Design and Analysis of Convolutional Neural Network for RF Signal Modulation Classification for In-Orbit Deployment

Cognitive Communications for Aerospace Applications (CCAA) Workshop
June 21, 2021

Chris Yakopcic (Prixarc), Tarek M. Taha (Brisk), Sanjeevi Sirisha Karri (Prixarc), Guru Subramanyam (Prixarc), Aaron D. Smith (NASA Glenn), Janette C. Briones (NASA Glenn)
CNN for Automatic Modulation Classification

- Automatic signal modulation classification (AMC) is a major research direction of signal recognition.

- AMC is the automatic identification of the modulation format of the transmitted signals by observing the received data samples which are corrupted by the noise and fading channels.

- It is an intermediate operation between the signal detection and the data demodulation

- AMC plays an important role in civilian and military applications such as software-defined radio, cognitive radio, dynamic spectrum management, interference identification and electronic warfare.
Deep Learning (DL) has been described as a universal learning approach that is able to solve many types of problems in different application domains.

Our focus is on implementing a DL engine in space that would enable Automatic Modulation Classification (AMC) outside of Earth’s atmosphere. Implementation of modulation recognition algorithm would allow for the deployment of real-time, high rate, low-power and useful neural network for RF communications.

We explored a Convolutional Neural Network (CNN), and a Convolution Neural Network that Implements Transfer Learning (CNN-TL) for the successful classification of different modulation schemes for data transmission.

The developed software was shown to successfully classify the modulation schemes using the open source Radio ML 2018 dataset.
CNN Algorithm

- Learns by extracting features from data samples using trainable convolution kernels (filters)
- Last layer is typical fully connected layer
- Very strong for image recognition and classification

Full CNN Training: All Layers are Optimized
CNN for Transfer Learning

- Learns by extracting features from data samples using trainable convolution kernels (filters)
- Last layer is typical fully connected layer
- Very strong for image recognition and classification

- Transfer Learning
  - Pre-train the convolution part of the network
  - Train only the fully connected layer with new data
  - Much simpler to implement in hardware

Feature Extraction

Classification

Convolution

Sub-sampling

Convolution

Sub-sampling

Fully Connected Layer

Outputs

Input

28×28

Feature maps

6@24×24

Feature maps

6@12×12

Feature maps

12@8×8

Feature maps

12@4×4

Pre-trained Layers

Transfer Layers

(Learning Occurs Here)

Full CNN Training: All Layers are Optimized
Radio ML Dataset

- Dataset used in this study
  - RadioML 2018
  - 24 Modulation Classes
  - SNR range: -20 to 30dB
- 4069 samples for each class in each SNR segment
- Sample size $2 \times 1024$ for CNN input
  - I channel
  - Q channel
- Little post processing within data
  - Cut sample lengths
  - Normalize to zero mean and unit variance

<table>
<thead>
<tr>
<th>Class Number</th>
<th>Mod. Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>OOK</td>
</tr>
<tr>
<td>2</td>
<td>4ASK</td>
</tr>
<tr>
<td>3</td>
<td>8ASK</td>
</tr>
<tr>
<td>4</td>
<td>BPSK</td>
</tr>
<tr>
<td>5</td>
<td>QPSK</td>
</tr>
<tr>
<td>6</td>
<td>8PSK</td>
</tr>
<tr>
<td>7</td>
<td>16PSK</td>
</tr>
<tr>
<td>8</td>
<td>32PSK</td>
</tr>
<tr>
<td>9</td>
<td>16APSK</td>
</tr>
<tr>
<td>10</td>
<td>32APSK</td>
</tr>
<tr>
<td>11</td>
<td>64APSK</td>
</tr>
<tr>
<td>12</td>
<td>128APSK</td>
</tr>
<tr>
<td>13</td>
<td>16QAM</td>
</tr>
<tr>
<td>14</td>
<td>32QAM</td>
</tr>
<tr>
<td>15</td>
<td>64QAM</td>
</tr>
<tr>
<td>16</td>
<td>128QAM</td>
</tr>
<tr>
<td>17</td>
<td>256QAM</td>
</tr>
<tr>
<td>18</td>
<td>AM-SSB-WC</td>
</tr>
<tr>
<td>19</td>
<td>AM-SSB-SC</td>
</tr>
<tr>
<td>20</td>
<td>AM-DSB-WC</td>
</tr>
<tr>
<td>21</td>
<td>AM-DSB-SC</td>
</tr>
<tr>
<td>22</td>
<td>FM</td>
</tr>
<tr>
<td>23</td>
<td>GMSK</td>
</tr>
<tr>
<td>24</td>
<td>OQPSK</td>
</tr>
</tbody>
</table>
CNN Transfer Learning

