A First Look at Deep Learning Apps on Smartphones

Mengwei Xu¹, Jiawei Liu¹, Yuanqiang Liu¹
Felix Xiaozhu Lin², Yunxin Liu³, Xuanzhe Liu¹

¹Peking University, ²Purdue University, ³Microsoft Research
Deep learning is popular

Google is using deep learning to change the world!
- summary of 1st Keynote

• Not only in CV, NLP, Robotics, but also Economics, Biology...

• In the Web Conference 2019: 16 (7%) papers are “deep”, 12 (5%) papers are connected to “neural” – only in title
DL inference on smartphones

**DL on Cloud**
- Centralized
- Powerful resources
- Well-studied
- Far from users

**DL on smartphones**
- Decentralized
- Low-end (challenging)
- Fewer prior efforts
- Close to users
DL inference on smartphones

**DL on Cloud**
- Centralized
- Powerful resources
- Well-studied
- Far from users

**DL on smartphones**
- Decentralized
- Low-end (challenging)
- Fewer prior efforts
- Close to users
DL inference on smartphones

• From academia
  • Cross-layer optimizations
  • Algorithm, System, Architecture

• From industrial
  • Android 8.1 officially introduces Neural Network APIs
  • There’re emerging DL frameworks for mobile: Facebook – Caffe2, Google – TF Lite, Apple – Core ML, Tencent – ncnn, Baidu – MDL
DL inference on smartphones

- From academia
  - Cross-layer optimizations
  - Algorithm, System, Architecture

- From industrial
  - Android 8.1 officially introduces Neural Network APIs

There're emerging DL frameworks for mobile: Facebook – Caffe2, Google – TF Lite, Apple – Core ML, Tencent – ncnn, Baidu – MDL

Early adopters have high influence or even decide the evolution of new technologies

- Diffusion of innovations theory, 1962
Research goal and challenges

• (The first) to understand how DL technique is utilized in real-world mobile applications
  • What are the characteristics and usage of DL apps?
  • How do the DL models look like?

1. How to identify DL apps?
2. How to extract DL models?

Android: the target platform
Identifying DL apps

Detecting DL functionality code? – Difficult 😢

- Heterogeneous: many ways to implement
- Binary code: written in C++ code and compiled in shared library
Deep learning frameworks: the popular choice

