DeepPHY

Yi Zhang, Akash Doshi, Peter Enescu, Chang Shi
EN CTO Innovation Labs
Manager: Rob Liston

CISCO

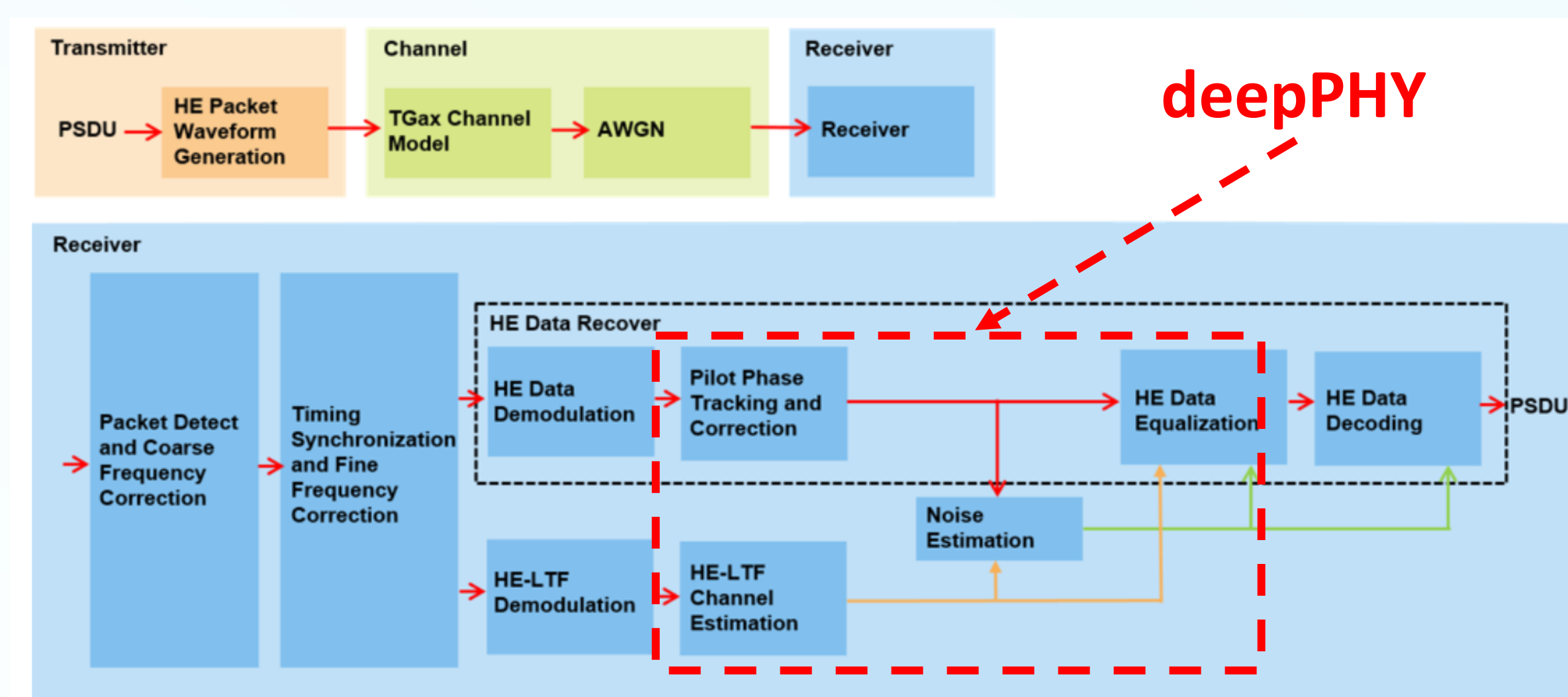Enterprise Networking

Background

