Discrete detector track-finding

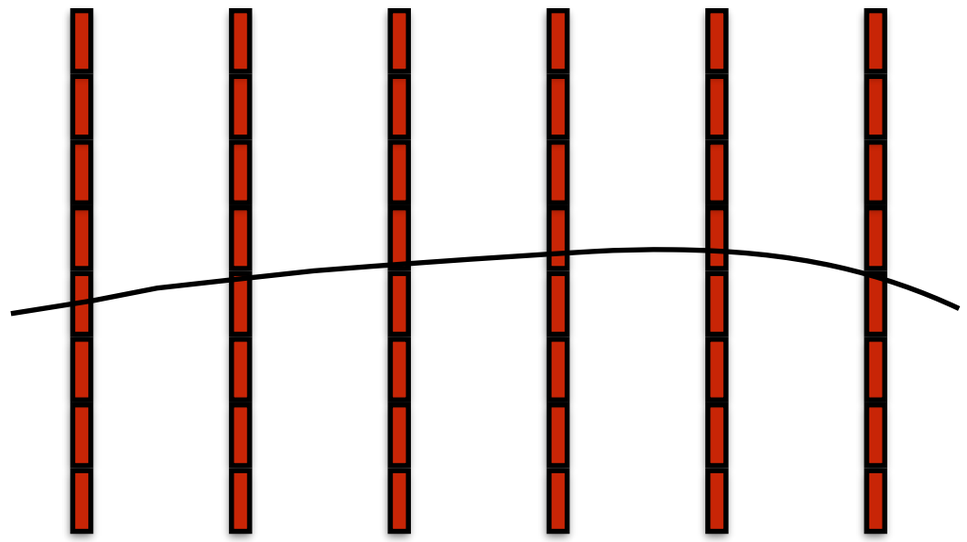
Steve Farrell

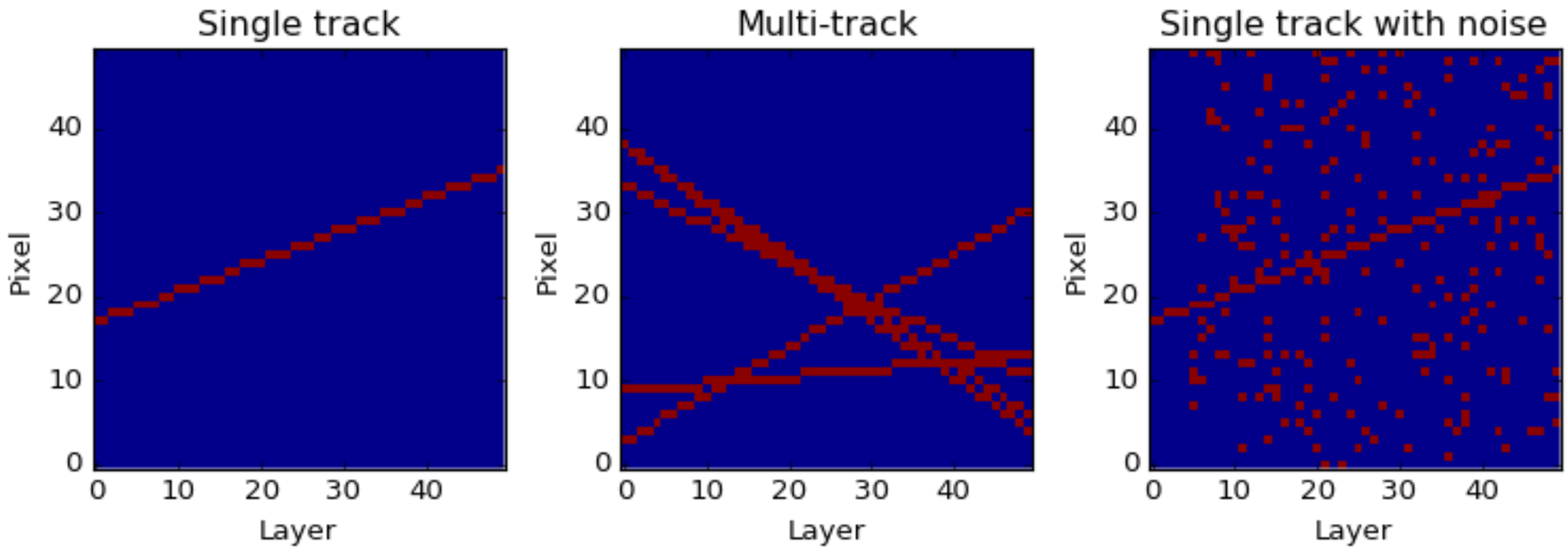
HEP.TrkX Project

Introduction

- Breadth-first search through some possible ideas to find tracks using *toy detector data*
  - KF-like extrapolations with RNNs (building off of Mayur's work)
    - discrete detector hit space (e.g. raw pixels)
    - Trying to handle hit combinatorics in one pass
- Not yet clear what will work and what won't
  - Just having fun at this point
  - Helps frame our discussions and stimulate brainstorming

