Wire-Cell ProtoDUNE Local Meeting

Yujin Park
Chung-Ang University
Jun03, 2026 (Wed)

Wire-Cell DNN Signal Processing
Reproducing PDHD Baseline model Training

To do this, I basically followed the training configuration from the DNNROI paper draft.

PDHD Raw Dataset: /nfs/data/1/hnam/train_data_PDHD_fixedbug_separateWC

Dead-channel Augmentation (`scripts/augment_dead_channels.py`)

- Replacing waveforms of randomly chosen channels by gaussian noise(applying new noise RMS)
- Using default configurations
  - ✓ **Replaced frame:** `frame_loose_lf0` only
  - ✓ **Random Seed:** 42
  - ✓ **Randomly chosen noise factor range:** [2, 15] -> New RMS = Noise factor * original RMS

Data Pre-processing (`scripts/build_training_dataset.py`)

- Channel Stacking
- Configurations
  - ✓ **Channels:** [loose_lf, mp2_roi, mp3_roi]
  - ✓ **Rebin factor :** 4
  - ✓ **Padding:** 1 rebinned tick
  - ✓ **Threshold for Truth labeling:** 150e- (after rebinning)

Q: For 6-channel training [loose_lf, mp2_roi, mp3_roi, tight_lf, decon_charge, gauss],
does dead-channel augmentation apply only to loose_lf, or to all frames except mproi?

Wire-Cell DNN Signal Processing
Reproducing PDHD Baseline model Training

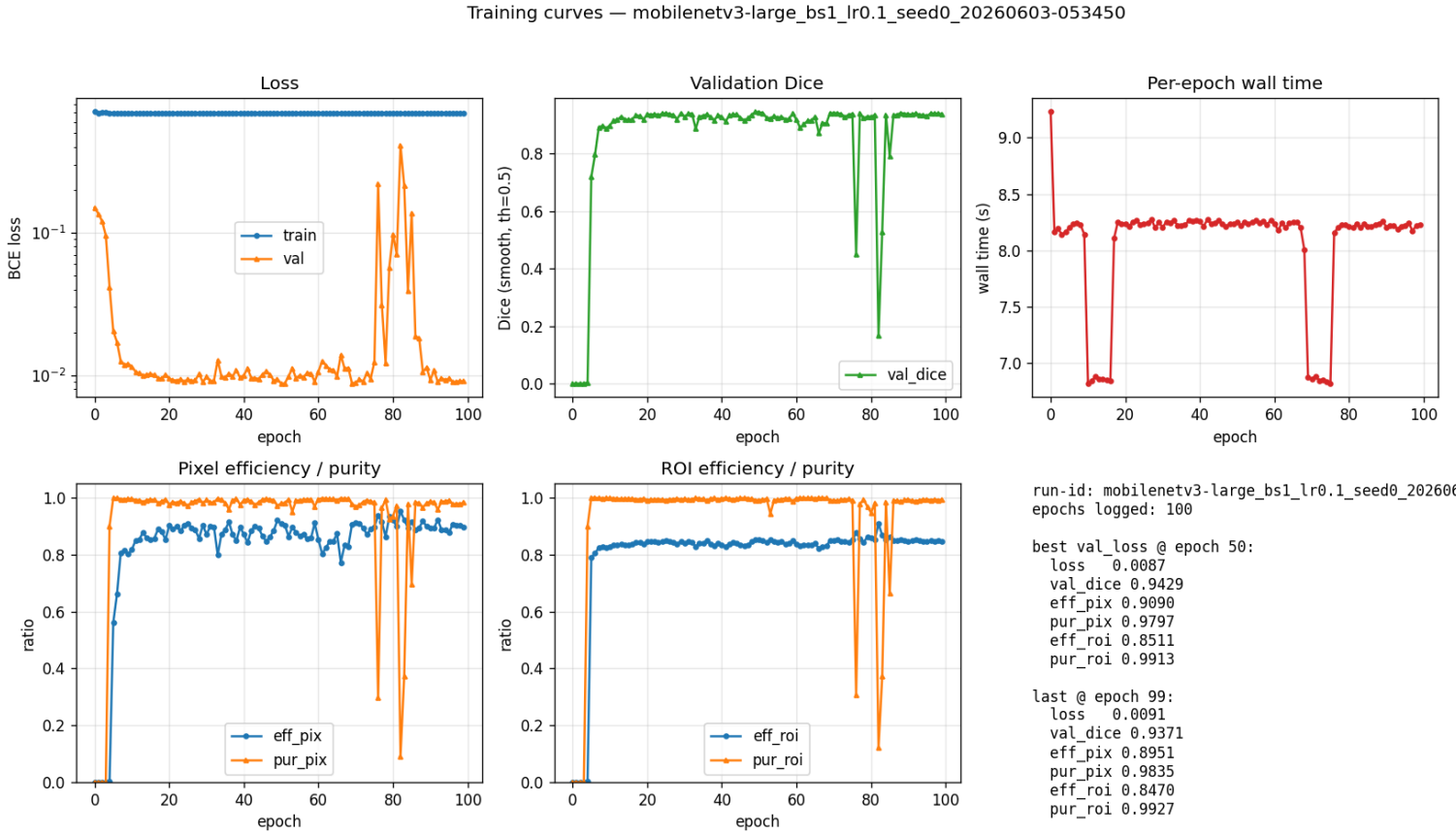
Data Pre-processing (`scripts/train.py & train.sh`)

