Deep Learning is Emerging

- Intrusion Detection
- Threat classification
- Facial recognition
- Imagery analysis
- Tumor Detection
- Medical data analysis
- Diagnosis
- Drug discovery
- Pedestrian / obstacle detection
- Navigation
- Street sign reading
- Speech recognition
- Image classification
- Speech recognition
- Language translation
- Document / database searching

Enabled by low-cost, highly capable general purpose graphics processing units (GPUs)

Radio Frequency Technology is Pervasive

Deep learning technology enabled and accelerated by GPU processors
- Has yet to impact design and applications in wireless and radio frequency systems
Where to Use Deep Learning in RF Systems

Spectrum / Network Centric Applications
- Spectrum monitoring (threats)
- Intelligent spectrum usage
- Electronic protection (anti-jam)
- Cognitive system control

Device / Basestation Centric Applications
- Advanced modulation techniques
- Adaptive waveforms
- Encryption and security

User App Centric Applications
- Voice / image recognition
- Multi-sensor fusion
- Decision making and data reduction
Deep Learning Comparison

Image and Video
- Multiple channels (RGB)
- x, y spatial dependence
- Temporal dependence (video)

Audio and Language
- Single channel
- Frequency, phase, amplitude
- Temporal dependence

Systems and Signals
- Multiple channels
- Frequency, phase, amplitude
- Temporal dependence
- Complex data (I/Q)
- Large Bandwidths
- Human engineered

Existing deep learning potentially adaptable to systems and signals
- Must contend with wideband signals and complex data types
# Hardware for Deep Learning in RF Systems

## Training vs. Inference

<table>
<thead>
<tr>
<th>Hardware</th>
<th>Training Pros</th>
<th>Training Cons</th>
<th>Inference Pros</th>
<th>Inference Cons</th>
</tr>
</thead>
</table>
| **CPU**  | • Supported by ML Frameworks  
• Lower power consumption | • Slower than GPU  
• Fewer software architectures | • Adaptable architecture  
• Software programmable  
• Medium latency | • Low parallelism  
• Limited real-time bandwidth  
• Medium power requirements |
| **GPU**  | • Supported by ML Frameworks  
• Widely utilized  
• Highly parallel / adaptable  
• Good throughput vs power | • Overall power consumption  
• Requires highly parallel algorithms | • Adaptable architecture  
• High real-time bandwidth  
• Software programmable | • Medium power requirements  
• Not well integrated into RF  
• Higher latency |
| **FPGA** | Not widely utilized, not well suited (yet) | | • High power efficiency  
• High real-time bandwidth  
• Low latency | • Long development / upgrades  
• Limited reprogrammability  
• Requires special expertise |
| **ASIC** | Not widely utilized, not well suited | | • Extremely power efficient  
• High real-time bandwidth  
• Highly reliable  
• Low latency | • Extremely expensive  
• Long development time  
• No reprogrammability  
• Requires special expertise |
### Critical Performance Parameters for Deep Learning in RF Systems

<table>
<thead>
<tr>
<th></th>
<th>Adaptability / Upgradability</th>
<th>Deployment Time</th>
<th>Lifecycle Cost</th>
<th>Real Time Bandwidth</th>
<th>Compute / Watt</th>
<th>Latency</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>Green</td>
<td>Green</td>
<td>Green</td>
<td>Red</td>
<td>Green</td>
<td>Red</td>
</tr>
<tr>
<td>GPU</td>
<td>Green</td>
<td>Green</td>
<td>Green</td>
<td>Yellow</td>
<td>Red</td>
<td>Yellow</td>
</tr>
<tr>
<td>FPGA</td>
<td>Yellow</td>
<td>Red</td>
<td>Green</td>
<td>Yellow</td>
<td>Red</td>
<td>Red</td>
</tr>
<tr>
<td>ASIC</td>
<td>Red</td>
<td>Red</td>
<td>Green</td>
<td>Red</td>
<td>Green</td>
<td>Red</td>
</tr>
</tbody>
</table>