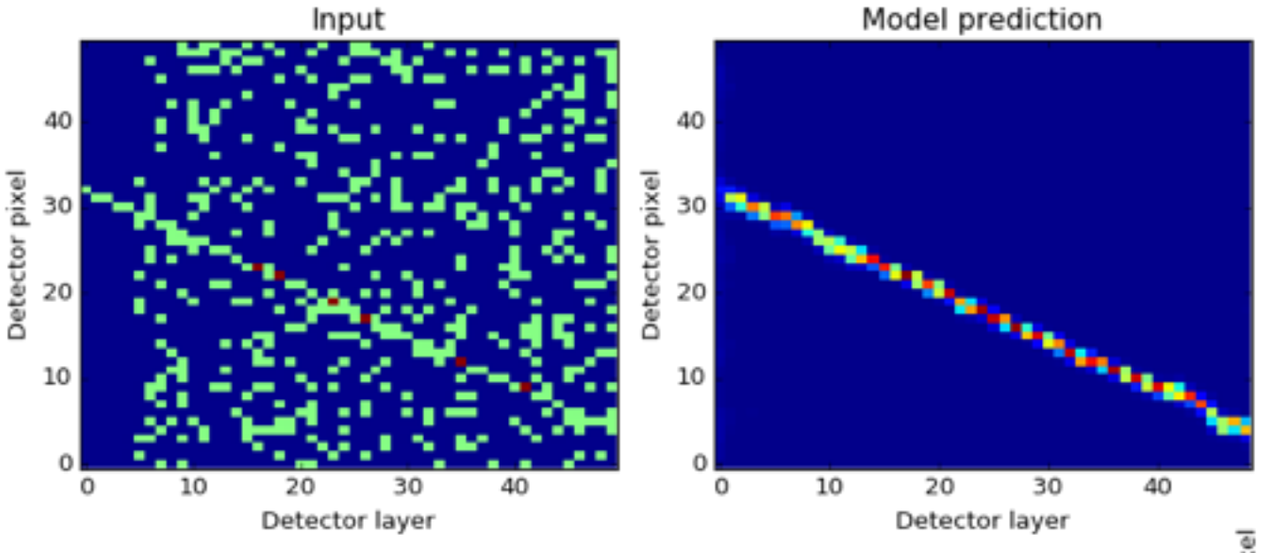
1D discrete detector layers





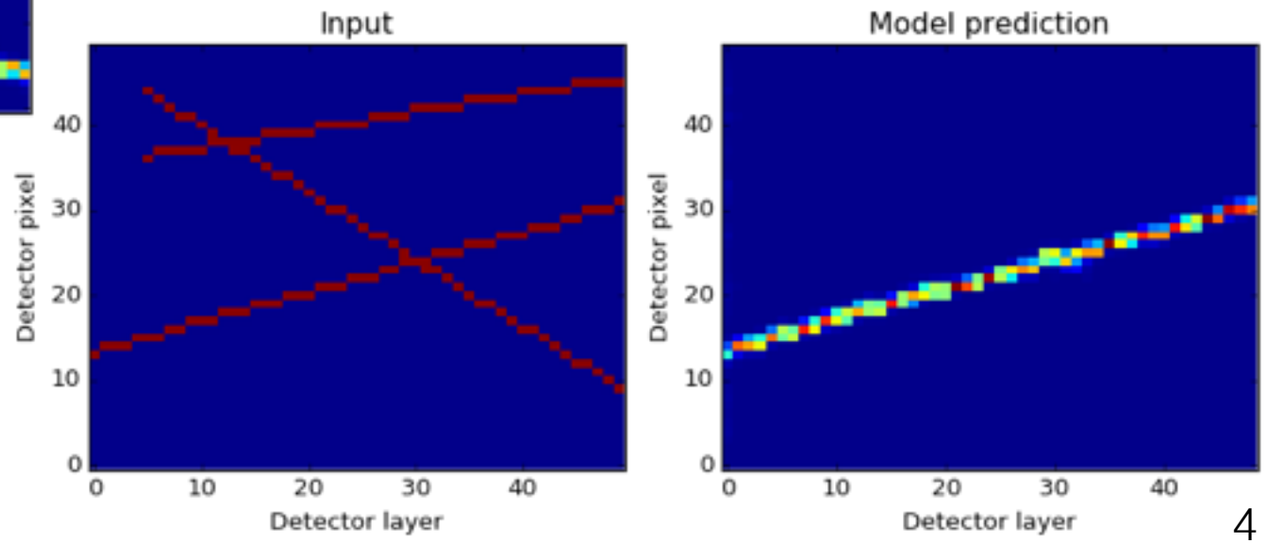
Fixed-width LSTM model

- Try to find one track at a time, seeded with some number of layers
- The sequence of detector layer arrays is fed into an LSTM, which tries to predict the hit location on each corresponding *next* layer
- With 50 pixels per layer and 50 layers, this is pretty easy to train



