Updates on RNN τ Decay Mode Classification

Bowen Zhang
16/August/2019

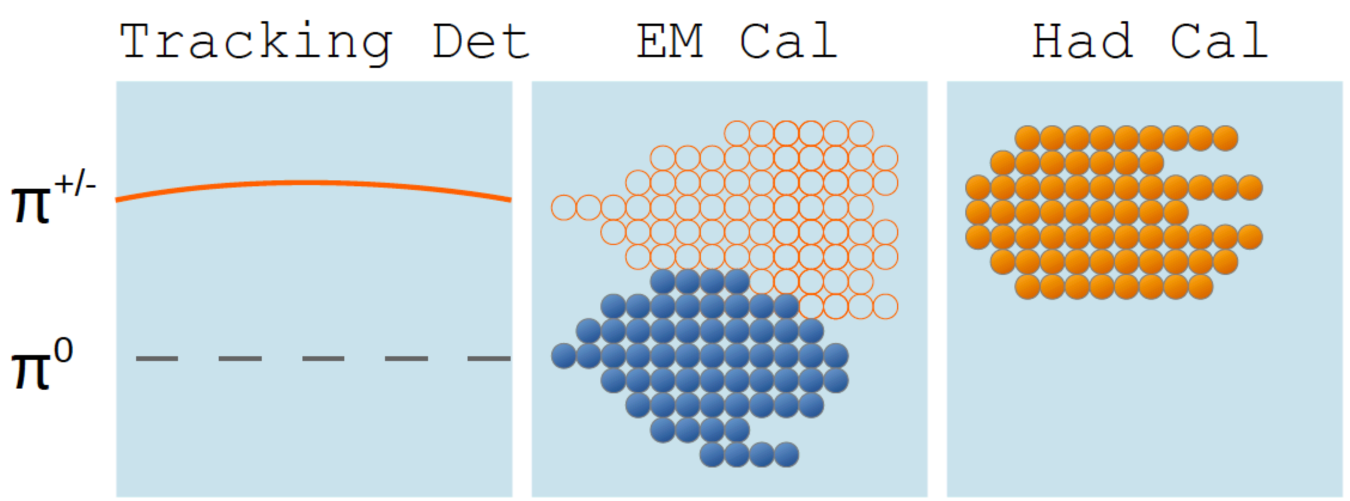
Outline

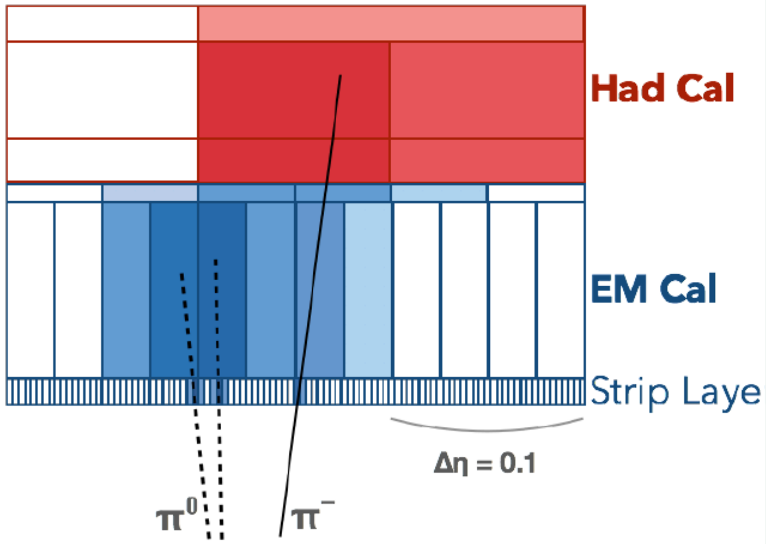
* Introduction to tau particle flow method
* BDT decay mode classification
* First try on RNN decay mode classification
* ToDos

Introduction to tau particle flow method

Lara's slides

Particle Flow





Idea: reconstruct all decay products to determine decay mode

- use tracks to identify $\pi^{+/-}$
- use energy deposition in Had Cal and tracking detector to determine the $\pi^{+/-}$ energy in the EM Cal
- subtract the $\pi^{+/-}$ energy from matched cluster in the EM Cal
- identify neutral particles from remnant clusters
- use strip-layer information to identify neutral pions (2 photon shots)

Introduction to tau particle flow method

Lara's slides

Challenges:

Convertion tracks, tracking inefficiency

- use tracks to identify $\pi^{+/-}$
- use energy deposition in Had Cal and tracking detector to determine the $\pi^{+/-}$ energy in the EM Cal
- subtract the $\pi^{+/-}$ energy from matched cluster in the EM Cal
- identify neutral particles from remnant clusters
- use strip-layer information to identify neutral pions (2 photon shots)

Distangle energy deposits from $h^{\pm}$ showers when reconstruct the energy of pi0

Lara Schildgen • Tau decay mode identification for Run 3 • Prague 06.06.19 3

1p 1n vs ≥2n: Pi0 fail selection, energy deposits merge into one cluster
3p 0n vs ≥1n: pi0s are typically soft with large overlapping $h^{\pm}$ deposits

