RNN in hls4ml

Aaron Wang
University of Washington
12th, July, 2021
APS DPF Presentation

Introduction

- Recursive Neural Network (RNN) models are the best to exploit the sequential structure in a dataset
- Increasing usage of RNN-based algorithms in the particle physics community
  - Fast inference of such algorithms on an FPGA will be crucial in the future
- Our goal is to support Keras/TensorFlow RNN models in hls4ml
  - Past presentations (in 2019) by Phill et. al: talk1 talk2

In this presentation

- Training and performance of some benchmark models
- hls4ml conversion: Top level design - with blocks and simpler code algo
- Design Constraints
- Results

Overview

- We want to support RNN models of different sizes
- Currently we are working on several **LSTM (Long Short-Term Memory)** and **GRU (Gated Recurrent Unit)** models

- Two benchmark models will be discussed

1. **Small model** with **5000k** trainable parameters
   a. Jet-tagging problem

   Dataset: CERNbox link

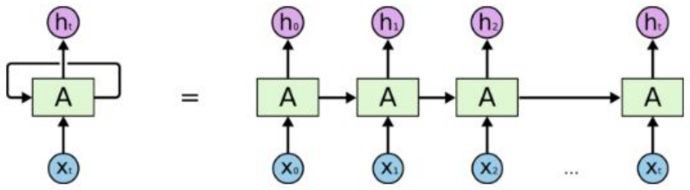
2. **Large model** with **100000k** trainable parameters
   a. QuickDraw model

   Dataset: quickdraw-dataset

Training code: RNN-HLS4ML-paper

What is an RNN?

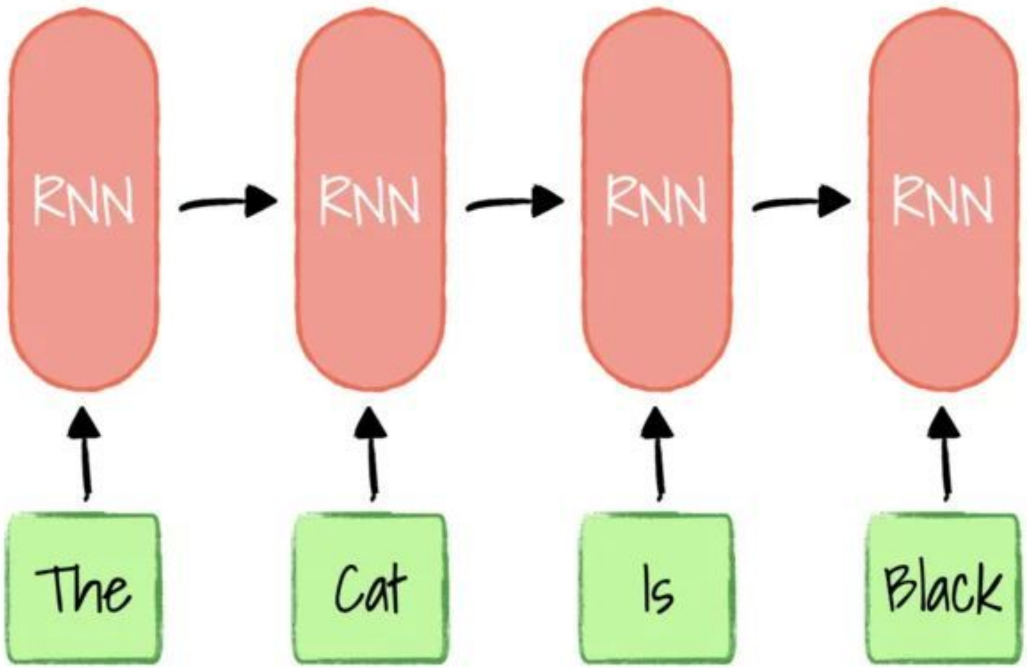
- RNN is a **Recurrent Neural Network**
  - Performs same function for every input of data
  - Remembers the immediate past and adds it to the present
  - Good at processing sequential data
    - Text, speech, strokes, etc



