backward	Forward	Backward	Forward	Backward
Baseline w/ `LMF-V6S`	684.981	1073.637	761.028	7940.50	1394.15	1.000x	1.000x
Baseline w/ `LMF-V7S`	684.981	752.383	647.108	7940.00	1394.59	1.427x	1.176x
Baseline w/ `LMF-V7P`	684.981	719.276	588.888	7940.50	1394.58	**1.493x**	**1.292x**

inability to maintain strict numerical equivalence, we ultimately **designate `LMF-V7S` as the more cost-effective choice**. It delivers over $3.366\times$ **forward acceleration** for LMF while exhibiting **lower memory footprint during backward** propagation—thereby freeing up computational resources for optimization of other modules.

A.4.3 Kernel Packing for Parallel MLDC

Motivation. The original MLDC (`MLDC-Ser`) executes per-order loops, launching two convolutions per order (real and imaginary components) following Eq. 31. This design increases kernel launch overhead and memory traffic, making MLDC a significant latency bottleneck.

Parallelization Strategy. The parallel MLDC (`MLDC-Par`) preserves mathematical behavior while reducing the number of convolution launches. Recall from Eq. 31 that each order-n LDC implements a causal convolution with kernel $\mathbf{H}^n \in \mathbb{R}^{C_K \times (n+1) \times 1}$. The key insight is that order-n kernels have varying heights $k_n = n + 1$. By using a common kernel height $k_{\max} = N + 1$ (where N is the maximum order) and top-padding each order kernel to $k_{\max}$, all kernels can be concatenated into a single weight bank. This enables:

$$\text{Launches: } 2 \times (N+1) \rightarrow 2 \text{ convolutions.} \tag{54}$$

Formally, for the original order-n LDC output following Eq. 31:

$$\tilde{\mathbf{X}}^n = \mathcal{H}^n_{\text{caus}} \left\{ \texttt{Pad}^n_{\text{caus}}(\hat{\mathbf{X}});\ \mathbf{H}^n \right\}, \tag{55}$$

the parallel version constructs padded kernels $\tilde{\mathbf{H}}^n \in \mathbb{R}^{C_K \times k_{\max} \times 1}$ with top-padding offset $\Delta_n = k_{\max} - k_n$, where $\tilde{\mathbf{H}}^n[:, \Delta_n + j, :] = \mathbf{H}^n[:, j, :]$ and remaining entries are zero. A shared top-padded input $\texttt{Pad}^N_{\text{caus}}(\hat{\mathbf{X}})$ preserves the same valid convolution window and output length for all orders.

The `MLDC-Par` output is then computed as:

$$\mathcal{H}^*_{\text{parallel}} \left\{ \texttt{Pad}^N_{\text{caus}}(\hat{\mathbf{X}});\ \{\tilde{\mathbf{H}}^n\}_{n=0}^N \right\} = \tilde{\mathbf{X}}^0 \oplus \tilde{\mathbf{X}}^1 \oplus \cdots \oplus \tilde{\mathbf{X}}^N, \tag{56}$$

where $\oplus$ denotes channel-wise concatenation as in Eq. 33.

Numerical Equivalence. Table 15 verifies that `MLDC-Par` preserves computation. The remaining difference is at the floating-point accumulation level and is negligible for practical engineering use.

Table 15: Numerical equivalence of MLDC parallel implementation.

Comparison	Max Abs Err	Mean Abs Err	Rel. Mean Abs Err
`MLDC-Ser` vs `MLDC-Par`	3.052e−5	2.113e−7	9.440e−9

Latency Impact. Table 16 shows the dramatic latency reduction at the module level and the corresponding end-to-end speedup in Table 17.

Table 16: MLDC Acceleration Results: MLDC-only Improvements

Variant	MLDC Compute (GMAC)	Latency (ms)		Peak Memory (MB)		Speedup vs Baseline	
		Forward	Backward	Forward	Backward	Forward	Backward
`MLDC-Ser`	38.655	736.453	101.112	7424.00	3072.00	1.000x	1.000x
`MLDC-Par`	51.540	17.677	49.140	7436.75	3072.00	41.665x	2.057x

Table 17: MLDC Acceleration Results: End-to-End Improvements

Variant	End2End Comp. (GMAC)	Latency (ms)		Peak Memory (MB)		Speedup vs Baseline	
		Forward	Backward	Forward	Backward	Forward	Backward
Baseline w/ `MLDC-Ser`	684.981	1055.609	757.457	7940.50	1394.15	1.000x	1.000x
Baseline w/ `MLDC-Par`	697.866	504.872	712.314	7952.75	1393.43	2.091x	1.063x

Summary for MLDC Acceleration. The overall **speedup in forward computation is** $2.091\times$. `MLDC-Par` almost completely removes MLDC as the primary runtime bottleneck, reducing end-to-end forward latency from $\sim$1056ms to $\sim$505ms per iteration. More importantly, this speedup is achieved with virtually no loss in numerical precision (*i.e.*, numerical equivalence is maintained) and almost no additional memory overhead, rendering it a *painless acceleration solution*.

A.4.4 Einsum-Based Projection for GDF

Motivation. From Table 10 we can see that, in fact, the computational demand of GDF is not particularly high, at merely 25.77 GMAC; however, its peak memory footprint is noticeably higher than that of LMF and the PKC-Segmentation head, despite their greater computational loads. This discrepancy is particularly pronounced during the backward propagation phase. This stems from the fact that the original grouped complex-domain convolution implementation, while mathematically strictly aligned with the physical interpretation of coherent accumulation, lacks operational compactness and does not constitute an optimal computational pathway—temporal projection over the pulse dimension rather than spatial sliding-window convolution.

Root Cause Analysis. Recall from Eq. 34 and Eq. 38 that GDF computes Doppler projection through four grouped convolution terms. For group n with input $\tilde{\mathbf{X}}^n \in \mathbb{C}^{C_{\text{H}}\times P\times R}$ and kernels $\mathbf{D}^{n\dagger} \in \mathbb{R}^{C_{\text{H}}\times 2\times P\times 1\times D}$, the real and imaginary outputs are:

$$\begin{aligned}\Re(\dot{\mathbf{X}}^n) &= \mathcal{D}^n\{\Re(\tilde{\mathbf{X}}^n);\ \mathbf{D}^{n\dagger}[:][1]\} + \mathcal{D}^n\{\Im(\tilde{\mathbf{X}}^n);\ \mathbf{D}^{n\dagger}[:][2]\},\\ \Im(\dot{\mathbf{X}}^n) &= \mathcal{D}^n\{\Im(\tilde{\mathbf{X}}^n);\ \mathbf{D}^{n\dagger}[:][1]\} - \mathcal{D}^n\{\Re(\tilde{\mathbf{X}}^n);\ \mathbf{D}^{n\dagger}[:][2]\},\end{aligned} \tag{57}$$

where each $\mathcal{D}^n$ represents a grouped convolution with kernel size $(P, 1)$ and stride $(P, 1)$. While the forward pass is relatively efficient, backward propagation requires gradient kernels for all four convolutions, thereby introducing additional memory fragmentation. In this shape regime (full pulse projection), the grouped-convolution backward pass is poorly matched to the actual batched projection operation.

Einsum Reformulation. The `einsum` implementation expresses the same computation as direct tensor contractions over channel and pulse axes (C_{H}, P). Let $\tilde{\mathbf{X}}_r^n, \tilde{\mathbf{X}}_i^n \in \mathbb{R}^{B\times C_{\text{H}}\times P\times R}$ denote the real and imaginary parts of group-n input, and $\mathbf{D}_c^n, \mathbf{D}_s^n \in \mathbb{R}^{C_{\text{H}}\times P\times D}$ denote the cosine and sine kernel components derived from Eq. 37. The GDF output is computed as:

$$\Re(\dot{\mathbf{X}}^n)[b,d,r] = \sum_{c=1}^{C_{\text{H}}}\sum_{p=1}^{P}\Big(\tilde{\mathbf{X}}_r^n[b,c,p,r]\cdot\mathbf{D}_c^n[c,p,d] + \tilde{\mathbf{X}}_i^n[b,c,p,r]\cdot\mathbf{D}_s^n[c,p,d]\Big) + b_c^n[d] + b_s^n[d], \tag{58}$$

$$\Im(\dot{\mathbf{X}}^n)[b,d,r] = \sum_{c=1}^{C_{\text{H}}}\sum_{p=1}^{P}\Big(\tilde{\mathbf{X}}_i^n[b,c,p,r]\cdot\mathbf{D}_c^n[c,p,d] - \tilde{\mathbf{X}}_r^n[b,c,p,r]\cdot\mathbf{D}_s^n[c,p,d]\Big) + b_c^n[d] - b_s^n[d], \tag{59}$$

where $b_c^n, b_s^n \in \mathbb{R}^D$ are the cosine and sine bias terms. This formulation enables a more compact computational process for GDF, thereby improving memory utilization efficiency. The complete GDF

output is obtained by concatenating all groups:

$$\mathcal{D}^{*\dagger}_{\text{einsum}}\left\{\{\tilde{\mathbf{X}}^n\}_{n=0}^N;\ \{\mathbf{D}^{n\dagger}\}_{n=0}^N\right\} = \dot{\mathbf{X}}^0 \oplus \dot{\mathbf{X}}^1 \oplus \cdots \oplus \dot{\mathbf{X}}^N. \tag{60}$$

Numerical and Gradient Consistency. As shown in Table 18, compared with Conv-based GDF (`GDF-Cnv`), Einsum-Based GDF (`GDF-Ens`) maintains excellent numerical computational equivalence: **forward outputs are bitwise-identical**; **backward gradients are consistent up to expected floating-point accumulation noise**.

Table 18: Numerical equivalence: Conv-based GDF vs Einsum-based GDF.

Quantity	**Max Abs Err**	**Mean Abs Err**	**Rel. Mean Abs Err**
output	0.0	0.0	0.0
input gradient	0.0	0.0	0.0
cos kernel gradient	9.313e−9	9.354e−10	2.199e−7
sin kernel gradient	8.382e−9	9.262e−10	2.198e−7
cos bias gradient	0.0	0.0	0.0
sin bias gradient	0.0	0.0	0.0

Peak Memory Efficiency Improvement. Table 19 demonstrates the memory optimization effects of Einsum for GDF: approximately 50% **reduction in forward** computation and **over** 70% **reduction in backward** propagation. Simultaneously, it maintains comparable computational speed. Given that the computational latency of GDF itself is not substantial, the modest increase in forward latency is essentially negligible.

Table 19: GDF Improvement: Memory-Optimized Einsum Implementation

Variant	**GDF Comp.**	**Latency (ms)**		**Peak Memory (MB)**		**Speedup vs Baseline**	
	(GMAC)	Forward	Backward	Forward	Backward	Forward	Backward
`GDF-Cnv`	25.770	18.706	45.378	1590.00	5316.59	1.000x	1.000x
`GDF-Ens`	25.770	23.103	43.416	786.00	1560.00	0.810x	1.045x

Summary for GDF Improvement. For training larger-scale EchMo models with larger-scale data, the benefits arising from improved memory efficiency are more significant—enabling **larger batch sizes, thereby indirectly achieving training acceleration**. Hence, for throughput-oriented EchMo training, `gdf_impl = "einsum"` should be adopted as the default.

A.4.5 Comprehensive Profiling for Improved EchMo

Improved EchMo (EchMo†) Setup. Following the systematic module-by-module modifications and comparative experiments described above, we finalize the improved EchMo† as `EchMo-LMF-V7S-LDC-Par-GDF-Ens`, with the detailed setup in Table 20.

EchMo vs. EchMo†. Table 21 presents an end-to-end comparison between the baseline EchMo and the improved EchMo in metrics including computational cost, forward/backward latency, and GPU memory footprint. The conclusions are evident: after applying module-by-module optimizations to the front end, the improved EchMo achieves over $5\times$ **forward speedup** and $1.26\times$ **backward speedup**, while maintaining comparable computational cost and numerical equivalence. Additionally, a 62.38% reduction in peak GPU memory usage during backward propagation is achieved.

These improvements substantially enhance EchMo's resource utilization efficiency, enabling us to explore scaling behavior with reduced training time and computational requirements. Detailed profiling of EchMo† is also provided in Table 22 to maintain consistency with the description of the baseline.