- CNN used for transfer learning test
  - 2 convolution layers
  - 2 fully connected layers

- Dataset broken into two groups
  - Fully train on one set
  - Transfer learn the other set

<table>
<thead>
<tr>
<th>Class</th>
<th>Number</th>
<th>Mod. Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>OOK</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>4ASK</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>8ASK</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>BPSK</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>QPSK</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>8PSK</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>16PSK</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>32PSK</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>16APSK</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>32APSK</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>64APSK</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>128APSK</td>
</tr>
<tr>
<td>13</td>
<td>13</td>
<td>16QAM</td>
</tr>
<tr>
<td>14</td>
<td>14</td>
<td>32QAM</td>
</tr>
<tr>
<td>15</td>
<td>15</td>
<td>64QAM</td>
</tr>
<tr>
<td>16</td>
<td>16</td>
<td>128QAM</td>
</tr>
<tr>
<td>17</td>
<td>17</td>
<td>256QAM</td>
</tr>
<tr>
<td>18</td>
<td>18</td>
<td>AM-SSB-WC</td>
</tr>
<tr>
<td>19</td>
<td>19</td>
<td>AM-SSB-SC</td>
</tr>
<tr>
<td>20</td>
<td>20</td>
<td>AM-DSB-WC</td>
</tr>
<tr>
<td>21</td>
<td>21</td>
<td>AM-DSB-SC</td>
</tr>
<tr>
<td>22</td>
<td>22</td>
<td>FM</td>
</tr>
<tr>
<td>23</td>
<td>23</td>
<td>GMSK</td>
</tr>
<tr>
<td>24</td>
<td>24</td>
<td>0QPSK</td>
</tr>
</tbody>
</table>

**Layer (type) | Output Shape | Param #**
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Conv2d-1</td>
<td>[-1, 32, 1024, 1]</td>
<td>1,632</td>
</tr>
<tr>
<td>ReLU-2</td>
<td>[-1, 32, 1024, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-3</td>
<td>[-1, 32, 512, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Conv2d-4</td>
<td>[-1, 64, 512, 1]</td>
<td>51,264</td>
</tr>
<tr>
<td>ReLU-5</td>
<td>[-1, 64, 512, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-6</td>
<td>[-1, 64, 256, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Dropout-7</td>
<td>[-1, 16384]</td>
<td>0</td>
</tr>
<tr>
<td>Linear-8</td>
<td>[-1, 1, 1000]</td>
<td>16,385,000</td>
</tr>
<tr>
<td>Linear-9</td>
<td>[-1, 24]</td>
<td>24,024</td>
</tr>
</tbody>
</table>

Total params: 16,461,920
Trainable params: 16,461,920
Non-trainable params: 0

Input size (MB): 0.01
Forward/backward pass size (MB): 1.38
Params size (MB): 62.80
Estimated Total Size (MB): 64.19
CNN Transfer Learning

- CNN used for transfer learning test
  - 2 convolution layers
  - 2 fully connected layers
- Dataset broken into two groups
  - Fully train on one set
  - Transfer learn the other set

<table>
<thead>
<tr>
<th>Layer (type)</th>
<th>Output Shape</th>
<th>Param #</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conv2d-1</td>
<td>[-1, 32, 1024, 1]</td>
<td>1,632</td>
</tr>
<tr>
<td>Relu-2</td>
<td>[-1, 32, 1024, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-3</td>
<td>[-1, 32, 512, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Conv2d-4</td>
<td>[-1, 64, 512, 1]</td>
<td>51,264</td>
</tr>
<tr>
<td>Relu-5</td>
<td>[-1, 64, 512, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-6</td>
<td>[-1, 64, 256, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Dropout-7</td>
<td>[1, 16384]</td>
<td>0</td>
</tr>
<tr>
<td>Linear-8</td>
<td>[1, 1000]</td>
<td>16,385,000</td>
</tr>
<tr>
<td>Linear-9</td>
<td>[1, 24]</td>
<td>24,024</td>
</tr>
</tbody>
</table>