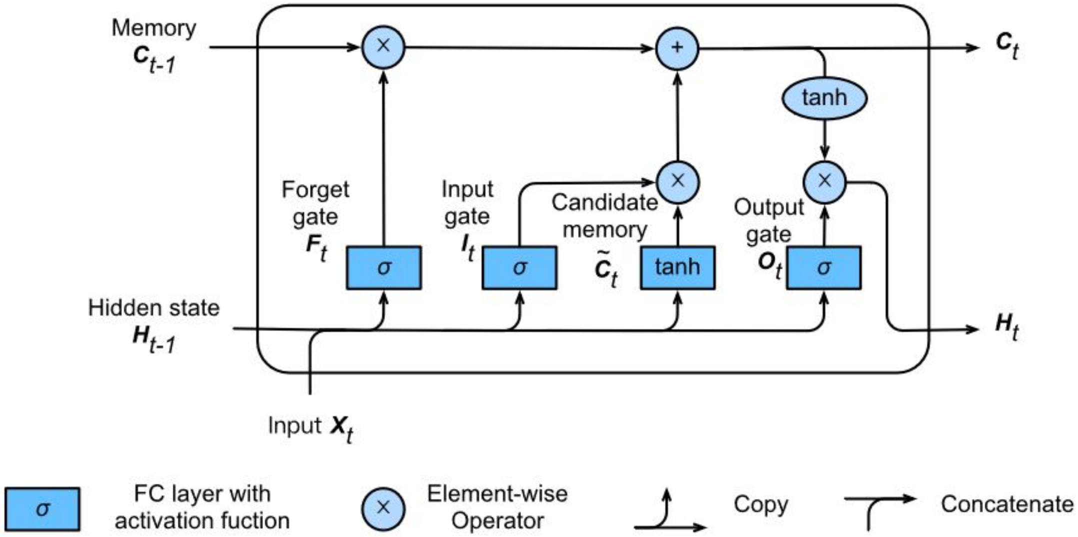
An unrolled recurrent neural network.

https://colah.github.io/posts/2015-08-Understanding-LSTMs/

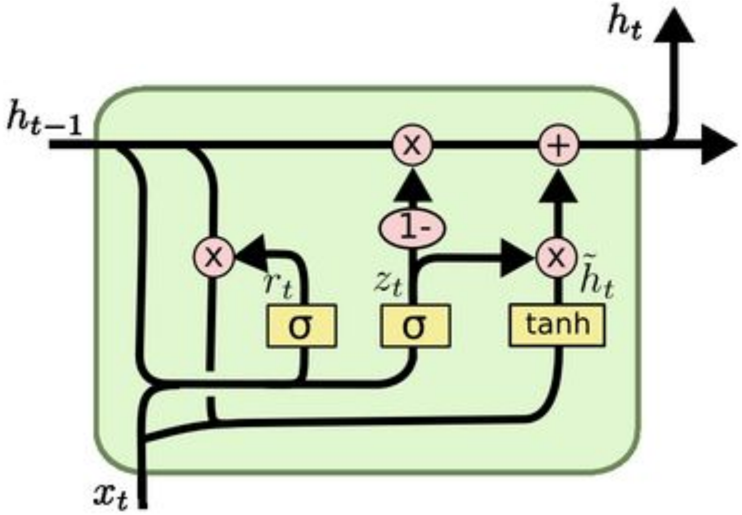
RNN based Encoder



Brief Introduction of RNN models - LSTM



Brief Introduction of RNN models - GRU



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

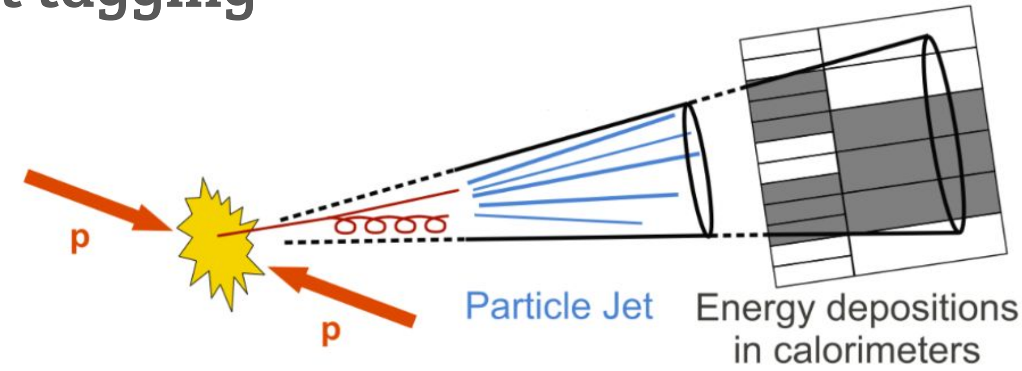
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

Benchmark Problem: LHC jet tagging

5-class Classifier

- **Five different categories** are considered

- Types of jets:
  - Quark (q)
  - Gluon (g)
  - W boson (W)
  - Z boson (Z)
  - Top (t)



- **Input features:** total 6
  - **pT** - Transverse Momentum
  - **eta** - Pseudo Rapidity
  - **phi** - Azimuthal Angle
  - **E** - Energy
  - **deltaR** - Relative angular distance w.r.t jet axis
  - **pdgID** - Particle identification information

Jet tagging LSTM/GRU model: Architecture

- 1 million jets total
- Standard scaled, and organized by particle
- Input:
  - Sequence of 20 particles with 6 features each
- Output:
  - Probability of 5 jet classes (q,g,W,Z,t)