A.5 Scaling Up EchMo

The ultimate goal of developing EchMo is to provide a **foundational architecture for radar echo perception** tasks, analogous to the role of *the Transformer in LLMs*. A critical prerequisite for such a foundational architecture is parameter scalability. In view of this, we investigate

Table 20: EchMo† Configuration for Computation Profiling

Attribute	Value	Tag in Repo.
Model family	EchMo-Improved	`EchMo-LMF-V7S-LDC-Par-GDF-Ens`
Model scale	∼15.7M	-
Input Specification		
Batch Size per GPU (B)	4	`batch_size`
Input channels (C)	1	`nb_input_channels`
Pulses (P)	128	`n_pulses`
Range bins (R)	1024	`n_range`
Input data type	`complex128`	`cmplx_flag=true`
Hyperparameters		
LMF chains (N_{chains})	8	`lmf_chains`
LMF layers/chain (N_{layers})	2	`lmf_layers`
LMF channels/chain (H_{lmf})	16	`lmf_channels`
LDC channels (C_{ldc})	128	`ldc_channels`
LDC orders (O_{ldc})	3	`ldc_orders`
GDF Doppler bins (N_{doppler})	128	`nb_doppler`
Baseline Implementation		
LMF	V7	`{lmf_impl:v7, lmf_chain_impl:serial}`
MLDC	Parallel	`mldc_impl:parallel`
GDF	Einsum	`gdf_impl:einsum`
FeatPack	Complex-to-real ops	`feat_pack:apiq, amp,angle,real,imag`

Table 21: End-to-End Comparison: Baseline EchMo vs. Improved EchMo†

Variant	End2End Comp. (GMAC)	Latency (ms)		Peak Memory (MB)		Speedup vs Baseline	
		Forward	Backward	Forward	Backward	Forward	Backward
EchMo	684.981	1041.676	746.699	7940.00	1394.54	1.000x	1.000x
EchMo†	697.865	204.889	589.958	7952.75	**524.61**	**5.083x**	**1.266x**

Table 22: Improved EchMo†: Forward-Backward Profiling Results

Module	Params	Compute (GMAC)	Input DType	Compute DType	Output DType	Peak Mem FWD (MB)	Peak Mem BWD (MB)
Normalize	0	N/A	`complex128`	`complex64`	`complex64`	24.52	32.72
LMF	1,745,664	457.481	`complex64`	`float32`	`complex64`	1166.36	180.64
MLDC	196,992	51.540	`complex64`	`float32`	`complex64`	7436.75	3072.00
GDF	12,583,680	25.770	`complex64`	`float32`	`complex64`	786.00	1559.61
FeatPack	0	N/A	`complex64`	`complex64`	`float32`	36.00	74.33
RDTranslator	1,198,023	163.075	`float32`	`float32`	`float32`	1746.27	528.36
Loss	0	N/A	`float32, int64`	`float32`	`float32`	18.00	24.00
End-to-End	**15,724,359**	**697.865**	`complex128`	`complex64, float32`	`float32`	**7952.75**	**524.61**

three complementary scaling strategies that extend EchMo along three orthogonal dimensions: depth, width, and temporal extent, while preserving the physical module semantics that underpin the framework's interpretability and optimization stability. To more rapidly investigate the scalability of EchMo under limited computational resources, we adopt the improved version EchMo† provided in Sec. A.4 as the basic backbone model. The configuration for the backbone EchMo† is in Table 23[9].

Table 23: Backbone EchMo† Configuration for Scaling

Attribute	Value	Tag in Repo.
Model family	EchMo-Improved	`EchMo-LMF-V7S-LDC-Par-GDF-Ens`
Model scale	9.3M	-
Input Specification		
Batch Size per GPU (B)	depends on GPU mem.	`batch_size`
Input channels (C)	1	`nb_input_channels`
Pulses (P)	128	`n_pulses`
Range bins (R)	1024	`n_range`
Input data type	`complex128`	`cmplx_flag=true`
Hyperparameters		
LMF chains (N_{chains})	8	`lmf_chains`
LMF layers/chain (N_{layers})	2	`lmf_layers`
LMF channels/chain (H_{lmf})	8	`lmf_channels`
LDC channels (C_{ldc})	64	`ldc_channels`
LDC orders (O_{ldc})	3	`ldc_orders`
GDF Doppler bins (N_{doppler})	128	`nb_doppler`
Baseline Implementation		
LMF	V7	`{lmf_impl:v7, lmf_chain_impl:serial}`
MLDC	Parallel	`mldc_impl:parallel`
GDF	Einsum	`gdf_impl:einsum`
FeatPack	Complex-to-real ops	`feat_pack:amp`, `amplitude only`

A.5.1 Deep Scaling via Multi-Resolution Stages

Motivation. When scaling EchMo to deeper architectures, simply stacking more layers within a single stage can lead to optimization instability and diminishing returns. The fundamental challenge is to increase model capacity while maintaining the stability of prior-guided behavior in the early processing stages. We propose **EchMoDeep**, a multi-resolution depth-scaling variant that introduces a second processing stage at reduced resolution, operating as a complementary refinement branch alongside the full-resolution anchor stage.

Architectural Design. EchMoDeep maintains the original physical-design modules (LMF, MLDC, GDF) unchanged within each stage, scaling model capacity by introducing a second stage at a different resolution. The architecture consists of two parallel branches:

† **Stage-1 (Anchor Branch):** A full-resolution EchMo† backbone that serves as the stable optimization anchor, preserving the theoretically optimal filtering behavior at the native input resolution.

† **Stage-2 (Refine Branch):** A reduced-resolution EchMo† backbone with independently scaled channels and chains, operating at lower computational cost while injecting complementary context.

Forward Formulation. Let $\mathbf{X} \in \mathbb{C}^{B\times 1\times P\times R}$ denote the complex echo input. The forward pass proceeds as follows:

Stage-1 full-resolution prediction:

$$\mathbf{Y}_1 = \mathcal{F}^{(1)}_{\text{EchMo}^\dagger}(\mathbf{X}) \in \mathbb{R}^{B\times K\times P\times R}, \tag{61}$$

where $\mathcal{F}^{(1)}_{\text{EchMo}^\dagger}$ denotes the stage-1 EchMo backbone with full resolution, and K is the number of segmentation classes.

[9]To further accelerate the scaling experiments, we only employ the amplitude operator, $|\cdot|$.

Stage-2 reduced-resolution prediction:

$$\mathbf{X}_2 = \mathcal{R}(\mathbf{X}), \quad \mathbf{Y}_2 = \mathcal{F}^{(2)}_{\text{EchMo}^\dagger}(\mathbf{X}_2), \tag{62}$$

where $\mathcal{R}(\cdot)$ denotes complex-valued bilinear resizing of the real and imaginary parts independently, mapping $\mathbf{X} \in \mathbb{C}^{B\times 1\times P\times R}$ to $\mathbf{X}_2 \in \mathbb{C}^{B\times 1\times P_2\times R_2}$ with $P_2 = \rho_p \cdot P$ and $R_2 = \rho_r \cdot R$, where ρ_p and ρ_r are the pulse and range scaling factors, respectively.

Upsampling and fusion:

$$\tilde{\mathbf{Y}}_2 = \mathcal{U}(\mathbf{Y}_2) \in \mathbb{R}^{B\times K\times P\times R}, \tag{63}$$

where $\mathcal{U}(\cdot)$ denotes bilinear upsampling to match the stage-1 output resolution. The final output combines both pathways through weighted summation and learnable logit fusion:

$$\mathbf{Y}_{\text{sum}} = \mathbf{Y}_1 + w_2 \cdot \tilde{\mathbf{Y}}_2, \tag{64}$$

$$\mathbf{Y}_{\text{fuse}} = \mathcal{K}_{1\times 1}\left\{[\mathbf{Y}_1, \tilde{\mathbf{Y}}_2];\ \mathbf{W}_{\text{fuse}}\right\}, \tag{65}$$

$$\mathbf{Y} = \mathbf{Y}_{\text{sum}} + \mathbf{Y}_{\text{fuse}}, \tag{66}$$

where w_2 is a learnable scalar weight (`stage2_weight`), $[\cdot,\cdot]$ denotes channel-wise concatenation, and $\mathcal{K}_{1\times 1}$ represents a 1×1 convolution with kernel $\mathbf{W}_{\text{fuse}}$ for logit fusion.

Stage-wise Internal Semantics. Each stage maintains the standard EchMo pipeline structure:

$$\mathbf{X} \xrightarrow{\text{LMF}} \hat{\mathbf{X}} \xrightarrow{\text{MLDC}} \tilde{\mathbf{X}} \xrightarrow{\text{GDF}} \dot{\mathbf{X}} \xrightarrow{\text{FeatPack}} \mathbf{Z} \xrightarrow{\text{PKC-Seg.}} \mathbf{Y}, \tag{67}$$

ensuring that EchMoDeep scales by *stacking stages at different resolutions* rather than replacing the LMF/MLDC/GDF design.

Scale Controls. Table 24 summarizes the key configuration parameters.

Table 24: EchMoDeep configuration parameters.

Parameter	Description	Usage	Value in Exp.
`deep_stage2_enable`	Enable/disable second stage	`true/false`	`true`
`deep_stage2_range_scale`	Stage-2 range scaling ρ_r	0.5 means $R_2 = 0.5R$	0.5
`deep_stage2_pulse_scale`	Stage-2 pulse scaling ρ_p	1.0 keeps P unchanged	1
`deep_stage2_chain_scale`	Stage-2 LMF chain scaling	Multiplier on N_{chains}	0.5
`deep_stage2_lmf_channel_scale`	Stage-2 LMF channel scaling	Multiplier on H_{lmf}	0.75
`deep_stage2_ldc_channel_scale`	Stage-2 MLDC channel scaling	Multiplier on C_{ldc}	0.75

Why It Scales Well. EchMoDeep offers two key advantages for deep scaling:

i. *Capacity increases through multi-resolution learning*, which both introduces behavior analogous to feature pyramids and enables more flexible compute/accuracy trade-offs.

ii. *Physical module semantics remain intact*, preserving interpretability and compatibility with the vanilla EchMo architecture.

A.5.2 Wide Scaling via Mixture of Experts

Motivation. As radar perception scenarios diversify—encompassing varying target types, clutter conditions, and interference patterns—a monolithic backbone may struggle to specialize effectively across all regimes. We propose **EchMo4MoE**, a structured width-scaling variant that employs a mixture-of-experts (MoE) architecture [31] where each expert is a full EchMo† backbone, enabling sample-dependent specialization while preserving the physical module semantics within each expert.

Architectural Design. EchMo4MoE scales model capacity via expert specialization with the following components:

† **Experts:** E parallel EchMo† backbones, each preserving the full LMF $\rightarrow$ MLDC $\rightarrow$ GDF $\rightarrow$ PKC-Segmenter pipeline.

† **Gate:** A lightweight statistics-driven routing network that predicts per-sample expert weights.

† **Fusion:** Top-k sparse soft weighting over expert logits for efficient computation.

Forward Formulation. Given input $\mathbf{X} \in \mathbb{C}^{B\times 1\times P\times R}$:

Expert logits:

$$\mathbf{Y}_e = \mathcal{F}_e(\mathbf{X}) \in \mathbb{R}^{B\times K\times P\times R}, \quad e = 1, \ldots, E, \tag{68}$$

where each $\mathcal{F}_e$ is an independent EchMo backbone.

Gate feature extraction: The gate derives features from complex-domain statistics:

$$\mathbf{g}(\mathbf{X}) = [\mu(|\mathbf{X}|),\ \sigma(|\mathbf{X}|),\ \max(|\mathbf{X}|),\ \mu(\Re(\mathbf{X})),\ \mu(\Im(\mathbf{X})),\ \sigma(\angle\mathbf{X})] \in \mathbb{R}^{B\times 6}, \tag{69}$$

where $\mu(\cdot)$, $\sigma(\cdot)$ denote mean and standard deviation computed over spatial dimensions, and $\angle\mathbf{X}$ denotes the phase angle.

Gate weights with temperature:

$$\mathbf{w} = \texttt{softmax}\left(\frac{\text{MLP}(\mathbf{g}(\mathbf{X}))}{\mathcal{T}}\right) \in \mathbb{R}^{B\times E}, \tag{70}$$

where $\mathcal{T}$ is the temperature parameter controlling routing sharpness.

Top-k sparsification and renormalization:

$$\tilde{\mathbf{w}} = \texttt{ReNorm}\left(\texttt{TopK_Mask}(\mathbf{w}, k) \odot \mathbf{w}\right), \tag{71}$$

where $\texttt{TopK_Mask}(\cdot, k)$ zeros out all but the top-k weights per sample, and $\texttt{ReNorm}(\cdot)$ renormalizes the remaining weights to sum to one.

Final fusion:

$$\mathbf{Y} = \sum_{e=1}^{E} \tilde{\mathbf{w}}_e \cdot \mathbf{Y}_e, \tag{72}$$

where $\tilde{\mathbf{w}}_e \in \mathbb{R}^{B\times 1\times 1\times 1}$ is broadcast across spatial and channel dimensions.

