Convolutional Networks for Mobile Applications

Gao Huang
Department of Automation, Tsinghua University
Deep Learning

**AlphaGo Beats Go Human Champ: Godfather Of Deep Learning Tells Us Do Not Be Afraid Of AI**

By Aaron Murrill, Tech Times | March 21, 10:18 AM

Last week, Google's artificial intelligence program AlphaGo dominated its match with South Korean world Go champion Lee Sedol, winning with a 4-1 score.

The achievement stunned artificial intelligence experts, who previously thought that Google's computer program would need at least 10 more years before developing enough to be able to beat a human world champion.

What could be scary regarding the computer program is that Google DeepMind CEO Demis Hassabis said that AlphaGo could still improve its performance, as the match with Sedol was able to expose some of its weaknesses.

Computers have long been winning against skilled humans in games - Deep Blue defeated chess legend Gary Kasparov two decades ago - but Microsoft and Google's success is new.

---

**CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning**

Pranav Rajpurkar*, Jeremy Irvin*, Kaylee Zhu, Brandon Yang, Rishita Matha, Tony Duan, Daisy Ding, Anqi Begel, Curtis Langlotz, Katie Shpanskaya, Matthew P. Lungren, Andrew Y. Ng

We develop an algorithm that can detect pneumonia from chest X-rays at a level exceeding practicing radiologists.

CheXNet is one of the few scalable methods for diagnosing pneumonia, placing it a crucial role in clinical practice and epidemiological studies. Pneumonia is responsible for more than 1.2 million hospitalizations and 50,000 deaths per year in the U.S. alone.

READ OUR PAPER

---

**AlphaFold: Using AI for scientific discovery**

Today we're excited to share DeepMind’s first significant advance in demonstrating how artificial intelligence can help solve some of the hardest and most important scientific discoveries. With a strong interdisciplinary approach to our work, AlphaFold has brought together experts from the fields of computer science, physics and structural biology to develop a new AI model that predicts the 3D structures of proteins based on their genetic sequences.

Our colleague, AlphaFold, which we have been working on for the past two years, builds on years of prior research in using AI to generate stable 3D protein structures. The 3D models of proteins that AlphaFold generates are far more accurate than any that have come before, making significant progress on one of the core challenges in biology.
1. Overview of CNN backbones
2. Architecture design for mobile CNNs
3. Dynamic CNNs for mobile applications
1. Overview of CNN backbones
2. Architecture design for mobile CNNs
3. Dynamic CNNs for mobile applications
Convolutional Networks

- **LeNet**
- **AlexNet**
- **VGG**
- **Inception**
- **ResNet**
- **DenseNet**
- **HRNet**
Why architecture matters?

- Representation power
- Optimization Characteristics
- Generalization
- Efficiency
Advances in CNN Architecture Design

- AlexNet
- ZF-Net
- DSN
- NIN
- VGG
- GoogleNet
- …

2012-2015
Fast developing stage
Aim for high accuracy

2015-2017
Mature stage
Aim for simple design principles

- Highway Networks
- FractalNet
- ResNet
- DenseNet
- ResNeXt
- Dual Path Network
- …

2017-Present
Prosperous stage
Aim for better accuracy-speed tradeoff

- Light-weighted models
  - MobileNet (V1, V2, V3)
  - CondenseNet
  - ShuffleNet (V1, V2)
  - …

- Neural Arch. Search
  - NASNet
  - DARTS
  - …

- Dynamic models
  - MSDNet
  - Block-Drop
  - Glance and Focus
  - …

- Transformers?!
  - ViT and its variants
  - …
CNNs for Mobile Applications

Goal:

- Low compute
- Low latency
- Low memory cost
1. Overview of CNN backbones
2. Architecture design for mobile CNNs
3. Dynamic CNNs for mobile applications
Group Convolution

Main Idea:
✓ Split convolution into multiple groups

Standard Convolution

Group Convolution

\( O(C \times C) \)

\( O\left(\frac{C \times C}{G}\right) \)

CNNs using Group Convolution:
✓ AlexNet (Krizhevsky et al, NIPS’12)
✓ ResNeXt (Xie et al, CVPR’17)
✓ CondenseNet (Huang et al, CVPR’18)
✓ ShuffleNet (Zhang et al, CVPR’18)
✓ …
Depth-wise Separable Convolution (DSC)

Main Idea:
✓ Split convolution into multiple groups, each group has one channel

Networks using DSC:
✓ Xception (Chollet, CVPR’17)
✓ MobileNet (Howard et al, CVPR’18)
✓ MobileNet V2 (Sandler et al, 2018)
✓ ShuffleNet V2 (Ma et al, CVPR’19)
✓ NasNet (Zoph, CVPR’18)
✓ …
MobileNets

