Wire-Cell DNN-based Signal Processing for ProtoDUNE : Final Status and Outlook

Yujin Park
Chung-Ang University, South Korea

on behalf of the Wire-Cell Team @ BNL

2026 Sep DUNE Collaboration Meeting Parallel Talk

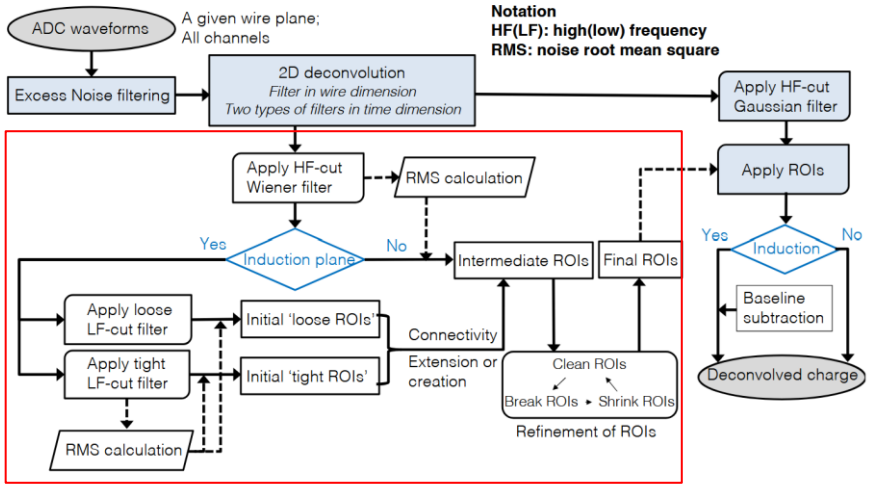
Outline

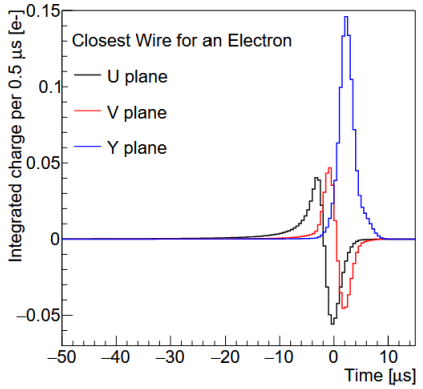
- Introduction & Motivation
- Baseline Model
  - Lightweight MobileNetV3 U-Net
  - Pre-processing / Training Setup
- Knowledge Distillation (KD) Strategies
  - Response Distillation
  - Architecture-aware Feature Distillation
- INT8 Quantization
  - Post-Training Quantization (PTQ)
  - Quantization-Aware Training (QAT)
- Performance Evaluation
- Summary

Introduction

Wire-Cell Signal Processing

- Wire-Cell signal processing pipeline
- Region of Interest (ROI) finding is a critical step
  - Identifying true signal regions, maximizing the signal-to-noise ratio.
  - It is important for induction planes whose signal is bipolar, whereas a simple threshold-based ROI finding is typically sufficient for collection planes.
- The traditional method relys on various software filters and heuristic finding algorithms.

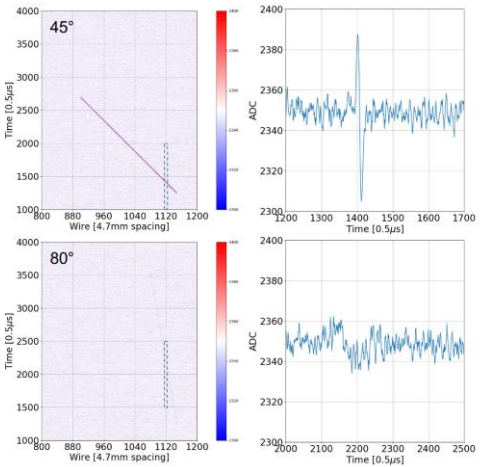


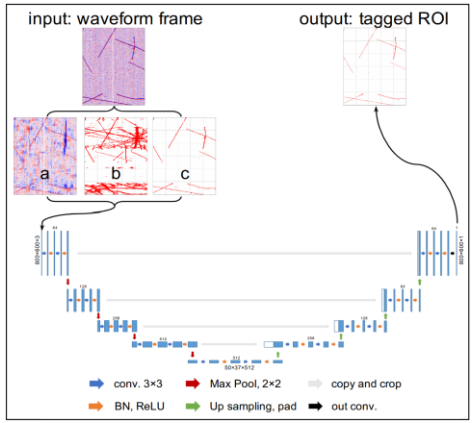


MicroBooNE Collaboration, 2018 *JINST* 13 P07006

Introduction

DNN based ROI Finding





H.W. Yu *et al* 2021 *JINST* 16 P01036

- Semantic Segmentation problem
- U-Net based CNN models

Hokyeong's study

- Reduced inference cost via MobileNetV3-based U-Net architecture.
- Reduced Wire-Cell signal processing memory consumption.

Limitations

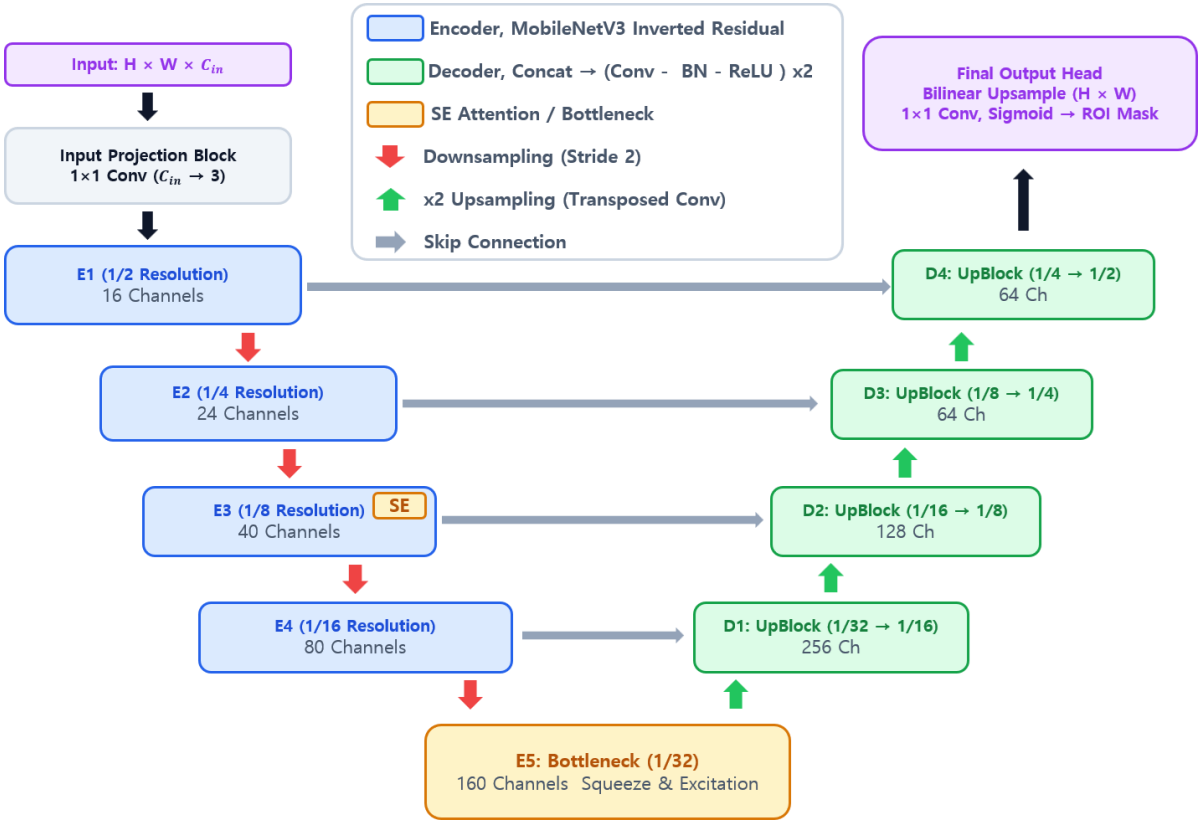
- Track angle dependent performances
- Track traveling normal to the wire planes(prolong tracks) are vulnerable
- Poor S/N ratio in induction wire planes