Total params: 16,461,920
Trainable params: 16,461,920
Non-trainable params: 0

Input size (MB): 0.01
Forward/backward pass size (MB): 1.38
Params size (MB): 62.88
Estimated Total Size (MB): 64.19
CNN Transfer Learning

- CNN used for transfer learning test
  - 2 convolution layers
  - 2 fully connected layers
- Dataset broken into two groups
  - Fully train on one set
  - Transfer learn the other set

<table>
<thead>
<tr>
<th>Metric</th>
<th>Train A (40 Epochs)</th>
<th>Train B (40 Epochs)</th>
<th>Train A Transfer B (40 + 40 Epochs)</th>
<th>Train B Transfer A (40 + 40 Epochs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>91.15%</td>
<td>80.40%</td>
<td>78.32%</td>
<td>83.34%</td>
</tr>
</tbody>
</table>

Learn Set A

Learn Set B, Transfer Learn Set A

<table>
<thead>
<tr>
<th>Metric</th>
<th>Train A (40 Epochs)</th>
<th>Train B (40 Epochs)</th>
<th>Train A Transfer B (40 + 40 Epochs)</th>
<th>Train B Transfer A (40 + 40 Epochs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy</td>
<td>91.15%</td>
<td>80.40%</td>
<td>78.32%</td>
<td>83.34%</td>
</tr>
</tbody>
</table>

Learn Set A

Learn Set B, Transfer Learn Set A
## CNN Optimization

- **Deeper CNN**
  - More convolution layers
  - More fully connected layers
  - Fewer parameters
  - Experiment is learning all 24 classes and testing using unique untrained data samples

<table>
<thead>
<tr>
<th>Convolution Layers</th>
<th>Filter Size</th>
<th>FC Layers</th>
<th>Epochs</th>
<th>Parameters</th>
<th>Training Accuracy (%)</th>
<th>Testing Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2→16→16→16</td>
<td>1 by 5</td>
<td>2048→250→24</td>
<td>40</td>
<td>521,042</td>
<td>82.0</td>
<td>74.8</td>
</tr>
<tr>
<td></td>
<td>1 by 3</td>
<td>512→250→24</td>
<td>40</td>
<td>519,954</td>
<td>78.9</td>
<td>71.7</td>
</tr>
<tr>
<td>2→16→16→16→16→16</td>
<td>1 by 3</td>
<td>512→128→24</td>
<td>40</td>
<td>137,522</td>
<td>77.4</td>
<td>75.9</td>
</tr>
<tr>
<td>2→16→16→16→16→16→16→16</td>
<td>1 by 3</td>
<td>256→128→24</td>
<td>80</td>
<td>40,024</td>
<td>71.4</td>
<td>70.28</td>
</tr>
<tr>
<td>2→12→12→12→12→12→12→12→12</td>
<td>1 by 3</td>
<td>192→128→24</td>
<td>80</td>
<td>30,104</td>
<td>66.3</td>
<td>65.2</td>
</tr>
<tr>
<td>2→12→12→12→12→12→12→12→12→12→12→12→12→12→12→12→12</td>
<td>1 by 3</td>
<td>192→64→24</td>
<td>160</td>
<td>16,216</td>
<td>77.8</td>
<td>76.5</td>
</tr>
<tr>
<td>2→8→8→8→8→8→8→8→8→8→8</td>
<td>1 by 3</td>
<td>128→64→24</td>
<td>80</td>
<td>10,872</td>
<td>68.3</td>
<td>67.4</td>
</tr>
<tr>
<td>2→8→8→8→8→8→8→8→8→8→8</td>
<td>1 by 3</td>
<td>128→24</td>
<td>160</td>
<td>4,152</td>
<td>54.6</td>
<td>54.06</td>
</tr>
</tbody>
</table>
## CNN Optimization