During pi0 reconstruction, the energy deposits from both photons typically merge into a single cluster

BDT decay mode classification

Tau Decay Mode Classification

Extended **PanTau** decay modes:

1p0n + 0 add
1p0n + X add
1p1n + 0 add
1p1n + X add
1pXn + X add
3p0n + 0 add
3p0n + X add
3pXn + X add

1p0n vs. 1p1n BDT

1p1n vs. 1pXn BDT

3p0n vs. 3pXn BDT

neutrals which were not tagged as π^0

π^0-tagged neutrals (PanTau result)

prong = charged pion

Decay mode test	$N(\pi^0_{\text{cand}})$	$N(\pi^0_{\text{ID}})$	Variables
$h^{\pm}\,\{0,1\}\pi^0$	≥ 1 1	0 1	S_1^{BDT}, $f_{\pi^0,1}$, $\Delta R(h^{\pm},\pi^0)$, $D_{h^{\pm}}$, N_{γ}
$h^{\pm}\,\{1,\geq 2\}\pi^0$	≥ 2 ≥ 2	1 ≥ 2	S_2^{BDT}, f_{π^0}, m_{π^0}, N_{π^0}, N_{γ}
$3h^{\pm}\,\{0,\geq 1\}\pi^0$	≥ 1 ≥ 1	0 ≥ 1	S_1^{BDT}, f_{π^0}, $\sigma_{E_T,h^{\pm}}$, $m_{h^{\pm}}$, N_{γ}

INPUT

from imperfect h± substraction, pile-up and underlying event

METHOD

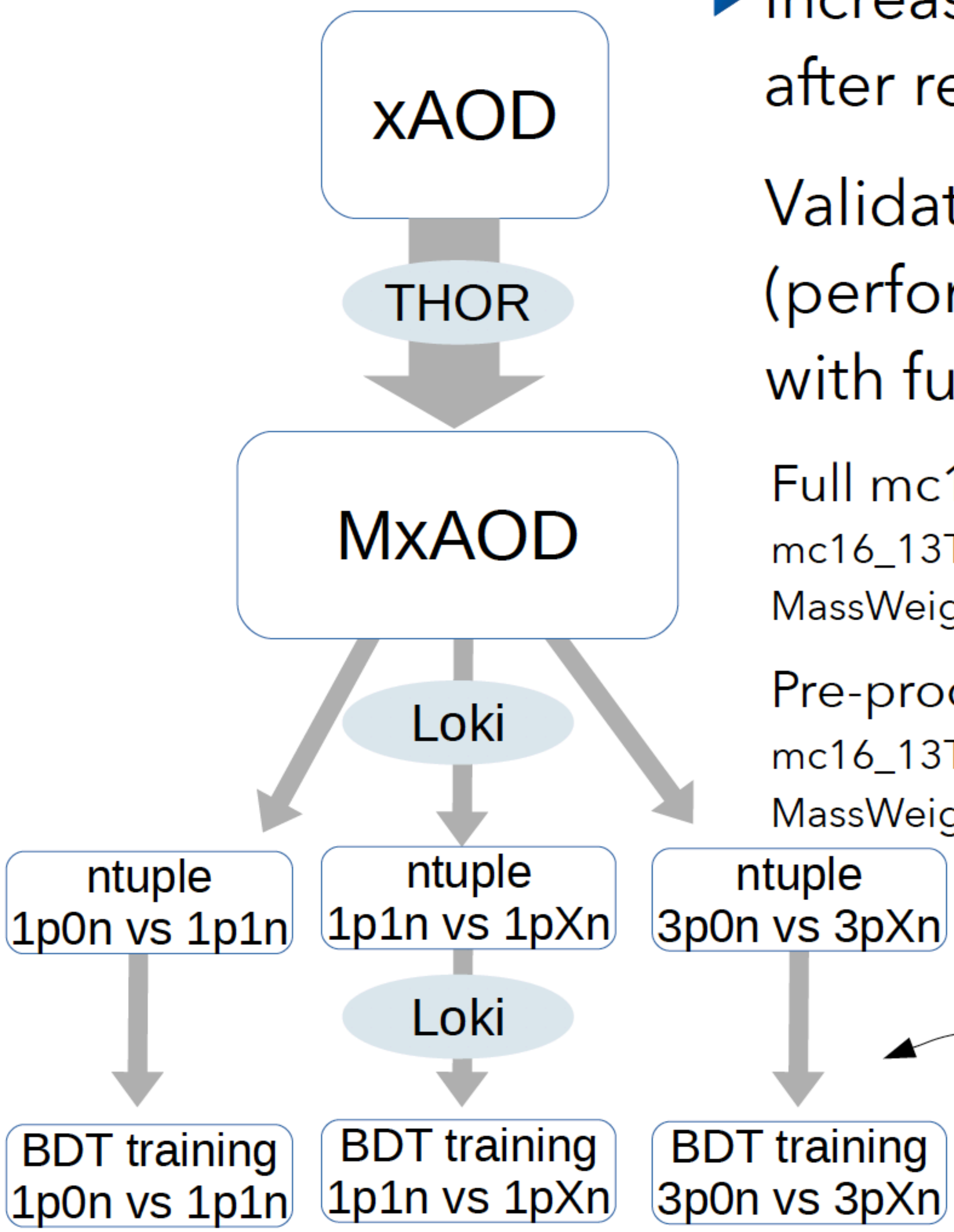
Improving decay mode classification with machine learning techniques in Boosted Decision Trees (BDT)