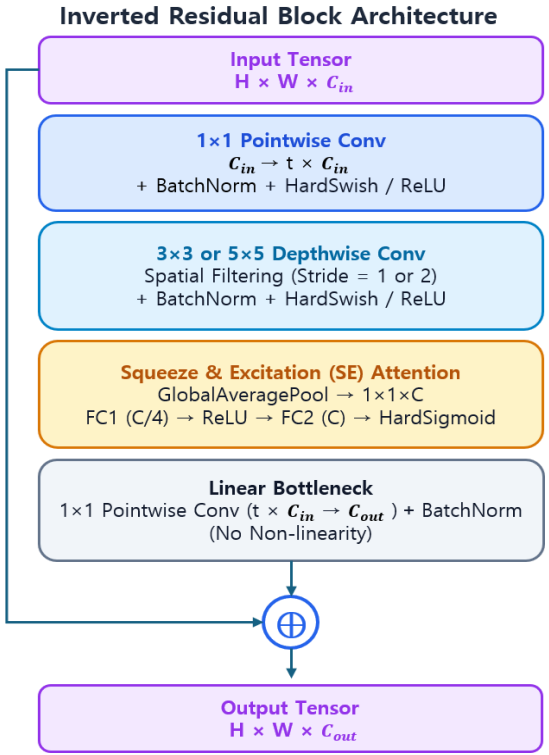
In this update,

- *Improved pre-processing*
- *Knowledge distillation / Quantization techniques*
- *Enhanced inference performance while maximizing resource efficiency*

Baseline Model: MobileNetV3 U-Net

Network Architecture



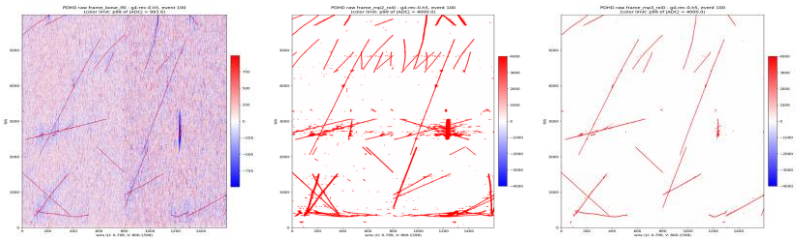


Baseline Model: MobileUNetV3 UNet

Model Input Channels

- The original work used only three input images, while there are additional images in the WireCell chain.

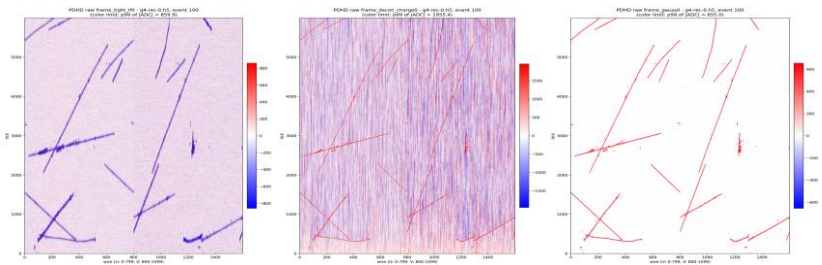
1. loose_lf **2. mp2_roi** **3. mp3_roi**



1. Deconvolved signal with a loose low-frequency filter
2. Two-plane (2-of-3) multi-plane-coincidence mask
3. Three-plane (3-of-3) multi-plane-coincidence mask

- In this work, we expand the inputs from 3 to 6 images to incorporate more comprehensive event information.

4. tight_lf **5. decon_charge** **6. gauss**



4. Deconvolved signal with a tight low-frequency filter
5. Deconvolved charge without any filters
6. Charge after gaussian filter and traditional ROI masking

* Per channel normalization is applied.

Baseline Model: MobileUNetV3 UNet

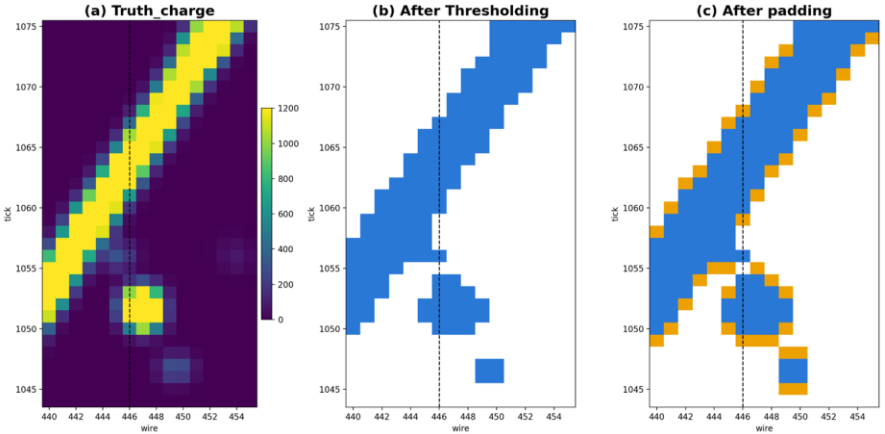
Truth Masking

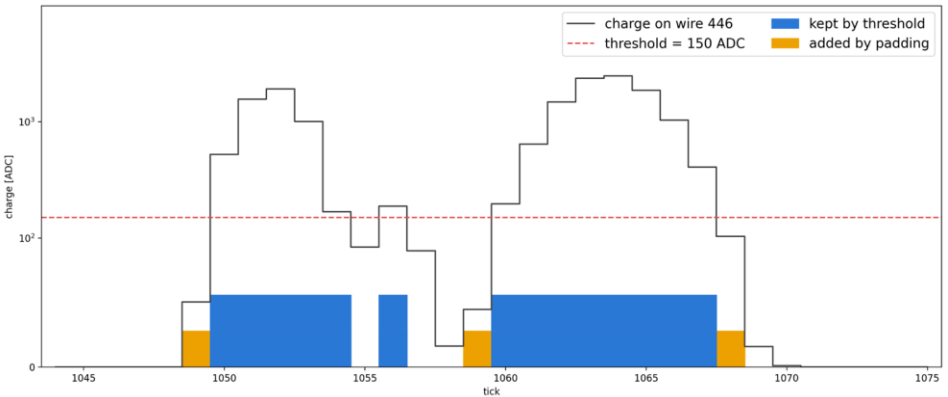
Thresholding

- Each rebinned pixel above the threshold is labeled as true.
- Threshold adjusted from 100 to 150 compared to the original work

Padding

- Compensates for the charge loss in the tails caused by thresholding.
- Pads ROI edges with adjacent pixels per wire
- skips padding if it would overlap with neighboring ROIs.





