Small is Big: Making DNNs faster & energy-efficient for low-power hardware

Deeplite Inc.

Davis Sawyer & Charles Marsh
Nov 3rd, 2020
<table>
<thead>
<tr>
<th>Date</th>
<th>Title</th>
<th>Host</th>
</tr>
</thead>
<tbody>
<tr>
<td>November 3, 2020</td>
<td>Small is big: Making Deep Neural Nets faster and energy-efficient on low power hardware</td>
<td>Deeplite</td>
</tr>
<tr>
<td>November 17, 2020</td>
<td>The Smart City in Motion - AI in intelligent transportation</td>
<td>Clever Devices, NXP, Arcturus</td>
</tr>
<tr>
<td>December 8th, 2020</td>
<td>Bringing Spatial AI to embedded devices</td>
<td>SLAMCore</td>
</tr>
</tbody>
</table>
Introduction

Davis Sawyer
Co-founder and CPO @ Deeplite

Charles Marsh
CCO @ Deeplite
Contents

• Introduction (3 Big Trends)
• tinyML is taking off
• Domain Specific Architectures and DNN Model Optimization
• Software stacks for edge AI
• Takeaways and Summary

• Open Mic Q&A Session
3 Big Things

• **tinyML**
  • Sensor data in IoT can be analyzed with compact AI models
  • Huge opportunity for battery powered devices
  • Just starting to scratch the surface of applications

• **Domain-Specific Architectures**
  • Optimize to do few a things very well
  • Codesigning compact AI models for efficient AI hardware
  • Unlock more efficient designs through new model design space exploration

• **Easy to use AI software**
  • Lower friction to adoption & deployment
  • Building out the SW infrastructure stack will unlock more applications
Part 1: tinyML is taking off

- Massive value unlocked by making AI applicable for low-power devices
- AI inference must meet strict memory, speed, cost and resource constraints

Q4 2019 shipped a record 6.4 billion Arm-based chips, including a record 4.2 billion Cortex-M processors for embedded and IoT devices
Part 1: tinyML is taking off

- Huge amount of mili/microwatt devices and use cases across many verticals
- Availability of compact, high performance algorithms for multiple data types
- Anomaly detection, visual detection, keyword spotting, audio classification etc becoming commonplace as we keep exploring applications

Benchmarking tinyML Systems: Challenges and Direction
Part 1: tinyML optimization example

- Design compact CNN model architecture for binary classification on low power MCU
  - Minimize # of parameters to fit on-chip memory (<256KB SRAM)
  - Reduce # of MAC to lower energy cost per inference (less is better)
  - Maintain model accuracy using design space exploration

Example Study: Optimize MobileNetv1-0.25x in TFLite Micro (INT8)
Visual Wake Words (VWW) dataset with different accuracy tradeoffs

<table>
<thead>
<tr>
<th>Name</th>
<th>Accuracy (%)</th>
<th>Cycle count</th>
<th>model size (Bytes)</th>
<th>Complexity (GMAC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Baseline) Org_0.25_FP32</td>
<td>84.69</td>
<td>-</td>
<td>1,192, 832</td>
<td>0.0079</td>
</tr>
<tr>
<td>(Baseline) Org_0.25_int8</td>
<td>83.71</td>
<td>18,997,450</td>
<td>298,208</td>
<td>0.0079</td>
</tr>
<tr>
<td>Deeplite_0.25_int8 opt1</td>
<td>83.71</td>
<td>17,606,718</td>
<td>176,208</td>
<td>0.0059</td>
</tr>
<tr>
<td>Deeplite_0.25_int8 opt2</td>
<td>81.73</td>
<td>16,621,617</td>
<td>130,624</td>
<td>0.0042</td>
</tr>
<tr>
<td>Deeplite_0.25_int8 opt3</td>
<td>79.91</td>
<td>16,476,406</td>
<td>123,304</td>
<td>0.0039</td>
</tr>
<tr>
<td>Initial-TF-Lite model (int8)</td>
<td>79.33</td>
<td>19,124,534</td>
<td>300,568</td>
<td>-</td>
</tr>
</tbody>
</table>
Part 2: Domain-Specific Architectures

What’s Left?

Since

- Transistors not getting much better
- Power budget not getting much higher
- Already switched from 1 inefficient processor/chip to N efficient processors/chip

Only path left is *Domain Specific Architectures*

- Just do a few tasks, but extremely well

*Domain-Specific Architectures for Deep Neural Networks, David Patterson, April 2019*

*Hyeenahs and Cheetahs: AI is awakening the chip industry’s animal spirits* – Economist 2018
We are witnessing the emergence of MANY candidates to run a variety of DNNs (CNN, RNN, LSTM, MLP) at a variety of form factors (Server, Desktop, Mobile, Edge/IoT)

Complimentary technologies (CNN accelerator, Arm Cortex-M4 and RISC-V CPU ie the MAX78000)

By 2025, 50% of new products with embedded MCUs will ship with tinyML for advanced AI (<1% in 2020)

Landscape keeps changing with M&A and new entrants

James Wang, ARK Invest Oct 2019
Part 2: Domain-Specific Architectures & DNN Model Optimization

- Model optimization for a DSA deep learning deployment (and re-training, continuous learning, if any) remains highly manual with limited support for different operations / frameworks

- Some optimizations / specializations like INT8 precision becoming standard (low precision networks to be implemented in the future)

- Many more opportunities with SW + HW co-design for model inference optimization (i.e., mixed low precision, layer fusing, sparsity, distillation, dw-conv and new operations/layer types etc.)