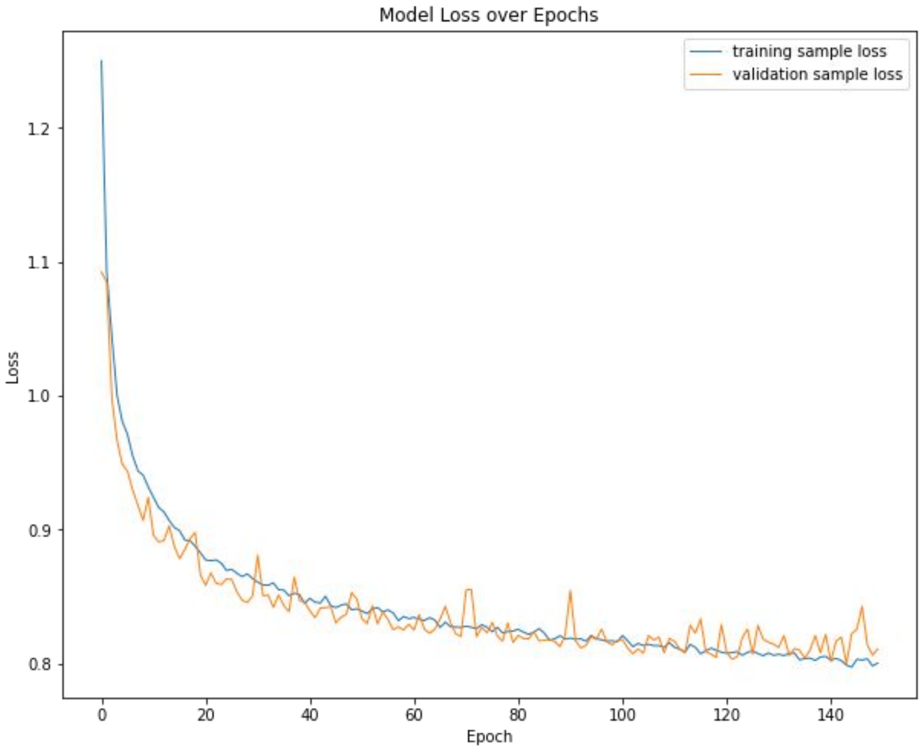
GRU

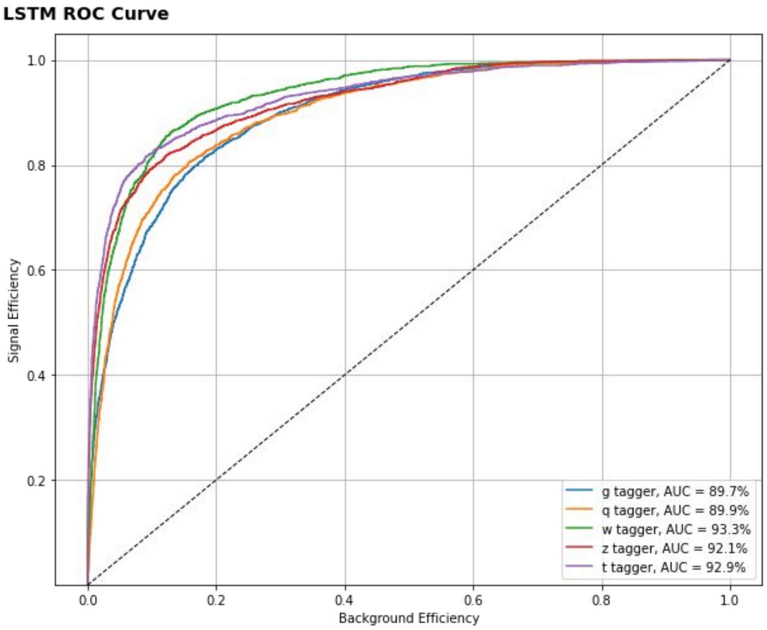
```
Model: "model_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_4 (InputLayer)         [(None, 20, 6)]           0
_________________________________________________________________
lstm1 (GRU)                  (None, 16)                1152
_________________________________________________________________
fc4 (Dense)                  (None, 64)                1088
_________________________________________________________________
dropout_3 (Dropout)          (None, 64)                0
_________________________________________________________________
fc7 (Dense)                  (None, 32)                2080
_________________________________________________________________
output_sigmoid (Dense)       (None, 5)                 165
=================================================================
Total params: 4,485
Trainable params: 4,485
Non-trainable params: 0
_________________________________________________________________
```