Expert Internal Semantics. Each expert maintains the standard EchMo pipeline:

$$\mathbf{X} \xrightarrow{\text{LMF}} \hat{\mathbf{X}} \xrightarrow{\text{MLDC}} \tilde{\mathbf{X}} \xrightarrow{\text{GDF}} \dot{\mathbf{X}} \xrightarrow{\text{FeatPack}} \mathbf{Z} \xrightarrow{\text{PKC-Seg.}} \mathbf{Y}_e, \tag{73}$$

ensuring that MoE scaling operates at the routing layer without replacing physical modules.

Key Configuration Parameters. Table 25 summarizes the key hyperparameters.

Table 25: EchMo4MoE configuration parameters.

Parameter	Description	Typical Range	Value in Exp.
`moe_num_experts`	Number of EchMo experts E	2–8	4
`moe_topk`	Number of active experts per sample k	1–E	2
`moe_gate_hidden`	Hidden width of gate MLP	32–128	32
`moe_temperature`	Routing softness temperature $\mathcal{T}$	0.1–1.0	1.0
`moe_dropout`	Gate dropout rate	0.0–0.1	0.0

Practical Considerations. Lower $\mathcal{T}$ produces sharper routing decisions; $k = 1$ tends toward hard expert selection while $k > 1$ enables smoother blending. Note that the current implementation computes all experts and fuses sparsely; future work may explore conditional computation for true sparse execution.

Why It Scales. EchMo4MoE offers three key advantages for wide scaling:

i. *Capacity grows with experts* while preserving LMF/MLDC/GDF inductive design in each expert backbone.

ii. *Routing enables sample-dependent specialization*, allowing different experts to handle distinct electromagnetic regimes.

iii. *Directly compatible* with the backbone training interface and output shape.

A.5.3 Temporal Scaling via Memory-Augmented Processing

Motivation. In many radar applications, the coherent processing interval (CPI) extends beyond a single frame, with targets exhibiting persistent signatures across multiple consecutive pulses. Processing each CPI frame independently ignores valuable temporal continuity. We propose **EchMoMemo**, a temporal-scaling variant that adds a lightweight memory state on top of the EchMo† backbone, enabling temporal accumulation capacity while keeping physical module semantics intact.

Architectural Design. EchMoMemo scales model capability along the CPI-time axis through:

† **Unchanged Backbone:** The core LMF $\rightarrow$ MLDC $\rightarrow$ GDF $\rightarrow$ PKC-Segmenter pipeline remains identical per frame.

† **Memory State:** Maintained in logit space across CPI frames via exponential moving average (EMA) [22].

† **Memory Injection:** The final output at each step combines current logits with a memory-based correction.

Forward Formulation. Given a CPI sequence input $\{\mathbf{X}_t\}_{t=1}^{T}$ with $\mathbf{X}_t \in \mathbb{C}^{B\times 1\times P\times R}$:

Backbone prediction per CPI:

$$\mathbf{Y}_t = \mathcal{F}_{\text{EchMo}^\dagger}(\mathbf{X}_t) \in \mathbb{R}^{B\times K\times P\times R}, \quad t = 1, \ldots, T. \tag{74}$$

EMA memory update:

$$\mathbf{h}_t = \beta \cdot \mathbf{h}_{t-1} + (1-\beta)\cdot \mathbf{Y}_t, \quad \mathbf{h}_1 = \mathbf{Y}_1, \tag{75}$$

where $\beta \in [0, 1]$ is the memory retention factor (`memo_beta`), controlling the balance between historical and current information.

Memory injection:

$$\hat{\mathbf{Y}}_t = \mathbf{Y}_t + \gamma \cdot \Phi(\mathbf{h}_t), \tag{76}$$

where:

- γ is the memory contribution strength (`memo_gamma`),
- Φ is either the identity mapping or a 1×1 projection (controlled by `memo_use_state_proj`).

The model returns the last-step output $\hat{\mathbf{Y}}_T$ as the final prediction.

Recurrent Formulation. The memory update can be expressed compactly in recurrent form:

$$\begin{aligned} \mathbf{h}_t &= \beta \cdot \mathbf{h}_{t-1} + (1-\beta)\cdot \mathcal{F}_{\text{EchMo}^\dagger}(\mathbf{X}_t), \\ \hat{\mathbf{Y}}_t &= \mathcal{F}_{\text{EchMo}^\dagger}(\mathbf{X}_t) + \gamma \cdot \Phi(\mathbf{h}_t). \end{aligned} \tag{77}$$

Input Shape Handling. EchMoMemo supports multiple input formats:

- $[B, T, P, R]$: Batch-first sequence format
- $[B, 1, T, P, R]$: Explicit channel dimension
- $[B, T, 1, P, R]$: Alternative channel placement

Each CPI frame is normalized to $[B, 1, P, R]$ before processing by EchMo†.

Key Configuration Parameters. Table 26 summarizes the key hyperparameters.

Why It Scales. EchMoMemo offers three key advantages for temporal scaling:

i. *Adds temporal capacity without changing LMF/MLDC/GDF internals*, preserving module interpretability.

ii. *Cost scales roughly with CPI count* while maintaining the same per-frame backbone computation.

Table 26: EchMoMemo configuration parameters.

Parameter	Description	Typical Range	Value in Exp.
`nb_input_channels`	The number of CPIs T	1–9	4
`memo_enable`	Enable/disable memory branch	`true/false`	`true`
`memo_beta`	EMA retention factor β	0.5–0.95	0.8
`memo_gamma`	Memory injection strength γ	0.1–1.0	0.4
`memo_use_state_proj`	Use 1×1 projection for Φ	`true/false`	`true`

Table 27: Summary of EchMo scaling strategies.

Strategy	Dimension	Core Mechanism	Key Hyperparameters	Prior Preserved
EchMoDeep	Depth	Multi-resolution stages	ρ_r, ρ_p, channel scales	Yes
EchMo4MoE	Width	Mixture of experts	$E, k, \mathcal{T}$	Yes
EchMoMemo	Temporal	Memory-augmented CPI	T, β, γ	Yes

Summary of Scaling Strategies. Table 27 provides a comparative overview of the three scaling strategies.

All three strategies preserve the core LMF/MLDC/GDF pipeline within their respective components, ensuring that the physical priors and interpretability of the base EchMo architecture remain intact regardless of scaling direction.

A.6 Dataset Details

Most existing radar datasets only provide signals after pulse compression; raw base-band echoes are rarely available. Furthermore, public radar datasets exhibit two key limitations: simulated data are often overly idealized and neglect real-world noise and interference, whereas real-world collected data typically lack precise annotations and diverse scenarios. To enable rigorous evaluation of our algorithm's robustness and generalization across environments, we constructed two echo datasets covering two representative scenarios faced by active radar systems: low-slow-small (LSS) target detection and target-jammer recognition under active electronic countermeasures:

- ⋆ **Syn-RED**: the **Syn**thetic **R**adar **E**cho **D**ataset. Syn-RED is constructed under the LSS target detection scenario. It uses precise electromagnetic simulation to generate controlled target scenes with varying motion characteristics and SNRs, ideal for ablation studies and initial performance screening.
- ⋆ **HiL-RED**: the **H**ardware-**i**n-the-**L**oop **RED**. HiL-RED is constructed for target-jammer recognition under active electronic countermeasures. It is built on a HiL platform that supports multiple radar jamming types and JSR levels, closely emulating practical scenarios.

Together, they form a progressive validation pipeline from ideal digital simulations to near-real-world scenarios, ensuring a comprehensive assessment of algorithm effectiveness and generalizability.

A.6.1 Syn-RED

Radar System Parameters. The radar system operates in the X-band with a fixed carrier frequency of $9.8\,\mathrm{GHz}$, transmitting an LFM coherent burst with a $16\,\mu\mathrm{s}$ pulse width, $8\,\mathrm{MHz}$ bandwidth, $100\,\mu\mathrm{s}$ PRI, and 128 pulses. The received signal is sampled at $10.24\,\mathrm{MHz}$.

Data Simulation Flow. Syn-RED employs a modular signal-level simulation pipeline to achieve high-fidelity modeling of the full radar signal generation chain. The data generation process comprises three core stages: geometric scenario calculation, target echo synthesis, and clutter and noise modeling.

i. **Geometric Scenario Calculation**. Based on a predefined scenario (e.g., target trajectories), the relative positions and motions of targets are first transformed into the radar coordinate system. A beam coverage check is then performed to determine which targets fall within the radar's transmit beam at each time step.

ii. **Target Echo Synthesis**. For each illuminated target, the echo signal is generated through a cascaded multiplicative process. This begins with baseband transmit waveform generation, producing a reference signal. This is followed by transmit antenna gain weighting based on the target's azimuth and elevation angles. The signal then undergoes RCS scaling to account for target reflectivity fluctuations, followed by two-way free-space path loss for range-dependent attenuation. Range-Doppler modulation introduces precise time delay and Doppler phase rotation according to the target's instantaneous radial velocity and range. Finally, receive antenna gain weighting completes the spatial filtering at the receiver.

iii. **Clutter and Noise Modeling**. To simulate realistic ground clutter, we employ a clutter simulation module based on the grid-mapping method: the illuminated area is partitioned into independent grids, scatterers within each grid are aggregated into equivalent point scatterers, and clutter is synthesized via coherent summation. Additionally, to model receiver characteristics, thermal noise and quantization noise are superimposed at the signal level, to reproduce the noise behavior of the receiver front end.

Range-Doppler Data Processing. The dataset captures the complete radar processing pipeline: raw complex baseband I/Q echoes, MF for range compression, MTI for clutter suppression, and MTD, which applies a slow-time Fourier transform to generate the complex RD spectrum. The final magnitude RD maps are derived directly from this complex RD output, preserving all intermediate representations for comprehensive analysis. Figure 9 shows intermediate results for a representative radar frame at each processing stage.

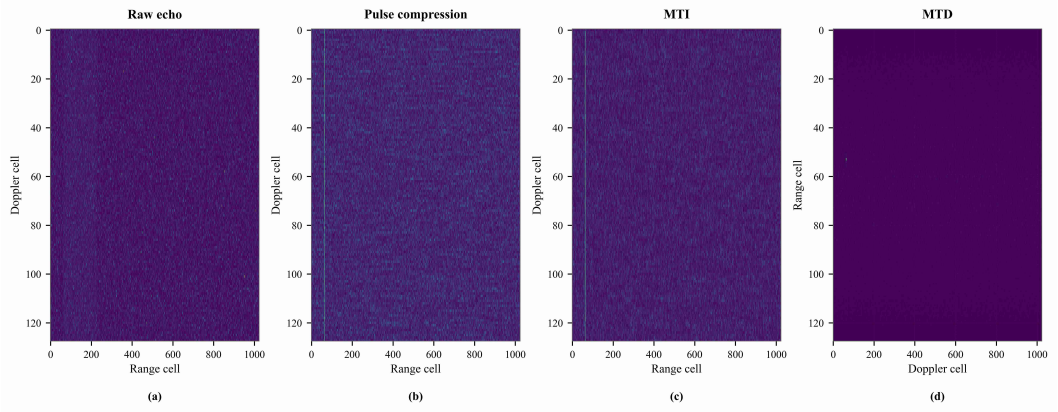


Figure 9: **Range-Doppler Data Processing**. Intermediate results at each stage for a representative radar frame.

Range-Doppler Data Annotation. Automatic dataset annotation leverages precisely computed ground truth from the simulation models. For each target, its position in the range-Doppler (RD) spectrum is determined by the range resolution $\Delta R = c/(2B)$ and Doppler resolution $\Delta v = \lambda/(2N \cdot \text{PRI})$, where c is the speed of light, B is the bandwidth, λ is the wavelength, N is the number of pulses, and PRI is the pulse repetition interval. Within the maximum unambiguous range $R_{\text{max}} = c \cdot \text{PRI}/2$ and velocity $v_{\text{max}} = \lambda/(4 \cdot \text{PRI})$, the range-bin index is computed as $r_{\text{bin}} = \lfloor r N_r / R_{\text{max}} \rfloor$ (equivalently $\lfloor 2 r N_r / (c \cdot \text{PRI}) \rfloor$), and the Doppler-bin index as $d_{\text{bin}} = \lfloor (v + v_{\text{max}}) N_d / (2 v_{\text{max}}) \rfloor \bmod N_d$, where N_r and N_d are the numbers of range and Doppler bins, respectively. Based on these coordinates, three types of annotations are generated to accommodate different algorithmic requirements: a 5×5 bounding box centered at $(d_{\text{bin}}, r_{\text{bin}})$ (stored in JSON format), a corresponding $N_d \times N_r$ binary mask for pixel-level dense annotation, and a single-point annotation marking only the peak bin. Figure 10 illustrates representative dataset annotations.