OUTPUT

Resulting information for analyses:
- decay mode
- preliminary τ 4-vector = 4-vectors of charged + neutral pions

BDT Training
mc16D

BDT Re-Training



- Increase of decay mode classification efficiency after re-training of the three PanTau BDTs for R21

  Validation of R21 PanTau BDT tuning (performed on pre-production sample) with full mc16D sample

  Full mc16D Gammatautau sample:
  mc16_13TeV.425200.Pythia8EvtGen_A14NNPDF23LO_Gammatautau_MassWeight.merge.AOD.e5468_s3126_r10201_r10210

  Pre-production Gammatautau sample used for R21 tuning:
  mc16_13TeV.425200.Pythia8EvtGen_A14NNPDF23LO_Gammatautau_MassWeight.merge.AOD.e5468_s2997_r8903_r8906

BDT training of 3 decay mode specific BDTs using extended tau decay mode

Migration Matrices

BDT Training mc16D

"pre-training" = trained on preprod sample

"post-training" = trained on full mc16D sample

evaluation: Gammatautau

pre-training

ATLAS Simulation Internal — Diagonal: 73.1% Efficiency

Reco Tau Decay Mode \ True Tau Decay Mode	1p0n	1p1n	1pXn	3p0n	3pXn
3pXn	0.0	0.5	0.5	5.6	58.5
3p0n	0.1	0.1	0.1	91.2	36.1
1pXn	2.0	11.1	40.0	0.5	1.7
1p1n	17.7	77.7	56.6	1.5	3.3
1p0n	80.3	10.5	2.8	1.2	0.4

post-training

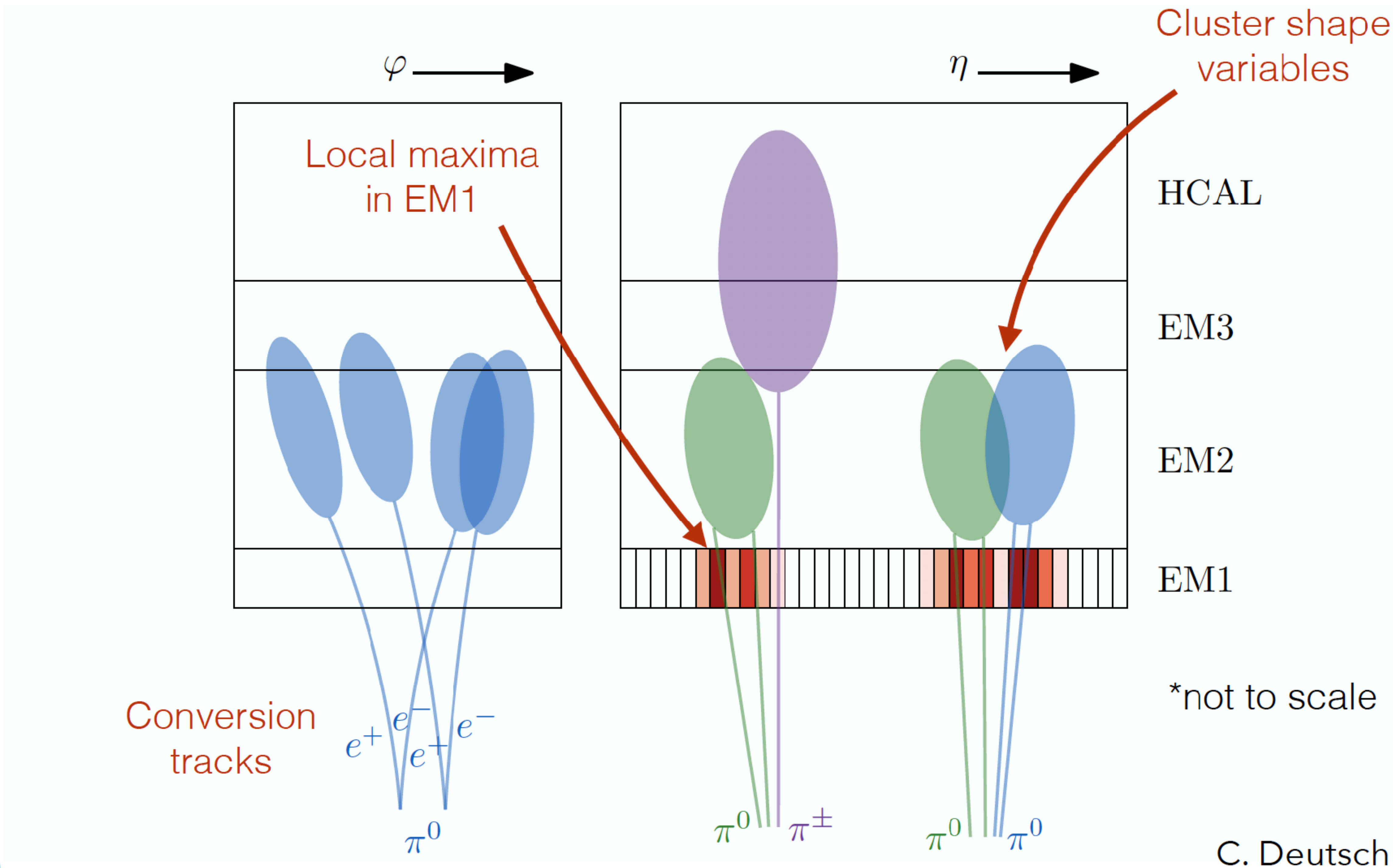
ATLAS Simulation Internal — Diagonal: 73.1% Efficiency