- Set the number of `epochs = 100` and the other hyperparameters are all default values
- ~ 15min with one GPU in wcgpu1

```
================================================================================
DNN-ROI trainer (DNN_ROI_SP/scripts/train.py)
================================================================================
  processed-path  : ./data/pdhd/processed/pdhd_rebin4_th150.h5
  model           : mobilenetv3-large pretrained=True
  device          : cuda  (NVIDIA GeForce RTX 4090)
  params          : 5.19M total, 5.19M trainable
  train samples   : [0, 90)  → n_train=90
  val samples     : [90, 100)  → n_val=10
  batch-size      : 1
  epochs          : [0, 100)
  lr / momentum / weight-decay : 0.1 / 0.9 / 5e-4   (SGD, no scheduler)
  pin_memory      : True
  cudnn.benchmark : False
  AMP             : True
  channels_last   : False
  mask-dtype      : uint8
  subset-only     : None
  num-workers     : 0
  seed            : 0
```

Wire-Cell DNN Signal Processing
Reproducing PDHD Baseline model Training

Training Result with 3 channels




```
run-id: mobilenetv3-large_bs1_lr0.1_seed0_20260603-053450
epochs logged: 100

best val_loss @ epoch 50:
  loss    0.0087
  val_dice 0.9429
  eff_pix 0.9090
  pur_pix 0.9797
  eff_roi 0.8511
  pur_roi 0.9913

last @ epoch 99:
  loss    0.0091
  val_dice 0.9371
  eff_pix 0.8951
  pur_pix 0.9835
  eff_roi 0.8470
  pur_roi 0.9927
```

three additional channels contribute a further small gain. The six-channel input is adopted for all subsequent training.

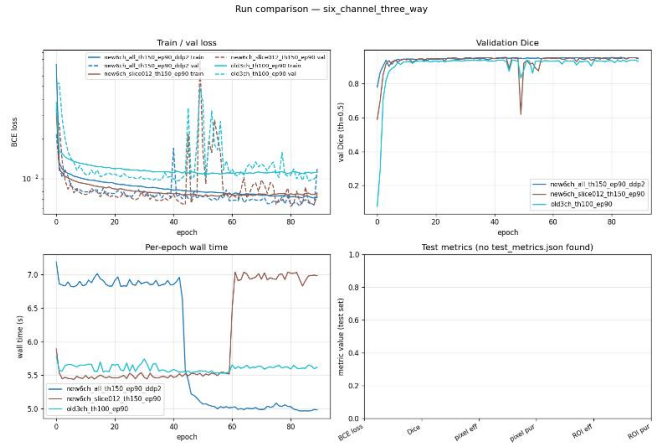


Figure 2. Three-way training comparison establishing the six-channel input: the original three-channel configuration, the same three channels drawn from the new sample, and the full six-channel input. The six-channel input does not degrade performance and yields a small additional gain.

Wire-Cell DNN Signal Processing
Reproducing PDHD Baseline model Training

To Do

- Training baseline model w/ 6 channels -> First
- Training Teacher models
- Applying Knowledge Distillations
- Also Quantization

172 three additional channels contribute a further small gain. The six-channel input is adopted for all
173 subsequent training.