- Range of CNN designs were evaluated to find tradeoff between number of parameters and accuracy
- Two bold networks show strong accuracy vs. throughput results

<table>
<thead>
<tr>
<th>Convolution Layers</th>
<th>Filter Size</th>
<th>FC Layers</th>
<th>Epochs</th>
<th>Parameters</th>
<th>Training Accuracy (%)</th>
<th>Testing Accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2→16→16→16→16</td>
<td>1 by 5</td>
<td>2048→250→24</td>
<td>40</td>
<td>521,042</td>
<td>82.0</td>
<td>74.8</td>
</tr>
<tr>
<td></td>
<td>1 by 3</td>
<td></td>
<td>40</td>
<td>519,954</td>
<td>78.9</td>
<td>71.7</td>
</tr>
<tr>
<td>2→16→16→16→16→16→16</td>
<td>1 by 3</td>
<td>512→250→24</td>
<td>40</td>
<td>137,522</td>
<td>77.4</td>
<td>75.9</td>
</tr>
<tr>
<td></td>
<td></td>
<td>512→128→24</td>
<td>40</td>
<td>72,008</td>
<td>71.4</td>
<td>70.28</td>
</tr>
<tr>
<td>2→16→16→16→16→16→16</td>
<td>1 by 3</td>
<td>256→128→24</td>
<td>40</td>
<td>40,024</td>
<td>76.7</td>
<td>75.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>80</td>
<td>40,024</td>
<td>82.7</td>
<td>81.4</td>
</tr>
<tr>
<td>2→12→12→12→12→12→12→12→12</td>
<td>1 by 3</td>
<td>192→128→24</td>
<td>40</td>
<td>30,104</td>
<td>66.3</td>
<td>65.2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>80</td>
<td>30,104</td>
<td>75.7</td>
<td>74.8</td>
</tr>
<tr>
<td>2→12→12→12→12→12→12→12→12</td>
<td>1 by 3</td>
<td>192→64→24</td>
<td>80</td>
<td>16,216</td>
<td>77.8</td>
<td>76.5</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>160</td>
<td>16,216</td>
<td>78.2</td>
<td>77.4</td>
</tr>
<tr>
<td>2→8→8→8→8→8→8→8→8→8→8</td>
<td>1 by 3</td>
<td>128→64→24</td>
<td>80</td>
<td>10,872</td>
<td>68.3</td>
<td>67.4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>160</td>
<td>10,872</td>
<td>72.5</td>
<td>71.7</td>
</tr>
<tr>
<td>2→8→8→8→8→8→8→8→8</td>
<td>1 by 3</td>
<td>128→24</td>
<td>160</td>
<td>4,152</td>
<td>54.6</td>
<td>54.06</td>
</tr>
</tbody>
</table>
CNN Optimization

- Deeper CNN
  - 6 convolution layers
  - 2 fully connected layers
  - More layers and fewer parameters

<table>
<thead>
<tr>
<th>Layer (type)</th>
<th>Output Shape</th>
<th>Param #</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conv2d-1</td>
<td>[-1, 12, 1824, 1]</td>
<td>72</td>
</tr>
<tr>
<td>ReLU-2</td>
<td>[-1, 12, 1824, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-3</td>
<td>[-1, 12, 512, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Conv2d-4</td>
<td>[-1, 12, 512, 1]</td>
<td>432</td>
</tr>
<tr>
<td>ReLU-5</td>
<td>[-1, 12, 512, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-6</td>
<td>[-1, 12, 256, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Conv2d-7</td>
<td>[-1, 12, 256, 1]</td>
<td>432</td>
</tr>
<tr>
<td>ReLU-8</td>
<td>[-1, 12, 256, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-9</td>
<td>[-1, 12, 128, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Conv2d-10</td>
<td>[-1, 12, 128, 1]</td>
<td>432</td>
</tr>
<tr>
<td>ReLU-11</td>
<td>[-1, 12, 128, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-12</td>
<td>[-1, 12, 64, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Conv2d-13</td>
<td>[-1, 12, 64, 1]</td>
<td>432</td>
</tr>
<tr>
<td>ReLU-14</td>
<td>[-1, 12, 64, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-15</td>
<td>[-1, 12, 32, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Conv2d-16</td>
<td>[-1, 12, 32, 1]</td>
<td>432</td>
</tr>
<tr>
<td>ReLU-17</td>
<td>[-1, 12, 32, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-18</td>
<td>[-1, 12, 16, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Dropout-19</td>
<td>[-1, 192]</td>
<td>0</td>
</tr>
<tr>
<td>Linear-20</td>
<td>[-1, 64]</td>
<td>12,288</td>
</tr>
<tr>
<td>Linear-21</td>
<td>[-1, 12]</td>
<td>768</td>
</tr>
</tbody>
</table>