Reco Tau Decay Mode \ True Tau Decay Mode	1p0n	1p1n	1pXn	3p0n	3pXn
3pXn	0.0	0.5	0.5	5.4	58.0
3p0n	0.1	0.1	0.1	91.4	36.6
1pXn	2.0	11.2	40.4	0.5	1.7
1p1n	16.6	77.2	56.1	1.4	3.3
1p0n	81.3	11.0	2.9	1.2	0.4

- Stable performance after PanTau BDT re-training on full mc16D sample (decay mode reconstruction efficiency differences <1%)
- Find all evaluation plots in the plotbook

RNN decay mode classification

- Based on Christopher Deutsch's study: link to documentation
- Using additional information:



Input variables

- Using additional information:

h± from track in inner detector → $\phi, \Delta\phi(\text{pfo}, \tau), \eta, \Delta\eta(\text{pfo}, \tau), p_{\text{T}}, p_{\text{T}}(\tau)$

π^0 from cluster in EM calorimeter → $\phi, \Delta\phi(\text{pfo}, \tau), \eta, \Delta\eta(\text{pfo}, \tau), p_{\text{T}}, p_{\text{T}}(\tau)$,

- xAOD::PFODetails::PFOAttributes::cellBasedNHitsInEM1, SECOND_R, ENG_FRAC_CORE, NPosECells_EM1, secondEtaWRTClusterPosition_EM1
- pfo->bdtPi0Score

$$\text{EnergyFracEM2} = \text{energy\_EM2/pfo->cluster(0)->e()}$$

$$\text{PtSubRatio} = \frac{\text{pfo->cluster(0)->pt()} - \text{pfo->pt()}}{\text{pfo->cluster(0)->pt()}}$$

From track classification → $\phi, \Delta\phi(\text{trk}, \tau), \eta, \Delta\eta(\text{trk}, \tau), p_{\text{T}}, p_{\text{T}}(\tau)$

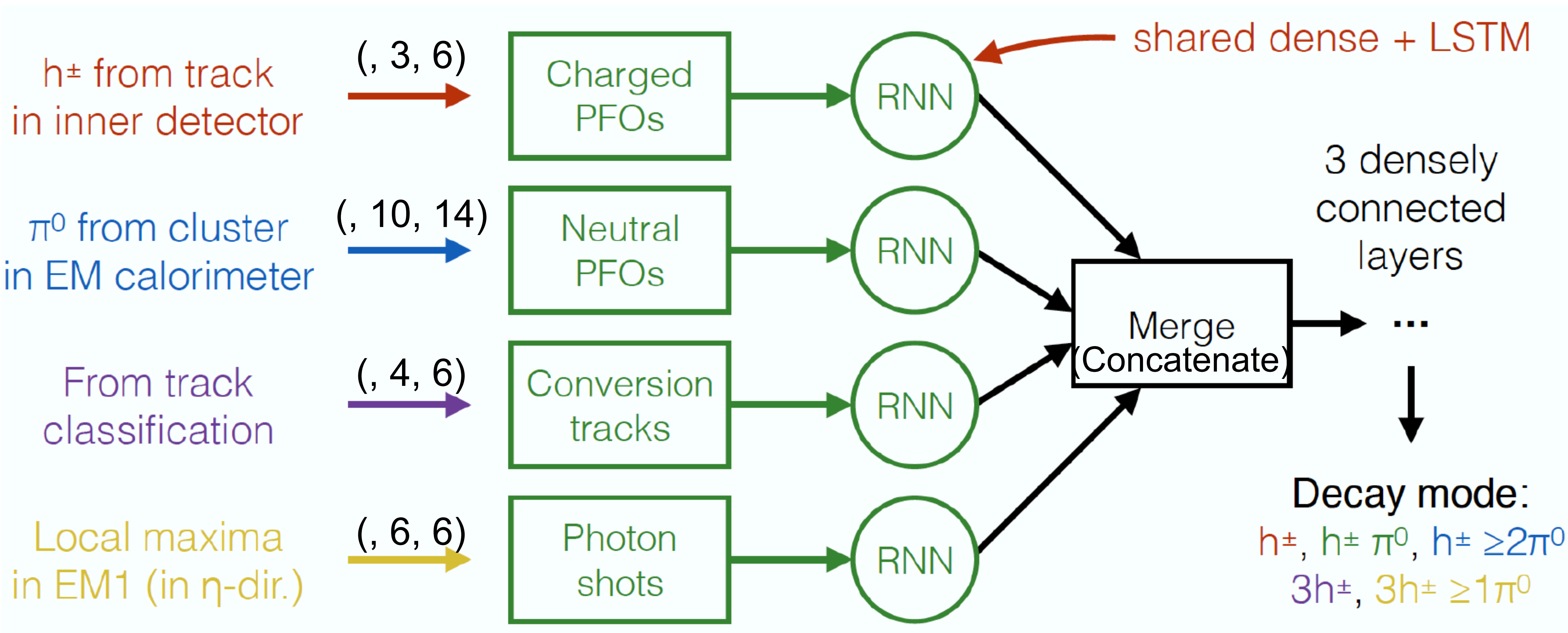
Local maxima in EM1 (in η-dir.) → $\phi, \Delta\phi(\text{pfo}, \tau), \eta, \Delta\eta(\text{pfo}, \tau), p_{\text{T}}, p_{\text{T}}(\tau)$