Scenario Design. The dataset spans target trajectory, detection range, and SNR, and comprises 27 independent sequences. Each sequence is defined as a unique combination of one trajectory type (constant velocity linear motion, S-turn maneuver, or orbit maneuver), one detection range (2 km, 3 km, or 5 km), and one SNR level (10 dB, 14 dB, or 18 dB). Target detection ranges cover typical medium-to-long-range radar surveillance scenarios. For brevity, we use Line, S, and O to denote constant-velocity, S-turn, and orbit trajectories, respectively, throughout the paper. Table 28 summarizes the detailed parameter configuration.

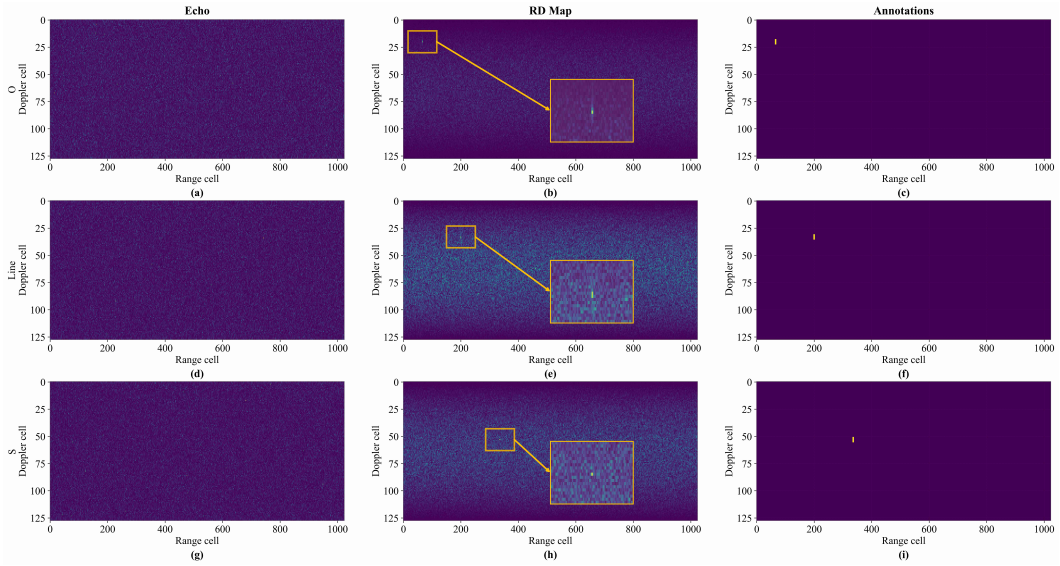


Figure 10: Syn-RED dataset examples for three maneuver types (O, Line, S). Columns: raw echo, RD map, annotation mask. Rows correspond to target trajectories, illustrating the annotation pipeline from raw data to pixel-level labels. In the annotation mask, different colors denote different classes. Purple: background, yellow: target.

Table 28: Parameter configuration of the Syn-RED dataset.

Line			S-Turn			Orbit		
Range	**SNR (dB)**	**Frames**	**Range**	**SNR (dB)**	**Frames**	**Range**	**SNR (dB)**	**Frames**
	10	1,000		10	1,000		10	1,000
5 km	14	1,000	5 km	14	1,000	5 km	14	1,000
	18	1,000		18	1,000		18	1,000
	10	1,000		10	1,000		10	1,000
3 km	14	1,000	3 km	14	1,000	3 km	14	1,000
	18	1,000		18	1,000		18	1,000
	10	1,000		10	1,000		10	1,000
2 km	14	1,000	2 km	14	1,000	2 km	14	1,000
	18	1,000		18	1,000		18	1,000
Total Configurations: 27								**Total Frames: 27,000**

A.6.2 HiL-RED

Radar System Parameters. HiL-RED employs a pulse width of $4\,\mu$s, 40 MHz bandwidth, $10.24\,\mu$s PRI, 100 MHz sampling rate, and 128 coherent pulses. While these differ from Syn-RED in pulse duration and bandwidth, the core processing chain remains consistent to ensure algorithmic comparability. The key distinction of HiL-RED lies in its hardware-in-the-loop architecture, comprising a Controller for system control and baseband signal exchange, and an IF Acquisition Card for intermediate frequency sampling and conversion.

Data Simulation Flow. As illustrated in Fig. 11, the HiL-RED platform operates through a closed-loop target-jammer co-simulation pipeline. The Controller generates baseband transmit signals, which the IF Acquisition Card upconverts to IF samples and feeds to the simulation modules. In the Target Simulation module, target RCS, time delay, Doppler shift, and amplitude attenuation are computed to generate target signals. Simultaneously, the Jamming Simulation module executes cognitive countermeasures: parameter measurement, intra-pulse feature extraction, emitter sorting and correlation, threat assessment, strategy matching, and jamming pattern selection (e.g., DRFM delay or repeat noise modulation). Target and jamming signals are digitally synthesized with noise, and the composite IF echo is fed back to the IF Acquisition Card for downconversion, yielding baseband digital echo data.

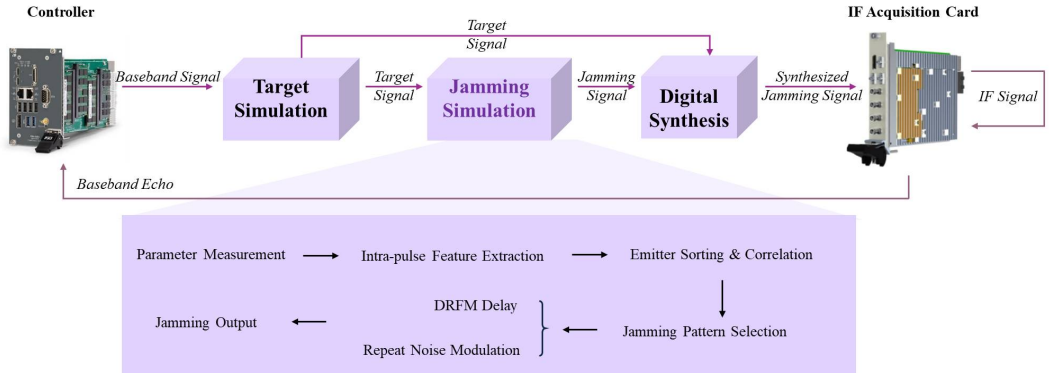


Figure 11: HiL-RED hardware-in-the-loop simulation architecture.

Jamming Pattern Modeling. Five jamming patterns are implemented based on cognitive electronic countermeasure strategies. The mathematical models for each jamming type are described as follows.

i. **Narrowband Barrage Jamming (NBJ).** NBJ generates narrowband Gaussian noise centered at the radar carrier frequency f_c with bandwidth B_j slightly wider than the radar bandwidth B to accommodate frequency measurement errors. The jamming signal is modeled as:

$$j_{\text{NBJ}}(t) = \sqrt{P_j} \cdot n(t) \cdot \exp(\text{j}2\pi f_c t), \tag{78}$$

where P_j is the jamming power, and $n(t)$ is complex Gaussian noise with bandwidth B_j.

ii. **Wideband Barrage Jamming (WBJ).** WBJ generates wideband Gaussian noise that covers the entire operational radar band with bandwidth B_j much larger than the radar bandwidth B (typically $B_j \gg B$, where B is the instantaneous radar bandwidth), ensuring jamming regardless of the radar's specific operating frequency within the band. The jamming signal is modeled as:

$$j_{\text{WBJ}}(t) = \sqrt{P_j} \cdot n(t) \cdot \exp(\text{j}2\pi f_c t), \tag{79}$$

where P_j is the total jamming power spread over bandwidth B_j, and $n(t)$ is complex Gaussian noise with flat power spectral density across B_j.

iii. **Dense False-Target Jamming (DFTJ).** DFTJ generates multiple delayed and amplitude-scaled replicas of the intercepted radar signal to create false targets. The jamming signal is:

$$j_{\text{DFTJ}}(t) = \sum_{i=1}^{N} \alpha_i \cdot s(t - \tau_i) \cdot \exp(\text{j}\phi_i), \tag{80}$$

where N is the number of false targets, α_i is the amplitude scaling factor, τ_i is the time delay for the i-th false target, ϕ_i is the random phase, and $s(t)$ is the intercepted radar signal.

iv. **Interrupted-Sampling Repeater Jamming (ISRJ).** ISRJ samples the radar signal intermittently and immediately repeats the sampled segments. The jamming signal is constructed as:

$$j_{\text{ISRJ}}(t) = \sum_{p=1}^{P} g(t - p \cdot T_{\text{PRT}}) \cdot s(t - p \cdot T_{\text{PRT}} - \tau_d) \cdot \exp(\text{j}\phi(t)), \tag{81}$$

where $g(t)$ is the gating function with duty cycle $\eta = T_g / T_{\text{PRT}}$ (sampling duration T_g over pulse repetition time T_{PRT}), P is the number of coherent pulses, τ_d is the processing delay, and $\phi(t)$ is the phase modulation.

v. **Slice Jamming (SIJ).** SIJ slices the intercepted pulse into multiple sub-pulses and reassembles them with modified timing. The jamming signal is:

$$j_{\text{SIJ}}(t) = \sum_{m=1}^{M} \text{rect}\left(\frac{t - m \cdot T_s - \tau_m}{T_w}\right) \cdot s(t - \tau_m) \cdot \exp(\text{j}\phi_m), \tag{82}$$

where M is the number of slices, T_s is the slicing interval, T_w is the sub-pulse width, τ_m is the delay for the m-th slice, and $\text{rect}(\cdot)$ denotes the rectangular window function.

Table 29: Jamming Parameter Configuration in HiL-RED.

Barrage Jamming				
Jamming Type	**Bandwidth**	**Center Freq.**	**Mod. Depth**	**Noise Type**
WBJ	80 MHz	0 MHz	1	Gaussian
NBJ	50 MHz	0 MHz	1	Gaussian
Deceptive Jamming				
Jamming Type	**Parameter**	**Setting**		
DFTJ	Number of targets	3		
	Target interval	0.5 μs		
ISRJ	Sampling time	0.5 μs		
	Sampling interval	1 μs		
	Repeat times	1		
SIJ	Slice width	0.5 μs		
	Slice period	1 μs		

Jamming Parameter Configuration. Table 29 summarizes the specific parameter settings for each jamming type in HiL-RED.

Range-Doppler Data Annotation. For HiL-RED, we employ the same data preprocessing and automatic annotation pipeline as Syn-RED, and thus omit the details here. Figure 12 illustrates representative dataset annotations.

Scenario Design. HiL-RED is constructed for radar target detection and jamming analysis scenarios. In addition to target echoes, it contains typical active jamming data, covering two categories: deceptive jamming and barrage jamming, with five types: DFTJ, ISRJ, SIJ, NBJ, and WBJ. For each jamming type, three JSR levels are set: -20 dB, -15 dB, and -10 dB. Each **sequence**, defined as a unique combination of jamming type and JSR level, comprises 10,000 frames, yielding a total of 150,000 frames. We uniformly sample 20% per sequence to construct **HiL-RED30K** for ablation studies and comparisons with milestone architectures, while the full HiL-RED is used for scaling experiments. Table 30 summarizes the detailed parameter configuration.

Table 30: Parameter configuration of the HiL-RED dataset.

Type	**JSR (dB)**	**Frames (CPIs)**	**Type**	**JSR (dB)**	**Frames (CPIs)**	**Type**	**JSR (dB)**	**Frames (CPIs)**
DFTJ	-20	10,000	ISRJ	-20	10,000	SIJ	-20	10,000
	-15	10,000		-15	10,000		-15	10,000
	-10	10,000		-10	10,000		-10	10,000
NBJ	-20	10,000	WBJ	-20	10,000			
	-15	10,000		-15	10,000		—	
	-10	10,000		-10	10,000			
Total Configurations: 15						**Total Frames: 150,000**		

An overview of the Syn-RED and HiL-RED datasets, including maneuver patterns, SNR/JSR distributions, range coverage, and jamming type distributions, is illustrated in Fig. 13.

A.6.3 Dataset Split Strategy

Both Syn-RED and HiL-RED employ **within-sequence** train/val/test splits. A *sequence* is defined as a contiguous collection of CPIs generated under fixed scenario settings: for Syn-RED, this means a unique combination of trajectory type, detection range, and SNR; for HiL-RED, a unique combination of jamming type and JSR level. Within each sequence, contiguous CPI ranges are assigned to the training, validation, and test partitions, ensuring **strictly disjoint CPIs** while retaining the same scenario settings across splits. The train/validation/test proportions are $70\%/20\%/10\%$ for Syn-RED and $70\%/10\%/20\%$ for HiL-RED.