- Works with background tracks

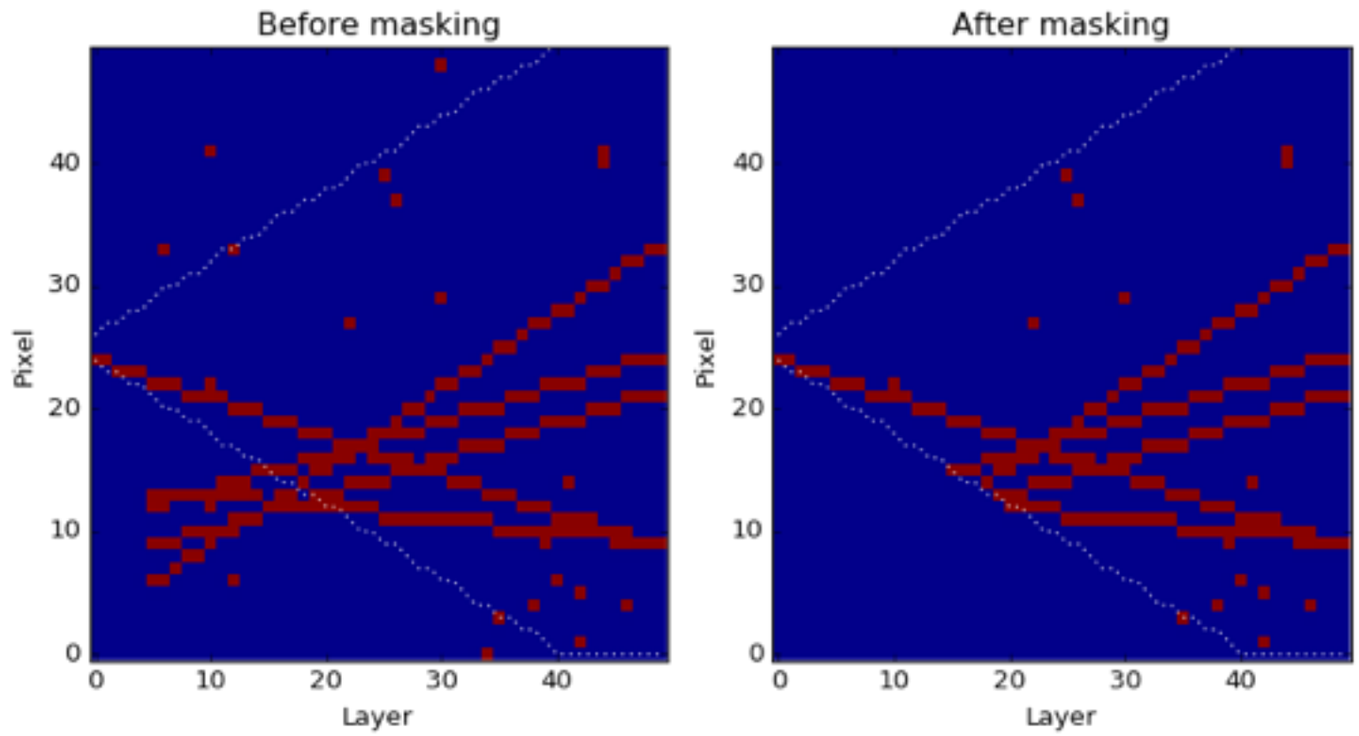
- Works with noise



Variable-width 1D data

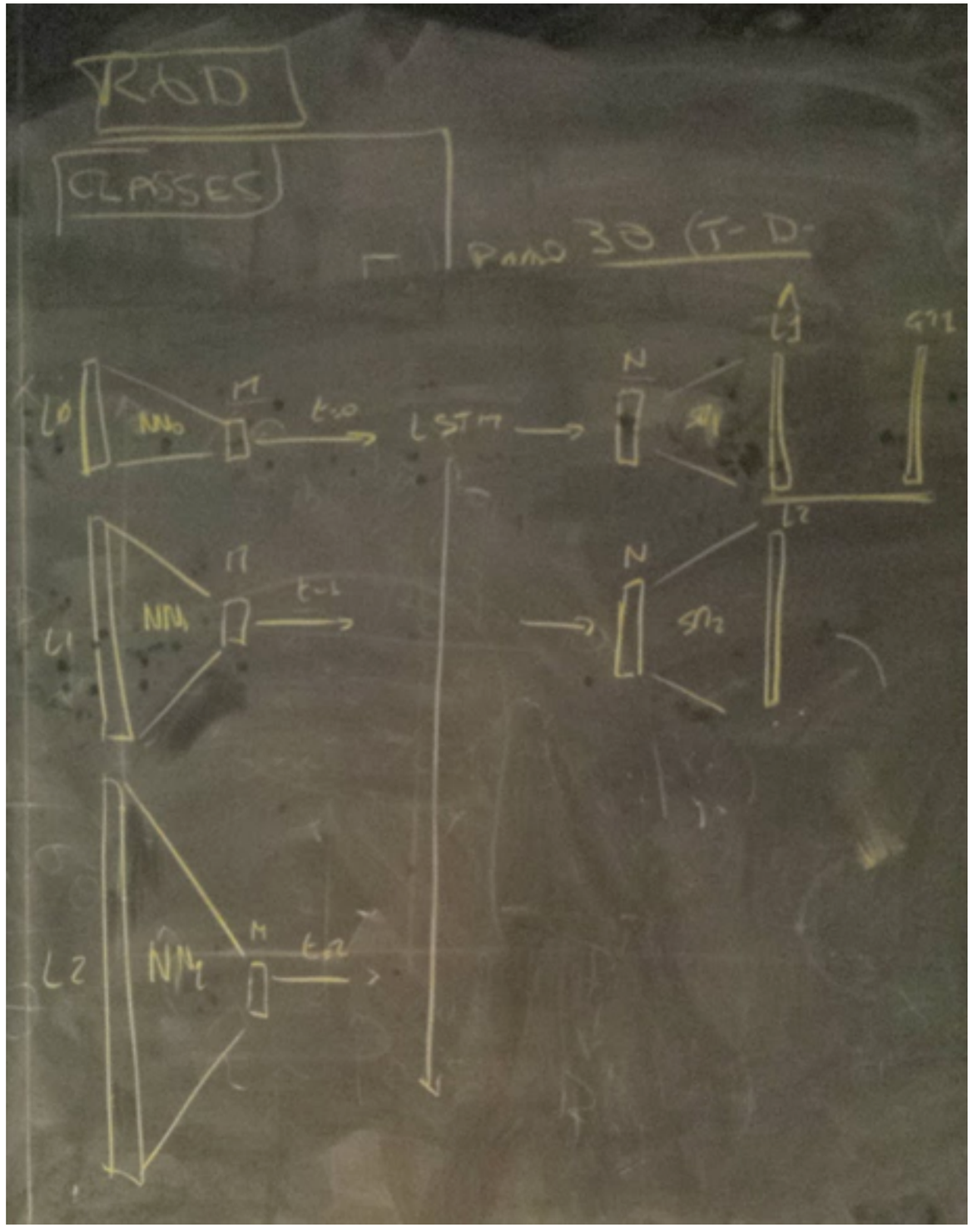
- Can the model be extended to work on variable sized layer data?

- Use the fixed-width toy data and cut out a shape:

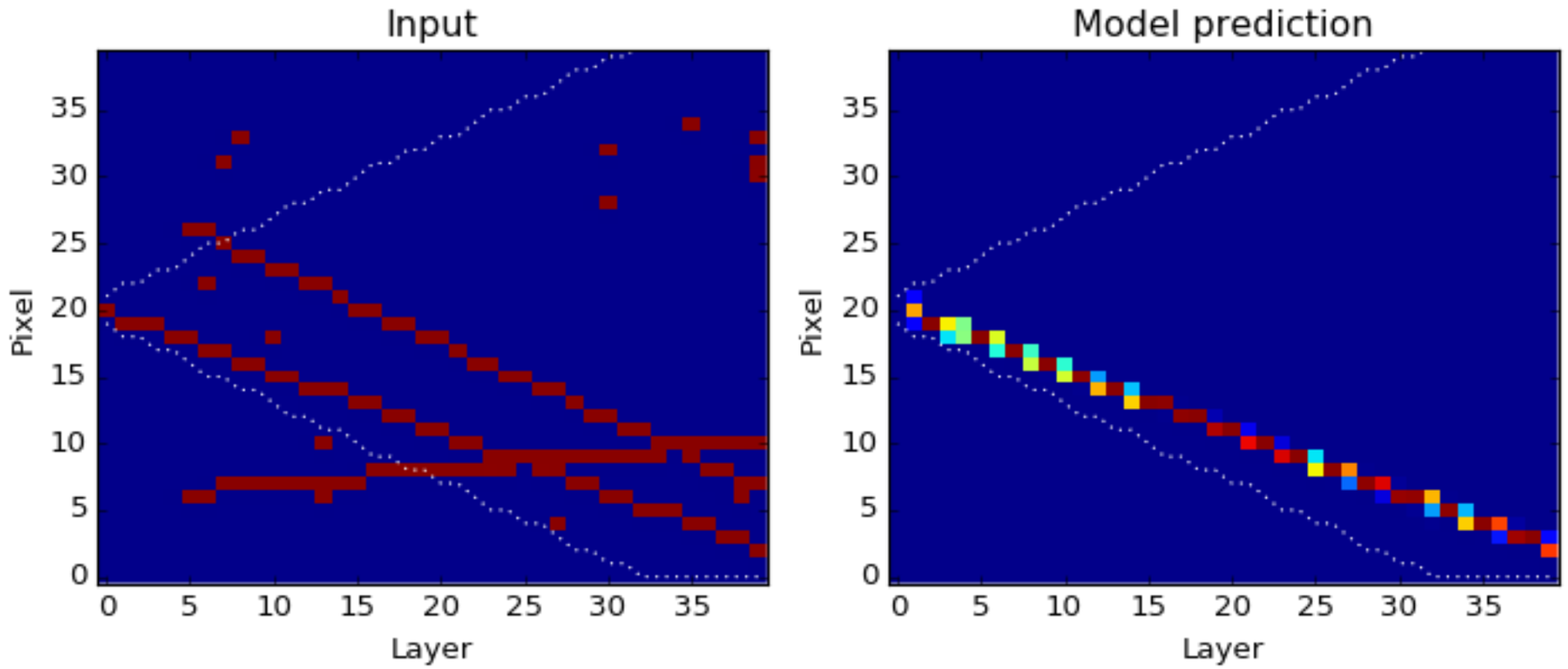


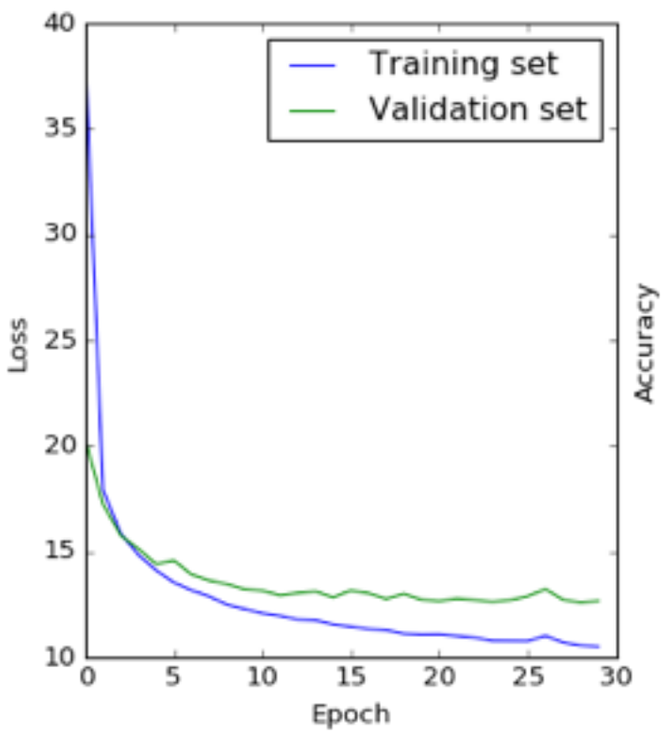
Variable-width LSTM model

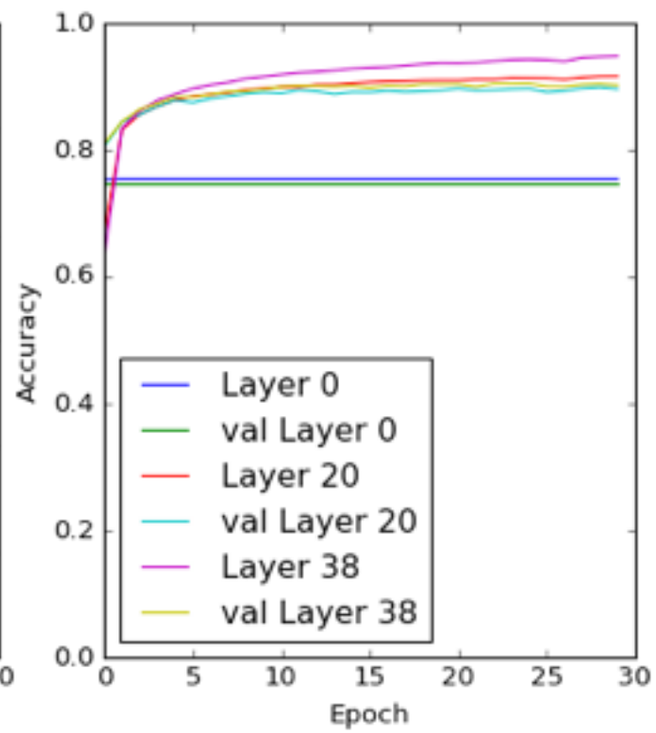
- Each layer transformed to a fixed-width representation by fully-connected layers
- The sequence of fixed-width layer arrays then fed into the LSTM as before
- The fixed-width output sequence of the LSTM is transformed to the target layer's size by fully-connected layers