Note: Needs to apply some preprocessing strategies (log, abs, scale, offset, …) to ensure stable training.

RNN Architecture

Shape: (n_samples, timesteps, features)

i.e. (n_samples, n_objects, n_variables)
objects are sorted in descending pT, fill zeros to control dimensions



RNN Architecture: technical details

Example: ChargedPFO branch:

Input Layer

- Three time steps (nChargedPFOs should be equal to nChargedTracks, then this is always 1 or 3)
- Six features

Masking Layer

- Skip the empty time steps (zeros array)

TimeDistribute Layer

- Apply a dense layer on each time step
- Extend features from 6 to 24, shared by all time steps
- Could be optimal for RNN

LSTM (Long-Short Term Memory) Layer

- 24 input neurons, 24 hidden neurons, 24 output neurons

Concatenate with other branches and feed to the final densely connected Layer

First try

- Train on mc16d Gammatautau sample: mc16_13TeV.425200.Pythia8EvtGen_A14NNPDF23LO_Gammatautau_MassWeight.merge.AOD.e5468_s3126_r10201_r10210
- Before the training, same selections are applied as BDT. (Finally after the selections and random shuffling, NumPy arrays are stored in a series of HDF5 files.)
- ~6M for training (10% validation), ~6M for testing
- ~12h training time (stop when the loss doesn't improve, use the model with the minimum loss)
- Hyper-parameters are detailed in backup

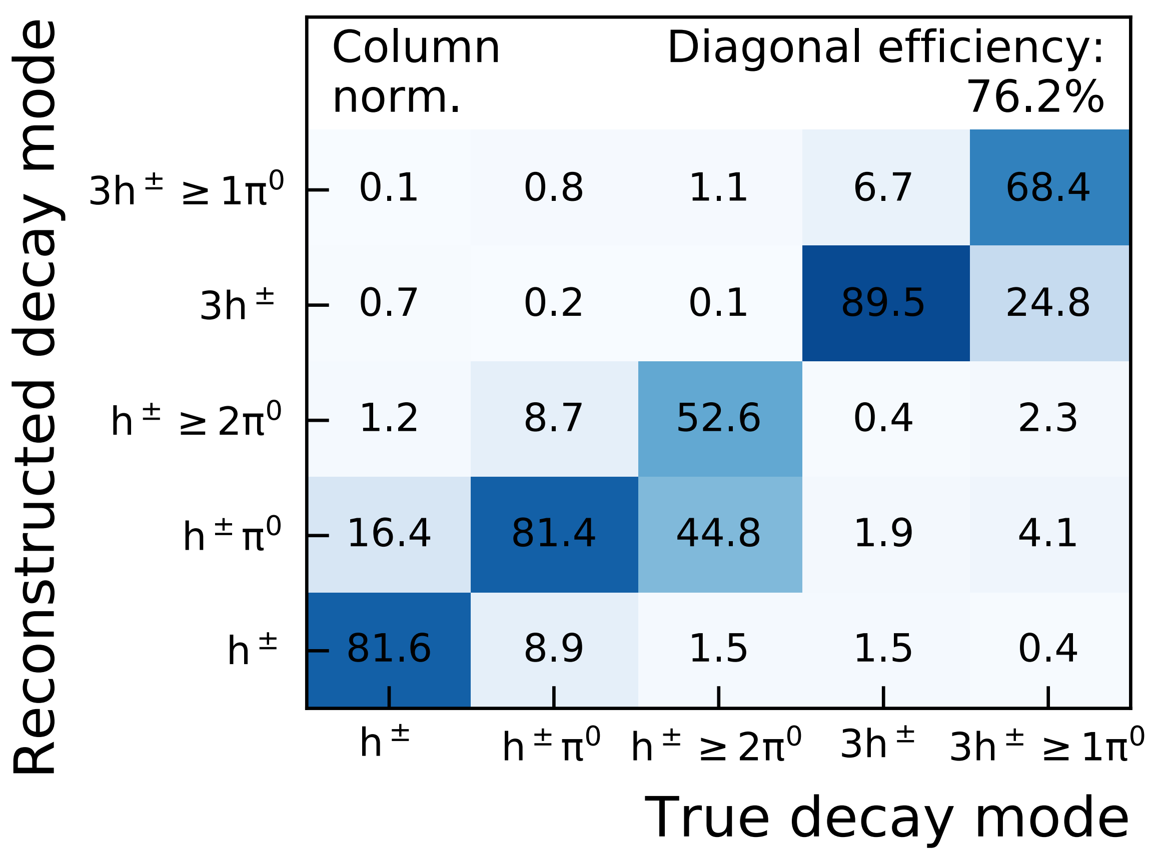
First try

- Noticed bug: Delta Phis and Delta Etas are zero (looks fine in the MxAOD, miscalculate in the code.) Could lose a lot of important information! (~40% information and may lead to strange behaviours of an RNN)
- NOT fixed yet..