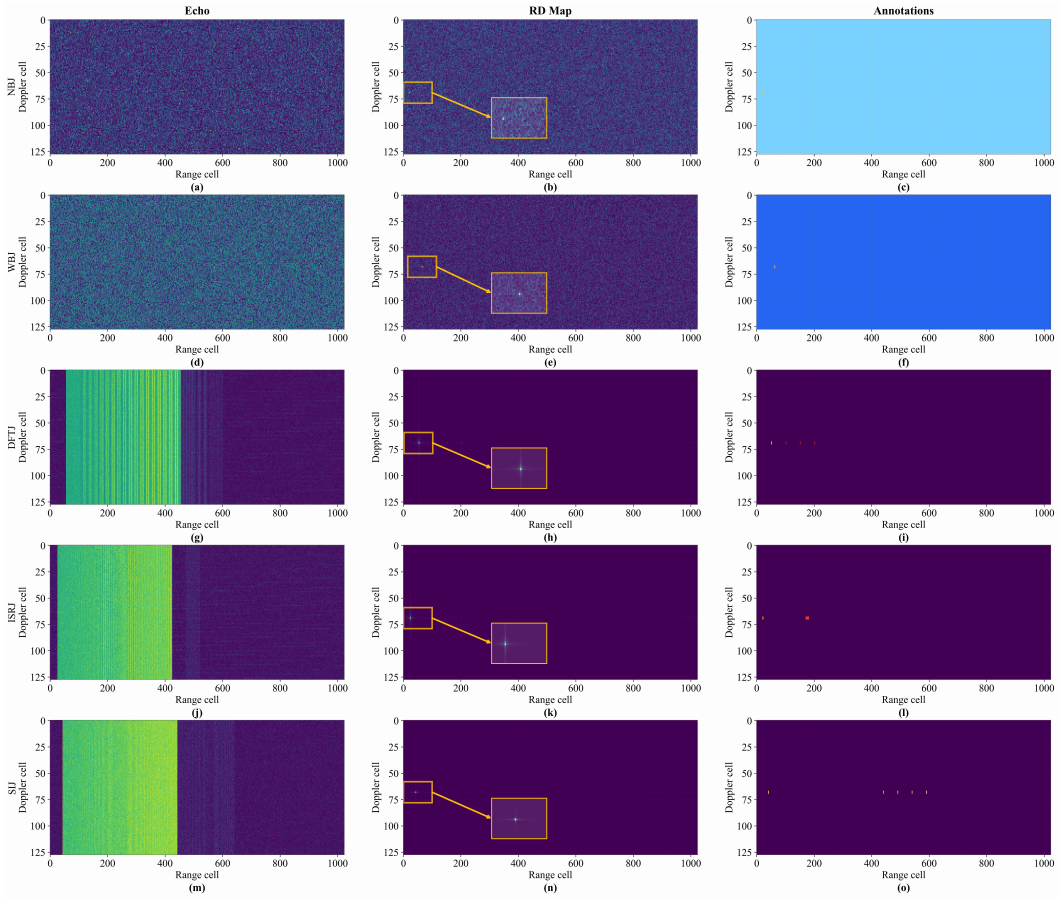


Figure 12: HiL-RED dataset visualization of five jamming types. **Columns**: raw echo, RD map, annotation mask. **Rows** (top to bottom): NBJ, WBJ, DFTJ, ISRJ, SIJ. In the annotation mask, different colors denote different classes. Purple: background, yellow: target, sky blue: NBJ, dodger blue: WBJ, deep red: DFTJ, red: ISRJ, light red: SIJ.

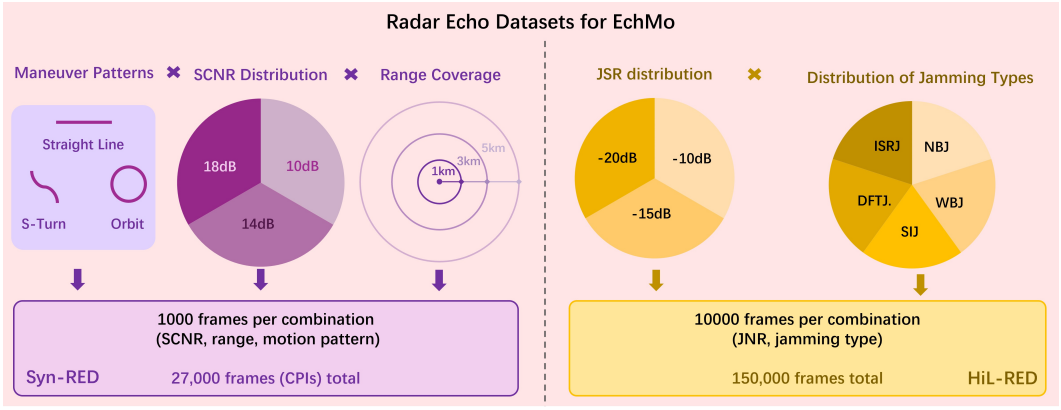


Figure 13: Radar echo datasets for EchMo: Syn-RED (left) with maneuver patterns, SNR distribution, and range coverage; HiL-RED (right) with JSR distribution and jamming type distribution.

Rationale. Within-sequence splitting is appropriate for the current study because radar echoes are highly stochastic. Even within a fixed SNR/JSR condition, individual CPIs exhibit random fluctuations in noise, clutter, and interference. Our primary goal is to validate whether EchMo can solve classical PD radar detection challenges (low-RCS small targets and anti-jamming) that conventional pipelines struggle with. Condition-level hold-out (e.g., excluding entire jamming types or SNR levels from training) would test few-shot generalization and is reserved for future work.

A.7 Supplementary Description of ReCK

The core idea of EchMo is to incorporate the expert prior knowledge in classical RSP into learnable network modules through computational flows and optimization starting points. It is therefore crucial to examine how this idea affects radar-specific deep learning, particularly how prior knowledge constrains the optimization of modern models. To this end, we introduce the concept of "**Re**laxation of **C**onstraint from Prior **K**nowledge (ReCK)". ReCK provides a unified ablation method for each carefully designed EchMo module. It defines four levels of progressively weaker prior-knowledge constraints:

† **ReCK-0**: the most tightly constrained learning strategy for the designed modules. For a module under investigation (*e.g.*, LMF, MLDC, or GDF), prior initialization is followed by parameter freezing. Thus, during EchMo training, the module parameters receive no backpropagation updates (equivalently, their gradients remain 0), as shown in Fig. 4-(a). ReCK-0 represents the lower bound of constraint relaxation: the module strictly follows both the prior computational flow and the prior parameters. It therefore isolates the effect of a specific prior under maximal trust in that prior.

† **ReCK-1**: a cautiously constrained update mode for the designed modules. After prior initialization, a given module is fine-tuned with a reduced learning rate $\eta^* = \alpha\eta$, where $0 < \alpha < 1$ and η is the global learning rate for the current EchMo epoch. Figure 4-(b) illustrates ReCK-1. This setting assumes that the prior initialization is close to, but not necessarily identical to, the desired solution.

† **ReCK-2**: a fully fine-tuned learning strategy for the designed modules and the default mode for EchMo. After prior initialization, all modules are trained with the global learning rate, as illustrated in Fig. 4-(c). This setting treats the network as a whole and allows it to adapt to the training data with a consistent learning rate while retaining beneficial prior parameter configurations.

† **ReCK-3**: the most relaxed constraint learning strategy for the designed modules. In this case, the module under investigation inherits only the prior computational flow, while its parameters are randomly initialized, as shown in Fig. 4-(d). This mode places the least trust in the prior and serves as the ablation control for prior-initialized modules.

A.8 Additional Experimental Results

A.8.1 Intermediate Feature Visualizations

To illustrate EchMo's behavior, we visualize intermediate outputs alongside conventional RSP (Figs. 14–19). On Syn-RED, EchMo modules exhibit structural correspondence with conventional LMF, MTI, and MTD operations, with refined feature clarity in select cases, confirming the incorporation and subtle enhancement of physical priors. On HiL-RED30K, conventional RSP exhibits limited capability: under deceptive jamming, targets and interfering signals are both detected yet remain indistinguishable, whereas under barrage jamming, the output is almost entirely overwhelmed by noise, leading to near-complete detection failure. In contrast, EchMo maintains distinct target and interference resolution across both scenarios. Notably, while EchMo's intermediate feature maps appear visually noisier than conventional counterparts under barrage jamming, this reflects the network's learned sensitivity to diverse jamming patterns; indeed, these seemingly cluttered representations encode richer discriminative information. This confirms that EchMo achieves practical robustness not through heuristic filtering, but through adaptive, data-driven parameter modulation that generalizes across jamming types.

A.8.2 Visualization and Analysis of Learned GDF Kernels

Fig. 20 visualizes the kernel weights of representative GDF frequency channels. In each subpanel, yellow dashed arrows depict individual sample vectors $(\cos_w[i], \sin_w[i])$ along the slow-time dimension, while the solid purple arrow indicates their vector sum $\Sigma = (\sum \cos_w, \sum \sin_w)$. For

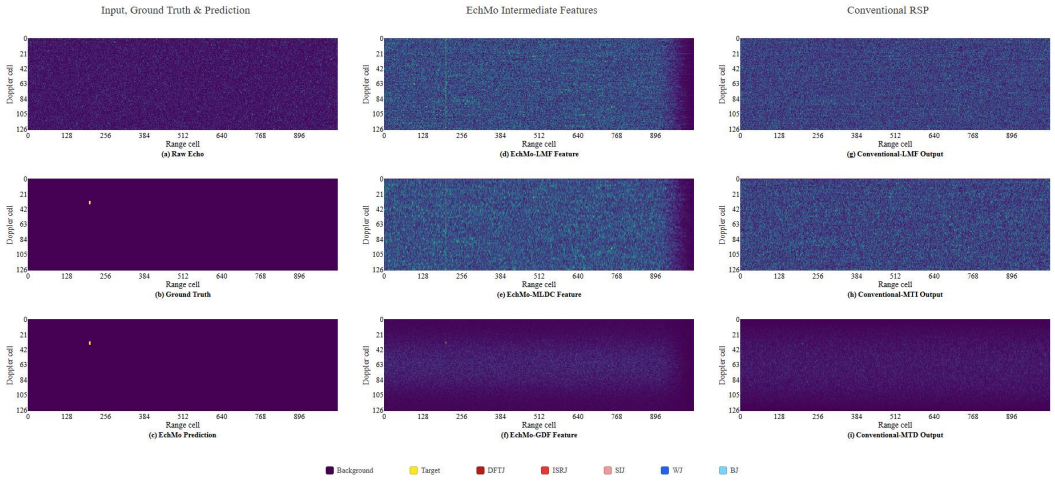


Figure 14: Intermediate outputs of EchMo modules and conventional RSP on Syn-RED.

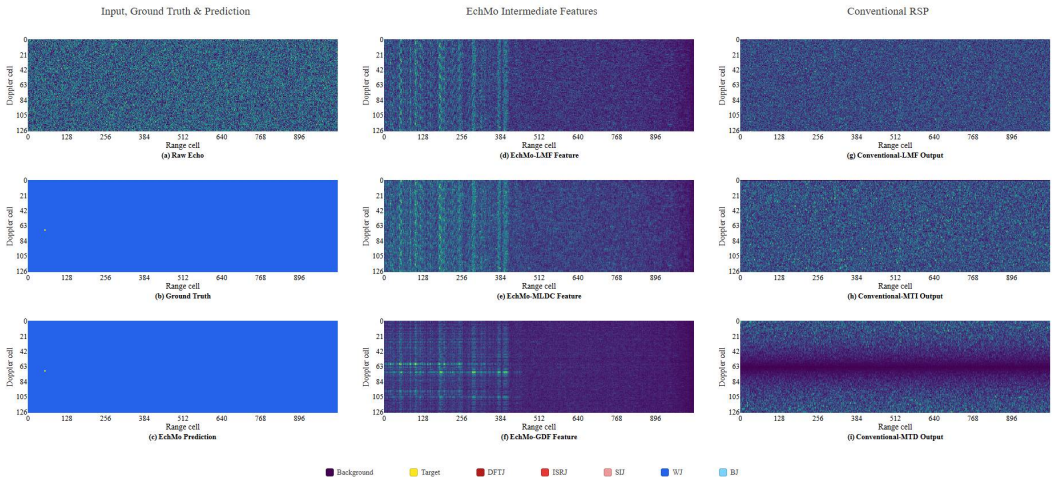


Figure 15: Intermediate outputs of EchMo modules and conventional RSP on HiL-RED under WBJ.

learned kernels, sample vectors are uniformly scaled when their maximum magnitude falls below 0.95 for visual clarity, while the vector sum remains unscaled and thus reflects the true net deviation.