LSTM

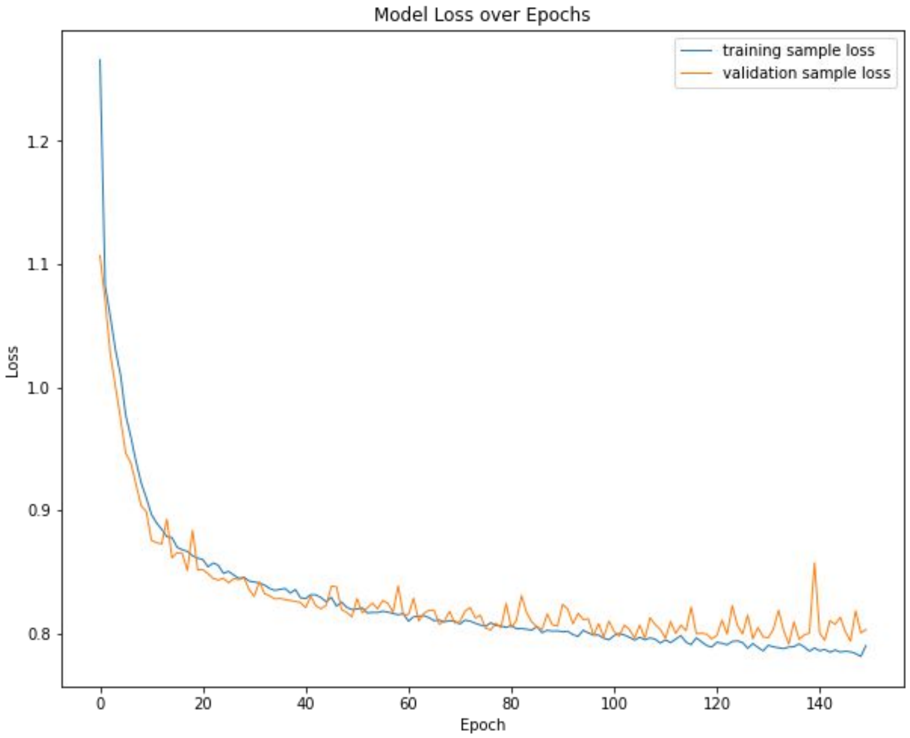
```
Model: "model_6"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_10 (InputLayer)        [(None, 20, 6)]           0
_________________________________________________________________
lstm1 (LSTM)                 (None, 20, 20)            2160
_________________________________________________________________
flatten_5 (Flatten)          (None, 400)               0
_________________________________________________________________
output_sigmoid (Dense)       (None, 5)                 2005
=================================================================
Total params: 4,165
Trainable params: 4,165
Non-trainable params: 0
```