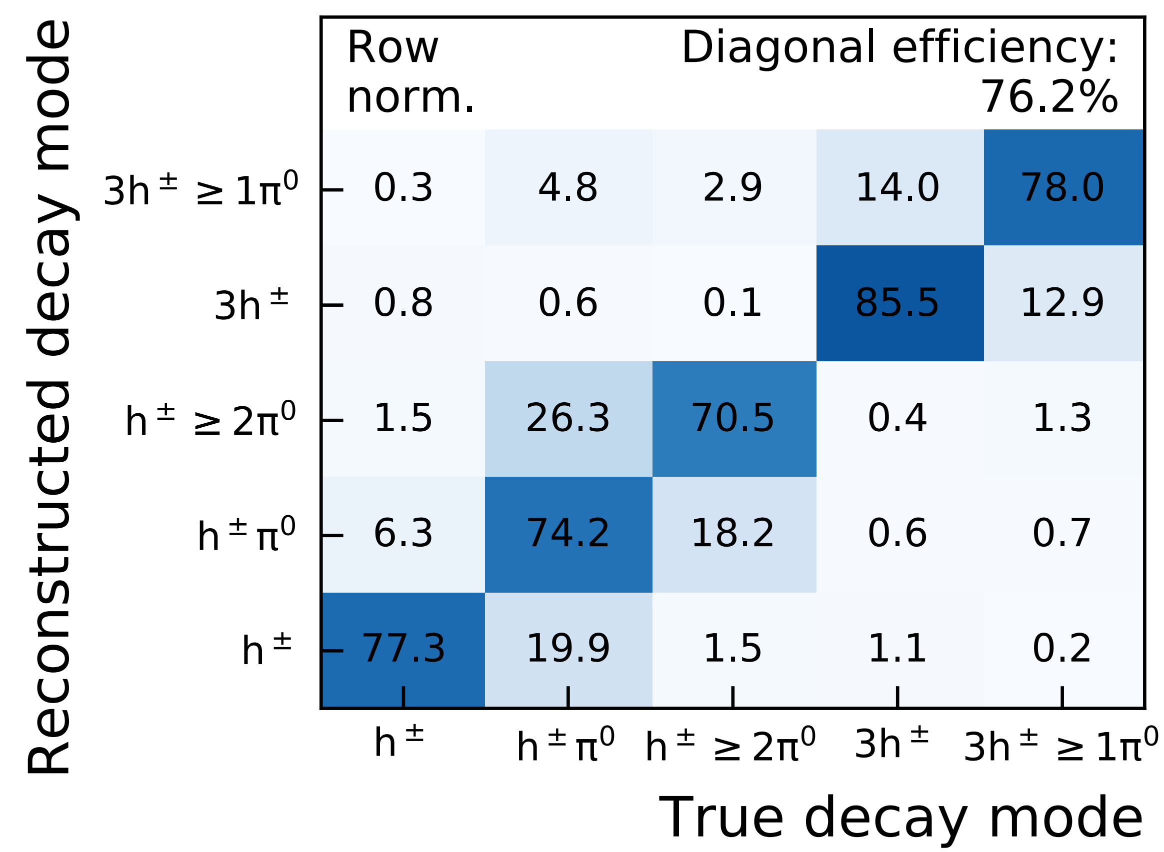
```
(Pdb) chrg_train.x[3]
array([[-0.15978548,  0.        ,  0.084272  ,  0.        ,  0.83250107,         1.98806074],
       [-0.15978548,  0.        ,  0.084272  ,  0.        ,  0.72135359,         1.98806074],
       [-0.15978548,  0.        ,  0.084272  ,  0.        , -0.2773456 ,         1.98806074]])
```

First try result: Migration matrices

Efficiency



Purity



ToDos

- Investigate input variables (current/new branch, TauJets variables, High level variables, how are they related to BDT inputs)
- Improve workflow
- Evaluation, get hidden information from the NN (what is remembered)
- NN architecture: …
- Harmonise with other algorithms