The top row shows the ideal Fourier basis: except for the Nyquist channel $f = 128$ where all phases align at π to yield $\Sigma = (128, 0)$, the remaining channels exhibit symmetrically distributed vectors on the unit circle, producing near-zero sums due to complete phase cancellation. The middle row (Syn-RED) reveals only mild deviations concentrated at low frequencies ($f = 1, 16$), with higher frequencies rapidly reconverging to the prior ($f \geq 32$ yields $\Sigma \approx \mathbf{0}$, while $f = 128$ reaches 123.44). This close adherence to the ideal basis is expected because Syn-RED is generated under controlled electromagnetic simulation with prescribed SNRs, where target echoes closely follow ideal signal models and the physical Fourier prior remains largely valid.

The bottom row (HiL-RED), however, displays pronounced non-zero vector sums across almost all frequency channels—for instance, $\Sigma = (-1.26, -1.18)$ at $f = 32$ and $(-0.30, 1.24)$ at $f = 80$. This substantial departure directly corresponds to the far greater complexity of HiL-RED, which involves hardware-in-the-loop active jamming, deceptive targets, and complex radio-frequency interactions that violate the idealized assumptions underlying classical DFT. Faced with such severe model mismatch, the network must substantially redistribute spectral energy across channels, producing the

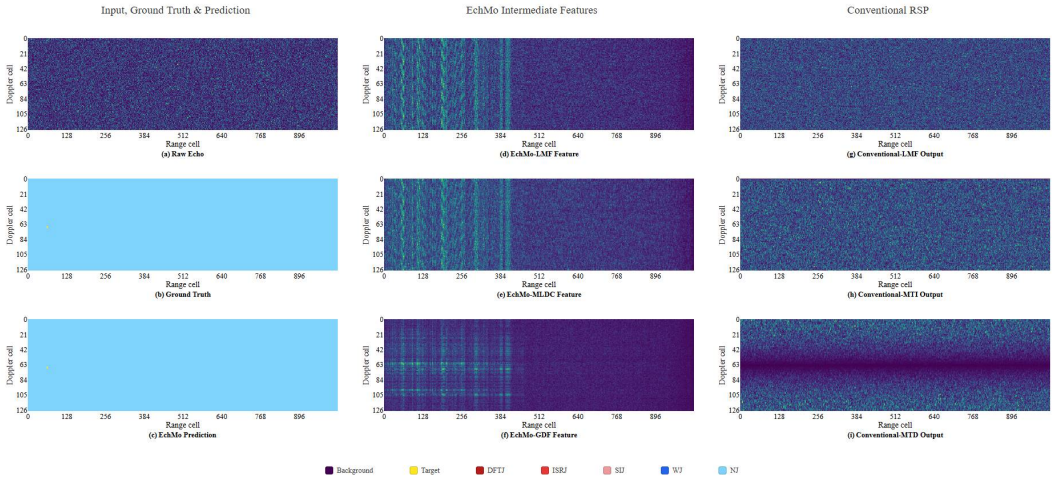


Figure 16: Intermediate outputs of EchMo modules and conventional RSP on HiL-RED under NBJ.

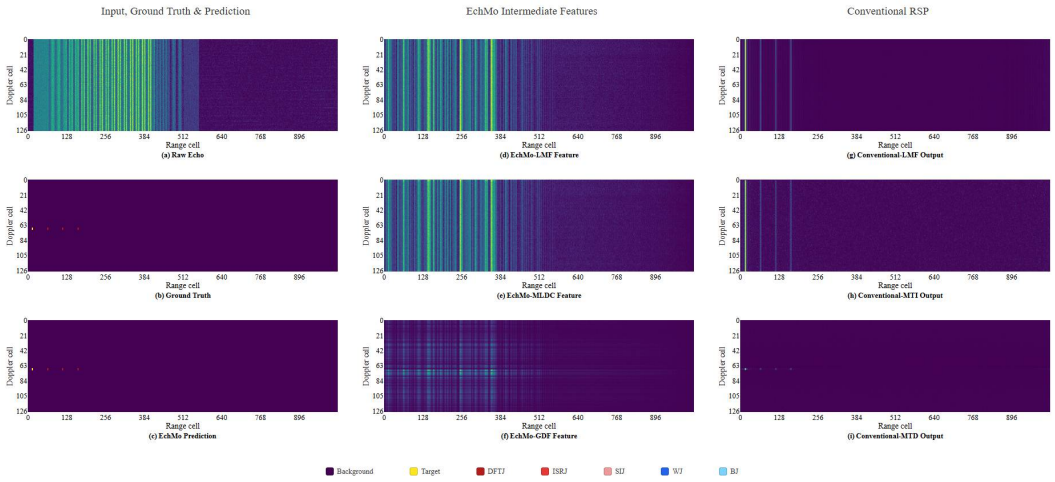


Figure 17: Intermediate outputs of EchMo modules and conventional RSP on HiL-RED under DFTJ.

observed non-orthogonal geometry. In this sense, the degree of kernel deviation from the ideal basis serves as a direct, interpretable indicator of how severely the actual data distribution departs from the physical assumptions embedded in the prior.

Importantly, this visible departure from orthogonality does not imply information loss or aliasing. GDF adopts an inherently overcomplete architecture; this structural redundancy helps the concatenated representation preserve input information even when individual filters become non-orthogonal. The learned non-uniform offsets can therefore be understood as an adaptive spectral reallocation that escapes the rigid uniform grid of classical DFT, rather than a degradation of the transform.

Thus, the non-orthogonality observed in HiL-RED is not a defect but an advantageous adaptation: the more complex the electromagnetic environment, the larger the learned deviation from the ideal basis, and the more the network relies on data-driven spectral reallocation rather than rigid physical priors.

A.8.3 Milestone Architectures Comparison Visualizations

Figures 22–27 extend visual comparisons to all interference scenarios. These results reveal a consistent performance hierarchy: EchMo maintains precise target-jamming discrimination across

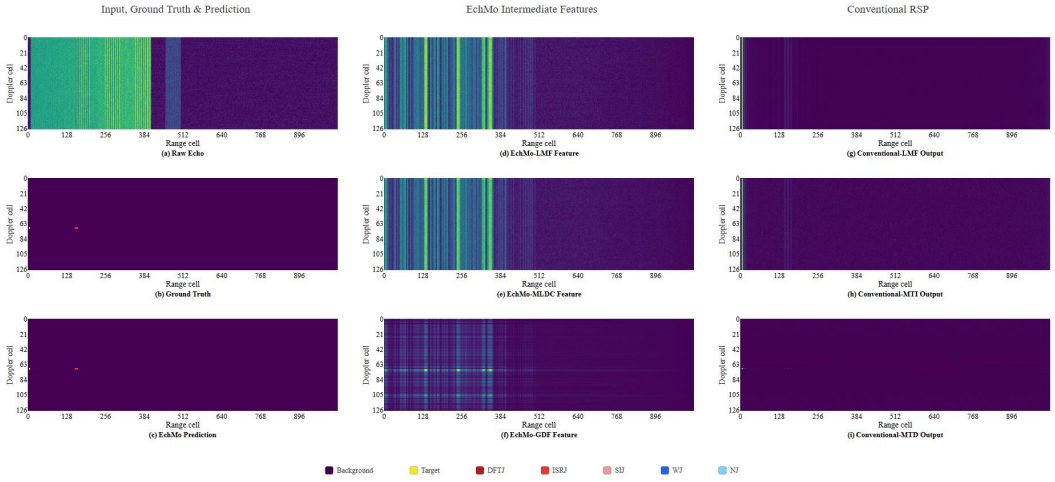


Figure 18: Intermediate outputs of EchMo modules and conventional RSP on HiL-RED under ISRJ.

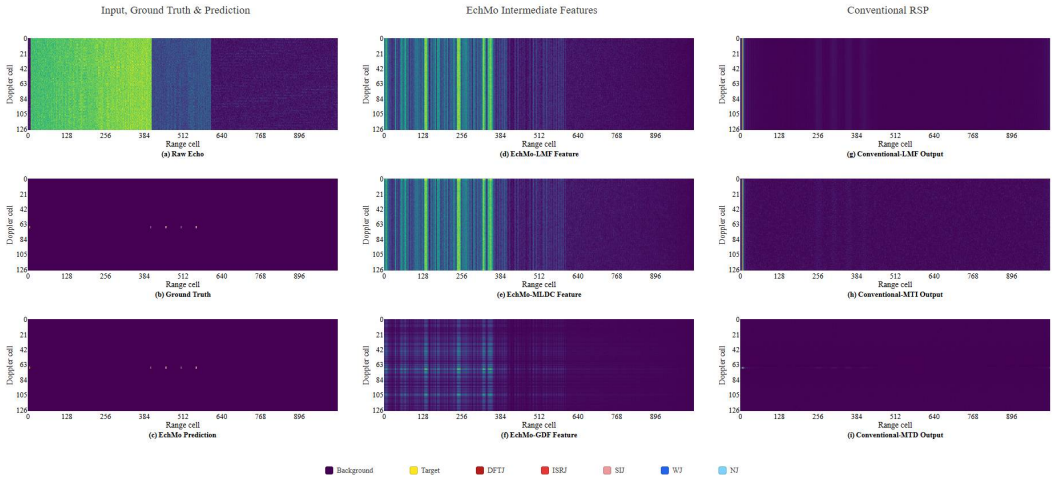


Figure 19: Intermediate outputs of EchMo modules and conventional RSP on HiL-RED under SIJ.

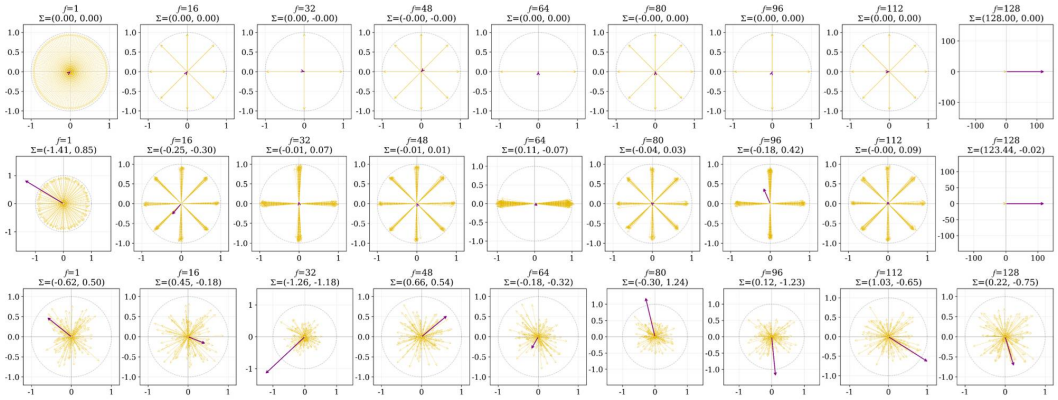


Figure 20: Vector analysis of learned GDF Fourier kernels.

jamming types, while RSP-dependent baselines exhibit catastrophic degradation under jamming and false-target confusion. This contrast validates that bypassing fixed physical pipelines enables robust adaptation in complex electronic countermeasure environments.

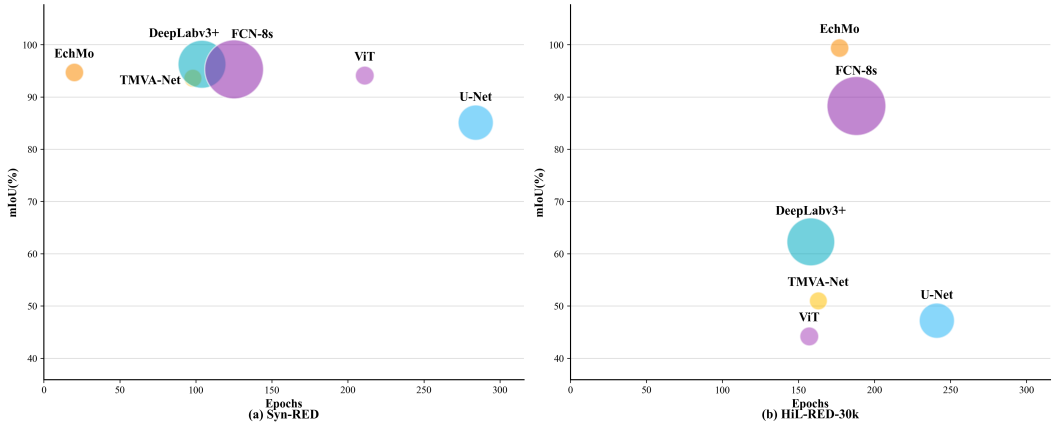


Figure 21: Model comparison on (a) Syn-RED and (b) HiL-RED30K. Bubble area indicates model parameters (larger = more parameters).