**GPU signal processing can provide wideband capability and software upgradability at lower cost and development time**

- Must contend with increased latency (~2 microsecond)
Outline

• Introduction to Deep Learning in RF
• Deepwave’s Technology
• Signal Detection and Classification
• Real-time Benchmarks on Embedded GPUs
• Summary
Why Has Deep Learning in RF Not Been Addressed

- **Bandwidth Limitations**
  - remote processing not possible

- **Limited Compute Resources**
  - at field site

- **Complicated Software**
  - for RF and AI independently

- AI requires large data sets
- Insufficient bandwidth to send to remote data center
- No RF systems exist with integrated AI computational processors
- Disjointed software
- Difficult to program and understand
Deepwave’s Software Defined Radio
A Platform for a Multitude of Applications

The Platform
Complete Edge-compute AI Platform for RF

The Software
Simply build AI into wireless technology

3rd Party Software
- WAVELEARNER Software
  - Deep / Machine Learning
  - GPU Parallel Computing
- AI Frameworks
- Signal Processing
- Hardware Abstraction
- AIR-T Hardware

*Patent Pending
Artificial Intelligence Radio Transceiver (AIR-T)

Hardware Specifications

- **2x2 MIMO Transceiver**
  - Analog Devices 9371 chip
  - Tunable from 300 MHz to 6 GHz
  - 100 MHz bandwidth per channel

- **Digital Signal / Deep Learning Processors**
  - Xilinx Artix 7 FPGA
  - NVIDIA Jetson TX2
    - ARM Cortex-A57 (quad-core)
    - Denver2 (dual core)
    - Nvidia Pascal 256 Core GPU
    - Shared GPU/CPU memory

- **Connectivity**
  - 1 PPS / 10 MHz for GPS Synchronization
  - External LO input
  - HDMI, USB 2.0/3.0, SATA, Ethernet, SD Card, GPIO

AIR-T

Mini ITX Form Factor

6.7 inch (17 cm)

6.7 inch (17 cm)
Incorporation of GPU in RF system allows for wideband processing of signal data in software environment
- Reduces development time and cost
Simplified Programming

Deep Learning
- TensorFlow
- Caffe
- Keras

Digital Signal Processing
- VHDL, Verilog
- NVIDIA
- CUDA
- Python
- Java
- C++

or

Custom Software
- GNU Radio
- TensorRT

or
FFT Performance Testing

- FPGA
  - PCIe
  - Shared Memory
  - Complex int16 to complex float32
  - Shared Memory
  - cuFFT
    - NVIDIA Tegra TX2

Real-time Signal Processing Measurements

![Graph showing signal bandwidth vs. Fourier transform size for different processors]
Inference at the Edge with GR-Wavelearner

Train Neural Network
Optimize Neural Network
Deploy Application
GR-Wavelearner Software

- Goal is to help the open source community easily deploy deep learning within signal processing applications
- Well documented README with dependency installation instructions to get started quickly
  - Ubuntu 16.04 recommended, Windows 10 supported
  - NVIDIA Docker Container 18.08*
- Signal classifier example provided:
  - GNU Radio Flowgraph
  - Python source code
  - PLAN files that are executable on the AIR-T and Maxwell
  - Signal data file example for testing
- Support for TensorRT 5.0
- Available at: deepwavedigital.com/wavelearner

[Image of GR-Wavelearner interface]

https://docs.nvidia.com/deeplearning/sdk/tensorrt-container-release-notes/rel_18.08.html
GNU Radio – Software Defined Radio (SDR) Framework

- Popular open source software defined radio (SDR) toolkit:
  - RF Hardware optional
  - Can run full software simulations

- Python API
  - C++ under the hood

- Easily create DSP algorithms
  - Custom user blocks

- Primarily uses CPU
  - Advanced parallel instructions
  - Recent development: RFNoC for FPGA processing

- Deepwave is integrating GPU support for both DSP and ML
GR-Wavelearner

• Out of tree (OOT) module for GNU Radio
• Allows users to easily incorporate deep learning into signal processing
• C++ and Python API
• Open source GPLv3 license
• Two blocks currently:
  • Inference – TensorRT wrapper for GNU Radio
  • Terminal Sink – Python module for displaying classifier output
Outline

