Compute Spectrum: AI

Cloud

PC

Large Edge
  > 128MB
  > 1GHz+

Tiny Edge
  ~1MB
  ~16MHz

TinyML
  (EdgeML+ELL, µTensor, cube.ai, ...)

- Microsoft CNTK
- TensorFlow
- ONNX
- PyTorch
- TensorFlow Lite
- Core ML
- AWS Greengrass
- Azure IOT
What, why, how?

• What type of devices are we talking about?

• Why do you care for these devices?

• How will we enable ML on these devices?
Resource-constrained IoT Devices

Freescale KL03 microcontroller
ARM® Cortex®-M0+ processor

48 MHz
32 KB Flash, 8KB boot ROM, 2 KB RAM
35μA/MHz low-power active mode
1 μA sleep mode

ARM Cortex M0+ at 48 MHz & 35 μA/MHz with 2 KB RAM & 32 KB read only Flash
Communication is more expensive than computation

- Wellness-centric wearables
  - Privacy
  - Battery
  - Latency

- Smart farms
  - Bandwidth
  - Battery

- Smart meter, smart city
  - Battery Cost

- Smart sports
  - Latency
  - Battery

- Smart factory
  - Battery
  - Bandwidth

- Smart appliance
  - Latency
  - Battery
  - Privacy
ML on microcontrollers need optimization on:
• All four fronts
• Deployment ease front (Microsoft’s ELL, Google+ARM’s micro-tensor, STM’s cube.ai...)
Broad approaches for TinyML

- **Search better architectures**
  (Proxyless NAS, EfficientNets, ...)
- **Compress existing architecture**
  (Deep Compression, Xnor net...)
- **Design new architectures/blocks**
  (SqueezeNet, MobileNet, EdgeML, ...)

---

- **Yujun, Bichen’s Talk**
- **Shih-Chii, Yujun, Bichen’s Talk**
- **This Talk, Andrew’s Talk**
Edge Machine Learning (EdgeML) – Objectives

- To build a library of machine learning algorithms
  - Which can be trained in the cloud
  - But which will run on tiny IoT devices

ARM Cortex M0+
Microsoft’s EdgeML Library

- Compact tree, kNN and RNN algorithms for classification, regression, ranking, time series etc.,

https://github.com/Microsoft/EdgeML
EdgeML Building Blocks

- **Bonsai**
- **ProtoNN**
- **EMI-RNN**
- **FastGRNN**
- **ShaRNN**
- **RnnPool**

**Feed-forward layer, GBDT/SVM...**

**LSTM/GRU + Streaming data**

**CNNs + Max/Avg Pooling**
Two key ideas:
• Sparse projection: reduce dimensionality and learn good distance metric
• Learn prototypes: reduce model size, prediction time

Parameters to learn:
• Z: Projection Matrix
• $b_1, ..., b_m$: prototypes
• $w_1, ..., w_m$: label vector for each prototype
Comparison to Uncompressed Methods

Accuracy (%)

Model Size (KB)

Compressed:
- Bonsai
- ProtoNN

Uncompressed:
- GBDT
- kNN
- RBF-SVM
- Neural Nets
Prediction Accuracy vs Model Size

CUReT-61

Eye-2

Accuracy (%) vs Model Size (KB)

- ProtoNN
- Bonsai
- GBDT
- Tree Pruning
- LDKL
- LDKL-L1
- NeuralNet Pruning
- SNC
- Decision Jungle
- BudgetRF
- PruneRF
Microsoft’s EdgeML Library

Bonsai, ProtoNN, EMI-RNN, FastGRNN, ShaRNN, Feed-forward layer, GBDT/SVM...

LSTM/GRU + Streaming data

CNNs + Max/Avg Pooling
Recurrent Neural Networks (RNNs)

- State-of-the-art for analyzing sequences & time series
- Training is unstable due to exploding & vanishing gradients

\[ h_t = \sigma(Wx_t + Uh_{t-1} + b) \]
Recurrent Neural Networks (RNNs)

- State-of-the-art for analyzing sequences & time series
- Training is unstable due to exploding & vanishing gradients

\[
\nabla = f(\ldots, U^T = Q \begin{bmatrix} 2^{100} & \ldots & 0.5^{100} \end{bmatrix} Q^T, \ldots)
\]
FastRNN