Evaluation Metrics

Pixel-level Metrics

- G : Truth mask pixel
- P : Predicted mask pixel where the model output exceeds a threshold

$$Dice = \frac{2|P \cup G|}{|P| + |G|} \qquad Efficiency_{pixel} = \frac{|P \cap G|}{|G|} \qquad Purity_{pixel} = \frac{|P \cap G|}{|P|}$$

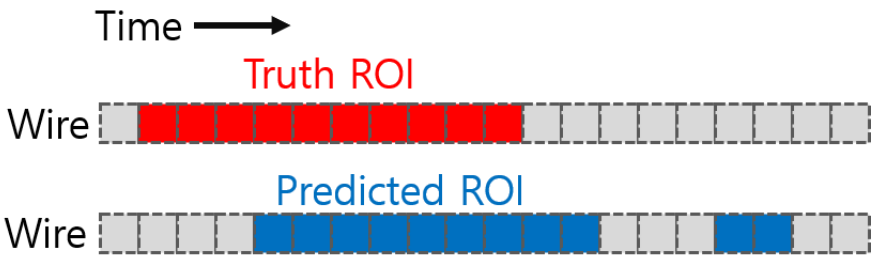
ROI-level Metrics

ROI : A contiguous run of truth/predicted mask pixels along the time axis of a given wire

- G' : Truth ROI
- P' : Predicted ROI where the model output exceeds a threshold
- If at least one pixel is overlapped, they are matched. (Numerators)
- ROI-level metrics reflect physical responses more realistically than pixel-level.

$$Efficiency_{ROI} = \frac{|P' \cap G'|}{|G'|}$$

$$Purity_{ROI} = \frac{|P' \cap G'|}{|P'|}$$



Baseline Model: MobileUNetV3 UNet

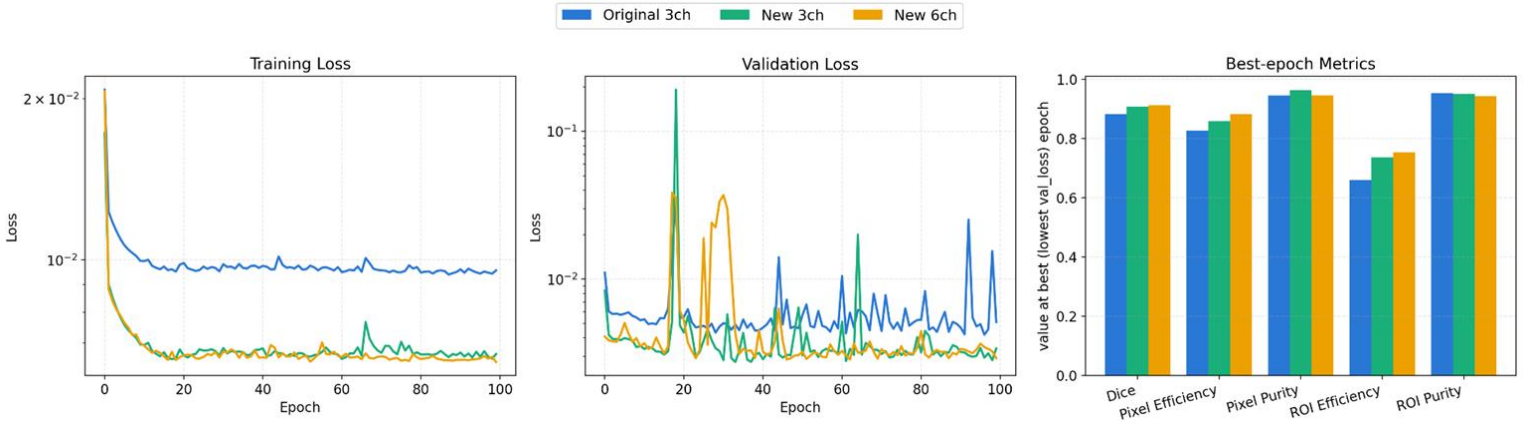
Dataset / Training Setup

Training Setup

- Optimizer: Stochastic gradient descent (SGD)
- Momentum: 0.9
- Learning rate: 0.1
- Weight Decay: 5×10^{-4}
- Loss: Binary cross-entropy (BCD)
- Training epochs: 100

Dataset split

- Total : 590 samples
- Train: 470 / Val: 60 / Held-out Test: 60



➢ Updated pre-processing shows slightly improved performances

Knowledge Distillation (KD)

What is the Knowledge Distillation?

Knowledge Distillation transfers representations learned by a teacher to a compact student

- Teacher model: Pre-trained, high-capacity model with rich feature representations
- Student model: Lightweight model optimized for low-resource inference

Three Teacher Models

1) U-Net

- The classic symmetric encoder/decoder with one skip connection per resolution level.

2) Nested U-Net

- A U-Net whose decoder is replaced by a dense grid of intermediate nodes
- Each node is fused by all earlier nodes at the same resolution

3) Transformer U-Net

- A U-Net encoder and decoder with a Transformer bottleneck
- The encoder feature map is flattened to a sequence, position-encoded, and processed by several self-attention blocks before being decoded

	# of parameters
MobileNetV3 U-Net	~ 5.2 M
U-Net	~13.4 M
Nest U-Net	~9.2 M
Transformer U-Net	~26 M

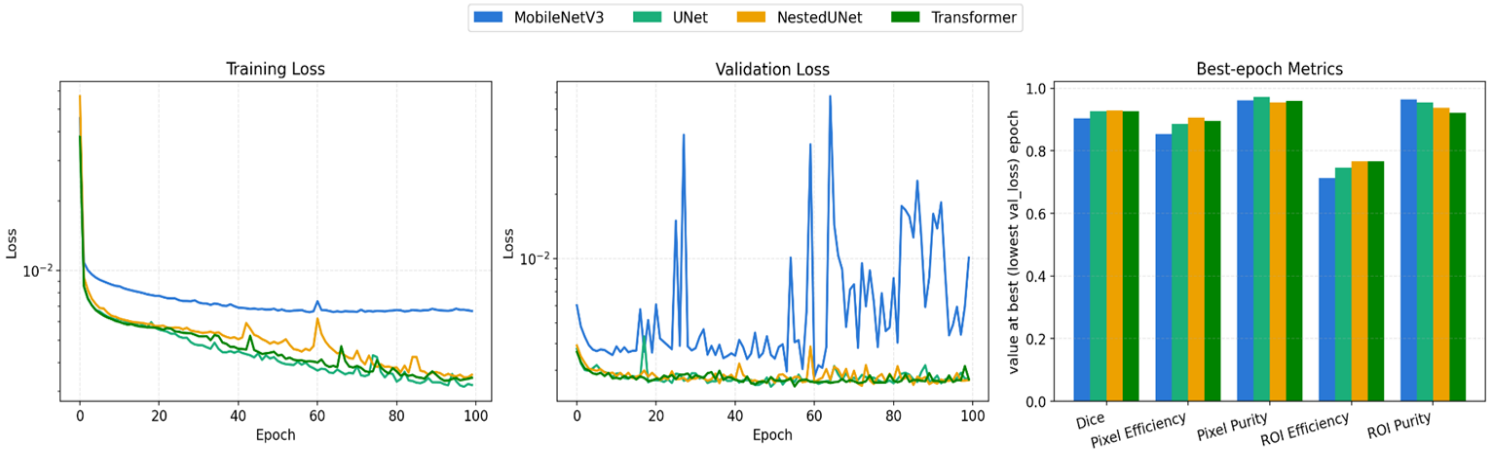
Knowledge Distillation
Motivation

MobileNetV3 U-Net serves as a conservative baseline

- High purity: Minimizes false positives by predicting only confident signals
- Low efficiency: Misses weak or ambiguous signals

➢ Spurious ROIs can be ruled out in downstream stages (e.g., 3D imaging, clustering), whereas missed signals cannot be recovered.