• Introduction to Deep Learning in RF
• Deepwave’s Technology
• Signal Detection and Classification
• Real-time Benchmarks on Embedded GPUs
• Summary
Multi-transmitter Environmental Scenario

AIR-T monitors congested spectrum using deep learning
Radar Signal Detector Model: Transmitted Signals

<table>
<thead>
<tr>
<th>Radar Waveform</th>
<th>Nothing</th>
<th>Interference</th>
<th>Surveillance</th>
<th>Ground (LEM1)</th>
<th>Ground (LEM2)</th>
<th>MTI</th>
<th>Airborne (Med PRF)</th>
<th>Airborne (High PRF)</th>
<th>Ground (Frank Code)</th>
<th>Nautical (Short Range)</th>
<th>Nautical (Long Range)</th>
<th>Ground (NLFM1)</th>
<th>Ground (NLFM2)</th>
<th>Ground (NLFM3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Linear Pulse</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Non-Linear Pulse</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Phase Coded Pulse</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Pulsed Doppler</td>
<td></td>
<td>X</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td></td>
<td></td>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

Technique demonstration shown with nominal radar signals
- Method applicable to communications, cellular, and other RF protocols
Dataset Overview

• Goal: Develop a deep learning classifier that detects signals below noise floor
  • Requires training on noisy data with and without interference

• Swept SNIR from -35 dB to 20 dB in 1 dB increments
  • 1000 training segments per SNIR
  • 500 inference segments per SNIR
  • Up to 3 interferers in each segment
Radar Signal Detector Model: Example Classifier

Signal Feature Extraction

Signal Classification

Signal Stream →

Convolution

Max Pool

Flatten

cuDNN
TensorFlow™
Training Process and Progress

• 1000 training segments per SNR
• 55 different SNR values
• Training on low SNR values increase detection sensitivity
• 100% accuracy not expected due to training at extremely low SNR values
• Softmax cross entropy
• Adam Optimizer
Detecting and Classifying Low Power Signals

Neural network starting to classify signal well below noise floor
Detecting and Classifying Low Power Signals

Near 100% classification probability with SNR > -10dB
Receiver Operating Characteristic (ROC) Curve

Probability of Correct Classification for One Radars

Decibel (dB) Refresher

<table>
<thead>
<tr>
<th>Signal-to-Noise Ratio (dB)</th>
<th>Receiver Noise Power (milliwatts)</th>
<th>Received Signal Power (milliwatts)</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>1</td>
<td>100</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>-10</td>
<td>1</td>
<td>0.1</td>
</tr>
<tr>
<td>-20</td>
<td>1</td>
<td>0.01</td>
</tr>
<tr>
<td>-30</td>
<td>1</td>
<td>0.001</td>
</tr>
</tbody>
</table>
Receiver Operating Characteristic (ROC) Curve

Probability of Correct Classification for All Radars

Decibel (dB) Refresher

<table>
<thead>
<tr>
<th>Signal-to-Noise Ratio (dB)</th>
<th>Receiver Noise Power (milliwatts)</th>
<th>Received Signal Power (milliwatts)</th>
</tr>
</thead>
<tbody>
<tr>
<td>20</td>
<td>1</td>
<td>100</td>
</tr>
<tr>
<td>10</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>-10</td>
<td>1</td>
<td>0.1</td>
</tr>
<tr>
<td>-20</td>
<td>1</td>
<td>0.01</td>
</tr>
<tr>
<td>-30</td>
<td>1</td>
<td>0.001</td>
</tr>
</tbody>
</table>
DNN appears to be randomly guessing at low SNR which will create unnecessary requirements on downstream processing.
Methodology for Testing False Positive Rate

![Diagram showing the methodology for testing false positive rate, with a flowchart indicating the process from Noise Source through CNN to Classifier, leading to the determination of whether something or nothing is present. A bar chart illustrates the false alarms for different sources, with Noise Source Only resulting in a total PFA of 0.41.]
Confusion Matrix and Signal to Noise Ratio