<table>
<thead>
<tr>
<th>Framework</th>
<th>Owner</th>
<th>Supported Mobile Platform</th>
<th>Mobile API</th>
<th>Is Open-source</th>
<th>Supported Model Format</th>
<th>Support Training</th>
</tr>
</thead>
<tbody>
<tr>
<td>TensorFlow [37]</td>
<td>Google</td>
<td>Android CPU, iOS CPU</td>
<td>Java, C++</td>
<td>✓</td>
<td>ProtoBuf (.pb, .pbtxt)</td>
<td>✓</td>
</tr>
<tr>
<td>TF Lite [38]</td>
<td>Google</td>
<td>Android CPU, iOS CPU</td>
<td>Java, C++</td>
<td>✓</td>
<td>FlatBuffers (.tflite)</td>
<td>x</td>
</tr>
<tr>
<td>Caffe [69]</td>
<td>Berkeley</td>
<td>Android CPU, iOS CPU</td>
<td>C++</td>
<td>✓</td>
<td>customized, json</td>
<td>✓</td>
</tr>
<tr>
<td>Caffe2 [9]</td>
<td>Facebook</td>
<td>Android CPU, iOS CPU</td>
<td>C++</td>
<td>✓</td>
<td>ProtoBuf (.pb)</td>
<td>✓</td>
</tr>
<tr>
<td>MxNet [49]</td>
<td>Apache Incubator</td>
<td>Android CPU, iOS CPU</td>
<td>C++</td>
<td>✓</td>
<td>customized, json</td>
<td>✓</td>
</tr>
<tr>
<td>ncnn [36]</td>
<td>Tencent</td>
<td>Android CPU, iOS CPU</td>
<td>C++</td>
<td>✓</td>
<td>customized (.params, .bin)</td>
<td>x</td>
</tr>
<tr>
<td>OpenCV [27]</td>
<td>OpenCV Team</td>
<td>Android CPU, iOS CPU</td>
<td>C++</td>
<td>✓</td>
<td>TesnorFlow, Caffe, etc</td>
<td>x</td>
</tr>
<tr>
<td>FeatherCNN [17]</td>
<td>Tencent</td>
<td>Android CPU, iOS CPU</td>
<td>C++</td>
<td>✓</td>
<td>customized (.feathermodel)</td>
<td>x</td>
</tr>
<tr>
<td>PaddlePaddle [25]</td>
<td>Baidu</td>
<td>Android CPU, iOS CPU &amp; GPU</td>
<td>C++</td>
<td>✓</td>
<td>customized (.tar)</td>
<td>✓</td>
</tr>
<tr>
<td>xNN [41]</td>
<td>Alibaba</td>
<td>Android CPU, iOS CPU</td>
<td>unknown</td>
<td>x</td>
<td>unknown</td>
<td>unknown</td>
</tr>
<tr>
<td>superID [35]</td>
<td>SuperID</td>
<td>Android CPU, iOS CPU</td>
<td>unknown</td>
<td>x</td>
<td>unknown</td>
<td>unknown</td>
</tr>
<tr>
<td>Parrots [31]</td>
<td>SenseTime</td>
<td>Android CPU, iOS CPU</td>
<td>unknown</td>
<td>x</td>
<td>unknown</td>
<td>unknown</td>
</tr>
<tr>
<td>MACE [24]</td>
<td>XiaoMi</td>
<td>Android CPU, iOS CPU</td>
<td>C++</td>
<td>✓</td>
<td>customized (.pb, .yml, .a)</td>
<td>x</td>
</tr>
<tr>
<td>SNPE [32]</td>
<td>Qualcomm</td>
<td>Qualcomm CPU, GPU, DSP</td>
<td>Java, C++</td>
<td>x</td>
<td>customized (.dle)</td>
<td>x</td>
</tr>
<tr>
<td>CNNDroid [74]</td>
<td>Oskouei et al.</td>
<td>Android CPU &amp; GPU</td>
<td>Java</td>
<td>✓</td>
<td>MessagePack (.model)</td>
<td>x</td>
</tr>
<tr>
<td>CoreML [12]</td>
<td>Apple</td>
<td>iOS CPU, GPU</td>
<td>Swift, OC</td>
<td>x</td>
<td>customized, ProtoBuf</td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td>Preferred Networks</td>
<td>/</td>
<td>/</td>
<td>✓</td>
<td>(.chainermodel)</td>
<td></td>
</tr>
<tr>
<td>Chainer [10]</td>
<td>Preferred Networks</td>
<td>/</td>
<td>/</td>
<td>✓</td>
<td>ProtoBuf (.proto)</td>
<td>✓</td>
</tr>
<tr>
<td>CNTK [23]</td>
<td>Microsoft</td>
<td>/</td>
<td>/</td>
<td>✓</td>
<td>customized (.dat)</td>
<td>✓</td>
</tr>
<tr>
<td>Torch [40]</td>
<td>Facebook</td>
<td>/</td>
<td>/</td>
<td>✓</td>
<td>customized, pickle</td>
<td>✓</td>
</tr>
<tr>
<td>PyTorch [30]</td>
<td>Facebook</td>
<td>/</td>
<td>/</td>
<td>✓</td>
<td>/</td>
<td></td>
</tr>
</tbody>
</table>
Identifying DL apps

**Detecting DL functionality code? — Difficult 😞**
- Heterogeneous: many ways to implement
- Binary code: written in C++ code and compiled in shared library

**Detecting DL framework usage? — Life is easier 😎**
- Most DL apps are developed by popular DL frameworks
- Those frameworks remain relatively stable across versions