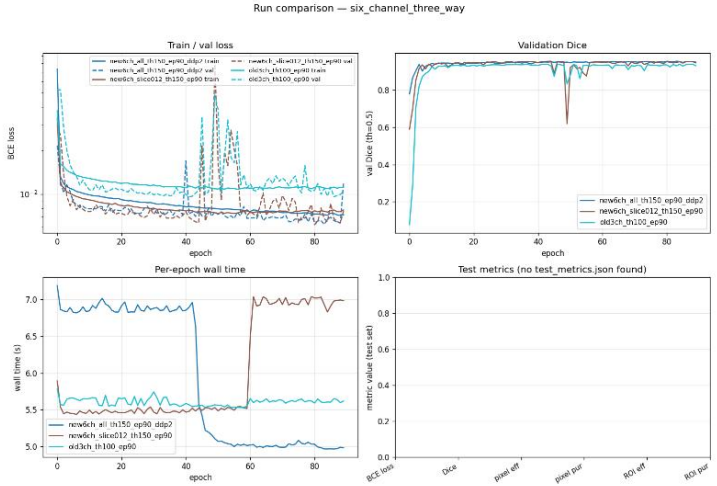


Figure 2. Three-way training comparison establishing the six-channel input: the original three-channel configuration, the same three channels drawn from the new sample, and the full six-channel input. The six-channel input does not degrade performance and yields a small additional gain.

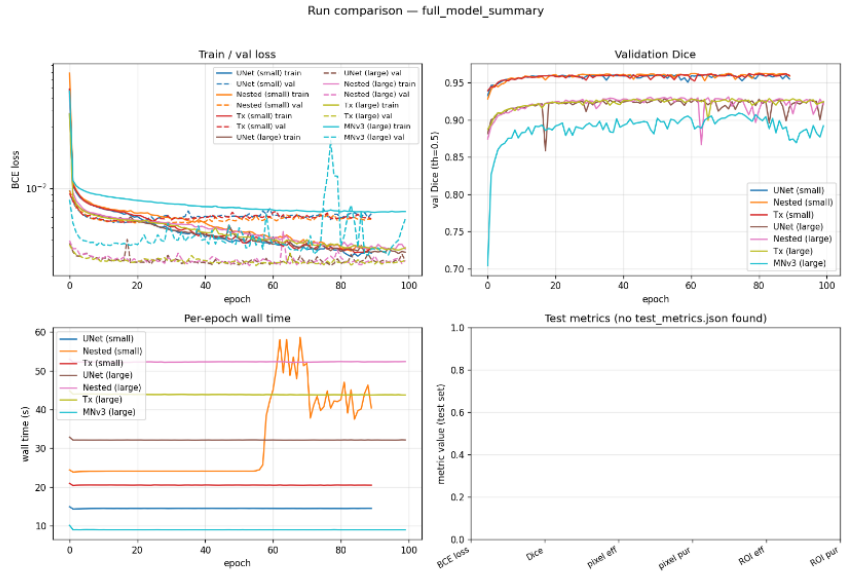


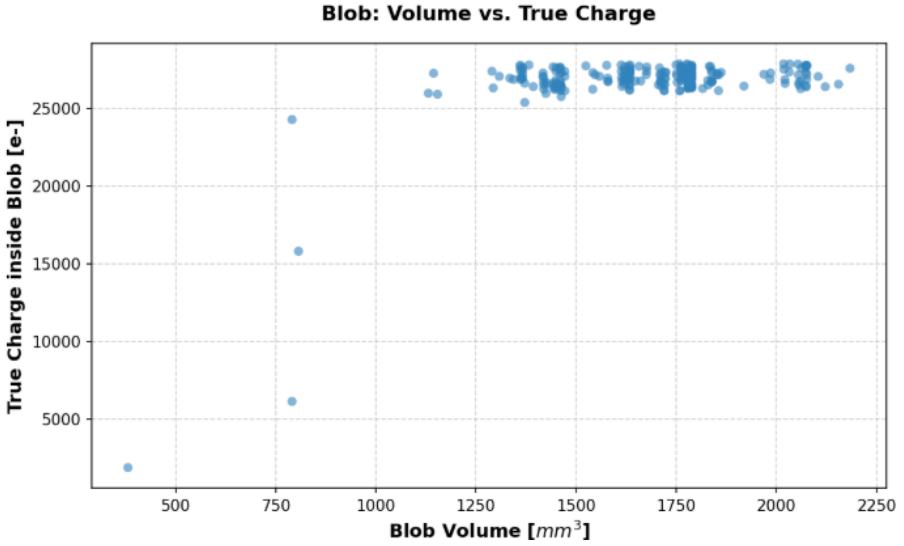
Figure 4. Training and validation curves — training loss, validation loss, validation Dice, and per-epoch wall time — for the teacher and student architectures trained on the six-channel sample.