➢ Since teacher models exhibit less conservative behavior (higher efficiency), the goal is to distill this characteristic into the baseline student.



Knowledge Distillation
Strategies

Strategy A: Response Distillation

- The teacher's response(model output) is distilled.
- The loss consists of two terms, balanced by **α**.
  - Hard-Target Loss (1st term): Anchors student predictions to the true labels
  - Soft-Target Loss (2nd term): Transfers the teacher's graded confidence
- T scaling
  - Exposes subtle relative probability differences in the background regime
  - Compensates for softened gradients, keeping distillation updates balanced with supervised loss
- During training, $\alpha = 0.5, T = 2.0$
- All three teachers are used.

$$L = \alpha\,\mathrm{BCE}(\sigma(s),\, y) + (1-\alpha)\,T^2\,\mathrm{BCE}(\sigma(s/T),\, \sigma(t/T))$$

Knowledge Distillation
Strategies

Architecture-aware Feature Distillation

- Beyond response distillation, the teacher's signature internal representations are also distilled.
- MSE term with weight **γ** (**γ** = 0.2)
  - 1X1 Conv Adapter: Matches student channel dimensions to the teacher during training (discarded at inference).
  - Mean Squared Error (MSE) between the adapted student feature map and the teacher's.

$$L = \alpha\,\mathrm{BCE}(\sigma(s), y) + (1-\alpha)\,T^2\,\mathrm{BCE}(\sigma(s/T), \sigma(t/T)) + \gamma\,\mathrm{MSE}(\mathrm{adapter}(f_s),\ f_t)$$

Strategy B (Nested U-Net)

- Distilled Location: Fully fused top-row node
- Multi-scale dense-skip refines the features

Strategy C (Transformer U-Net)

- Distilled Location: Post-attention bottleneck
- Long-range correlations along tracks can be utilized

INT8 Quantization

Improving computational resource efficiencies

- The model weights and activations are usually stored in **32-bit floating-point (FP32)**; 4 bytes.
- Transition from FP32 to **8-bit integers (INT8**) for both weights and activations; 1 byte.
- Smaller model file size, reduced memory usage, and accelerated inference time.

Post-Training Quantization (PTQ)

- No additional training for quantization
- Directly quantizing the trained floating-point model by calibrating the activation ranges on a small set of events.

Quantization-Aware Training (QAT)

- Inserts fake-quantization operators into the model during training, making it aware of INT8 rounding errors
- For KD, the unquantized teacher continuously provides continuous soft targets to guide the student in preserving critical signal representations under quantization constraints

Performance Evaluation

Choosing the Best Model

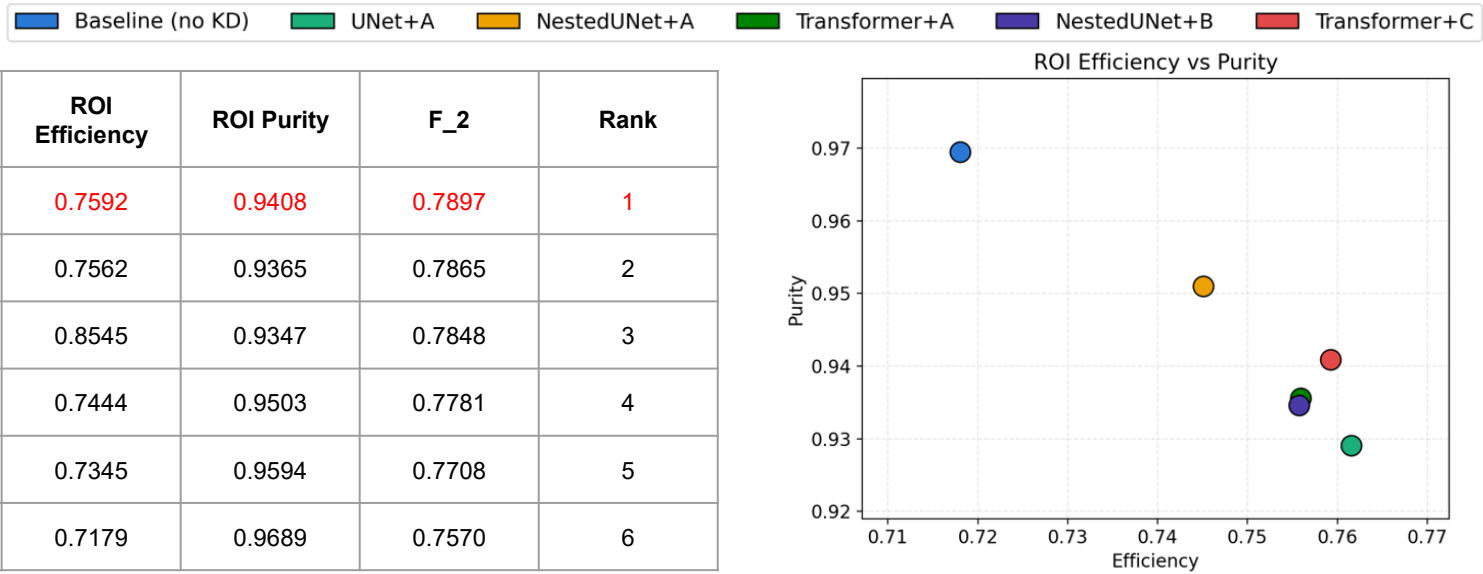
$$F_\beta = (1+\beta^2)\frac{Purity \cdot Efficiency}{(\beta^2 \cdot Purity) + Efficiency}$$

F_β score

- The metric which combines Purity & Efficiency, balancing their trade-off using a weighting parameter β
- β>1 (β=2.0): Prioritizes Efficiency

➢ **Best Model : Transformer + C**

	Dice	ROI Efficiency	ROI Purity	F_2	Rank
Trans + C	0.9129	0.7592	0.9408	0.7897	1
Trans + A	0.9116	0.7562	0.9365	0.7865	2
Nested + B	0.9068	0.8545	0.9347	0.7848	3
Nested + A	0.9099	0.7444	0.9503	0.7781	4
UNet + A	0.9101	0.7345	0.9594	0.7708	5
Baseline	0.9069	0.7179	0.9689	0.7570	6



Performance Evaluation
Quantization Comparison

Transformer + C with two quantization strategies

- Comparing resource usage for held-out test set
- Substantial reductions in both time and memory.

Quantization Strategy	Weight Memory (MB)	Median Inference Time (ms)	Peak RSS(MB)	Speedup	Peak RSS Decrement
FP32	19.9	757.9	6192	-	-
INT8 PQT	5.0	70.5	4882	10.75 x	21.2%
INT8 QAT	10.03	64.5	4417	11.76 x	28.7%

➢ Quantized models have significant resource gains.