Is deep learning effective on RF signals?
- 64-QAM OFDM receiver
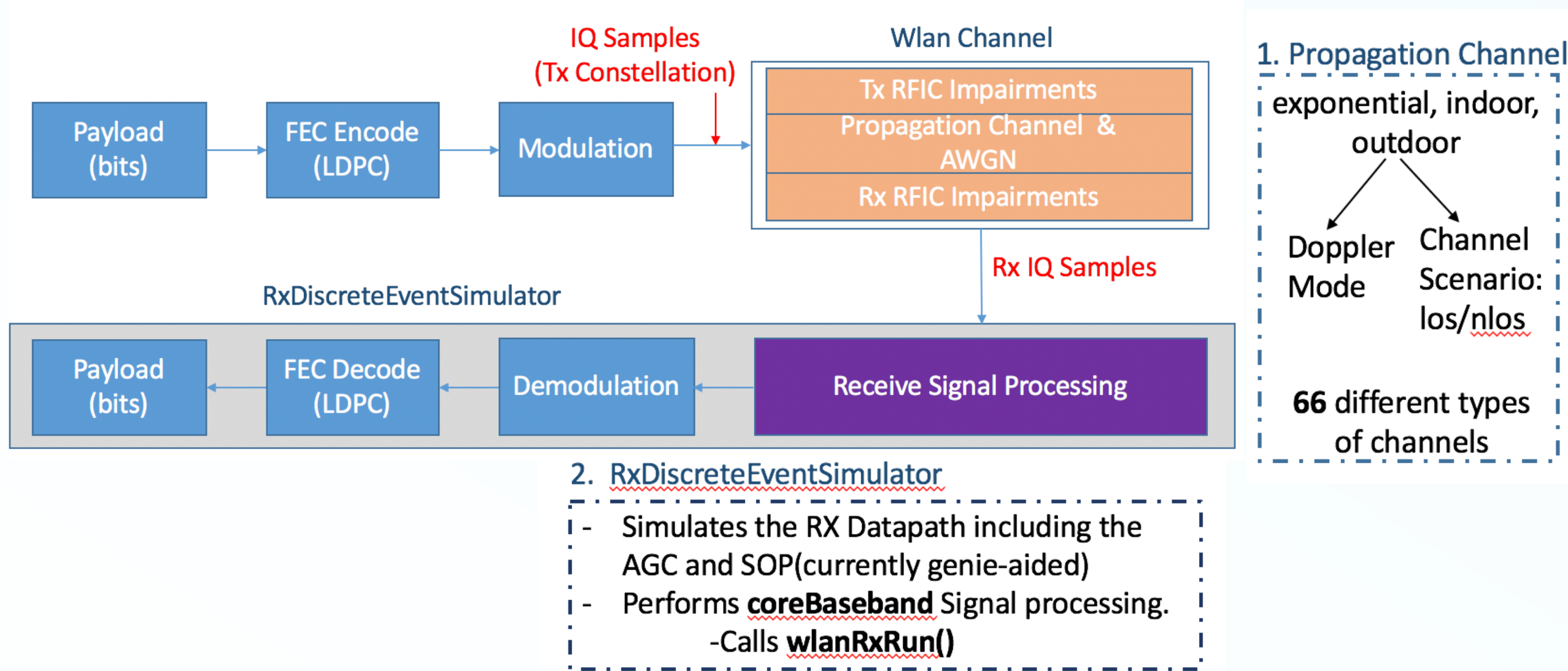- predict 64-QAM constellation given RF baseband transceiver samples

