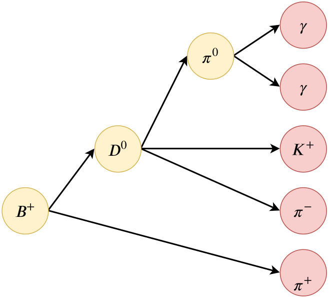
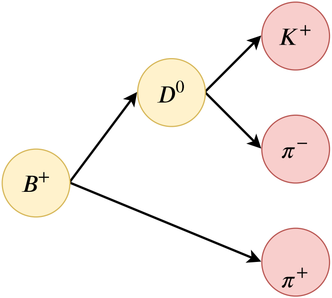
Deep Full Event Interpretation

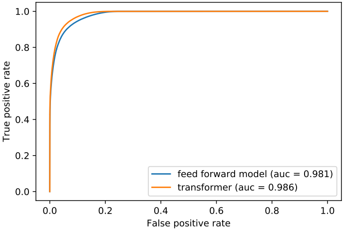
Deep Learning Hackathon Dresden

William Sutcliffe, Jochen Gemmler, Moritz Bauer, Tobias Böckh, Jeffrey Kelling | 13.09.2019

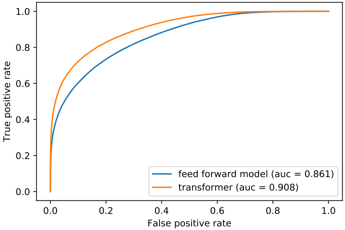

B^+
D^0
π^0
γ
γ
K^+
π^-
π^+

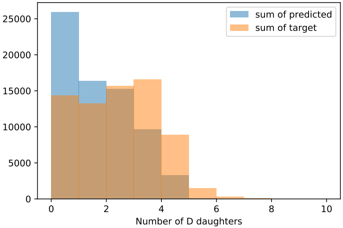

B^+
D^0
K^+
π^-
π^+

$\mathbf{B} \to \mathbf{D}(\to \mathbf{K}\pi\pi^0)\pi$ $\mathbf{B} \to \mu\nu$



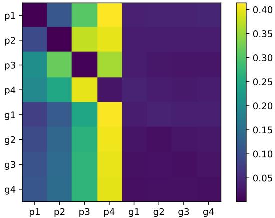
$B \to D(\to K\pi\pi^0)\pi$



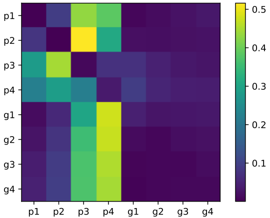


Attention Maps

- accuracy self-attention network: 0.9379
- accuracy feed forward network: 0.8292



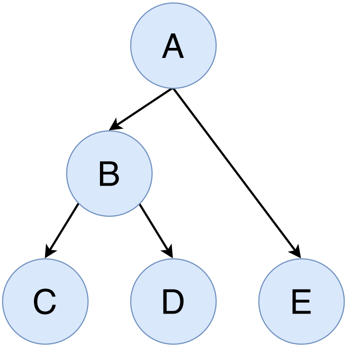
(a) $D \rightarrow K\pi$

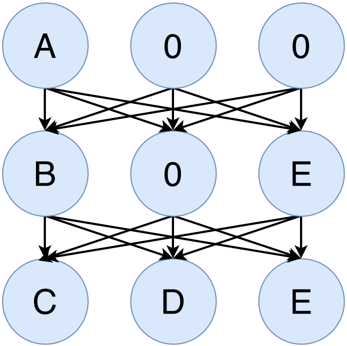


(b) $D \rightarrow K\pi\pi^0$

New architecture

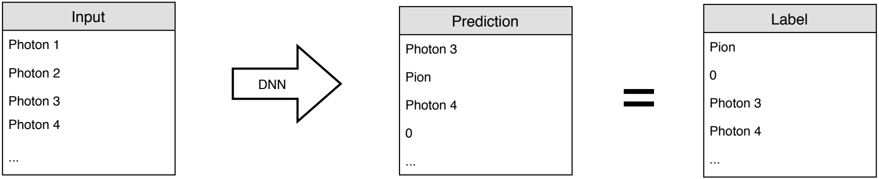
- predict physical values
- predict different particles with same layer





Permutation invariant loss function

Problem: Loss should be invariant under row swaps between label and prediction (particle order doesn't matter).



Permutation invariant loss function

Solution: Custom loss function.
Not enough time to implement Hungarian algorithm in TF so wrapping a scipy function.

```
def linassign(y_true, y_pred):

    loss = np.matmul(np.square(y_true - y_pred), np.ones(yt.shape[::-1]))
    _ , c = scipy.optimize.linear_sum_assignment(loss)

    return y_true[c]

def assignInvLoss(y_true, y_pred):

    return mean_squared_error(y_pred,tf.py_func(linassign, inp=[y_true, y_pred], Tout=y_true.dtype))
```

First results