Status report on

# DNNROI sigproc

Hokyeong Nam
Chung-Ang University

# Following up Xin's progress

- Lots of progress made by Xin for DNN ROI studies including new techniques and bug fix
- All the scripts and documents are in private git repo: https://github.com/lastgeorge/DNN_ROI_SP
- Key changes
  - Create pre-processed training data separately from DNN training
  - Flexible DNN inputs: [loose_lf, mp2_roi, mp3_roi] + [**tight_lf, gauss, decon_charge]**
  - New techniques: Knowledge Distillation (KD), Quantization-Aware Training (QAT), INT8
  - Field response file for PD-VD is updated: protodunevd_FR_imbalance3p_260501.json.bz2
  - For PD-VD, training samples now include TDE data
  - DNN ROI threshold is changed from 0.5 to 0.2
  - Wire-Cell-Toolkit now supports DNNROIFindingMultiPlane
- Notes
  - Training time decreased (pre-processed training sample + upload full sample to CPU memory)
  - 6ch input + new techniques showed better performance than baseline (3ch + MobileU-Net V3) in terms of ML metrics
  - PD-VD model trained on BDE + TDE showed better performance than baseline (BDE only)

# Following up Xin's progress

- Additional bug fix required for sp-filters.jsonnet in cfg/experiment/protodunevd/

```
 97        hf('Wiener_tight_U_b', { sigma: 0.148788  * wc.megahertz, power: 3.76194 }),
 98        hf('Wiener_tight_U_t', { sigma: 0.148788  * wc.megahertz, power: 3.76194 }),
 99        hf('Wiener_tight_V_b', { sigma: 0.1596568 * wc.megahertz, power: 4.36125 }),
100        hf('Wiener_tight_V_t', { sigma: 0.1596568 * wc.megahertz, power: 4.36125 }),
101        hf('Wiener_tight_W_b', { sigma: 0.13623   * wc.megahertz, power: 3.35324 }),
102        hf('Wiener_tight_W_t', { sigma: 0.13623   * wc.megahertz, power: 3.35324 }),
```

- Parameters are from PD-SP

- Optimized filter parameters for PD-VD BDE
- Which one is used for recent training samples?

```
hf('Wiener_tight_U', {
  sigma: 0.286735  * wc.megahertz,
  power: 8.07074,
}),
hf("Wiener_tight_V", {
  sigma: 0.289041 * wc.megahertz,
  power: 9.94129 }),
hf('Wiener_tight_W', {
  sigma: 0.207127 * wc.megahertz,
  power: 4.13956,
}),
```

# Following up Xin's progress

- Trained baseline models with different inputs 3ch/6ch and converted into TorchScript files
- Training data: /nfs/data/1/xqian/toolkit-dev/DNN_ROI_SP/data/pdvd/processed/pdvd_trainval_1200ev_6ch.h5
- Updated to_torchscript.py to support input-channels (e.g. 3 or 4 or 5 or 6)

```
[ckpt] checkpoints/pdvd_base_mbv3_3ch_1200ev/CP25.pth  (epoch 25)
[data] test slice [960, 1200)  x=(240, 3, 952, 1600)  y=(240, 952, 1600)
[model] mobilenetv3  input_channels=3
[eval] all               n= 240  loss=0.0175  dice=0.7376  eff_pix=0.6949  pur_pix=0.7995  eff_roi=0.7242  pur_roi=0.8249
[eval] A (crp1,crp2)     n=  60  loss=0.0128  dice=0.7657  eff_pix=0.7326  pur_pix=0.8085  eff_roi=0.7443  pur_roi=0.8237
[eval] B (crp3,crp4)     n=  60  loss=0.0213  dice=0.6759  eff_pix=0.6314  pur_pix=0.7430  eff_roi=0.6634  pur_roi=0.7681

metric       all               A (crp1,crp2)       B (crp3,crp4)
------------------------------------------------------------------------
loss         0.0175            0.0128              0.0213
dice         0.7376            0.7657              0.6759
eff_pix      0.6949            0.7326              0.6314
pur_pix      0.7995            0.8085              0.7430
eff_roi      0.7242            0.7443              0.6634
pur_roi      0.8249            0.8237              0.7681
n            240               60                  60
```

```
[ckpt] checkpoints/pdvd_base_mbv3_6ch_1200ev/CP11.pth  (epoch 11)
[data] test slice [960, 1200)  x=(240, 6, 952, 1600)  y=(240, 952, 1600)
[model] mobilenetv3  input_channels=6
[eval] all               n= 240  loss=0.0172  dice=0.7548  eff_pix=0.7084  pur_pix=0.8092  eff_roi=0.7292  pur_roi=0.8333
[eval] A (crp1,crp2)     n=  60  loss=0.0123  dice=0.7686  eff_pix=0.7225  pur_pix=0.8222  eff_roi=0.7450  pur_roi=0.8315
[eval] B (crp3,crp4)     n=  60  loss=0.0212  dice=0.6976  eff_pix=0.6533  pur_pix=0.7505  eff_roi=0.6717  pur_roi=0.7772

metric       all               A (crp1,crp2)       B (crp3,crp4)
------------------------------------------------------------------------
loss         0.0172            0.0123              0.0212
dice         0.7548            0.7686              0.6976
eff_pix      0.7084            0.7225              0.6533
pur_pix      0.8092            0.8222              0.7505
eff_roi      0.7292            0.7450              0.6717
pur_roi      0.8333            0.8315              0.7772
n            240               60                  60
```

# Back Up