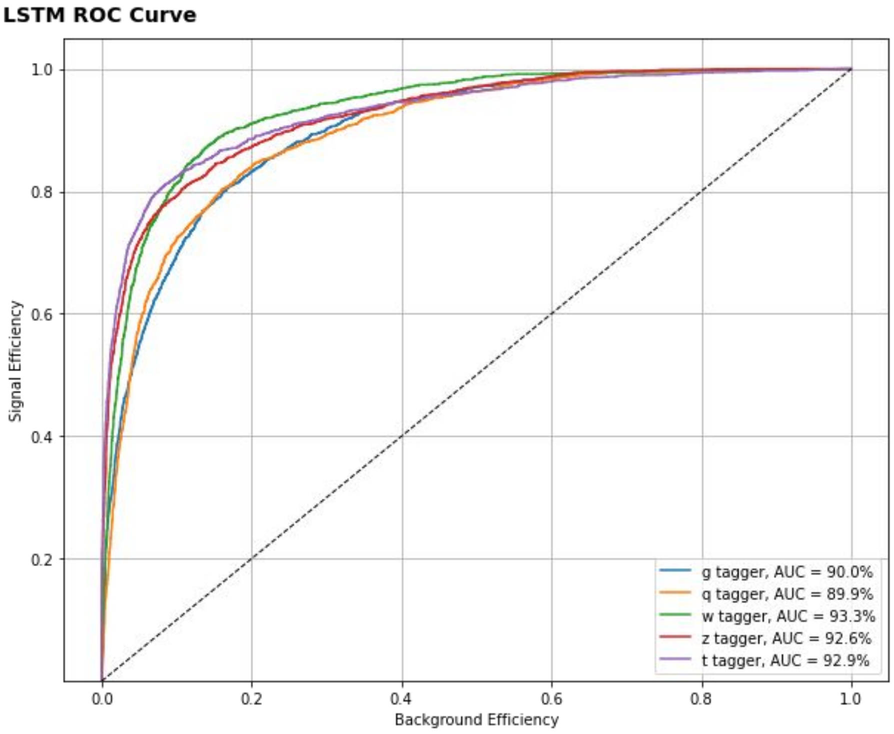
Jet tagging LSTM model: performance





Jet tagging GRU model: Performance

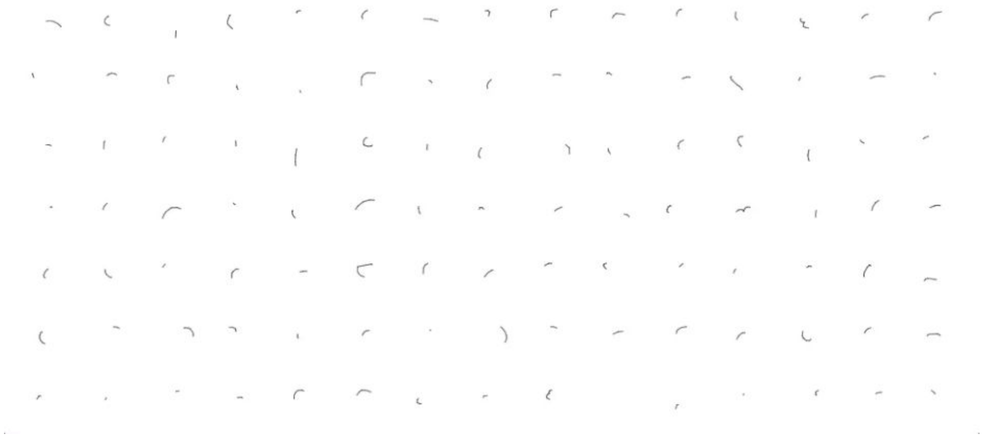




Benchmark Problem: QuickDraw image classification

5-class Classifier

- Types of images:
  - Bees
  - Butterflies
  - Mosquitos
  - Snails
  - Ants



- Input features(per stroke):
  - Pixel coordinates(x)
  - Pixel coordinates(y)
  - Time (t)

QuickDraw LSTM/GRU model: Architecture

- Input:
  - Sequence of up to 100 strokes with 3 coordinates (x,y, t)
- Output:
  - Probability of 5 picture types (Ants, bees, butterflies, mosquitos, snails)

GRU

Layer (type)	Output Shape	Param #
input_16 (InputLayer)	[(None, 100, 3)]	0
gru_13 (GRU)	(None, 128)	51072
dropout_15 (Dropout)	(None, 128)	0
dense_27 (Dense)	(None, 256)	33024
dropout_16 (Dropout)	(None, 256)	0
dense_28 (Dense)	(None, 128)	32896
rnn_densef (Dense)	(None, 5)	645