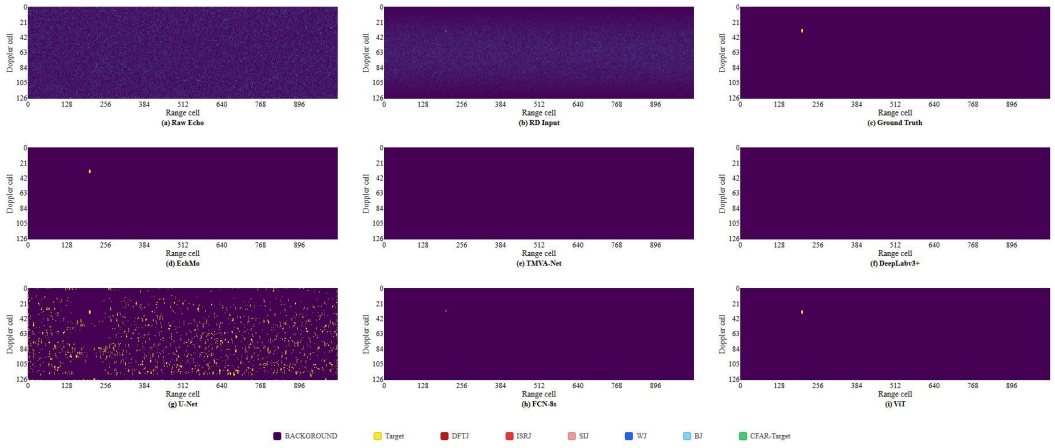


Figure 22: Visual comparison of different models on Syn-RED.

A.8.4 Raw-I/Q Baseline Comparisons

This section presents baseline comparisons between mainstream segmentation models with different input paradigms under strictly unified experimental settings. All methods use identical raw I/Q echo data as input and complete the full pipeline from raw radar signals to segmentation predictions, ensuring a fair comparison. The methods differ only in their feature-transformation paths from raw I/Q data to high-dimensional representations, not in the source data.

This comparison validates the core motivation of this work: exploring whether a fully learnable end-to-end framework can replace conventional manually designed RSP pipelines for pulse-Doppler radar segmentation. Table 31 reports the quantitative results on the Syn-RED dataset. Experimental results show that most general segmentation networks exhibit mIoU degradation of approximately 25–27 percentage points when fed raw I/Q echoes, except U-Net, which shows a smaller decrease of 7.7 percentage points. This demonstrates that conventional vision models do not adapt well to radar I/Q signals. In contrast, the proposed EchMo achieves superior segmentation performance directly on raw I/Q inputs. This further verifies the effectiveness and necessity of our physics-informed design for radar segmentation tasks.

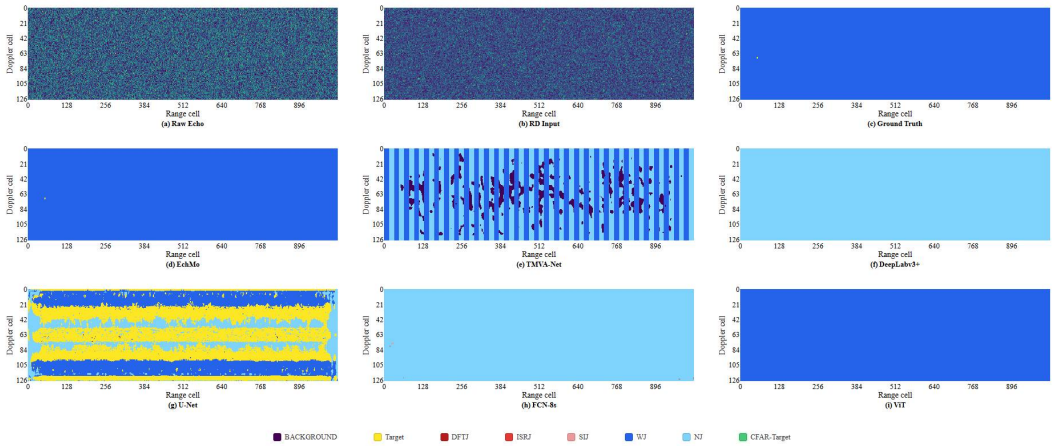


Figure 23: Visual comparison of different models on HiL-RED under WBJ.

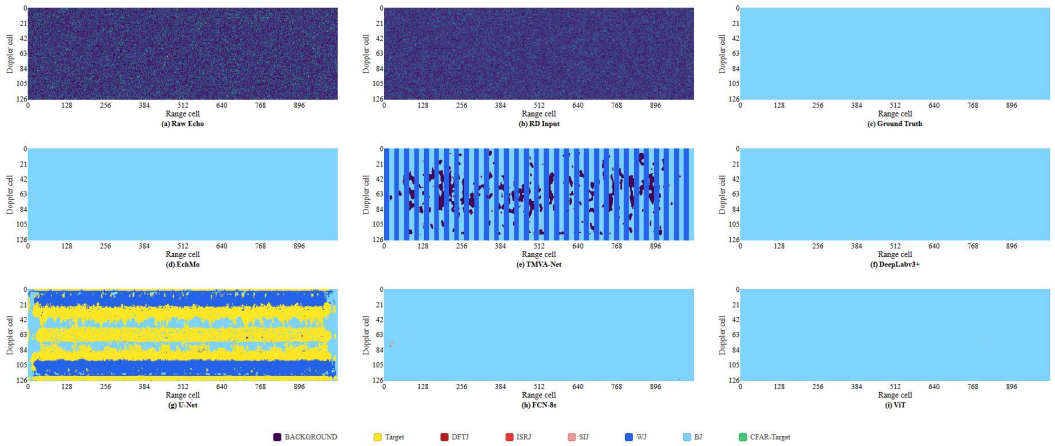


Figure 24: Visual comparison of different models on HiL-RED under NBJ.

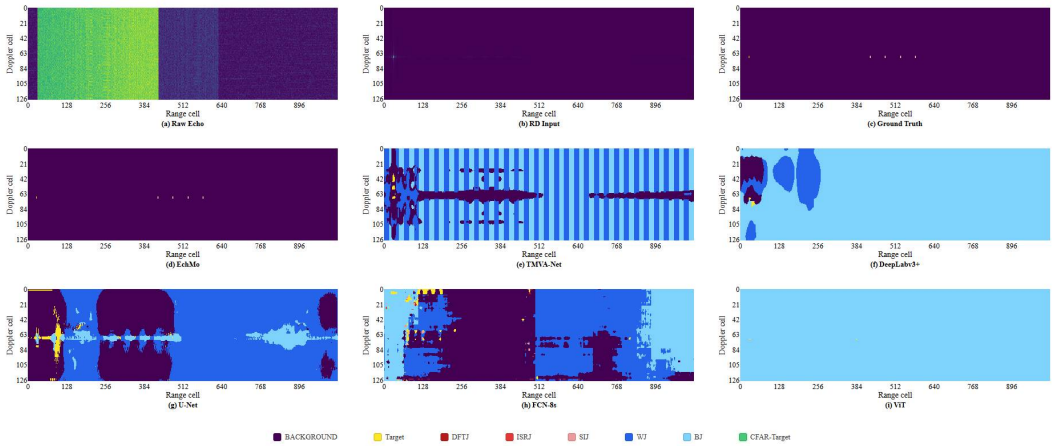


Figure 25: Visual comparison of different models on HiL-RED under SIJ.

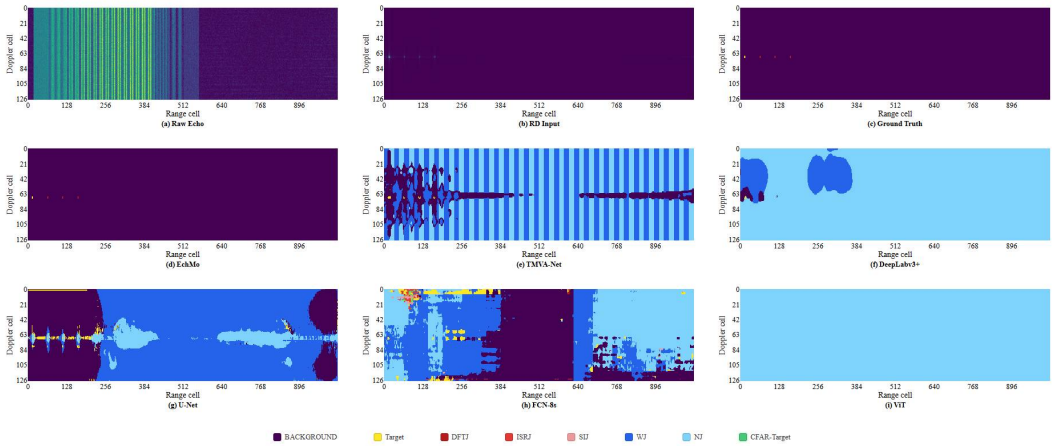


Figure 26: Visual comparison of different models on HiL-RED under DFTJ.

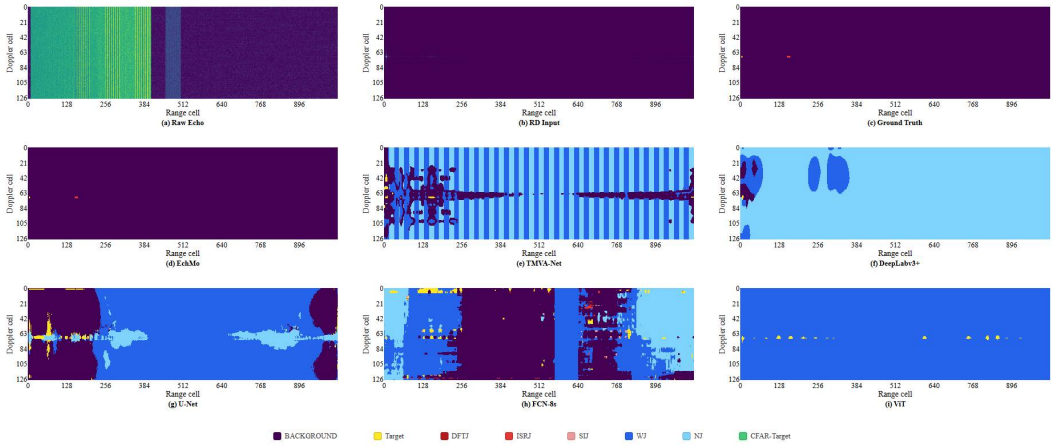


Figure 27: Visual comparison of different models on HiL-RED under ISRJ.

Table 31: Raw-I/Q vs. RD-map baseline comparison on Syn-RED.

Method	**mIoU**	**mDice**	**Params**	**Epochs**	**Input**
FCN	69.2%	77.8%	∼134.3M	161	I/Q
FCN	95.3%	97.6%	∼134.3M	206	RD
U-Net	77.4%	85.4%	∼17.2M	284	I/Q
U-Net	85.1%	91.2%	∼17.2M	284	RD
DeepLabV3+	71.2%	79.7%	∼58.7M	293	I/Q
DeepLabV3+	96.8%	98.3%	∼58.7M	124	RD
ViT	67.3%	75.8%	∼1.40M	254	I/Q
ViT	94.1%	96.9%	∼1.40M	211	RD
TMVA-Net	68.2%	76.7%	∼1.15M	84	I/Q
TMVA-Net	93.6%	96.6%	∼1.15M	98	RD
EchMo	**95.8%**	**97.8%**	**∼1.29M**	**20**	I/Q

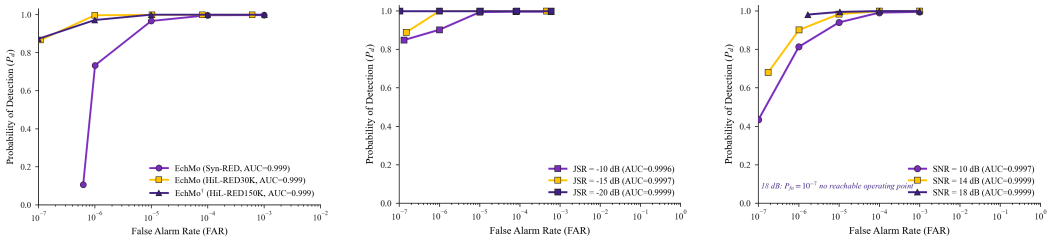
A.8.5 ROC and P_d–FAR Analysis

EchMo detection performance. We derive the probability of detection (P_d) and false-alarm rate (FAR) from segmentation outputs by treating target-class predictions as binary detection decisions. The ROC curve (Fig. 28) is constructed by ranking cells by their predicted target score, computing TPR and FPR at each threshold, and integrating via the trapezoidal rule to obtain AUC. EchMo achieves near-perfect discrimination across all datasets, with AUC consistently reaching 0.999. EchMo tightly calibrates its operating point to the preset P_{fa} threshold on jamming-contaminated HiL datasets; at the strictest preset $P_{fa} = 10^{-7}$, its actual FAR differs from the preset value by a factor of less than 1.09. By contrast, CA-CFAR inflates actual FAR by one to two orders of magnitude under jamming, as its Gaussian noise assumption is violated. The Syn-RED curve further reveals the intrinsic difficulty of low-RCS small-target detection. While P_d remains above 0.73 down to $P_{fa} = 10^{-6}$, it drops sharply to 0.105 at 10^{-7}, indicating that extremely low false-alarm tolerances impose a fundamental limit on small-target recovery. Figures 28(b–c) corroborate this robustness by showing stable high-detection performance across diverse JSR and SNR conditions.