Wire-Cell 3D Imaging & Clustering
Test Metric for Blob Size

$$\text{Blob Compactness}_i = \frac{\text{True charge in Blob}_i}{\text{Volume of Blob}_i} \cdot \frac{1}{\text{Total true charge}} = \frac{\text{Charge Density of Blob}_i}{\text{Total true charge}} \; [Length^{-3}]$$

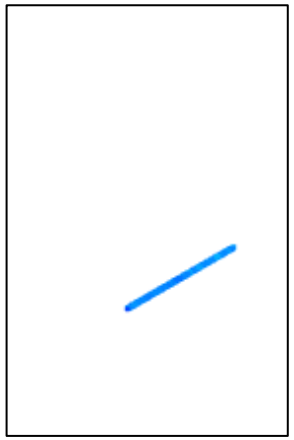
Relative Blob Compactness

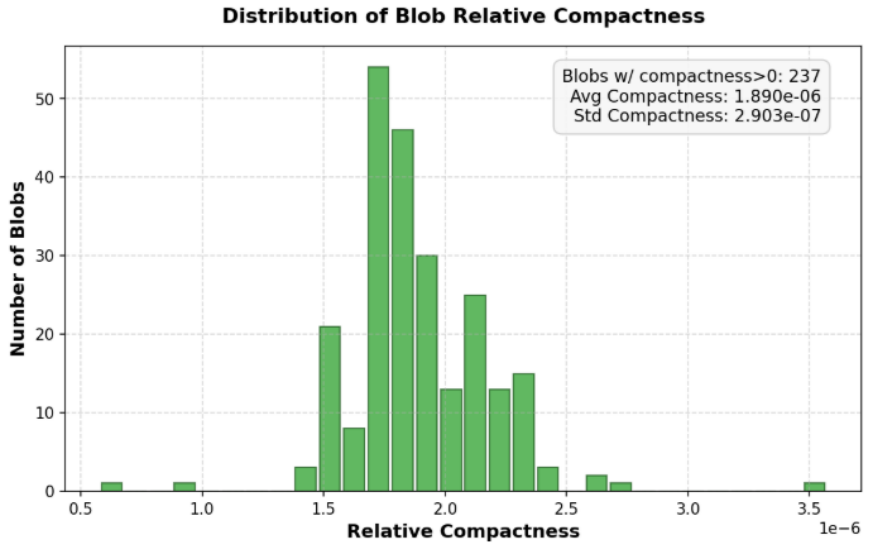
- Calculate true charge density.
- Normalized by the total true charge.
- Calculate for each blob
- Metrics from Statistical info(mean, std, etc) of the distribution



Considerations

- Relative quantity
- Factors that can affect blob size
  - Detector geometry(wire pitch, slice span, etc)
  - Track Topology(e.g. isochronous track)