Significant false alarm rate limits algorithm’s applicability and creates non-zero probability of correct classification (PCC) at low SNR values.
Deepwave Training Method to Reduce False Alarms

- Method makes probability of false alarm a training hyperparameter
- Example shows false alarm rate reduced from 41% to 0.01%

**False Alarms (Noise Only)**

**Probability of Correct Classification for Various Radars**
• Introduction to Deep Learning in RF
• Deepwave’s Technology
• Signal Detection and Classification
• Real-time Benchmarks on Embedded GPUs
• Summary
Critical Performance Parameters

• What makes a DNN model “good?”
  • **High Sensitivity** – detects low powered signals
  • **Low false alarm rate** – minimize false positives
  • **High real time bandwidth**
  • **Low computational requirements**
  • **Low latency**

• Most of these critical performance parameters are adversarial
Performance Benchmarking Test Setup

Define Model Structure

Train Model

Measure Sensitivity

Measure Real Time Throughput

Repeat for multiple models

Model Tuning Variables

<table>
<thead>
<tr>
<th>Model Tuning Variables</th>
<th>Min Val</th>
<th>Max Val</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNN Stride</td>
<td>1</td>
<td>16</td>
<td>9</td>
</tr>
<tr>
<td>Number of Filters</td>
<td>4</td>
<td>256</td>
<td>7</td>
</tr>
<tr>
<td>Classifier Layer 1 Width</td>
<td>64</td>
<td>128</td>
<td>3</td>
</tr>
<tr>
<td>Classifier Layer 2 Width</td>
<td>32</td>
<td>64</td>
<td>3</td>
</tr>
<tr>
<td>Classifier Layer 3 Width</td>
<td>0</td>
<td>64</td>
<td>2</td>
</tr>
<tr>
<td>Batch Size</td>
<td>1</td>
<td>256</td>
<td>8</td>
</tr>
<tr>
<td>Total Model Combinations Tested</td>
<td></td>
<td></td>
<td>728</td>
</tr>
</tbody>
</table>
Performance Benchmarking Test Setup

- 1000 training segments per SNR
  - 55 different SNR values
- Softmax cross entropy
- Adam Optimizer
- Quadro GP100 GPU
- Create UFF File for each model
Performance Benchmarking Test Setup

- Compute receiver operating characteristic (ROC) curve for each model
- Define sensitivity to be where median PCC = 50% for all signal types
Performance Benchmarking Test Setup

- Create TensorRT PLAN file for each platform tested
- Load signal data into RAM
- Stream unthrottled data to gr-wavelearner

- Measure data rate at two locations:
  1. Aggregate data rate for entire process
     - Number of bytes processed / wall time
  2. Computation data rate in work() function
     - Number of bytes process / computation time
Data Rate Benchmark for AIR-T (Tegra TX2)

- Tested 91 different CNN classifier models
- Maximum real-time inference data rate for 8 different batch sizes
- Able to achieve 200 MSPS (real samples) with AIR-T
Data Rate Benchmark for Desktop (Quadro P100)

- Tested 91 different CNN classifier models
- Maximum real-time inference data rate for 8 different batch sizes
- Using unified memory will increase throughput
Wall Time vs. Compute Time for AIR-T

Real time data rate limited by GNU Radio overhead
Model Accuracy Benchmarks

![Graph showing model accuracy benchmarks with batch size and real-time data rate on the x-axis and classification SNR sensitivity on the y-axis. The graph includes multiple lines representing different batch sizes (1, 2, 4, 16, 32, 64, 128, 256).]
Deepwave Inference Display
Summary

- Deep learning within signal processing is emerging
  - Algorithms may be applied to signal’s data content or signal itself

- High bandwidth requirements driving edge solutions

- Deepwave developed AIR-T
  - Edge-compute inference engine with MIMO transceiver
  - FPGA, CPU, GPU

- GR-Wavelearner
  - Open source inference engine for signal processing
  - Available now on our GitHub page

- Benchmarking analysis demonstrates AIR-T with GR-Wavelearner capable of signal classification inference at 200 MSPS real-time data rates
  - Improvements likely in future release

More info at www.deepwavedigital.com/sdr