- Provably stable training with 2 additional scalars
- Accuracy: RNN ≪ Unitary RNNs < FastRNN < Gated RNNs

\[ \tilde{h}_t = \sigma(Wx_t + Uh_{t-1} + b) \]

\[ h_t = \alpha \tilde{h}_t + \beta h_{t-1} \]
FastRNN

- Provably stable training with 2 additional scalars
- Accuracy: RNN ≪ Unitary RNNs < FastRNN < Gated RNNs

\[ \nabla = f(\ldots, (\alpha UD + \beta I)^T) = Q \begin{bmatrix} (\beta + \alpha \|UD\|)^T \\ \vdots \\ (\beta - \alpha \|UD\|)^T \end{bmatrix} Q^T, \ldots \]
FastGRNN

- Make \( \mathbf{U} \) and \( \mathbf{W} \) low-rank (L), sparse (S) and quantized (Q)
- Model Size: \( \text{FastGRNN} \ll \text{RNN} \approx \text{Unitary RNNs} < \text{Gated RNNs} \)
Comparison to Gated Architectures

- Uncompressed FastGRNN is as accurate as a GRU/LSTM
- FastGRNN is almost as accurate as a GRU/LSTM (within 1%)
- FastGRNN is 20-80x smaller than a GRU/LSTM

<table>
<thead>
<tr>
<th>Model Size (KB)</th>
<th>Google-12</th>
<th>Google-30</th>
<th>Yelp-5</th>
<th>DSA-19</th>
<th>Pixel-MNIST-10</th>
</tr>
</thead>
<tbody>
<tr>
<td>FastGRNN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Uncompressed FastGRNN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FastRNN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RNN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UGRNN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRU</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSTM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Accuracy (%)</th>
<th>Google-12</th>
<th>Google-30</th>
<th>Yelp-5</th>
<th>DSA-19</th>
<th>Pixel-MNIST-10</th>
</tr>
</thead>
<tbody>
<tr>
<td>FastGRNN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Uncompressed FastGRNN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>FastRNN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RNN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UGRNN</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>GRU</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSTM</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Prediction on Edge Devices

- None of the other RNNs fit on an Arduino Uno
- FastGRNN can be 25-132x faster at prediction on the MKR1K
Key Technical Take-aways

• Why RNN?
  – Time-series analysis is critical for TinyML applications

• How to deploy RNNs?
  – Faster and more efficient RNN cells (FastGRNN)

• How to enable end-to-end architecture on tiny devices?
  – Parallelize RNNs, CNN+RNNs, ... (EMI-RNN, ShaRNN, RnnPool)
Time Series

• Time series are the most frequently occurring types of signals found in the IoT domain

Hey, Cortana

Sprinklers

Soil moisture during a day

Smart farm
RNNs(ShaRNN) vs CNN

- CNN(10, 2) Accuracy: 0.81, Memory: 100kB
- CNN(10, 4) Accuracy: 0.85, Memory: 100kB
- CNN(20, 2) Accuracy: 0.83, Memory: Not shown
- CNN(20, 4) Accuracy: 0.88, Memory: Not shown
- ShaRNN Accuracy: 0.91, Memory: 100kB
Key Technical Take-aways

• Why RNN?
  – Time-series analysis is critical for TinyML applications

• How to deploy RNNs?
  – Faster and more efficient RNN cells (FastGRNN)

• How to enable end-to-end architecture on tiny devices?
  – Parallelize RNNs, CNN+RNNs, ... (EMI-RNN, ShaRNN, RnnPool)
Key Technical Take-aways

• Why RNN?
  – Time-series analysis is critical for TinyML applications

• How to deploy RNNs?
  – Faster and more efficient RNN cells (FastGRNN)

• How to enable end-to-end architecture on tiny devices?
  – Parallelize RNNs, CNN+RNNs, ... (EMI-RNN, ShaRNN, RnnPool)
“When I was traveling by metro, holding the cane in one hand and the railing (support) in the other hand. I am getting calls, before it was not possible (to answer them). Now I can talk using the cane.” —P2
Conclusions

- Resource-aware ML is critical for real-world deployment of AI
  - IoT devices +ML provides many high-impact opportunities

- Applications:
  - Time-series critical
  - RNN a memory efficient tool, but requires careful orchestration

- Microsoft’s EdgeML Library (https://github.com/Microsoft/EdgeML)
  - Bonsai, ProtoNN, FastGRNN, ShaRNN, RnnPool & EMI-RNN
    - Few Kilobytes of memory, milliseconds latency
    - Have state-of-the-art prediction accuracies
Compressing Cloud Models

Design new tiny ML models from scratch!