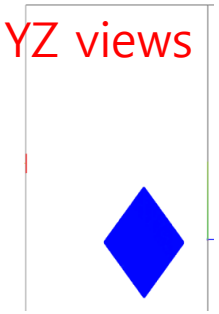


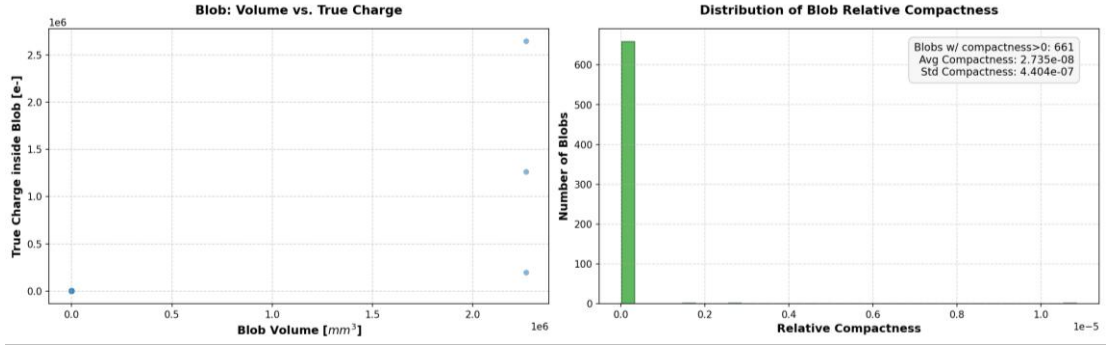
Wire-Cell 3D Imaging & Clustering

Test Metric for Blob Size

Length 100cm tracks with different thetaXZs

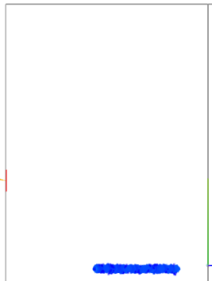
YZ views

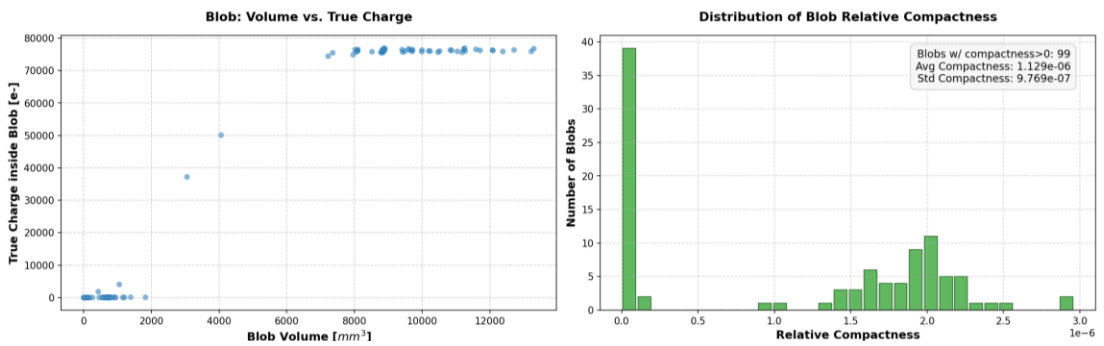




thetaXZ=0

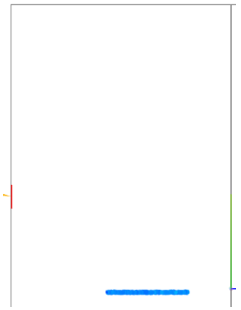
- # of blobs = 849
- Total true charge[-e] = 4.10e+6
- # of blobs w/ zero volume = 0
- # of blobs w/ zero true charge = 188
- mean = 2.735e-8
- std = 4.404e-7

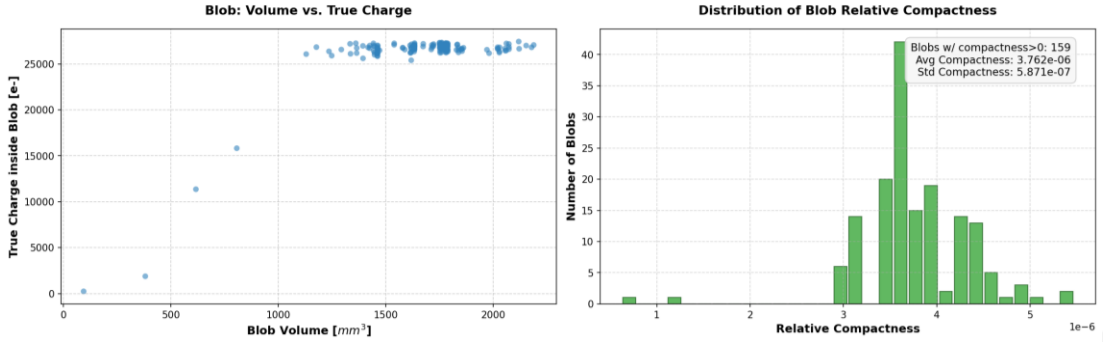




thetaXZ=10

- # of blobs = 99
- Total true charge[-e] = 4.13e+6
- # of blobs w/ zero volume = 0
- # of blobs w/ zero true charge = 0
- mean = 1.129e-6
- std = 9.769e-7