Backup

BDT inputs

π^0 identification score of the first π^0_{cand}, S_1^{BDT} π^0 identification score of the π^0_{cand} with the highest π^0 identification score
E_T fraction of the first π^0_{cand}, $f_{\pi^0,1}$ E_T of the π^0_{cand} with the highest π^0 identification score, divided by the E_T-sum of all π^0_{cand}'s and $h^\pm$'s
Hadron separation, $\Delta R(h^\pm, \pi^0)$ ΔR between the $h^\pm$ and the π^0_{cand} with the highest π^0 identification score
$h^\pm$ distance, $D_{h^\pm}$ E_T-weighted ΔR between the $h^\pm$ and the $\tau_{\text{had-vis}}$ axis, which is calculated by summing the four-vectors of all $h^\pm$'s and π^0_{cand}'s
Number of photons, N_γ Total number of photons in the $\tau_{\text{had-vis}}$, as reconstructed in Section 3.3
π^0 identification score of second π^0_{cand}, S_2^{BDT} π^0 identification score of the π^0_{cand} with the second-highest π^0 identification score
π^0_{cand} E_T fraction, f_{π^0} E_T-sum of π^0_{cand}'s, divided by the E_T-sum of π^0_{cand}'s and $h^\pm$'s
π^0_{cand} mass, m_{π^0} Invariant mass calculated from the sum of π^0_{cand} four-vectors
Number of π^0_{cand}, N_{π^0}
Standard deviation of the $h^\pm$ p_T, $\sigma_{E_T,h^\pm}$ Standard deviation, calculated from the p_T values of the $h^\pm$'s for $\tau_{\text{had-vis}}$ with three associated tracks
$h^\pm$ mass, $m_{h^\pm}$ Invariant mass calculated from the sum of $h^\pm$ four-vectors

Table 4: Variables used in the BDTs for the $\tau_{\text{had-vis}}$ decay mode classification. They are designed to discriminate against additional misidentified π^0_{cand}'s, which usually come from imperfect subtraction, pile-up or the underlying event.

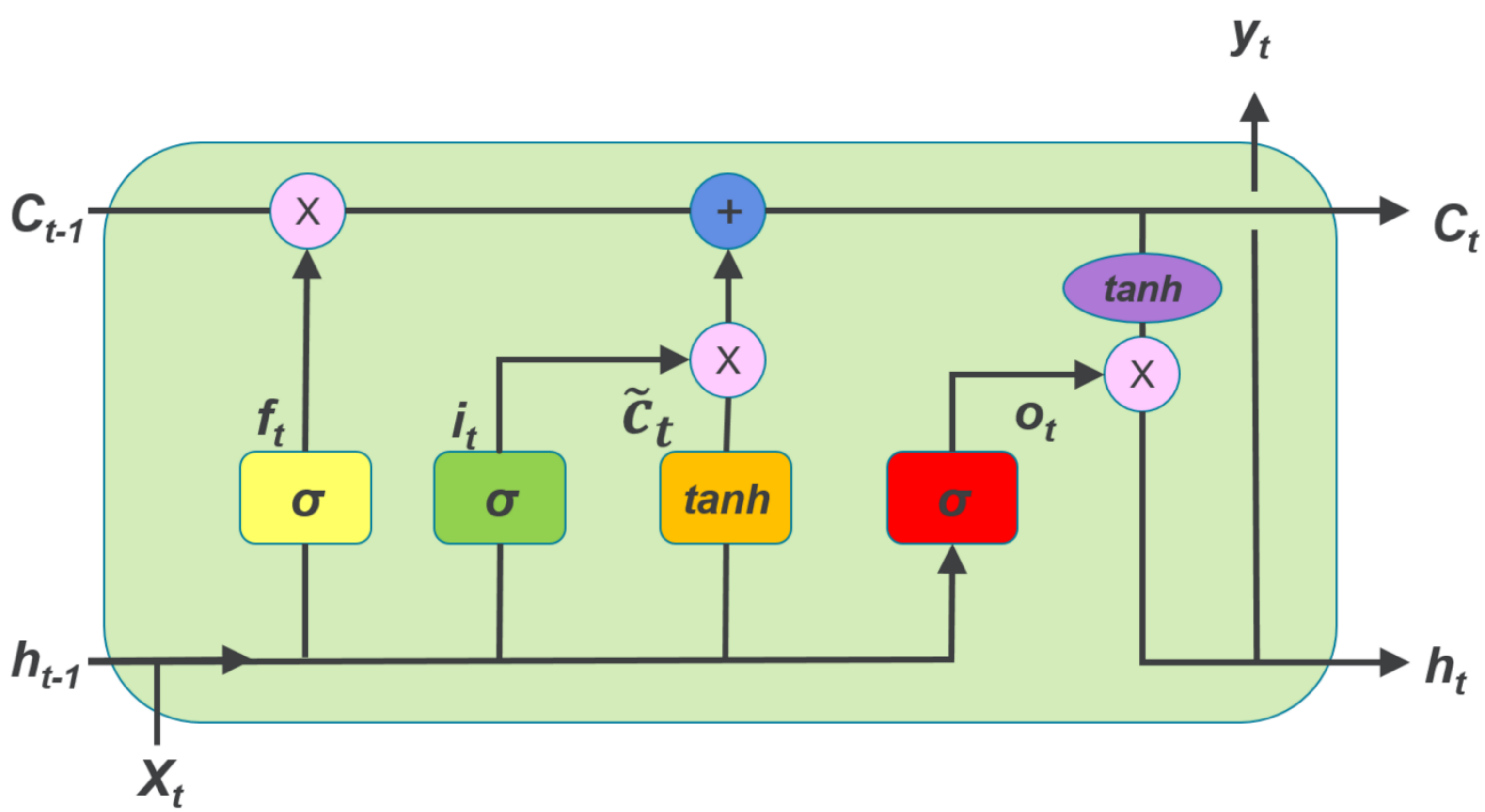
Model

```
INFO:main:____________________________________________________________________________________________________
INFO:main:Layer (type)                     Output Shape          Param #     Connected to
INFO:main:====================================================================================================
INFO:main:input_1 (InputLayer)             (None, 3, 6)          0
INFO:main:____________________________________________________________________________________________________
INFO:main:input_2 (InputLayer)             (None, 10, 14)        0
INFO:main:____________________________________________________________________________________________________
INFO:main:input_3 (InputLayer)             (None, 6, 6)          0
INFO:main:____________________________________________________________________________________________________
INFO:main:input_4 (InputLayer)             (None, 4, 6)          0
INFO:main:____________________________________________________________________________________________________
INFO:main:masking_1 (Masking)              (None, 3, 6)          0           input_1[0][0]
INFO:main:____________________________________________________________________________________________________
INFO:main:masking_2 (Masking)              (None, 10, 14)        0           input_2[0][0]
INFO:main:____________________________________________________________________________________________________
INFO:main:masking_3 (Masking)              (None, 6, 6)          0           input_3[0][0]
INFO:main:____________________________________________________________________________________________________
INFO:main:masking_4 (Masking)              (None, 4, 6)          0           input_4[0][0]
INFO:main:____________________________________________________________________________________________________
INFO:main:time_distributed_1 (TimeDistrib (None, 3, 24)          168         masking_1[0][0]
INFO:main:____________________________________________________________________________________________________
INFO:main:time_distributed_2 (TimeDistrib (None, 10, 24)         360         masking_2[0][0]
INFO:main:____________________________________________________________________________________________________
INFO:main:time_distributed_3 (TimeDistrib (None, 6, 16)          112         masking_3[0][0]
INFO:main:____________________________________________________________________________________________________
INFO:main:time_distributed_4 (TimeDistrib (None, 4, 16)          112         masking_4[0][0]
INFO:main:____________________________________________________________________________________________________
INFO:main:lstm_1 (LSTM)                    (None, 24)            4704        time_distributed_1[0][0]
INFO:main:____________________________________________________________________________________________________
INFO:main:lstm_2 (LSTM)                    (None, 24)            4704        time_distributed_2[0][0]
INFO:main:____________________________________________________________________________________________________
INFO:main:lstm_3 (LSTM)                    (None, 16)            2112        time_distributed_3[0][0]
INFO:main:____________________________________________________________________________________________________
INFO:main:lstm_4 (LSTM)                    (None, 16)            2112        time_distributed_4[0][0]
INFO:main:____________________________________________________________________________________________________
INFO:main:concatenate_1 (Concatenate)      (None, 80)            0           lstm_1[0][0]
INFO:main:                                                                   lstm_2[0][0]
INFO:main:                                                                   lstm_3[0][0]
INFO:main:                                                                   lstm_4[0][0]
INFO:main:____________________________________________________________________________________________________
INFO:main:dense_5 (Dense)                  (None, 64)            5184        concatenate_1[0][0]
INFO:main:____________________________________________________________________________________________________
INFO:main:dense_6 (Dense)                  (None, 32)            2080        dense_5[0][0]
INFO:main:____________________________________________________________________________________________________
INFO:main:dense_7 (Dense)                  (None, 5)             165         dense_6[0][0]
INFO:main:====================================================================================================
INFO:main:Total params: 21,813
INFO:main:Trainable params: 21,813
INFO:main:Non-trainable params: 0
INFO:main:____________________________________________________________________________________________________
```