Performance Evaluation

Quantization Comparison

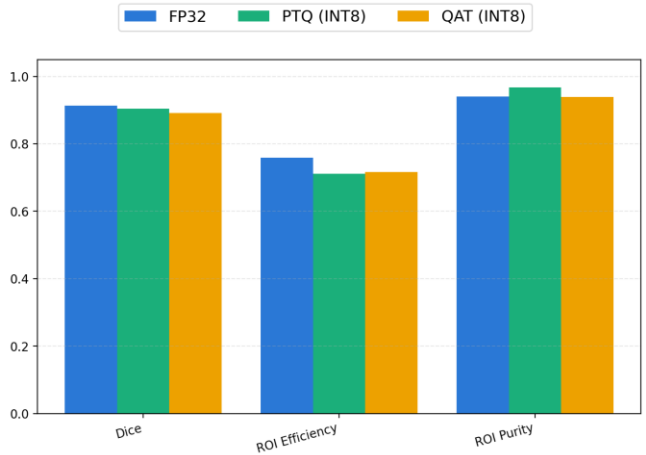
Transformer + C with two quantization strategies

- Comparing held-out test evaluation results
- The performance of quantized models become worse.
  - Purity: Remains comparable or slightly increases.
  - Efficiency: Drops more significantly, driving the overall performance loss.

➢ **The final model for production signal processing chain is floating-point distilled model.**

➢ **However, the QAT model is provided as a lower resource alternative.**

	Dice	ROI Efficiency	ROI Purity
FP32	0.9129	0.7592	0.9408
PTQ	0.8832	0.7108	0.9668
QAT	0.8918	0.7167	0.9390



Performance Evaluation

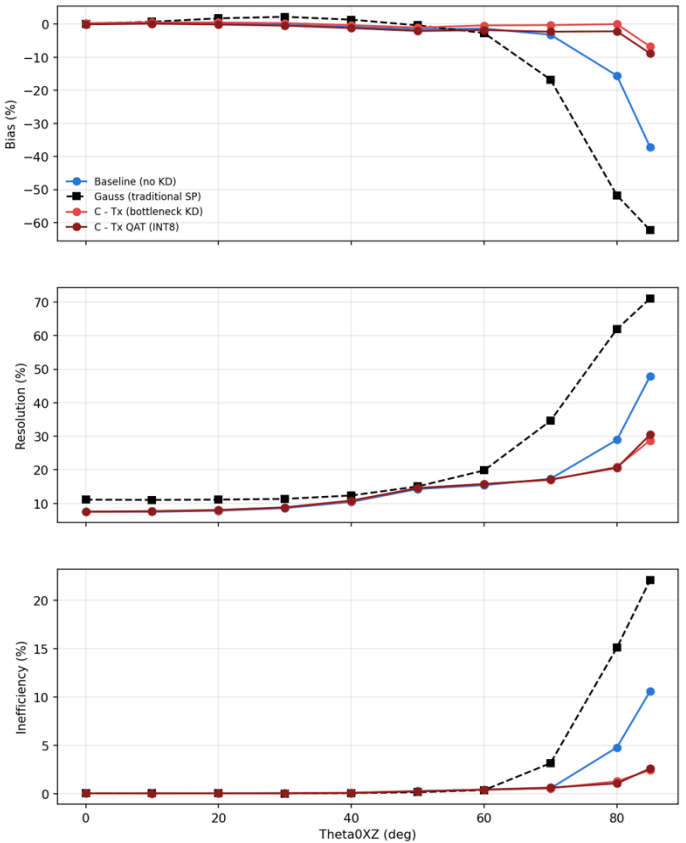
Physical(Charge) Performance Evaluation

$$Charge\ Bias\quad(\%) = \{mean\left(\frac{Charge_{reco}}{Charge_{true}}\right) - 1\} \times 100$$

$$Charge\ Resolution\quad(\%) = \frac{RMS(Charge_{reco}/Charge_{true})}{mean(Charge_{reco}/Charge_{true})} \times 100$$

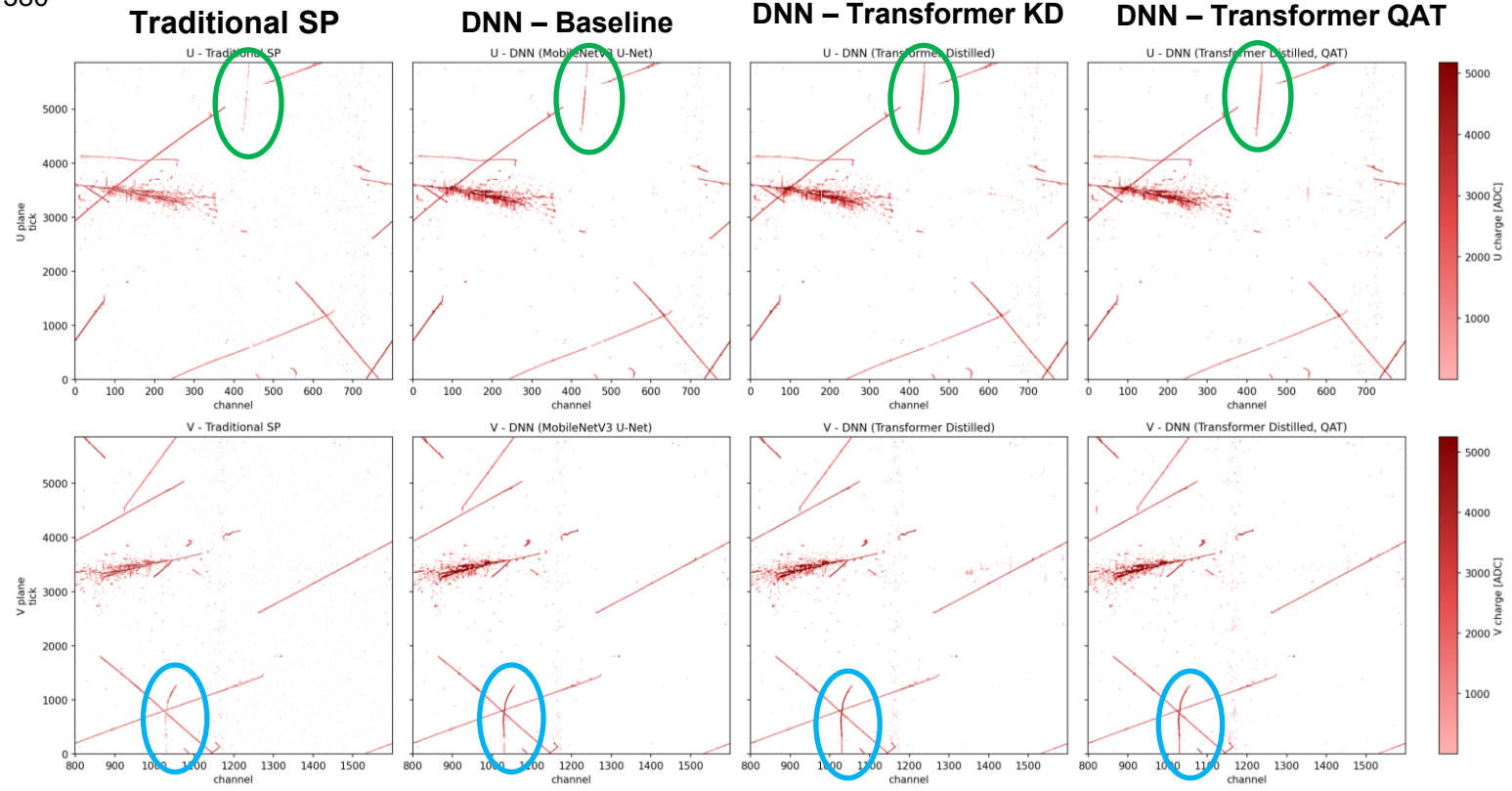
$$Inefficiency\quad(\%) = \frac{\#\ of\ bad\ channels}{\#\ of\ true\ channels} \times 100$$

- For physical performance evaluation,
  - Distilled models consistently outperform the baseline.
  - The QAT model exhibits a slight degradation compared to the FP32, preserving physics performance.