$\texttt{readelf -p .rodata libMagicClean.so} \# \text{from adobe app}$

The .rodata section of \texttt{libMagicClean.so} in \texttt{adobe} app
- \texttt{apkttool + readelf}
- \texttt{TensorFlow detected}
Extracting DL models

**Scanning apk & check file format**
- Tflite models use protobuf format
- Ncnn models start with a magic number: 7767517

**Complementary-1: install apk and check local files**
- Some models are fetched from network

**Complementary-2: decrypt files**
- Some models are encrypted
Analysis workflow

- top 500 apps for each 33 categories: 16,500 in total
- Jun. 2018 and Sep. 2018

static analysis

https://github.com/xumengwei/MobileDL
How many DL apps?

• There’re ???/16,500 DL apps found in Sep. 2018.
How many DL apps?

• There’re **211/16,500** DL apps found in Sep. 2018.
  • only 1.3%
How many DL apps?

• There’re 211/16,500 DL apps found in Sep. 2018.
  • only 1.3%

• DL apps are downloaded by 13M (11.9%) Jun. 2018 – Sep. 2018
• DL apps receive 9.6M (10.5%) reviews

★ ★ ★ ★ ★ 5,616,407  ★ ★ ★ ★ ★ 4.2  ★ ★ ★ ★ ★ 5,616,407 total
How many DL apps?

• There’re 211/16,500 DL apps found in Sep. 2018.
  • only 1.3%

• DL apps are downloaded by 13M (11.9%) Jun. 2018 – Sep. 2018
• DL apps receive 9.6M (10.5%) reviews

<table>
<thead>
<tr>
<th></th>
<th>Downloads (median)</th>
<th>Reviews (median)</th>
</tr>
</thead>
<tbody>
<tr>
<td>DL apps</td>
<td>~5,000,000</td>
<td>~100,000</td>
</tr>
<tr>
<td>Non-DL apps</td>
<td>~40,000</td>
<td>~1,000</td>
</tr>
</tbody>
</table>

DL apps vs. non-DL apps
How many DL apps?

- There’re **211/16,500** DL apps found in Sep. 2018.
  - only 1.3%

- DL apps are downloaded by **13M (11.9%)**  
  - DL apps receive **9.6M (10.5%)** reviews  
  - Early adopters of DL are top apps
How many DL apps?

• There’re 211/16,500 DL apps found in Sep. 2018.
  • only 1.3%

• DL apps are downloaded by 13M (11.9%) ⌋ Jun. 2018 – Sep. 2018
• DL apps receive 9.6M (10.5%) reviews

• Early adopters of DL are top apps

• DL apps are increasing rapidly
  • ↑27% within 3 months

DL apps change over time
Which DL frameworks are most popular?

- **Mobile-oriented DL frameworks are gaining traction**
  - e.g., TFLite, ncnn, Caffe2, etc
Which DL frameworks are most popular?

- **Mobile-oriented DL frameworks are gaining traction**
  - e.g., TFLite, ncnn, Caffe2, etc

### Number of DL apps by framework

<table>
<thead>
<tr>
<th>Framework</th>
<th>Jun. 2018</th>
<th>Sep. 2018</th>
</tr>
</thead>
<tbody>
<tr>
<td>caffe2</td>
<td>10</td>
<td>12</td>
</tr>
<tr>
<td>TFLite</td>
<td>12</td>
<td>31</td>
</tr>
<tr>
<td>ncnn</td>
<td>21</td>
<td>28</td>
</tr>
<tr>
<td>Parrots</td>
<td>22</td>
<td>22</td>
</tr>
<tr>
<td>Caffe</td>
<td>25</td>
<td>26</td>
</tr>
<tr>
<td>TF</td>
<td>47</td>
<td>51</td>
</tr>
</tbody>
</table>

#### Most new adoption
- TFLite

### Library size (MB)

<table>
<thead>
<tr>
<th>Framework</th>
<th>Library size (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TF</td>
<td>15.3</td>
</tr>
<tr>
<td>Caffe</td>
<td>10.1</td>
</tr>
<tr>
<td>ncnn</td>
<td>2.5</td>
</tr>
<tr>
<td>TFLite</td>
<td>2.1</td>
</tr>
<tr>
<td>Caffe2</td>
<td>4.1</td>
</tr>
</tbody>
</table>
What’s the usage of DL in smartphone?