- 10-15% loss in accuracy, if model size <100KB needed for tiny edge
FastGRNN

A Fast, Accurate, Stable & Tiny (Kilobyte Sized) Gated RNN

A. Kusupati (MSRI), M. Singh (IITD), K. Bhatia (Berkeley), A. Kumar (Berkeley), P. Jain (MSRI) & M. Varma (MSRI)
Recurrent Neural Networks (RNNs)

- State-of-the-art for analyzing sequences & time series
- Training is unstable due to exploding & vanishing gradients

\[ h_t = \sigma(Wx_t + Uh_{t-1} + b) \]
Unitary RNNs – uRNN, SpectralRNN, ...

- Unitary RNNs force all the eigenvalues of $U$ to be $\approx 1$
- Unfortunately, they are expensive to train & lack accuracy

\[ \nabla = f(\ldots, U^T = Q \begin{bmatrix} 1^T & \cdots & 1^T \end{bmatrix} Q^T, \ldots) \]
Gated RNNs – LSTM, GRU, ...

- Add extra parameters to stabilize training
- Have increased prediction costs on IoT microcontrollers
- Have intuitive explanations but lack formal guarantees
FastGRNN

- Extend $\alpha$ & $\beta$ from scalars to vector gates
- Accuracy: $\text{RNN} \ll \text{Unitary RNNs} < \text{Gated RNNs} \approx \text{FastGRNN}$

\[
\beta_t = \sigma_\beta(Wx_t + Uh_{t-1} + b_\beta); \quad \tilde{h}_t = \sigma_h(Wx_t + Uh_{t-1} + b_h)
\]
\[
\alpha_t \approx 1 - \beta_t; \quad h_t = \alpha_t \odot \tilde{h}_t + \beta_t \odot h_{t-1}
\]
<table>
<thead>
<tr>
<th>Dataset</th>
<th># Train</th>
<th># Features</th>
<th># Time Steps</th>
<th># Test</th>
</tr>
</thead>
<tbody>
<tr>
<td>Google-12</td>
<td>22,246</td>
<td>3,168</td>
<td>99</td>
<td>3,081</td>
</tr>
<tr>
<td>Google-30</td>
<td>51,088</td>
<td>3,168</td>
<td>99</td>
<td>6,835</td>
</tr>
<tr>
<td>Wakeword-2</td>
<td>195,800</td>
<td>5,184</td>
<td>162</td>
<td>83,915</td>
</tr>
<tr>
<td>Yelp-5</td>
<td>500,000</td>
<td>38,400</td>
<td>300</td>
<td>500,000</td>
</tr>
<tr>
<td>PTB-10000</td>
<td>929,589</td>
<td>---</td>
<td>300</td>
<td>82,430</td>
</tr>
<tr>
<td>HAR-2</td>
<td>7,352</td>
<td>1,152</td>
<td>128</td>
<td>2,947</td>
</tr>
<tr>
<td>DSA-19</td>
<td>4,560</td>
<td>5,625</td>
<td>125</td>
<td>4,560</td>
</tr>
<tr>
<td>Pixel-MNIST-10</td>
<td>60,000</td>
<td>784</td>
<td>784</td>
<td>10,000</td>
</tr>
</tbody>
</table>
Testimonials

- Constrained environments:
  - "When I was traveling by metro, holding the cane in one hand and the railing (support) in the other hand. I am getting calls, before it was not possible (to answer them). Now I can talk using the cane.” —P2

- Situational awareness:
  - If I want to go from office to Parngipalla, sometime I will get confuse in the crosses, if I check the location it was saying the cross 27th main 18th cross. So I was able to easily find out I am in this particular cross and I can navigate well. —P3
Copyright Notice

The presentation(s) in this publication comprise the proceedings of tinyML® Summit 2020. The content reflects the opinion of the authors and their respective companies. This version of the presentation may differ from the version that was presented at the tinyML Summit. The inclusion of presentations in this publication does not constitute an endorsement by tinyML Foundation or the sponsors.

There is no copyright protection claimed by this publication. However, each presentation is the work of the authors and their respective companies and may contain copyrighted material. As such, it is strongly encouraged that any use reflect proper acknowledgement to the appropriate source. Any questions regarding the use of any materials presented should be directed to the author(s) or their companies.

tinyML is a registered trademark of the tinyML Foundation.

www.tinyML.org