MACHINE LEARNING CONCEPTS

Ali Osman Ors
Director, AI ML Strategy and Technologies

JUNE 2021
Artificial Intelligence
- The very broad concept of using machines to do “smart” things and act intelligently like a human

Machine Learning
- One way to implement AI
- Give machines a lot of data and they can learn how to do smart things (supervised and unsupervised)

Deep Learning
- One of many ways to implement machine learning
- Uses Neural Networks that can learn and make intelligent decisions on its own
- Needs a lot of data
WHY MACHINE LEARNING IS MOVING TO THE EDGE

- Safeguard Privacy
- Experience
- Higher Reliability
- Enable Real Time
- Increase Security
- Reduce Cloud Costs Including: Bandwidth, Processing, Storage
MACHINE LEARNING USE CASES, DATA INPUTS AND COMPUTE ACCELERATORS

**Vision**
- Face and still image recognition, person detection (images)
- Multi-face recognition, object detection (video)
- Live video face and object recognition
- Multi-object surveillance (people, cars, animals)

**Audio**
- Wake word, 10 Word speech, speaker recognition
- Automatic speech recognition (basic command phrases)
- 40,000 words vocabulary, multiple speaker recognition
- Speech accents interpretation

**Time Series Data**
- Anomaly detection (environmental sensors)
- Pose estimation
- Gesture recognition
- Complex real-time motion analysis

- Cortex-M → 4x Cortex-A53 → GPU → Neural Processing Unit (NPU) → 3 (TOPS)
VISION BASED MACHINE LEARNING APPLICATION TERMINOLOGY

Image Classification

Classification with Localization

Object Detection

Instance Segmentation

Apple

Apple

Apples and Orange

Apples and Orange

Facial Detection

Facial Recognition

This is a person’s face

This is NXP CEO Kurt Sievers
**VERY SIMPLIFIED NEURAL NETWORK MODEL**

Node Value \( a_x = \text{ReLU}(w_1a_1 + w_2a_2 + w_3a_3 \ldots - b) \)

Weights \( (w_x) \)

Bias

**Model**

Input Layer

Hidden Layer(s)

Output Layer

- Armadillo
- Dog
- Cat
- Dodo

**Example**

\[ \text{Node Value } a_x = \text{ReLU}(w_1a_1 + w_2a_2 + w_3a_3 \ldots - b) \]

**Weights**

- \( w_1 \) = 0.01
- \( w_2 \) = 0.05
- \( w_3 \) = 0.93
- \( w_4 \) = 0.01
TYPICAL LAYER AND FUNCTION TYPES IN CNN

• Convolutional Layer – performs feature detection
• Max Pooling Layer – scales down input images
• Fully Connected Layer – performs classification
• ReLU- Rectified linear activation function, non-linear function that outputs zero if the input is less than zero
• Softmax Layer – estimates probability for categories
• Linear Layer – performs regression, estimates object position
• Dropout Layer – reduces overfitting
NEURAL NETWORK MODELS REPRESENTS A SEQUENCE OF COMPUTATIONS

- Neural network model comprised of multiple layers of operations (e.g. convolution, activation functions, pooling)
- Optimizations include layer merging, operator replacements, pruning, quantization, etc.

ONE CLASS SUPPORT VECTOR MACHINES

- Used for anomaly detection
- This algorithm indicates if sample is part of a known population or not.

We are using a Gaussian Kernel

\[ f_{svm}(x) = \sum_{i=1}^{n} \alpha_i e^{-\frac{||x-sv_i||^2}{2\sigma^2}} \]

a sample is considered True (1) if pdf \( f_{svm}(x) > \text{threshold} \) and False (-1) otherwise

\[ \text{sgn}[f_{svm}(x) - \text{thr}] \]

This can even run on a Cortex®-M4F device!
MACHINE LEARNING DEVELOPMENT WORK FLOW

Data Preparation & Training

Validation

Deployment

Inference

- Dataset
- Model Code
- Train
- Validate
- Optimize/Convert
- Converted Model
- Input Image/data
- Converted Model
- Deployment
- Inference
HOW ARE MODELS DESIGNED? – MODEL FRAMEWORKS

• A training framework provides proven APIs and utilities to design, analyze, train, test, validate and deploy models.

• Each framework has their own APIs and methodologies

• Allows developers to focus on overall logic of model, instead of the details of how to implement algorithms or link layers together
MODEL FRAMEWORKS

• There are several popular model frameworks in use today.
• This is a constantly changing list as new software is released:

- **TensorFlow**  
  Framework offered by Google

- **Keras**  
  Higher level API, usually built on top of TensorFlow

- **PyTorch**  
  Framework offered by Facebook

- **ONNX**  
  Not a deep learning framework, but an exchange format library providing framework interoperability between models and software tools

• Python is used extensively for ML frameworks (interact, build, and train)
## Build and Train Models Using Cloud Services (e.g. Amazon SageMaker)