- Exciting possibilities with new IP for variety of use case complexities
Part 2: Domain-Specific Architectures & DNN Model Optimization

<table>
<thead>
<tr>
<th>Sample Applications</th>
<th>Model</th>
<th>Compression$^3$</th>
<th>Complexity Reduction (FLOPs)$^3$</th>
<th>Accuracy Drop (%)</th>
<th>Dataset</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Original Size</td>
<td>Optimized Size</td>
<td>Improvement</td>
<td></td>
</tr>
<tr>
<td>Image classification</td>
<td>VGG19</td>
<td>80MB</td>
<td>2.16MB</td>
<td>x37</td>
<td>x5</td>
</tr>
<tr>
<td></td>
<td>Mobilenet-v1.0</td>
<td>12.8MB</td>
<td>530KB</td>
<td>x22</td>
<td>x5</td>
</tr>
<tr>
<td></td>
<td>Custom use case$^1$</td>
<td>45MB</td>
<td>1.8MB</td>
<td>x27</td>
<td>x5.3</td>
</tr>
<tr>
<td></td>
<td>ResNet50</td>
<td>97MB</td>
<td>26MB</td>
<td>x3.72</td>
<td>x2.5</td>
</tr>
<tr>
<td>Object Detection</td>
<td>ResNet50-SSD300</td>
<td>54MB</td>
<td>18MB</td>
<td>x3</td>
<td>x3</td>
</tr>
<tr>
<td></td>
<td>ResNet34-SSD300</td>
<td>32MB</td>
<td>11MB</td>
<td>x3.75</td>
<td>x4.16</td>
</tr>
<tr>
<td></td>
<td>Yolov3</td>
<td>235MB</td>
<td>28MB</td>
<td>x8</td>
<td>x2.6</td>
</tr>
</tbody>
</table>

1 Based on ResNet18 architecture  
2 Used 8x GPU for faster optimization process  
3 Results obtained purely using Deeplite’s content-aware optimization (models in FP32). Further memory, speedup and energy savings available using platform-aware optimizations (INT8, mixed precision, binary weights etc.) and Deeplite inference engine
Part 3: Software stacks for Edge AI

- Nascent AI infrastructure exists for deploying cloud AI in production
- Even less infrastructure exists for edge AI and tinyML in production

Running Deep Learning Models is complicated and here is why

Cadence may be less for edge devices, but still necessary

I keep hearing a lot of misperceptions about ML in production. Here are a few.

1. Deploying ML models is hard
   Deploying a model for friends to play with is easy. Export trained model, create endpoints, build a simple app.

2. Deploying it reliably is hard. Serving 1000s of requests with ms latency is hard. Keeping it up all the time is hard.

3. You only have a few ML models in production
   Booking/eBay has 100s models in prod. An app has multiple features, each might have one or multiple models for different data slices.

4. If nothing happens, model performance remains the same
   ML models perform best right after training. In prod, ML systems degrade quickly because of concept drift.

Tip: train models on data generated 6 months ago & test on current data to see how much worse they get.

- You won’t need to update your models as much
  Mindboggling fact about DevOps: Etsy deploys 50 times/day. Netflix 1000s times/day. AWS every 11.7 seconds.

MLOps isn’t exempt. For online ML systems, you want to update them as fast as possible.

Deploying isn’t just about getting ML models to the end-users. It’s about building an infrastructure so teams can be quickly alerted when something goes wrong, figure out what went wrong, test in production, roll-out/rollback updates.
Part 3: Software stacks for Edge AI

- Multiple levels of optimization, interoperability and automation are essential

**Training**

1. **Data labeling**
2. **DNN architecture design**
3. **AI Frameworks**
   - mxnet
   - PyTorch
   - TensorFlow
   - Keras
4. **Hyper parameter tuning**
   - SIGOPT
   - Auptimizer

**Inference**

1. **Replace Manual/Traditional Optimization (Pruning, Neural Architecture Search, etc.)**
2. **DeepLITE**
3. **Computational Graph Optimization**
   - Full and/or 8-bit Precision
   - Mixed Low-precision networks
3. **DeepLITE**

**Design and Train models**

**Model Architecture Optimization**

**Platform Aware Optimization Compilers and Low-Level**

**Target Hardware**

- CPU (ARM, RISC-V) | GPU | FPGA | ASIC ETC.
- CPU (ARM, x86, RISC-V) | MCU's
Part 3: Software stacks for Edge AI

- Training happens once, Inference is forever
- Model inference optimization and ease-of-implementation is key to scaling production up and across different edge devices

Your Trained Model

80MB (FP32)

DNN optimization with Deeplite is as simple as:
- Write one line of code
- Press “run”
- Watch Deeplite do the work for you!

Your Optimized Model

2.2MB (FP32)

Demo the software now at deeplite.ai
Get a 30-day Trial
deeplite.ai/demo
Takeaways & Summary

- New caliber of models, datasets and frameworks as the trend shifts from bigger to smaller DNNs for edge AI & tinyML

- New chips for low power inferencing require optimization and SW+HW co-design to seize the opportunity for new products

- More to come from AI software ecosystem as models, methods and best practices keep evolving
Open Mic Q&A Session
Thank you!

Tweet us: @ArmSoftwareDev

Check out our Arm YouTube channel and our Arm Software Developers YouTube channel

Signup now for our next AI Virtual Tech Talk here

Don’t forget to fill out our survey to be in with a chance of winning an Arduino Nano 33 BLE board