YES! With a big enough model and sufficient training data, deep learning could exceed state of the art performance!
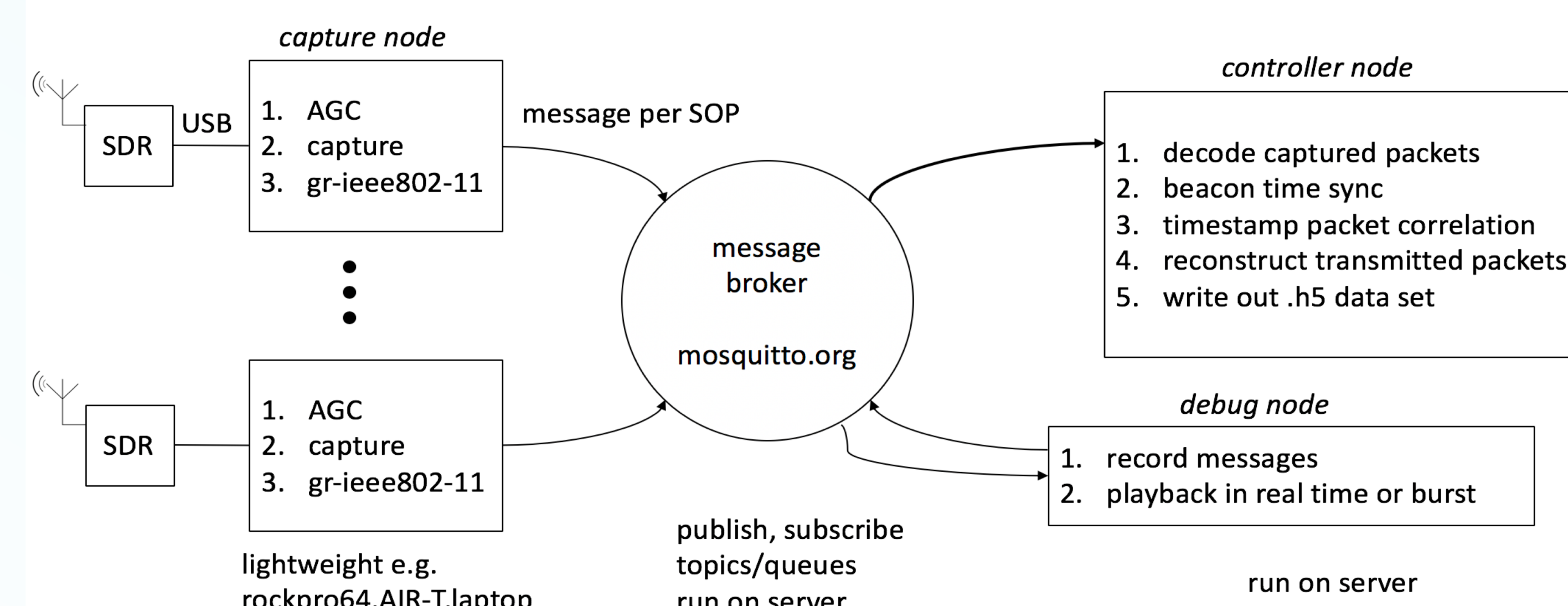
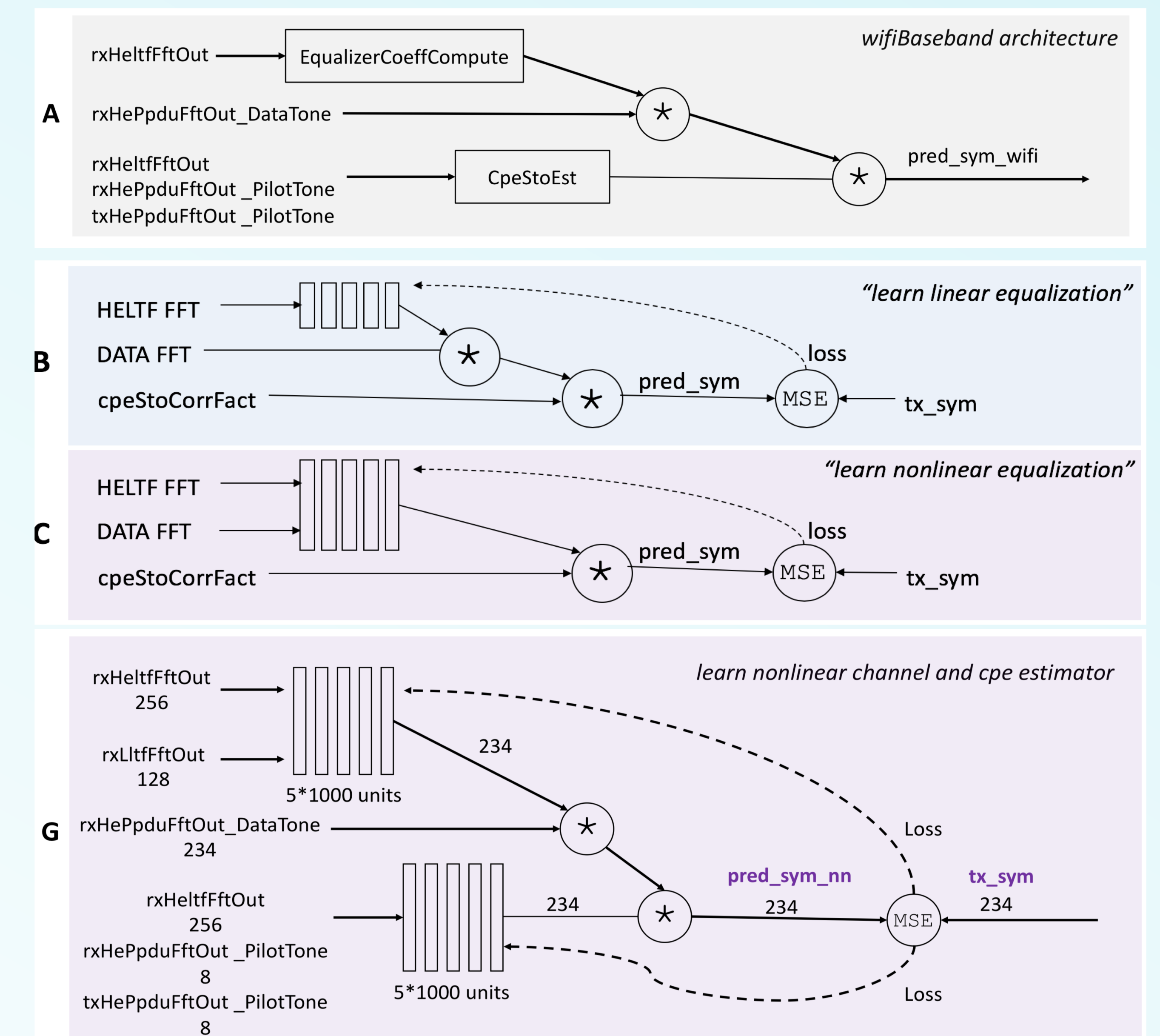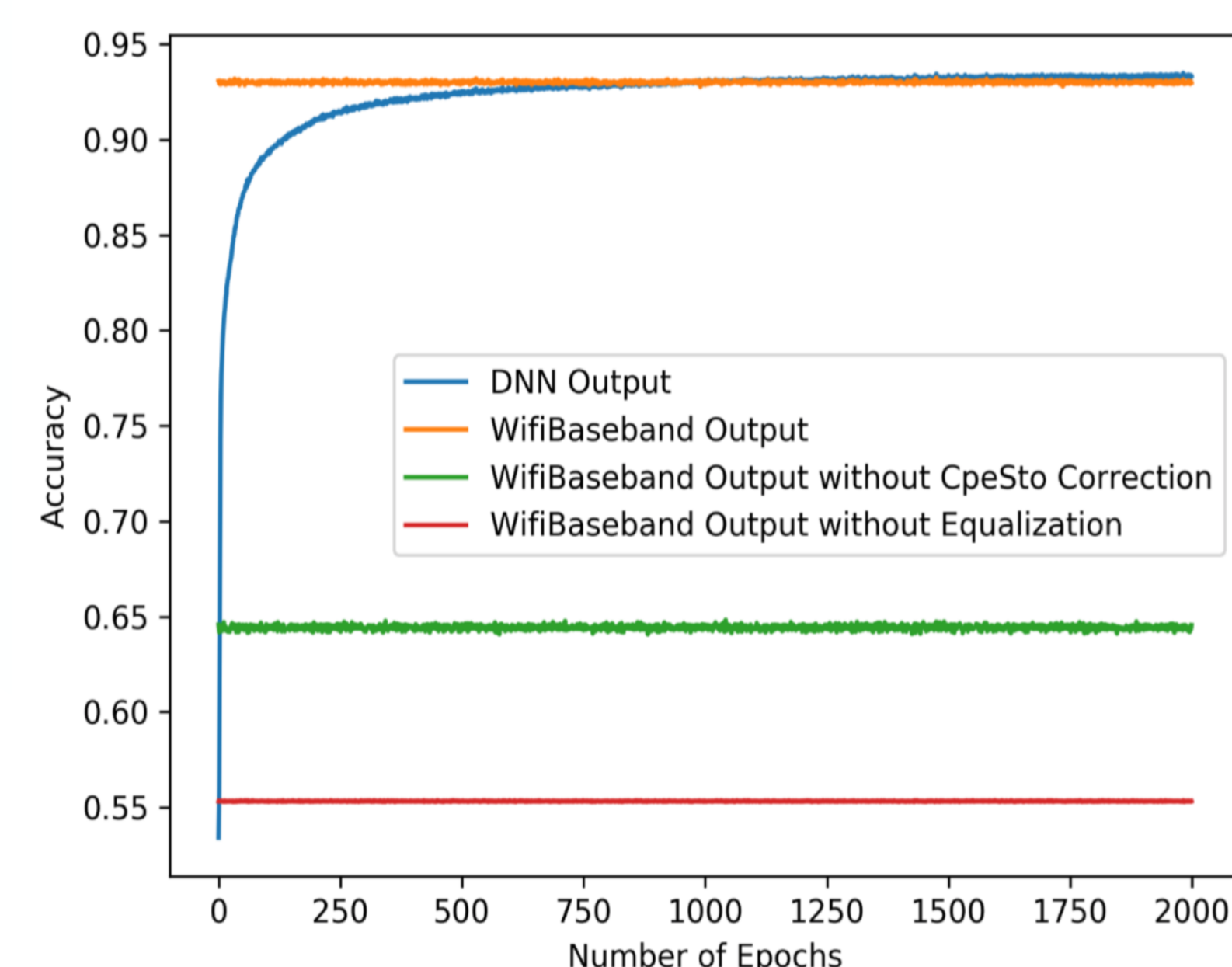## Workflow & Baseline