thetaXZ=30

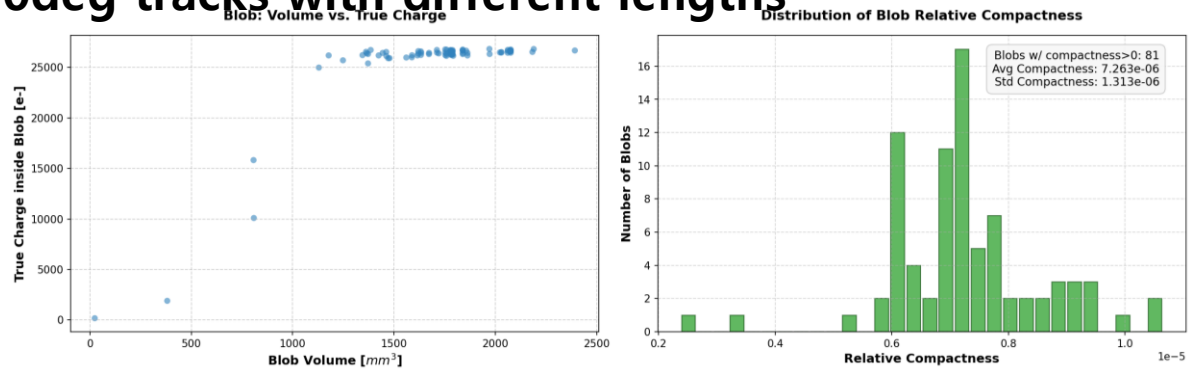
- # of blobs = 159
- Total true charge[-e] = 4.17e+6
- # of blobs w/ zero volume = 0
- # of blobs w/ zero true charge = 0
- mean = 3.762e-6
- std = 5.871e-7

Wire-Cell 3D Imaging & Clustering

Test Metric for Blob Size

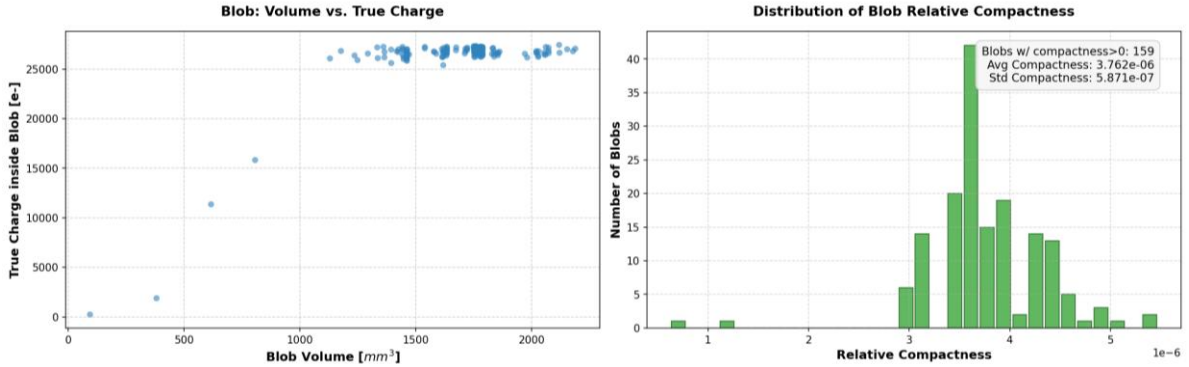
ThetaXZ=30deg tracks with different lengths