- The usage is diverse – of course

<table>
<thead>
<tr>
<th>Image</th>
<th>Audio</th>
<th>Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>photo beauty</td>
<td>Speech recognition</td>
<td>word prediction</td>
</tr>
<tr>
<td>face detection</td>
<td>Sound recognition</td>
<td>audio-correct</td>
</tr>
<tr>
<td>AR</td>
<td></td>
<td>Translation</td>
</tr>
<tr>
<td>face identification</td>
<td></td>
<td>text classification</td>
</tr>
<tr>
<td>Text recognition</td>
<td></td>
<td>Smart reply</td>
</tr>
<tr>
<td>action</td>
<td>Other recommendation</td>
<td>movement tracking</td>
</tr>
<tr>
<td></td>
<td>action detection</td>
<td></td>
</tr>
</tbody>
</table>
What’s the usage of DL in smartphone?

- The usage is diverse – of course
- What’s the most popular use case?

<table>
<thead>
<tr>
<th>Image</th>
<th>Audio</th>
<th>Text</th>
</tr>
</thead>
<tbody>
<tr>
<td>photo beauty</td>
<td>Speech recognition</td>
<td>word prediction</td>
</tr>
<tr>
<td>face detection</td>
<td>Sound recognition</td>
<td>audio-correct</td>
</tr>
<tr>
<td>AR</td>
<td></td>
<td>Translation</td>
</tr>
<tr>
<td>face identification</td>
<td></td>
<td>text classification</td>
</tr>
<tr>
<td>Text recognition</td>
<td>Other</td>
<td>Smart reply</td>
</tr>
<tr>
<td></td>
<td>recommendation movement tracking</td>
<td></td>
</tr>
<tr>
<td></td>
<td>action detection</td>
<td></td>
</tr>
</tbody>
</table>
What’s the usage of DL in smartphone?

- The usage is diverse – of course
- What’s the most popular use case?

<table>
<thead>
<tr>
<th>Image: 149</th>
<th>Audio: 24</th>
<th>Text: 26</th>
</tr>
</thead>
<tbody>
<tr>
<td>photo beauty</td>
<td>Speech recognition</td>
<td>word prediction</td>
</tr>
<tr>
<td>face detection</td>
<td>Sound recognition</td>
<td>audio-correct</td>
</tr>
<tr>
<td>AR face identification</td>
<td></td>
<td>Translation</td>
</tr>
<tr>
<td>Text recognition</td>
<td>recommendation</td>
<td>text classification</td>
</tr>
<tr>
<td></td>
<td>movement tracking</td>
<td></td>
</tr>
<tr>
<td></td>
<td>action detection</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Smart reply</td>
</tr>
</tbody>
</table>
What’s the usage of DL in smartphone?

• The usage is diverse – of course
• What’s the most popular use case?

<table>
<thead>
<tr>
<th>Image: 149</th>
</tr>
</thead>
<tbody>
<tr>
<td>photo beauty: 94</td>
</tr>
<tr>
<td>face detection</td>
</tr>
<tr>
<td>AR</td>
</tr>
<tr>
<td>face identification</td>
</tr>
<tr>
<td>Text recognition</td>
</tr>
</tbody>
</table>

• 44.5% DL apps use DL for photo beauty
What’s the usage of DL in smartphone?

• The usage is diverse – of course
• What’s the most popular use case?

Image: 149
- photo beauty: 94
- face detection
- AR
- face identification
- Text recognition

• 29% DL apps in photography category

BeautyCam
Meitu, Inc.  Photography
Everyone
This app is compatible with your device.
What’s the usage of DL in smartphone?

• The usage is diverse – of course
• What’s the most popular use case?