Total params: 15,288
Trainable params: 15,288
Non-trainable params: 0

Input size (MB): 0.01
Forward/backward pass size (MB): 0.46
Params size (MB): 0.06
Estimated Total Size (MB): 0.53
Optimized CNN Transfer Learning

- Transfer Learning to Add Class
  - Step 1: Train CNN to learn 4 modulation types

<table>
<thead>
<tr>
<th>Layer (type)</th>
<th>Output Shape</th>
<th>Param #</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conv2d-1</td>
<td>[-1, 12, 1024, 1]</td>
<td>72</td>
</tr>
<tr>
<td>ReLU-2</td>
<td>[-1, 12, 1024, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-3</td>
<td>[-1, 12, 512, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Conv2d-4</td>
<td>[-1, 12, 512, 1]</td>
<td>432</td>
</tr>
<tr>
<td>ReLU-5</td>
<td>[-1, 12, 512, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-6</td>
<td>[-1, 12, 256, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Conv2d-7</td>
<td>[-1, 12, 256, 1]</td>
<td>432</td>
</tr>
<tr>
<td>ReLU-8</td>
<td>[-1, 12, 256, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-9</td>
<td>[-1, 12, 128, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Conv2d-10</td>
<td>[-1, 12, 128, 1]</td>
<td>432</td>
</tr>
<tr>
<td>ReLU-11</td>
<td>[-1, 12, 128, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-12</td>
<td>[-1, 12, 64, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Conv2d-13</td>
<td>[-1, 12, 64, 1]</td>
<td>432</td>
</tr>
<tr>
<td>ReLU-14</td>
<td>[-1, 12, 64, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-15</td>
<td>[-1, 12, 32, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Conv2d-16</td>
<td>[-1, 12, 32, 1]</td>
<td>432</td>
</tr>
<tr>
<td>ReLU-17</td>
<td>[-1, 12, 32, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-18</td>
<td>[-1, 12, 16, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Dropout-19</td>
<td>[-1, 192]</td>
<td>0</td>
</tr>
<tr>
<td>Linear-20</td>
<td>[-1, 64]</td>
<td>12,288</td>
</tr>
<tr>
<td>Linear-21</td>
<td>[-1, 12]</td>
<td>768</td>
</tr>
</tbody>
</table>

Total params: 15,288
Trainable params: 15,288
Non-trainable params: 0

Input size (MB): 0.01
Forward/backward pass size (MB): 0.46
Params size (MB): 0.06
Estimated Total Size (MB): 0.53
Optimized CNN Transfer Learning

- Transfer Learning to Add Class
  - Step 2: Test with 4 learned modulations in addition to a new unlearned class

<table>
<thead>
<tr>
<th>Layer (type)</th>
<th>Output Shape</th>
<th>Param #</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conv2d-1</td>
<td>[-1, 12, 1024, 1]</td>
<td>72</td>
</tr>
<tr>
<td>ReLU-2</td>
<td>[-1, 12, 1024, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-3</td>
<td>[-1, 12, 512, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Conv2d-4</td>
<td>[-1, 12, 512, 1]</td>
<td>432</td>
</tr>
<tr>
<td>ReLU-5</td>
<td>[-1, 12, 512, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-6</td>
<td>[-1, 12, 256, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Conv2d-7</td>
<td>[-1, 12, 256, 1]</td>
<td>432</td>
</tr>
<tr>
<td>ReLU-8</td>
<td>[-1, 12, 256, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-9</td>
<td>[-1, 12, 128, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Conv2d-10</td>
<td>[-1, 12, 128, 1]</td>
<td>432</td>
</tr>
<tr>
<td>ReLU-11</td>
<td>[-1, 12, 128, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-12</td>
<td>[-1, 12, 64, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Conv2d-13</td>
<td>[-1, 12, 64, 1]</td>
<td>432</td>
</tr>
<tr>
<td>ReLU-14</td>
<td>[-1, 12, 64, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-15</td>
<td>[-1, 12, 32, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Conv2d-16</td>
<td>[-1, 12, 32, 1]</td>
<td>432</td>
</tr>
<tr>
<td>ReLU-17</td>
<td>[-1, 12, 32, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-18</td>
<td>[-1, 12, 16, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Dropout-19</td>
<td>[-1, 192]</td>
<td>0</td>
</tr>
<tr>
<td>Linear-20</td>
<td>[-1, 64]</td>
<td>12,288</td>
</tr>
<tr>
<td>Linear-21</td>
<td>[-1, 12]</td>
<td>768</td>
</tr>
</tbody>
</table>