XZ views



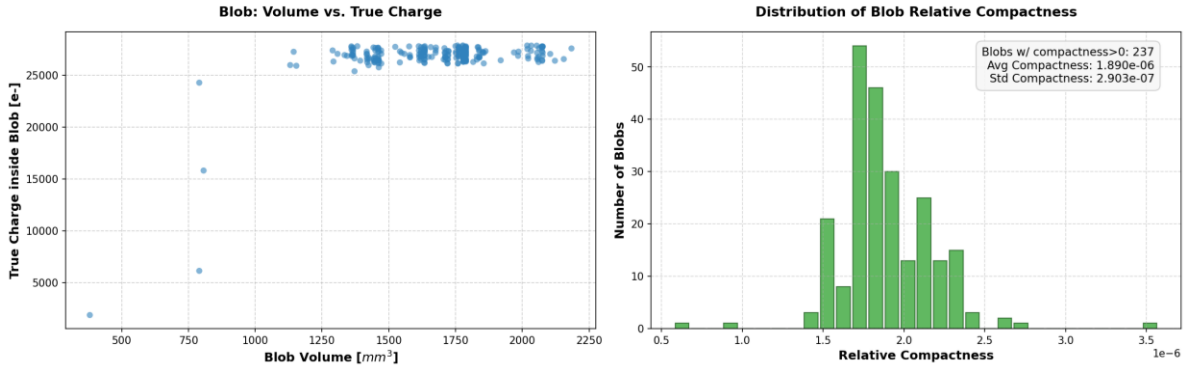
length = 50cm

- # of blobs = 81
- Total true charge[-e] = 2.06e+6
- # of blobs w/ zero volume = 0
- # of blobs w/ zero true charge = 0
- mean = 7.263e-6
- std = 1.313e-6



length = 100cm

- # of blobs = 159
- Total true charge[-e] = 4.17e+6
- # of blobs w/ zero volume = 0
- # of blobs w/ zero true charge = 0
- mean = 3.762e-6
- std = 5.871e-7



length = 200cm

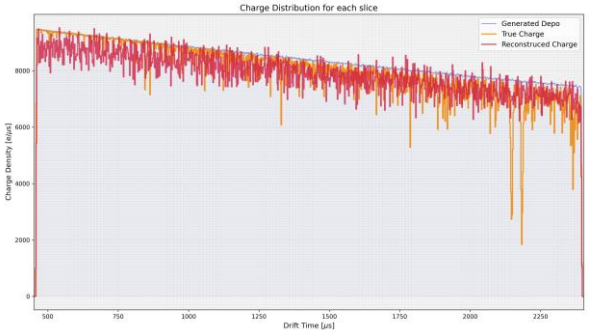
- # of blobs = 237
- Total true charge[-e] = 6.34e+6
- # of blobs w/ zero volume = 0
- # of blobs w/ zero true charge = 0
- mean = 1.890e-6
- std = 2.903e-7

Wire-Cell 3D Imaging & Clustering
Performance Evaluation with a simple track

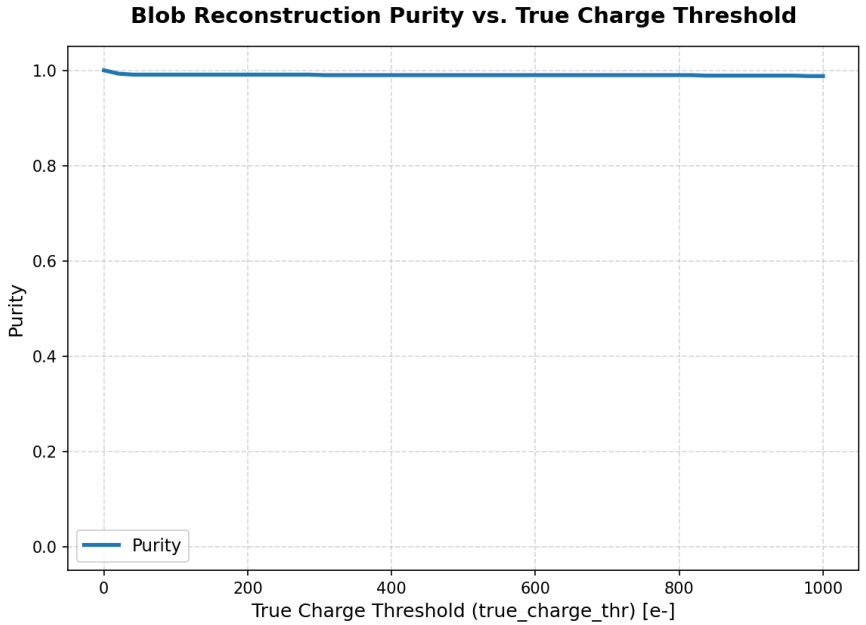
$$\text{Purity} = \frac{\text{Number of blobs with non-zero true charge}}{\text{Total number of reconstructed blobs}}$$

For a single long track (x : 10cm – 320cm)

Reco blobs **True Charge**



$$\text{Purity} = \frac{\text{Number of blobs above charge threshold}}{\text{Total number of reconstructed blobs}} \cdot 100\ [\%]$$