Table 32: Global Pd–FAR on Syn-RED, HiL-RED30K, and HiL-RED150K.

P_{fa} **(preset)**	**CA-CFAR (Syn-RED)**		**EchMo(∼1.29M) (Syn-RED)**		**EchMo(∼1.29M) (HiL-RED30K)**		**EchMo†(∼9.3M) (HiL-RED150K)**	
	P_d	**FAR**	P_d	**FAR**	P_d	**FAR**	P_d	**FAR**
10^{-3}	0.499	2.03×10^{-2}	0.997	1.00×10^{-3}	0.999	6.13×10^{-4}	0.999	1.02×10^{-3}
10^{-4}	0.445	9.41×10^{-3}	0.996	1.00×10^{-4}	0.998	8.09×10^{-5}	0.999	1.00×10^{-4}
10^{-5}	0.402	4.71×10^{-3}	0.966	1.00×10^{-5}	0.998	1.03×10^{-5}	0.999	1.00×10^{-5}
10^{-6}	0.367	2.50×10^{-3}	0.733	1.01×10^{-6}	0.996	1.00×10^{-6}	0.971	1.00×10^{-6}
10^{-7}	0.339	1.40×10^{-3}	0.105	6.21×10^{-7}	0.866	1.09×10^{-7}	0.872	1.00×10^{-7}

ROC AUC: 0.999 for EchMo on all datasets. CA-CFAR uses guard=2, ref=8.



(a) EchMo P_d–FAR across the three datasets.

(b) EchMo†(∼9.3M) P_d–FAR under varying JSR on HiL-RED30K.

(c) EchMo(∼1.29M) P_d–FAR under varying SNR levels on Syn-RED.

Figure 28: ROC curves of EchMo under different experimental configurations.

Comparison with Conventional CFAR. Table 33 reports the AUC-optimal configuration from an exhaustive search over 45 configurations: five pulse-compression windows (none, Hann, Hamming, Kaiser, Taylor), three CFAR variants (CA, GO, SO), and three preset P_{fa} thresholds $(10^{-4}, 10^{-5}, 10^{-6})$. On Syn-RED, Kaiser + SO-CFAR achieves the highest AUC (0.852); on HiL-RED30K, Hamming + SO-CFAR edges out Kaiser + SO-CFAR (0.852 vs. 0.847) despite a slightly lower P_d, owing to a more favorable FAR trade-off. EchMo improves AUC from 0.852 to 0.999 on both datasets, while reducing FAR by over two orders of magnitude.

A.8.6 Per-Class Metrics

Table 34 provides complete per-class breakdowns for Syn-RED (2-class) and HiL-RED (7-class), including mIoU, mDice, FAR, P_d, and AUC. EchMo delivers consistent per-class performance across datasets, with only the SIJ class exhibiting a measurable drop on HiL-RED150K (mIoU 75.1% vs. 96.6% on 30K), indicating that sparse, bursty interference remains the primary bottleneck for large-scale generalization.

Table 33: AUC-optimal conventional RSP + CFAR configuration per dataset.

Configuration	Dataset	P_{fa}	P_d	FAR	AUC
Kaiser + SO-CFAR	Syn-RED	10^{-4}	0.773	3.27×10^{-2}	0.852
Hamming + SO-CFAR	HiL-RED30K	10^{-4}	0.622	1.14×10^{-2}	0.852
EchMo($\sim$1.29M)	Syn-RED	10^{-4}	**0.996**	1.00×10^{-4}	**0.999**
	HiL-RED30K	10^{-4}	**0.998**	8.09×10^{-5}	**0.999**

Table 34: Per-class metrics across Syn-RED and HiL-RED.

	Syn-RED		HiL-RED30K		HiL-RED150K	
Class	**mIoU**	**mDice**	**mIoU**	**mDice**	**mIoU**	**mDice**
Bkg	100%	100%	100%	100%	100%	100%
Tgt	91.6%	95.6%	99.4%	99.7%	85.9%	92.4%
DFTJ	—	—	99.8%	99.9%	96.9%	98.4%
ISRJ	—	—	99.8%	99.9%	96.1%	98.0%
SIJ	—	—	96.6%	98.2%	75.1%	85.7%
WBJ	—	—	100%	100%	100%	100%
NBJ	—	—	100%	100%	100%	100%
FAR (Tgt)	1.00×10^{-5}		1.03×10^{-5}		1.00×10^{-5}	
P_d (Tgt)	0.966		0.998		0.999	
AUC (Tgt)	0.999		0.999		0.999	

A.8.7 Robustness and Sensitivity Analysis

Multi-Seed Uncertainty Estimates. Table 35 reports results across three independent random seeds (42, 123, 3407) for EchMo on Syn-RED and EchMo† on HiL-RED150K. On Syn-RED, mIoU and mDice have standard deviations of 1.25 and 0.78 percentage points, respectively; on HiL-RED150K, both metrics have a standard deviation of 0.12 percentage points across different platforms (RTX 4090 and L40S). These small standard deviations indicate that EchMo is stable and reproducible.

Table 35: Multi-seed uncertainty estimates.

Model	Seed	mIoU	mDice	Dataset	Platform
EchMo($\sim$1.29M)	42	95.8%	97.8%	Syn-RED	RTX 4090
	123	93.3%	96.4%	Syn-RED	RTX 4090
	3407	94.5%	96.5%	Syn-RED	RTX 4090
EchMo†($\sim$9.3M)	42	93.4%	96.3%	HiL-RED150K	RTX 4090
	123	93.6%	96.5%	HiL-RED150K	RTX 4090
	3407	93.6%	96.5%	HiL-RED150K	L40S

Loss Weight Sensitivity. Table 36 compares three loss-weighting strategies on HiL-RED150K: inverse pixel frequency (default), equal uniform weights, and empirically unified weights $[0.01, 1.0, 1.0, 1.0, 1.0, 0.01, 0.01]$. EchMo maintains stable performance across a $100\times$ weight range, with the equal-weight configuration yielding comparable accuracy to the default (93.7% vs. 93.4% mIoU). This weak sensitivity to loss hyperparameters is consistent with the robust optimization conferred by the prior-guided initialization analyzed through ReCK.

A.9 Broader Impacts and Responsible Use

EchMo is intended to support civilian research and public-safety applications, including autonomous navigation, airspace monitoring, and disaster response, where robust radar perception can improve safety under adverse weather and lighting conditions. The authors also recognize potential dual-use risks, particularly unauthorized surveillance and military offensive use. To mitigate these risks, the released code, checkpoints, and demo are provided subject to a responsible-use restriction that prohibits such applications; the same restriction will accompany the repository release.

Table 36: Loss weight sensitivity on HiL-RED150K.

Strategy	**mIoU**	**mDice**	**Tgt** P_d	**Epochs**	**Stability**
Inverse frequency (default)	93.4%	96.3%	0.999	48/50	Stable
Equal uniform weights	93.7%	96.4%	0.951	50/50	Stable
Empirically unified	96.4%	98.1%	0.999	47/50	Stable

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims made in the abstract and introduction accurately reflect the paper's contributions and scope. The core contributions of this work are: (1) establishing EchMo, an end-to-end semantic segmentation architecture that directly processes pulse-Doppler radar I/Q raw echoes; (2) designing the ReCK strategy to systematically quantify the constraint effects of physical priors; (3) constructing the Syn-RED and HiL-RED datasets targeting low-slow-small target detection and complex electromagnetic interference environment recognition, respectively; and (4) achieving an accelerated version, EchMo†, through equivalent computational graph substitution with significant efficiency gains and favorable scalability. The research scope is explicitly confined to I/Q echo semantic segmentation under the pulse-Doppler radar regime. Furthermore, the Conclusion explicitly discusses generalization to other radar modalities and independently collected real-world data, lower-precision training, and dataset split design, ensuring alignment between the proposed claims and the actual contributions and expected generalizability.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Limitations are discussed in Sec. 5: i) generalization to other radar modalities and independently collected real-world data from different radar platforms and acquisition conditions; ii) stable training under lower-precision computation, such as BF16; and iii) the scope of the current dataset split, which evaluates within-condition generalization but not transfer to unseen conditions.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.

- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: The theoretical results of this paper are primarily presented in the form of equivalence derivations and learnable module definitions. Sec. 2 establishes a unified notation system and signal model (Eq. 1), clarifies all parametric assumptions for pulse-Doppler radar, and rigorously provides the mathematical formulation of the conventional RSP pipeline (Eq. 2–5), laying a complete foundation for subsequent derivations. Section 3 systematically elaborates the theoretical constructions of LMF, MLDC, and GDF (Eq. 6–14), each presenting in the main text the complete transformation process from classical forms to learnable forms along with initialization rationales. The appendix further provides detailed methodological derivations for each module (Appendix A.2.2–A.2.4), ensuring the completeness of the derivations stated in the main text.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The EchMo architecture and training procedure are fully described in Sec. 3 and Sec. 4.1. Source code, lightweight checkpoints, and representative Syn-RED and HiL-RED-30K examples are available in `https://github.com/zlw9161/PeCoRSS`. The `README` provides execution commands and environment setup instructions. An interactive visualization app is available at `https://huggingface.co/PeCoLab/EchMo-Vis`, and model checkpoints are available at `https://huggingface.co/PeCoLab/EchMo-ckpts`. The full Syn-RED and HiL-RED datasets are being prepared for public release because of their large scale.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.

- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Source code, model checkpoints, representative data examples, and an interactive demo are openly available at `https://github.com/zlw9161/PeCoRSS`, `https://huggingface.co/PeCoLab/EchMo-ckpts`, and `https://huggingface.co/PeCoLab/EchMo-Vis`, respectively. The full Syn-RED and HiL-RED datasets will also be made publicly available. Owing to their large size, additional time is required to prepare a dedicated access portal. Release schedules and access instructions will be announced through our GitHub repository and Hugging Face pages.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.

- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The overall training configuration is described in Sec. 4.1, with experiment-specific hyper-parameters detailed in Appendix A.4–A.5.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Appendix A.8.7 reports multi-seed uncertainty estimates over three independent random seeds, including standard deviations for mIoU and mDice on Syn-RED and HiL-RED150K (Table 35).

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Detailed computing resources, including GPU models, per-iteration latency, and memory usage, are reported in the experimental section and Appendices A.4 and A.5 to support reproducibility.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: This work uses only digital synthetic and radio-frequency hardware-in-the-loop synthetic radar data. No human subjects, PII, or ethically sensitive content are involved.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: EchMo targets long-range target perception in complex electromagnetic environments and can be widely applied to civilian domains such as autonomous navigation safety, airspace monitoring, ground-air collaborative detection, low-slow-small UAV management, and disaster search and rescue, enhancing environmental understanding and public safety under adverse weather and lighting conditions. Meanwhile, the authors fully recognize the potential risks of radar perception technology being misused for unauthorized surveillance or military offensive purposes. Appendix A.9 discusses the positive and negative societal impacts of EchMo and includes a responsible-use statement that prohibits unauthorized surveillance and military offensive purposes. The same restriction will accompany the repository release.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.

- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [Yes]

    Justification: Appendix A.9 describes the safeguards for responsible release, including a responsible-use restriction that limits the model to positive societal applications and prohibits military offensive purposes and unauthorized surveillance. The same restriction will accompany the repository release.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Only self-developed datasets are used. All third-party software libraries and code are properly cited and comply with their open-source licenses.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: Two new radar echo datasets (Syn-RED and HiL-RED) are introduced. Dataset construction and specifications are detailed in Appendix A.6. The full Syn-RED and HiL-RED datasets are being prepared for public release because of their large scale.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve human subjects or crowdsourcing.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The paper does not involve human subjects or crowdsourcing.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: All core methods in this paper are originally proposed by the authors, with no use of LLMs involved.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.