Variable-width LSTM model



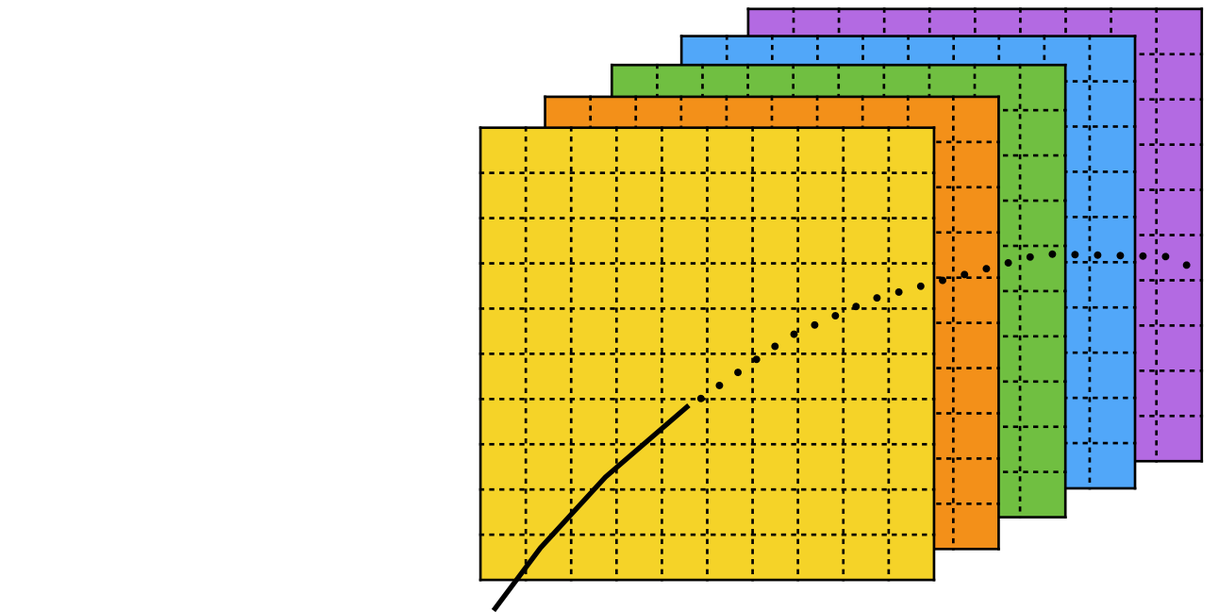


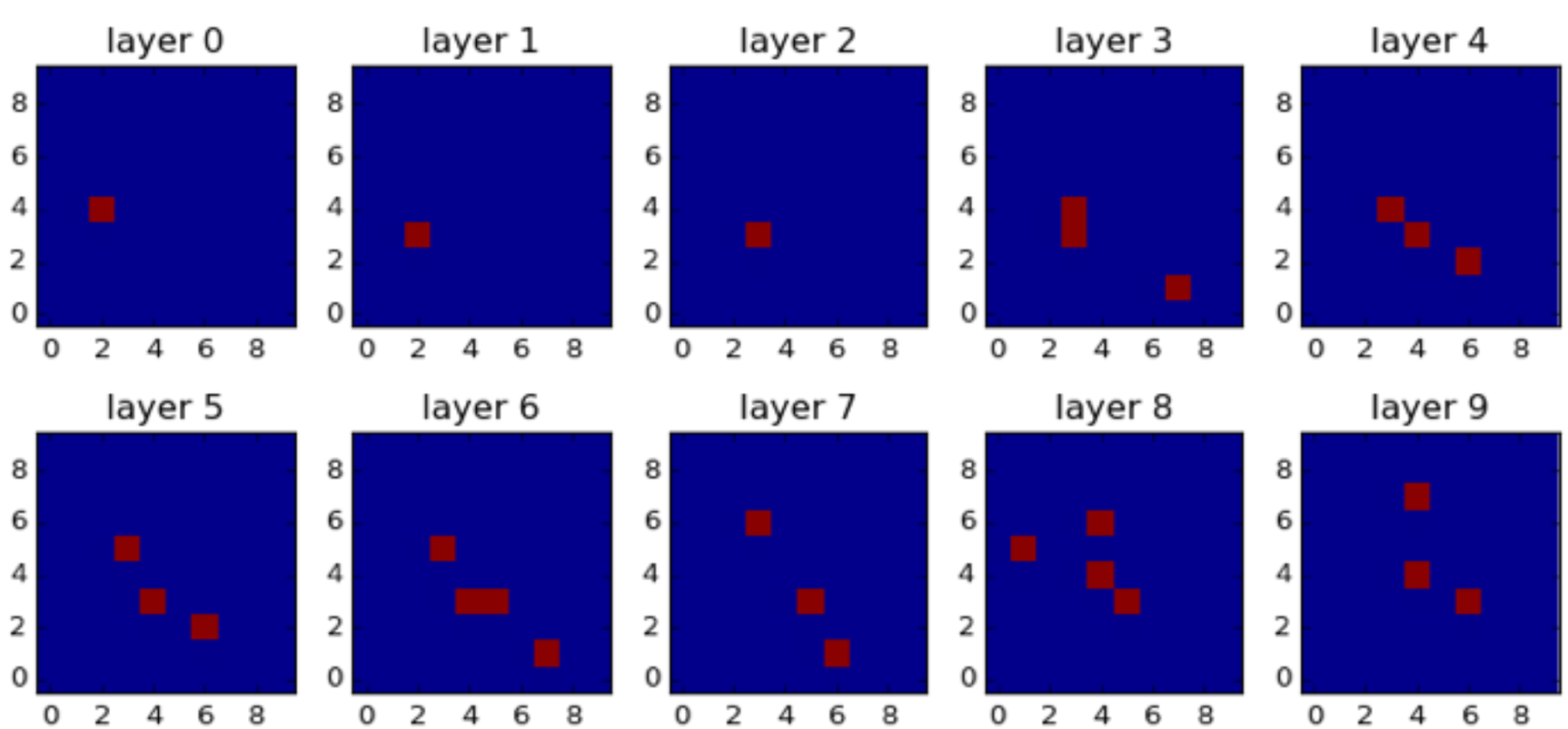


It works pretty well on this “easy” data

- 90% accuracy on prediction layers

2D discrete detector layers



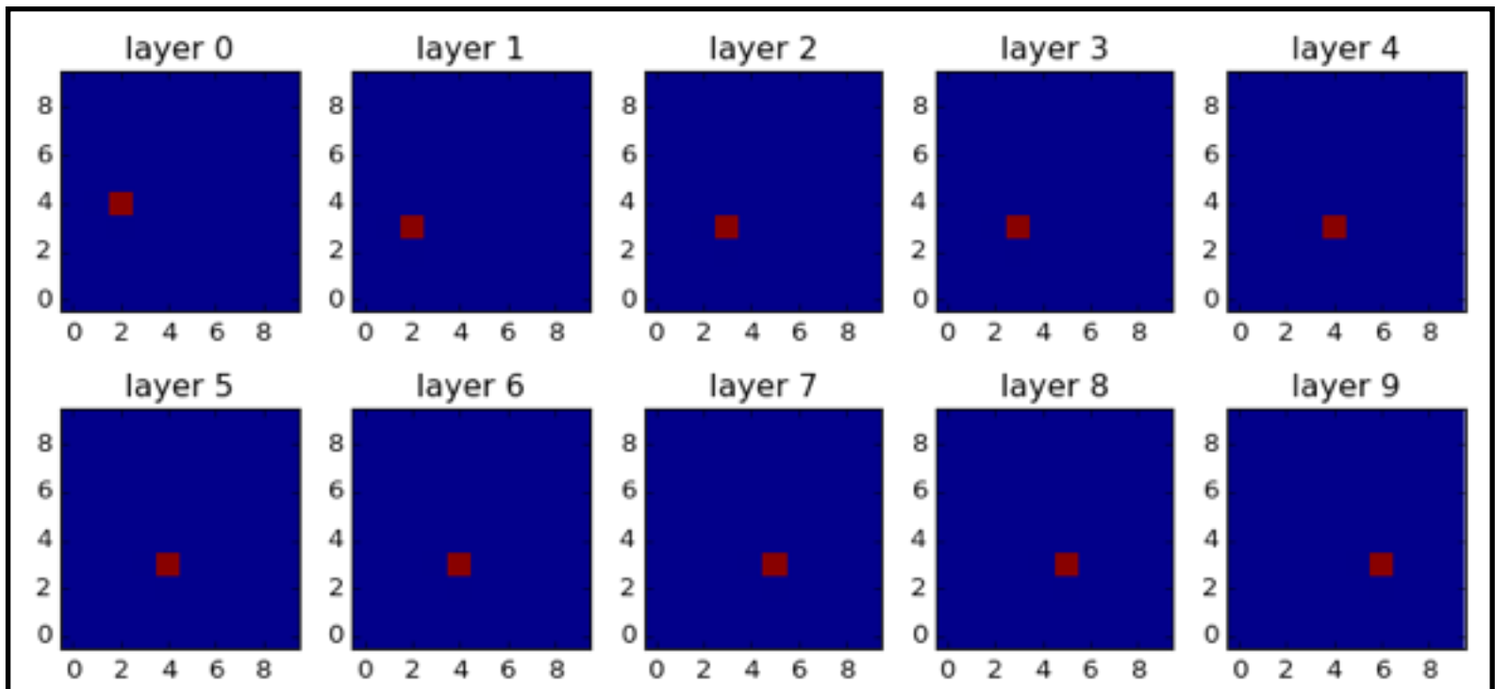


Fixed-shape LSTM model

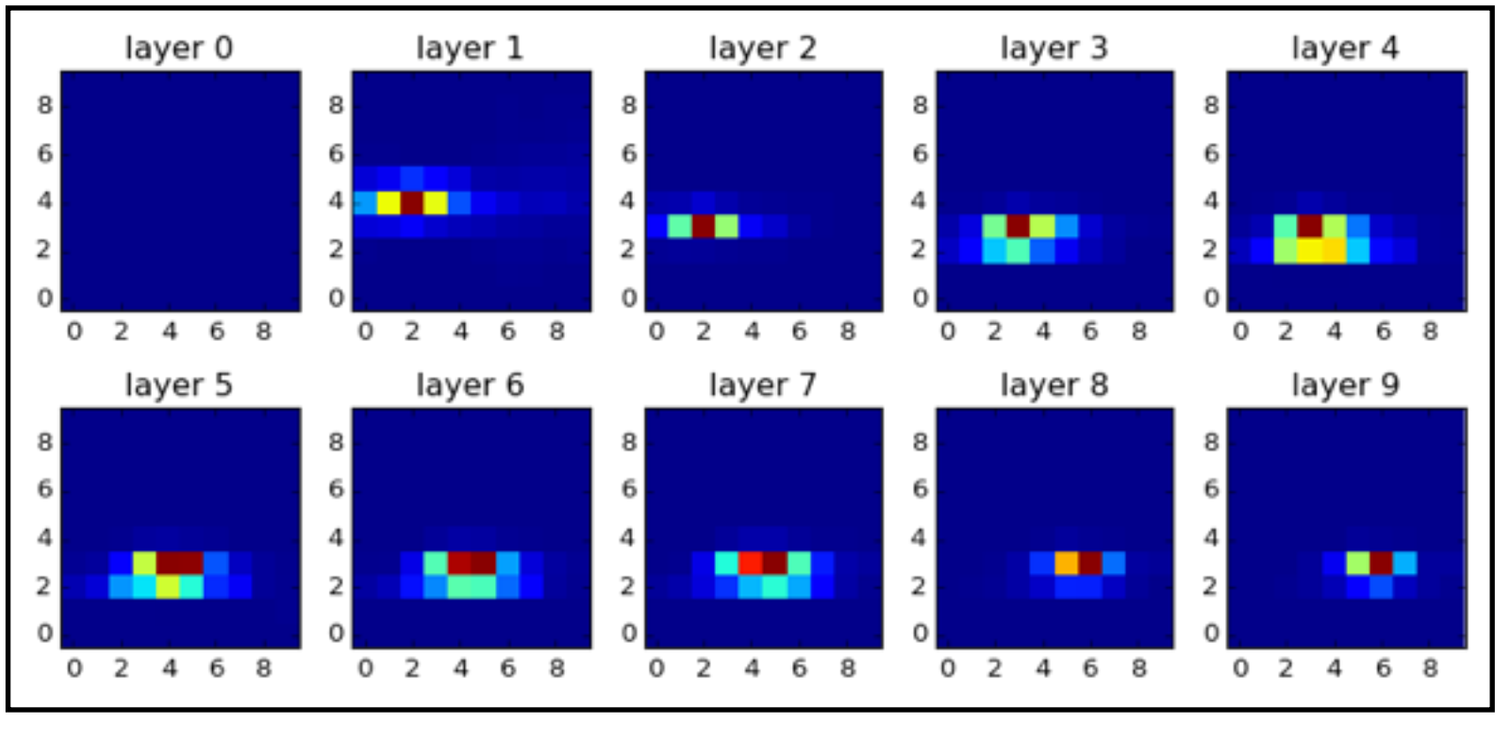
- Straightforward extension of the 1D fixed-width model to 2D detector layers
  - layers are unrolled or flattened into a 1D array, then fed into the LSTM as before