Total params: 15,288
Trainable params: 15,288
Non-trainable params: 0

Input size (MB): 0.01
Forward/backward pass size (MB): 0.46
Params size (MB): 0.06
Estimated Total Size (MB): 0.53
Optimized CNN Transfer Learning

- **Transfer Learning to Add Class**
  - Step 3: Use transfer learning to train only the fully connected layers
  - Step 4: Test if the network is able to learn all 5 modulations

---

### Training Here

<table>
<thead>
<tr>
<th>Layer</th>
<th>Output Shape</th>
<th>Param #</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conv2d-1</td>
<td>[-1, 12, 1024, 1]</td>
<td>72</td>
</tr>
<tr>
<td>ReLU-2</td>
<td>[-1, 12, 1024, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-3</td>
<td>[-1, 12, 512, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Conv2d-4</td>
<td>[-1, 12, 512, 1]</td>
<td>432</td>
</tr>
<tr>
<td>ReLU-5</td>
<td>[-1, 12, 512, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-6</td>
<td>[-1, 12, 256, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Conv2d-7</td>
<td>[-1, 12, 256, 1]</td>
<td>432</td>
</tr>
<tr>
<td>ReLU-8</td>
<td>[-1, 12, 256, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-9</td>
<td>[-1, 12, 128, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Conv2d-10</td>
<td>[-1, 12, 128, 1]</td>
<td>432</td>
</tr>
<tr>
<td>ReLU-11</td>
<td>[-1, 12, 128, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-12</td>
<td>[-1, 12, 64, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Conv2d-13</td>
<td>[-1, 12, 64, 1]</td>
<td>432</td>
</tr>
<tr>
<td>ReLU-14</td>
<td>[-1, 12, 64, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-15</td>
<td>[-1, 12, 32, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Conv2d-16</td>
<td>[-1, 12, 32, 1]</td>
<td>432</td>
</tr>
<tr>
<td>ReLU-17</td>
<td>[-1, 12, 32, 1]</td>
<td>0</td>
</tr>
<tr>
<td>MaxPool2d-18</td>
<td>[-1, 12, 16, 1]</td>
<td>0</td>
</tr>
<tr>
<td>Propout-19</td>
<td>[-1, 12]</td>
<td>12,288</td>
</tr>
<tr>
<td>Linear-20</td>
<td>[-1, 12]</td>
<td>768</td>
</tr>
<tr>
<td>Linear-21</td>
<td>[-1, 12]</td>
<td>0</td>
</tr>
</tbody>
</table>

**Total params:** 15,288
**Trainable params:** 15,288
**Non-trainable params:** 0

**Input size (MB):** 0.01
**Forward/backward pass size (MB):** 0.46
**Params size (MB):** 0.06
**Estimated Total Size (MB):** 0.53
Conclusion and Future Work

• **Summary**
  - CNN for AMC
  - Low power deployment of signal modulation classification
    - Through CNN optimization
  - Transfer learning makes system adaptable
    - Also reduces complexity of training if deployed on custom hardware

• **Future Work**
  - Hardware Survey
    - Best options for low SWaP deployment
    - Algorithm refinement
      - Optimize throughput and classification accuracy
    - Dataset improvement
      - Generate custom dataset using SDR for real world examination