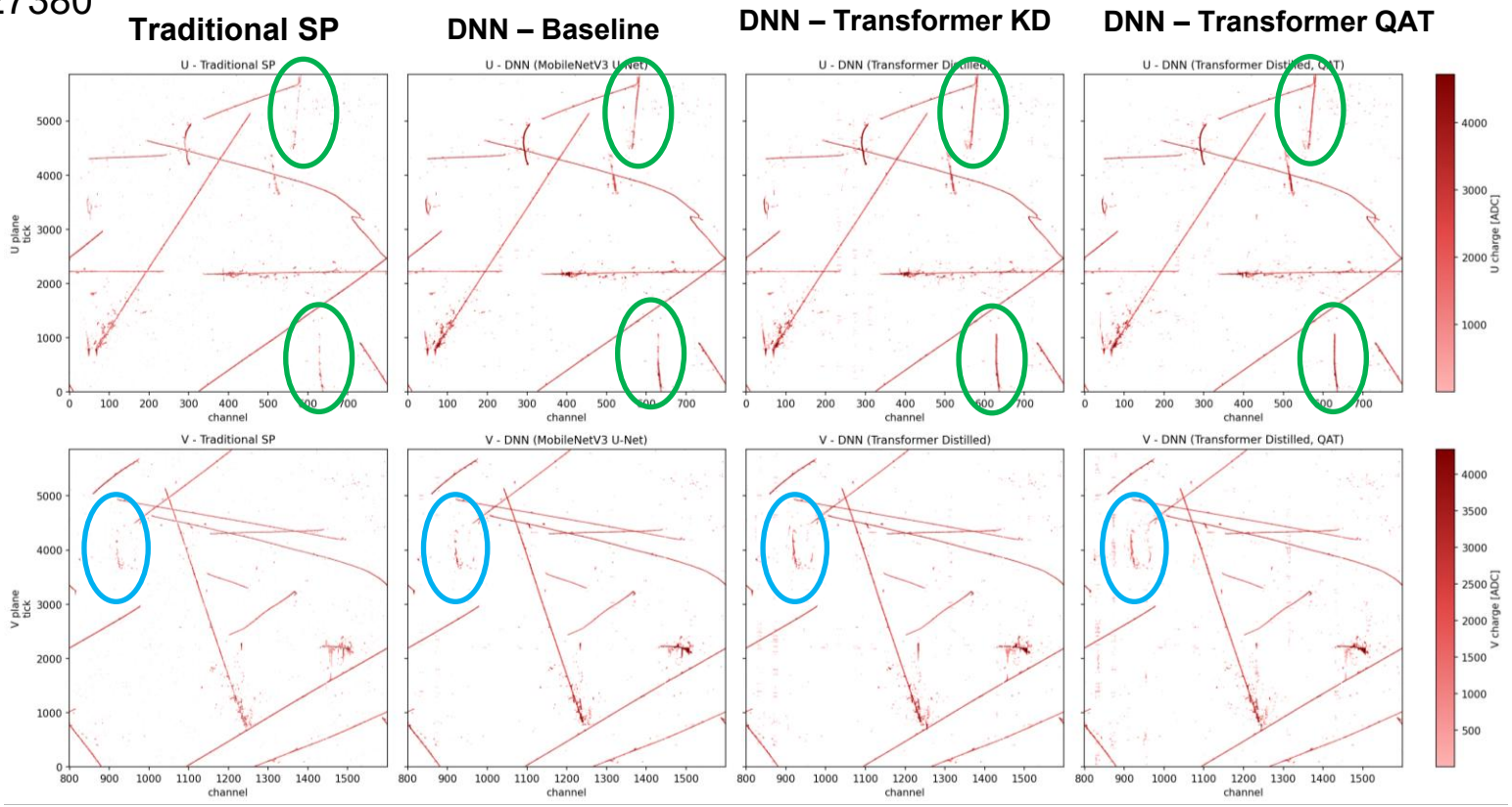
Performance Evaluation
PDHD Real Data Processing

Run 27380



Performance Evaluation
PDHD Real Data Processing

Run 27380



Summary

Enhanced Performance via Pre-processing and Knowledge Distillation

- Expanding input channels and refining threshold/padding enhanced feature extraction.
- Alleviated the conservative bias of the baseline student, recovering efficiency.
- **Transformer U-Net Bottleneck Distillation** achieved top performance.

Efficient Production Deployment with Quantization

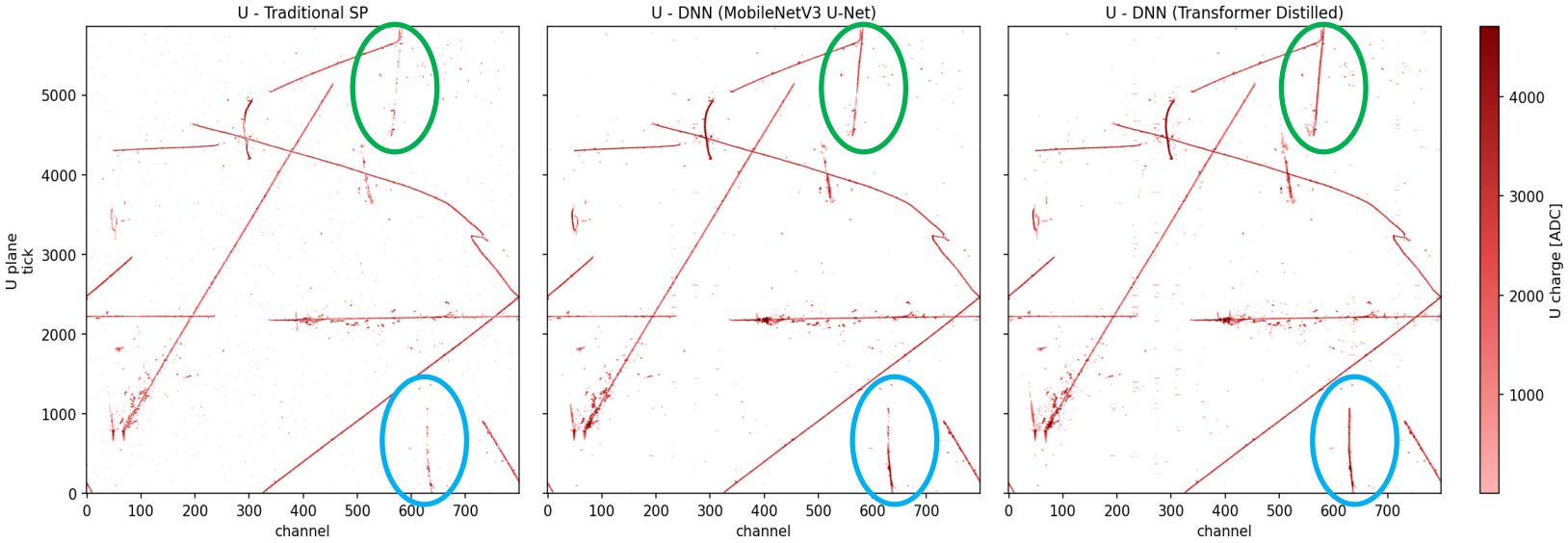
- Quantized models achieve **11–12 times speedup** and over **~20% memory reduction**.
- **Floating-point Distilled Model (Transformer+C)** serves as the primary production model, with the **QAT model** providing a high-fidelity, low-resource alternative.