Flowchart of general 802.11ax PHY

deepPHY

Transmitter — PSDU → HE Packet Waveform Generation
Channel — TGax Channel Model → AWGN
Receiver — Receiver

Receiver:
Packet Detect and Coarse Frequency Correction → Timing Synchronization and Fine Frequency Correction → HE Data Recover (HE Data Demodulation → Pilot Phase Tracking and Correction → HE Data Equalization → HE Data Decoding) → PSDU
HE-LTF Demodulation → HE-LTF Channel Estimation → Noise Estimation

WifiBaseband Overview

IQ Samples (Tx Constellation)

Wlan Channel
- Tx RFIC Impairments
- Propagation Channel & AWGN
- Rx RFIC Impairments

Payload (bits) → FEC Encode (LDPC) → Modulation → Rx IQ Samples

RxDiscreteEventSimulator
Payload (bits) → FEC Decode (LDPC) → Demodulation → Receive Signal Processing

1. Propagation Channel
exponential, indoor, outdoor
Doppler Mode | Channel Scenario: los/nlos
66 different types of channels

2. RxDiscreteEventSimulator
- Simulates the RX Datapath including the AGC and SOP(currently genie-aided)
- Performs **coreBaseband** Signal processing.
  -Calls **wlanRxRun()**

Testbed Setup

RF Agile Transceiver
AD9364
FUNCTIONAL BLOCK DIAGRAM

★ transmitter
● receiver

USRP B200-mini

capture node
1. AGC
2. capture
3. gr-ieee802-11

SDR — USB

message per SOP

message broker
mosquitto.org

controller node
1. decode captured packets
2. beacon time sync
3. timestamp packet correlation
4. reconstruct transmitted packets
5. write out .h5 data set

SDR
1. AGC
2. capture
3. gr-ieee802-11

debug node
1. record messages
2. playback in real time or burst

publish, subscribe topics/queues run on server

lightweight e.g. rockpro64,AIR-T,laptop

run on server

Experiment

A — wifiBaseband architecture
rxHeltfFftOut → EqualizerCoeffCompute
rxHePpduFftOut_DataTone → ⊛
rxHeltfFftOut, rxHePpduFftOut_PilotTone, txHePpduFftOut_PilotTone → CpeStoEst → ⊛ → pred_sym_wifi

B — "learn linear equalization"
HELTF FFT, DATA FFT, cpeStoCorrFact → ⊛ → pred_sym → loss → MSE ← tx_sym

C — "learn nonlinear equalization"
HELTF FFT, DATA FFT, cpeStoCorrFact → ⊛ → pred_sym → loss → MSE ← tx_sym

G — learn nonlinear channel and cpe estimator
rxHeltfFftOut 256, rxLltfFftOut 128, rxHePpduFftOut_DataTone 234 → 5*1000 units → 234
rxHeltfFftOut 256, rxHePpduFftOut_PilotTone 8, txHePpduFftOut_PilotTone 8 → 5*1000 units → 234 → ⊛ → pred_sym_nn → MSE ← tx_sym 234

Results

Accuracy vs Number of Epochs
- DNN Output
- WifiBaseband Output
- WifiBaseband Output without CpeSto Correction
- WifiBaseband Output without Equalization

Setting	Value
Training size	47,308,800(~347GB)
Testing size	9,470,600 (~70GB)
Learning rate	1e-05
Batch size	1000
Training time	≈ 10 h

Method	Accuracy (1-BER)
wifiBaseband	0.92976946
DeepPHY	0.93323037
wifibaseband_nocpeSto	0.64351735
wifibaseband_nodataCoeff	0.55314769

BER vs SNR Pre-FEC
PER Performance Comparison
- DNN
- WifiBaseband

PER vs SNR Post-FEC
PER Performance Comparison
- DNN
- WifiBaseband

Future Directions

1. Finetuning the architecture to improve performance for **low SNR** and **multipath** scenarios
2. Extension to **MIMO**
3. Real data collection via **SDR** and Train NN with real data