• What’s the role played by DL? Important or dispensable?
What’s the usage of DL in smartphone?

• The usage is diverse – of course
• What’s the most popular use case?

• What’s the role played by DL? Important or dispensable?
• ??% apps use DL as core feature

What makes a core feature?
• Hot: invoked frequently
• Essential: needed badly

Text recognition in different apps
• ✔ Scanner app (Adobe) ✔
• ✗ Payment app (Alipay) ✗
What’s the usage of DL in smartphone?

• The usage is diverse – of course
• What’s the most popular use case?

• What’s the role played by DL? Important or dispensable?
• 81% apps use DL as core feature: a strong motivation for future research to improve DL experience
Developers and bad practice

• **Most DL apps come from big companies**
  - Google *, Facebook, Adobe, Meitu, etc

• **Some DL apps use more than one DL frameworks (bad practice?)**
  - 24 apps; 5.4MB (13.6%) storage “overhead”
  - Can be avoided as DL frameworks can be used interchangeably
DL model resource footprint

- **DL models are very lightweight**
  - vs. state-of-the-art research models
  - The execution time of 80% models are less than 15ms on typical smartphone

<table>
<thead>
<tr>
<th></th>
<th>memory usage (MB)</th>
<th>Inference complexity (FLOPs)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>In the wild</strong></td>
<td>2.5</td>
<td>10M</td>
</tr>
<tr>
<td><strong>ResNet-50</strong></td>
<td>200</td>
<td>4G</td>
</tr>
<tr>
<td><strong>MobileNet</strong></td>
<td>54</td>
<td>500M</td>
</tr>
</tbody>
</table>
DL model optimizations

- **DL models are rarely optimized**
  - Only 6% models have been compressed

<table>
<thead>
<tr>
<th></th>
<th>1-bit Quan.</th>
<th>8-bit Quan.</th>
<th>16-bit Quan.</th>
<th>Sparsity</th>
</tr>
</thead>
<tbody>
<tr>
<td>TF</td>
<td>unsupported</td>
<td>4.78%</td>
<td>0.00%</td>
<td>0.00%</td>
</tr>
<tr>
<td>TFLite</td>
<td>unsupported</td>
<td>66.67%</td>
<td>unsupported</td>
<td>unsupported</td>
</tr>
<tr>
<td>Caffe</td>
<td>unsupported</td>
<td>0.00%</td>
<td>unsupported</td>
<td>unsupported</td>
</tr>
<tr>
<td>Caffe2</td>
<td>unsupported</td>
<td>0.00%</td>
<td>unsupported</td>
<td>unsupported</td>
</tr>
<tr>
<td>ncnn</td>
<td>unsupported</td>
<td>0.00%</td>
<td>unsupported</td>
<td>unsupported</td>
</tr>
<tr>
<td>Total</td>
<td>0.00%</td>
<td>6.32%</td>
<td>0.00%</td>
<td>0.00%</td>
</tr>
</tbody>
</table>
DL model protection

• For intellectual property and security

stop sign -> speed limit sign
DL model protection

• For intellectual property and security

• However, only small number of apps protect their models
  • 19.2% models are encrypted
  • 39.2% models are obfuscated

• Very few frameworks (2) provide model protection
  • strip text information, convert to C++ code
Implications

• To app developers
  - DL is affordable on smartphones: feel free to try it
  - Prefer mobile DL frameworks: they are often faster and more lightweight

• To DL framework developers
  - Make optimizations for smartphones (speed, memory, etc)
  - Build model protection as a functionality

• To DL researchers
  - Validate your brilliant ideas on lightweight models as well

Summary: DL is becoming increasingly popular and important on smartphones, but its power is still underexploited – *we are at dawn now*
Conclusion

• The first empirical study to understand how deep learning is utilized in real-world mobile applications.

• We find that deep learning is becoming increasingly popular and important on smartphones, but its power is still underexploited.

Thank you for attention!
Limitations and Future Work

- Our tool has false positive and false negative

- More platforms, longer time ranges