Model

```
opt = SGD(lr=0.01, momentum=0.9, nesterov=True)
model.compile(loss="categorical_crossentropy", optimizer="adam",
              metrics=["categorical_accuracy"])

callbacks = []

early_stopping = EarlyStopping(
    monitor="val_loss", min_delta=0.0001, patience=10, verbose=1)
callbacks.append(early_stopping)

model_checkpoint = ModelCheckpoint(
    args.model, monitor="val_loss", save_best_only=True, verbose=1)
callbacks.append(model_checkpoint)


hist = model.fit(
        [chrg_train.x, neut_train.x, shot_train.x, conv_train.x],
        chrg_train.y, sample_weight=chrg_train.w,
        validation_data=([chrg_test.x, neut_test.x, shot_test.x,
conv_test.x], chrg_test.y, chrg_test.w),
        epochs=100, batch_size=256, callbacks=callbacks,
        verbose=1)
```

LSTM



$$\tilde{c}^{<t>} = \tanh(W_c[a^{<t-1>}, x^{<t>}] + b_c)$$

$$\Gamma_u = \sigma(W_u[\, a^{<t-1>}, x^{<t>}] + b_u)$$

$$\Gamma_f = \sigma(W_f[\, a^{<t-1>}, x^{<t>}] + b_f)$$

$$\Gamma_o = \sigma(W_o[\, a^{<t-1>}, x^{<t>}] + b_o)$$

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + \Gamma_f * c^{<t-1>}$$

$$a^{<t>} = \Gamma_o * c^{<t>}$$