Total params: 117,637
Trainable params: 117,637
Non-trainable params: 0

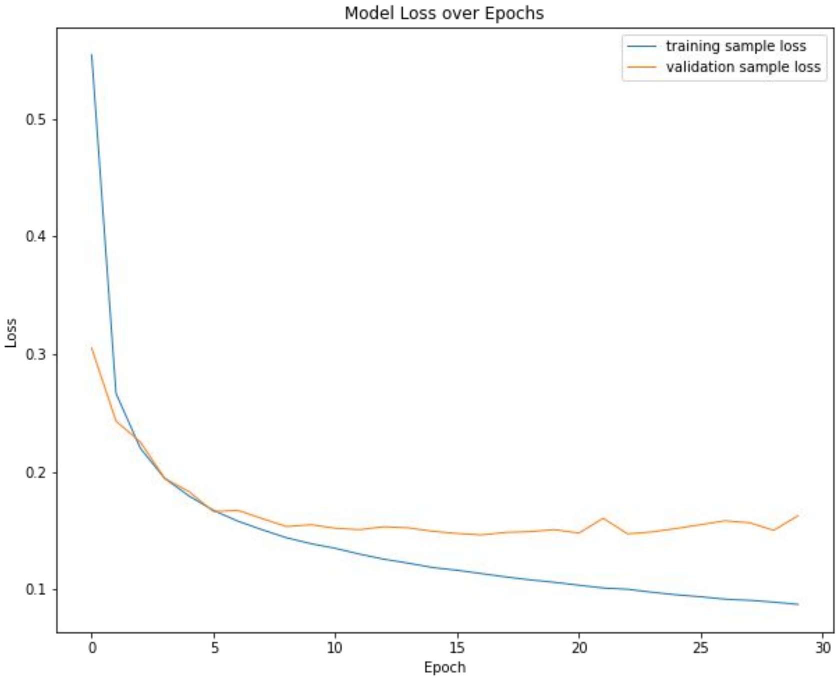
LSTM

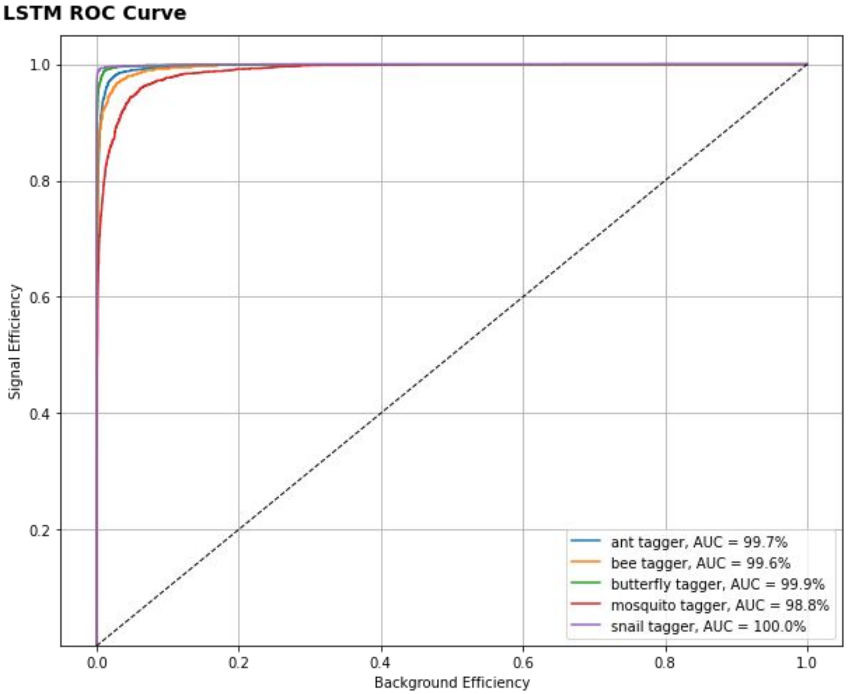
Model: "model_1"

Layer (type)	Output Shape	Param #
input_4 (InputLayer)	[(None, 100, 3)]	0
lstm_4 (LSTM)	(None, 100, 128)	67584
dropout_1 (Dropout)	(None, 100, 128)	0
lstm_5 (LSTM)	(None, 64)	49408
rnn_densef (Dense)	(None, 5)	325

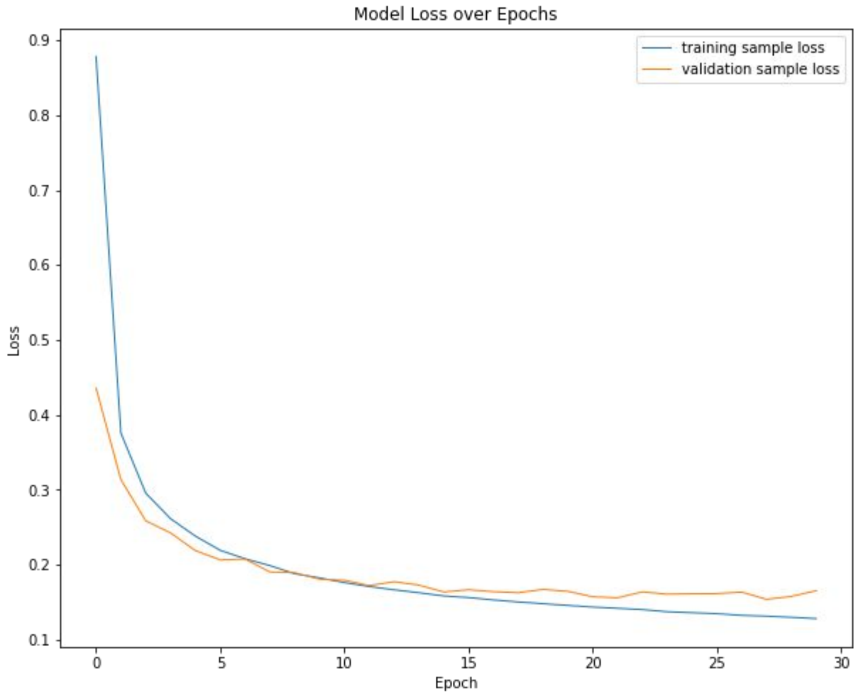
Total params: 117,317
Trainable params: 117,317
Non-trainable params: 0