<table>
<thead>
<tr>
<th>Label</th>
<th>Build</th>
<th>Train &amp; Tune</th>
<th>Deploy &amp; Manage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Amazon SageMaker Ground Truth</td>
<td>Amazon SageMaker Studio</td>
<td>Amazon SageMaker Autopilot</td>
<td>Amazon SageMaker Neo</td>
</tr>
<tr>
<td>Build and manage training data sets</td>
<td>Integrated development environment (IDE)</td>
<td>Automatically build and train models</td>
<td>Train once, deploy anywhere</td>
</tr>
<tr>
<td></td>
<td>for machine learning</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AWS Marketplace</td>
<td>Amazon SageMaker Debugger</td>
<td></td>
<td>Amazon Augmented AI</td>
</tr>
<tr>
<td>Pre-built algorithms and models</td>
<td>Debug and profile training runs</td>
<td></td>
<td>Add human review of model predictions</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Automatic Model Tuning</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>One-click hyperparameter optimization</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
MODEL ZOOS HELP PROVIDE STARTING POINT IN MODEL DEVELOPMENT

TensorFlow Model Garden

TensorFlow Official Models

The TensorFlow official models are a collection of models that use TensorFlow's high-level APIs. They are intended to be well-maintained, tested, and kept up to date with the latest TensorFlow API. They should also be reasonably optimized for fast performance while still being easy to read. These models are used as end-to-end tests, ensuring that the models run with the same or improved speed and performance with each new TensorFlow build.
MODEL ACCURACY

• Every time an inference is run, most models will give a confidence percentage
  - This is why last (or one of last) layers in a classification model is often “softmax” which converts final result to be between 0 and 1

• Every model has limitations

• Goal of Machine Learning
  + Accuracy
  - Size
  - Inference time
MODEL ACCURACY CONTINUED

• Models are not perfect. Especially when scaling down models to fit on embedded systems

<table>
<thead>
<tr>
<th>Model</th>
<th>Million MACs</th>
<th>Million Parameters</th>
<th>Top-1 Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>MobileNet_v1_1.0_224</td>
<td>569</td>
<td>4.24</td>
<td>70.9</td>
</tr>
<tr>
<td>MobileNet_v1_1.0_192</td>
<td>418</td>
<td>4.24</td>
<td>70.0</td>
</tr>
<tr>
<td>MobileNet_v1_1.0_180</td>
<td>291</td>
<td>4.24</td>
<td>68.0</td>
</tr>
<tr>
<td>MobileNet_v1_1.0_128</td>
<td>186</td>
<td>4.24</td>
<td>65.2</td>
</tr>
<tr>
<td>MobileNet_v1_0.75_224</td>
<td>317</td>
<td>2.59</td>
<td>68.4</td>
</tr>
<tr>
<td>MobileNet_v1_0.75_192</td>
<td>233</td>
<td>2.59</td>
<td>67.2</td>
</tr>
<tr>
<td>MobileNet_v1_0.75_160</td>
<td>162</td>
<td>2.59</td>
<td>65.3</td>
</tr>
<tr>
<td>MobileNet_v1_0.75_128</td>
<td>104</td>
<td>2.59</td>
<td>62.1</td>
</tr>
<tr>
<td>MobileNet_v1_0.50_224</td>
<td>150</td>
<td>1.34</td>
<td>63.3</td>
</tr>
<tr>
<td>MobileNet_v1_0.50_192</td>
<td>110</td>
<td>1.34</td>
<td>61.7</td>
</tr>
<tr>
<td>MobileNet_v1_0.50_160</td>
<td>77</td>
<td>1.34</td>
<td>59.1</td>
</tr>
<tr>
<td>MobileNet_v1_0.50_128</td>
<td>49</td>
<td>1.34</td>
<td>56.3</td>
</tr>
<tr>
<td>MobileNet_v1_0.25_224</td>
<td>41</td>
<td>0.47</td>
<td>49.8</td>
</tr>
<tr>
<td>MobileNet_v1_0.25_192</td>
<td>34</td>
<td>0.47</td>
<td>47.7</td>
</tr>
<tr>
<td>MobileNet_v1_0.25_160</td>
<td>21</td>
<td>0.47</td>
<td>45.5</td>
</tr>
<tr>
<td>MobileNet_v1_0.25_128</td>
<td>14</td>
<td>0.47</td>
<td>41.5</td>
</tr>
</tbody>
</table>
QUANTIZATION AND PRUNING

Quantization
• Transform 32-bit floating point weights $\rightarrow$ 8-bit fixed point weights
  - Reduces model size by 4x
  - Fixed point math quicker than floating point
  - Usually little loss of accuracy

Pruning
• Remove low importance weights and biases from a neural network
  - Recommended to retrain model after pruning
REFERENCES AND HELPFUL LINKS

• Embedding Intelligence at the Edge
  (https://www.nxp.com/ai)

• eIQ™ ML Software Development Environment
  (https://www.nxp.com/eig)

• eIQ Community

• Embedded Linux for i.MX Applications Processors
  (https://www.nxp.com/design/software/embedded-software/i-mx-software/embedded-linux-for-i-mx-applications-processors:IMX LINUX)

• MCUXpresso Software and Tools