Charge thresholds: [0,1000]

thr=0

Category	# of Blobs
Total Reco Blobs	983
Blobs w/ True Charge(> 0)	983
FINAL PURITY	100.000000%

thr=250

Total Reco Blobs	983
Blobs w/ True Charge(> 250)	974
FINAL PURITY	99.084435%

thr=1000

Total Reco Blobs	983
Blobs w/ True Charge(> 1000)	971
FINAL PURITY	98.779247%

$$\text{Efficiency} = \frac{\text{Total true charge in blob}}{\text{Total true charge}} \cdot 100\ [\%]$$

Category	Charge Value [e-]
Total True Charge (Depos)	1.63e+07
True Charge in Blobs	1.56e+07
Lost True Charge	7.69e+05
FINAL EFFICIENCY	95.29%

Backup

Wire-Cell DNN Signal Processing
New DNN Training

MobileUnetV3 -> Student
- Light and fast inference Architecture
- High ROI purity & Low ROI efficiency *(Need to check)*

Heavier Architectures -> Teachers
- High ROI efficiency *(Need to check)*
- Much more parameters & High inference cost

- **Knowledge Distillation**
  - ➢ Several Teacher Networks and a Student Network
  - ➢ Student is supervised both by the truth mask and by the teacher's output
  - ➢ Teacher's behaviour is transferred into the compact student during training
  - ➢ Student's small size and fast inference while gaining accuracy
  - ➢ Distillation cost during the training only

Models

Role	Architecture	Class
Student	MobileNetV3-UNet	`models/mobilenetv3_unet.py`
Teacher 1	NestedUNet	`models/nestedunet.py`
Teacher 2	Transformer-UNet	`models/transformer_unet.py`

Strategy A (Response distillation)

Strategy B (Nested U-Net Architecture-aware feature distillation)

Strategy C (Transformer U-Net Architecture-aware feature distillation)

Wire-Cell DNN Signal Processing
New DNN Training

INT8 Quantization

- Related to the data type of saved parameters and the model size
- 32-bit floating point(FP32) -> 8-bit314integers (INT8)
  - Reduces the model file size and memory
  - Accelerates CPU inference
  - Some potential cost in accuracy

Post-training quantization (PTQ)

- Pros: Some memory and speed gain
- Cons: Decreases ROI purity due to the round

Quantization-aware training (QAT)

- Pros: Recovers ROI purity
- Cons: Decreases ROI efficiency decrease

Quantization-aware training with Disllition (QAT-KD)

- Recoveres ROI efficiency

Wire-Cell DNN Signal Processing
Reproducing PDHD Baseline model Training

/nfs/data/1/yujin/DNN_ROI_SP/scripts/losses.py

```
def forward(self, pred, target):
    if target.dim() == pred.dim() - 1:
        target = target.unsqueeze(1)
    # loss = F.binary_cross_entropy(pred.reshape(-1), target.reshape(-1))
    loss = F.binary_cross_entropy_with_logits(pred.reshape(-1), target.reshape(-1).float())
    if self.dice_weight > 0.0:
        loss = loss + self.dice_weight * soft_dice_loss(pred, target)
    if self.cldice_weight > 0.0:
        loss = loss + self.cldice_weight * soft_cldice_loss(
            pred, target, self.cldice_iters)
    return loss
```

Wire-Cell 3D Imaging & Clustering
Relative Blob Comacptness

Some example cases

Scenario	True Charge in Blob	Volume of Blob	Charge Density of Blob	Total True Charge	Blob Compactness	Interpretations
1. Optimal (Low Charge)	50	100	0.5	100	0.005	Perfect Baseline: The algorithm tightly bounds a blob size, establishing a clear baseline for spatial compactness.
2. Optimal (High Charge)	500	100	5.0	1,000	0.005	Bias Corrected: Even though the local charge density is 10x higher due to physical interactions, the final score remains identical (0.005) after being normalized by the total true charge.
3. Inflated (Spatial Smearing)	50	1,000	0.05	1,00	0.0005	Accurate Volume Penalty: Blob inflation drastically drops the local charge density, decreasing by 10x, successfully capturing the spatial smearing whit 1/10X compactness.
4. Leakage (Under-reco)	20	50	0.2	1,00	0.002	Accurate Leakage Penalty: The local density seems normal, but because this blob only contains a tiny fraction of depos, the total charge normalization penalizes the score.
5. Pure Ghost (False Positive)	0	100	0.0	1,00	0.0	Zero Score: The reconstructed blob yields a charge density of absolute zero. Successfully categorized as a pure ghost.