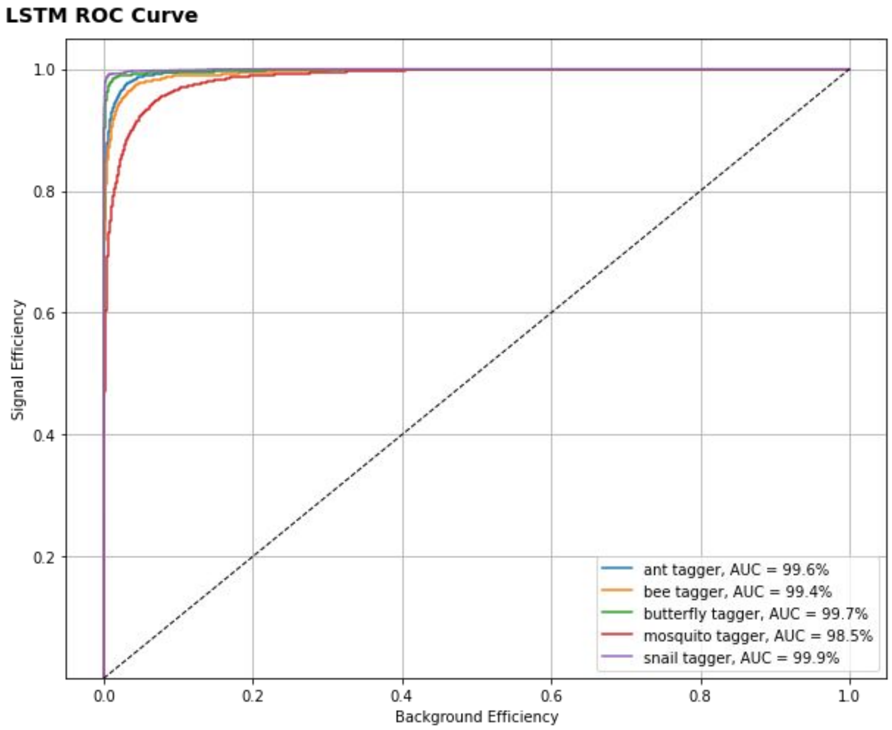
QuickDraw LSTM model: Performance



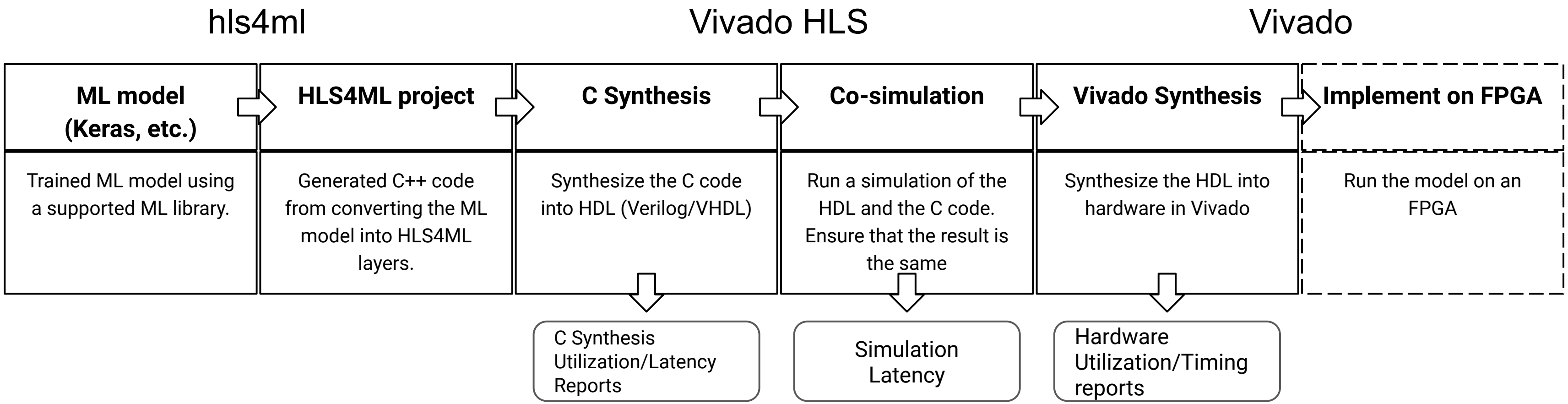


QuickDraw GRU model: Performance





hls4ml Flow



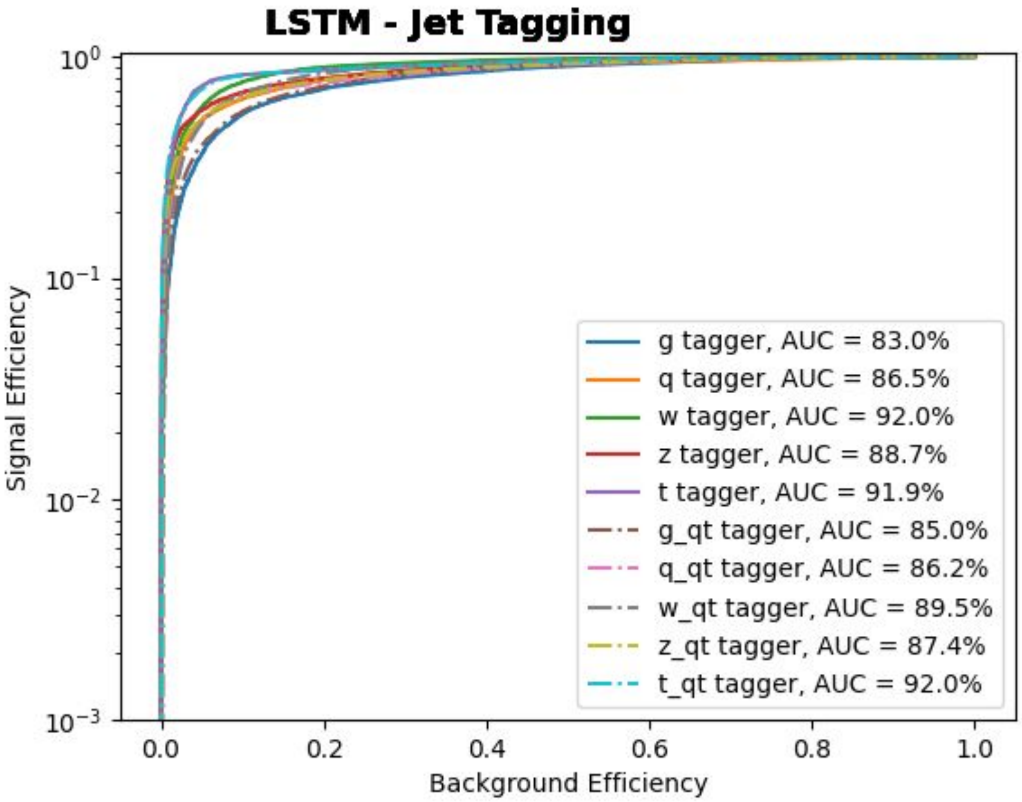
- We want to minimize resource usage and latency
  - Large models use lots of resources, while smaller models have higher latency
  - Want to find a balance
- Find a precision where resource usage is reasonable and AUC curves look similar