Note) Integration / Publication

- **Key Achievement: Integrated into `dunereco` starting from release `v10_24_00d00`.**
- The DUNE technical paper is currently in preparation.

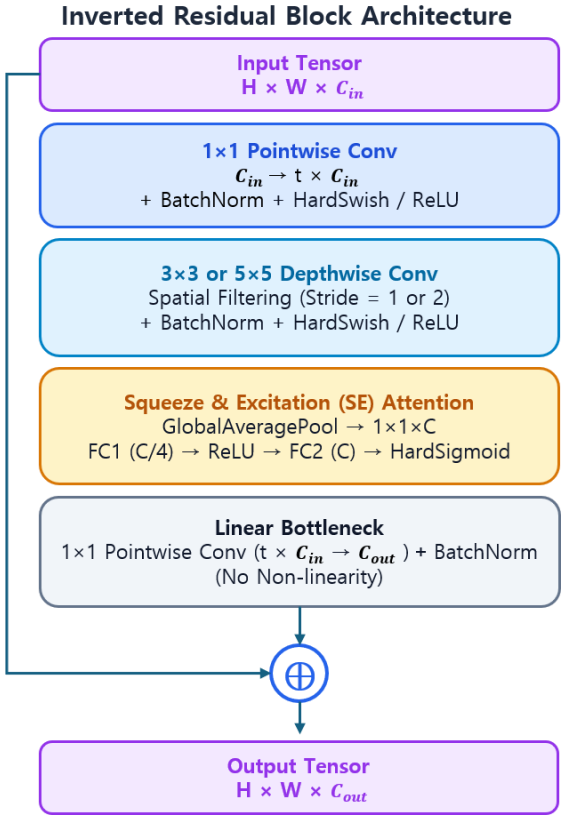
Backup

Run 27380, event 3



Baseline Model: MobileNetV3 U-Net

Network Architecture



1. Inverted Expansion

- Expands into high-dimensional space
- Prevents information loss during non-linear filtering.

2. Depthwise Separable

- Decouples spatial convolutions from channel
- Reduced computational cost (~10%)

3. Squeeze & Excitation

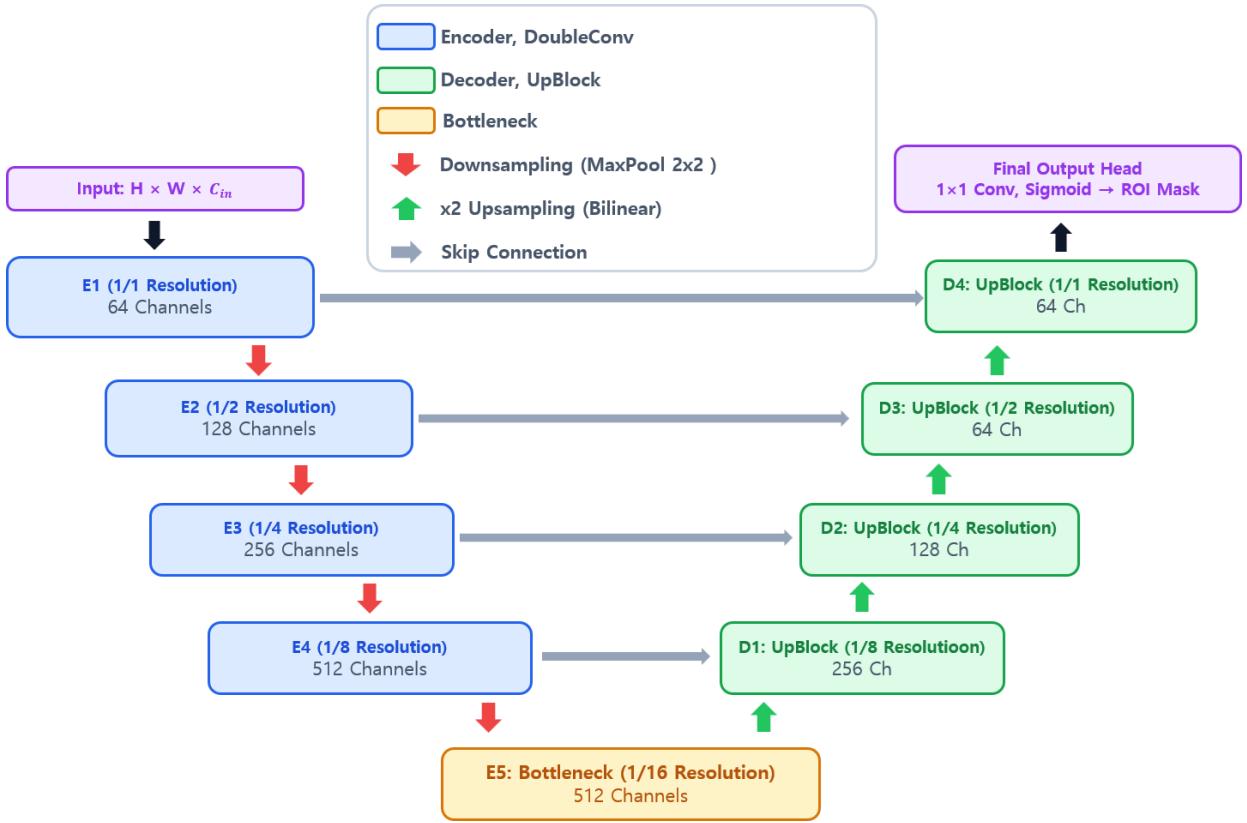
- Squeeze: Summarize global spatial information into a 1×1 feature map per channel
- Excitation: Compute channel-wise weights and scale each channel

4. Linear Bottleneck

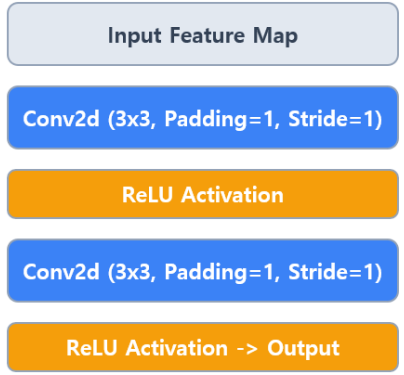
- Restores into low dimensional space
- Omits Non-linearity to prevent manifold destruction in low-dimensional space.