- The model and data quickly grow beyond what easily fits into our GPU with 6 GB memory
  - unable to do the many-layer “easy” problem
  - use a more realistic 10-layer setup

Fixed-shape LSTM model

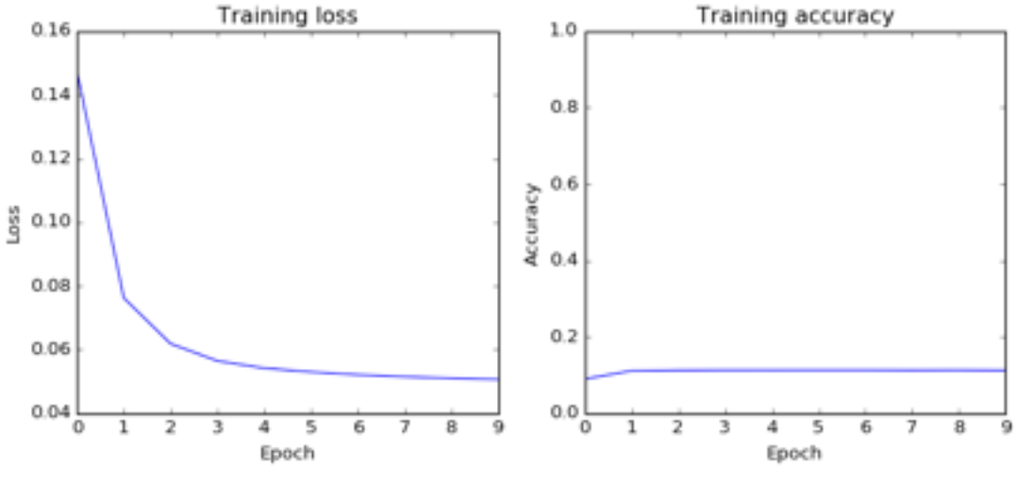
Input



Predictions



- The output is at least somewhat sensible sometimes
- Significantly poorer prediction accuracy than 1D case
  - 12% without any backgrounds (not a great metric, though)
- Predictions have an overly-horizontal shape
  - might be a bug



To do

- Understand, debug, then tune the 2D layer performance
  - disentangle effects from changing dimensionality, number of layers, granularity of layers
- Move to ACTS cylindrical detector data
  - and/or increase realism of toy data (e.g. curved tracks)
- Try out other ideas
  - Convolutional modeling across layers instead of LSTM
  - Bi-directional RNN
  - Abandon hit-predicting and just try hit-classifying