https://docs.google.com/presentation/d/1NXxKLoVqjLSqIccJLfTTqAwe0s5Tnn1gN6DFynOyp94/edit?usp=sharing

Results

Some of the experimental observations are presented in the following slides

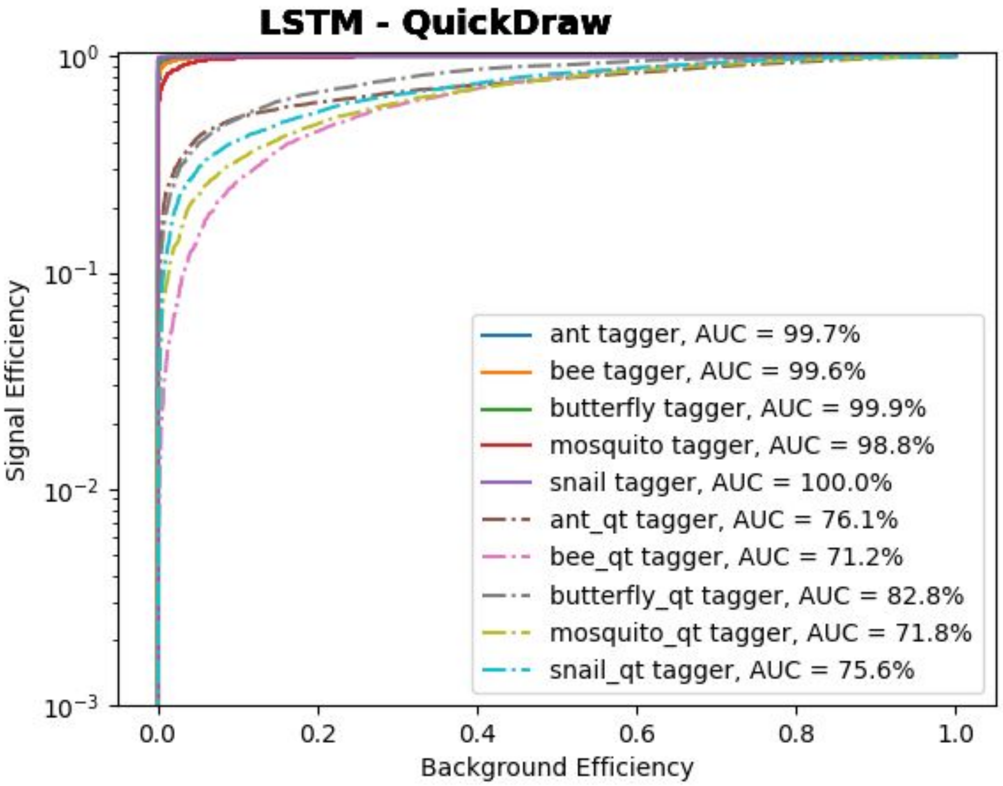
ROC curve for the jet identification task



**Solid Line for the floating point predictions*

**Dash-Dot Line for the quantized predictions at <16, 6>*

ROC curve for the QuickDraw classification task



**Solid Line for the floating point predictions*

**Dash-Dot Line for the quantized predictions at <16, 6>*

Summary

- Most of the necessary changes are implemented to support Keras LSTM and GRU models
- We discussed two benchmark models with 5k and 100k trainable parameters
- Converted hls4ml models perform very similar to the Keras/TensorFlow models
- Support for streaming IO-type has been added to enable HLS synthesis of larger models (tested up 120K parameters)

Next Steps

- The plan is to use three models (small, medium and large) as benchmark
- Currently we are also working on profiling a medium LSTM model for flavour tagging
  - 40k or more parameters
- Once we get consistent performance with the flavour tagging (medium) model, we will start wrapping up these studies

Thank You!