U-Net

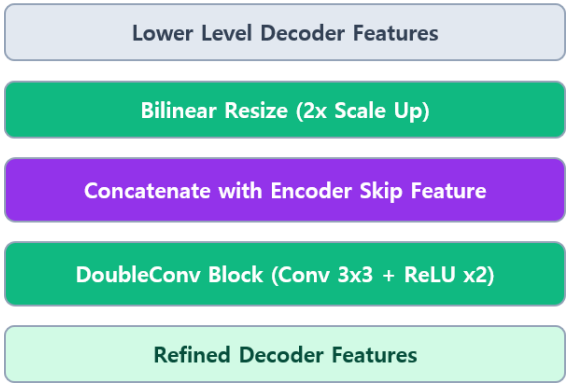
Network Architecture



DoubleConv Sub-Block Architecture

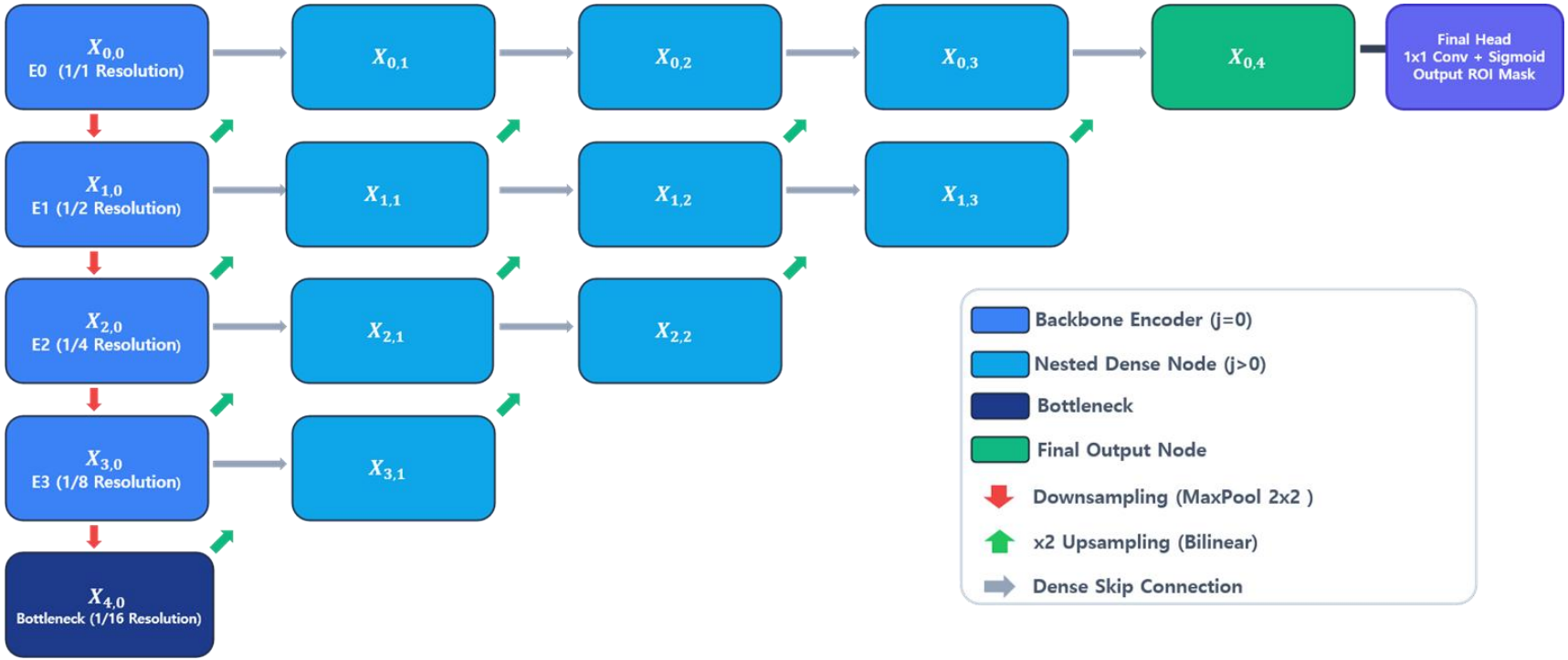


UpBlock (Decoder Step) Architecture

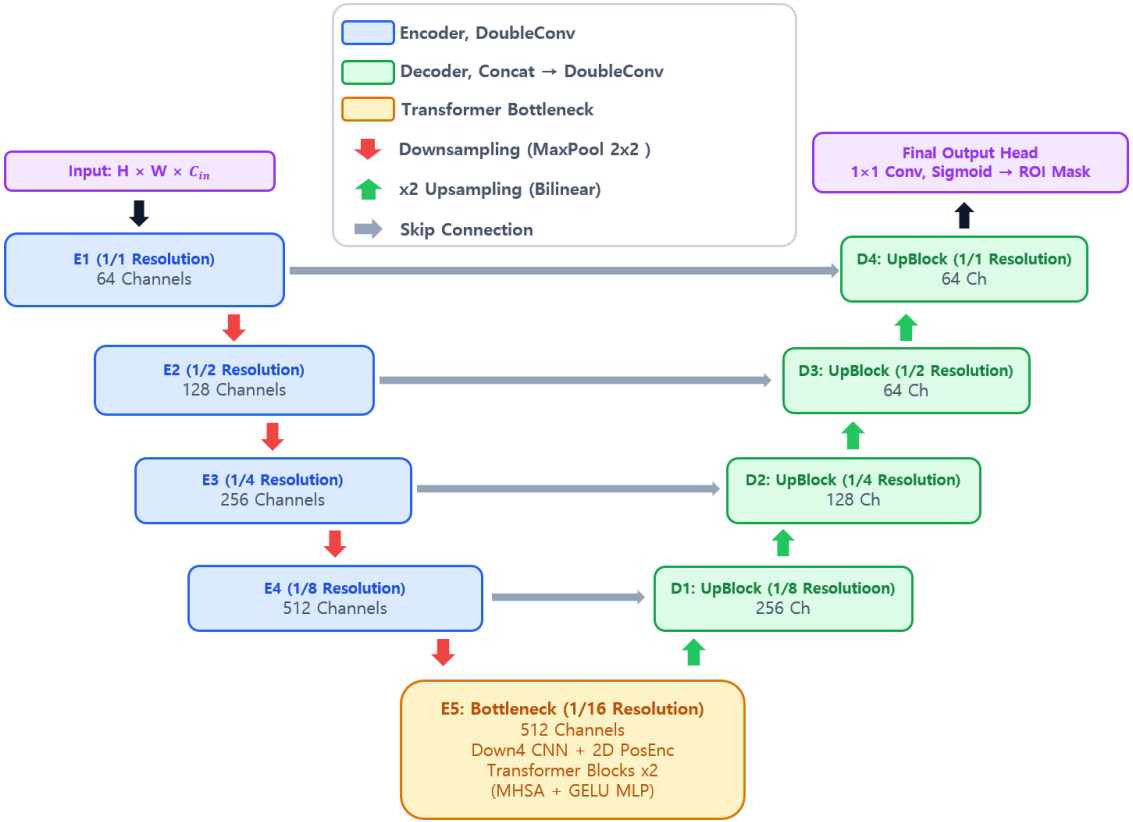


Nested U-Net

Network Architecture



Transformer U-Net
Network Architecture



Performance Evaluation

Physical(Charge) Performance Evaluation

$$Charge\ Bias\quad(\%) = \{mean\left(\frac{Charge_{reco}}{Charge_{true}}\right) - 1\} \times 100$$

$$Charge\ Resolution\quad(\%) = \frac{RMS(Charge_{reco}/Charge_{true})}{mean(Charge_{reco}/Charge_{true})} \times 100$$

$$Inefficiency\quad(\%) = \frac{\#\ of\ bad\ channels}{\#\ of\ true\ channels} \times 100$$