**MobileNet v1** [Howard et al, CVPR’18]
✓ Depth-wise separable convolution

**MobileNet v2** [Sandler et al, CVPR’19]
✓ Inverted Residuals and Linear Bottlenecks

**MobileNet v3** [Howard et al, CVPR’19]
✓ Introducing neural architecture search

Inverted Residuals in MobileNet v2

Comparison of MobileNet v3 and other models
ShuffleNets

**ShuffleNet v1** [Zhang et al, CVPR’18]
- ✓ Consecutive group convolution with channel shuffling

**ShuffleNet v2** [Ma et al, ECCV’18]
- ✓ Feature reuse with dense connection
- ✓ Special design for hardware efficiency
CondenseNets

CondenseNets v1 [Huang et al, CVPR’18]
✓ Sparsified dense connections
✓ Learned group convolution

CondenseNets v2 [Yang et al, CVPR’21]
✓ Feature reactivation
1. Overview of CNN backbones
2. Architecture design for mobile CNNs
3. Dynamic CNNs for mobile applications
Why do we need *dynamic* neural networks?
Accuracy-Time Tradeoff

Classification Results on ImageNet

- ResNets (He et al., 2015)
- DenseNets (Huang et al., 2017)
- GoogLeNet (Szegedy et al., 2015)
- AlexNet (Krizhevsky et al., 2012)
Bigger is better

Bigger models are needed for those noncanonical images.

*Photo Courtesy of Pixel Addict (CC BY-ND 2.0)
Bigger is better

*Photo Courtesy of Willian Doyle (CC BY-ND 2.0)
Why do we use the **same** expensive model for all images?
The model architecture (depth, width, etc) should be 
conditioned on the input!
1. Overview of CNN backbones
2. Architecture design for mobile CNNs
3. Dynamic CNNs for mobile applications
   A. Sample-wise Dynamic Networks
   B. Spatial-wise Dynamic Networks
   C. Temporal-wise Dynamic Networks
Sample-wise Dynamic Neural Networks

- Dynamic Networks
  - Sample-wise Dynamic Networks
    - Spatial-wise Dynamic Networks
    - Temporal-wise Dynamic Networks
  - Dynamic Architecture
    - Dynamic Depth
    - Dynamic Width
    - Dynamic Routing
  - Dynamic Parameter
    - Parameter Adjustment
    - Parameter Prediction
  - Dynamic Features
    - Dynamic Activation
  - Early Exiting
  - Layer Skipping
  - Skip Neurons
  - Skip Channels
  - Skip Branches
  - Attention on Weights
  - Kernel Shape Adaptation
  - Channel-wise
  - Spatial-wise
  - Dynamic Activation
Sample-wise Dynamic Neural Networks

Dynamic Architecture

- Dynamic Depth
  - Early Exiting
  - Layer Skipping
- Dynamic Width
  - Skip Neurons
  - Skip Channels
- Dynamic Routing
  - Skip Branches
Sample-wise Dynamic Neural Networks

Dynamic Architecture

- Dynamic Depth
  - Early Exiting
  - Layer Skipping
- Dynamic Width
  - Skip Neurons
  - Skip Channels
- Dynamic Routing
  - Skip Branches

Tsinghua University
Dynamic Depth: Early Exiting
Dynamic Depth: Early Exiting

Horse Down-sampling

Down-sampling

Linear

Horse
Early Exiting: Two Implementations

(a) Cascading of models.

(b) Network with intermediate classifiers.

Dynamic Depth: Early Exiting

A challenge: Intermediate classifiers may interfere with each other

Classifiers only work well on coarse-scale feature maps

Nearly all computation has been done before getting a coarse feature
Dynamic Depth: Early Exiting

Solution: Multi-scale Architecture
Dynamic Depth: Early Exiting

Solution: Multi-scale Architecture

- Fine-level features
- Mid-level features
- Coarse-level features

Dynamic Depth: Early Exiting

Solution: Multi-scale Architecture

Classifiers only operate on high level features!

Multi-scale densenet

Test Input

Classifier 1  Classifier 2  Classifier 3  Classifier 4  ...

...
Multi-scale denseNet
Multi-Scale DenseNet

Results

2x-5x speedup over DenseNet
Visualization

Test Input

Classifier 1

Classifier 2

Classifier 3

Classifier 4
Visualization

Class: red wine

Class: volcano

"easy"
(exit at first classifier)

"hard"
(exit at last classifier)
Dynamic Depth: Layer Skipping

A regular residual block
Dynamic Depth: Layer Skipping

Gating Module

Conv

+
Dynamic Depth: Layer Skipping

Gating Module

Conv

0
Layer Skipping Based on Policy Networks

Sample-wise Dynamic Neural Networks

Dynamic Architecture

- Dynamic Depth
  - Early Exiting
  - Layer Skipping

- Dynamic Width
  - Skip Neurons
  - Skip Channels

- Dynamic Routing
  - Skip Branches
Sample-wise Dynamic Neural Networks

Dynamic Architecture

- Dynamic Depth
  - Early Exiting
  - Layer Skipping

- Dynamic Width
  - Skip Neurons
  - Skip Channels
  - Skip Branches

Dynamic Routing
Dynamic Width
Skip Channels

Input

Output

Diagram showing Skip Channels with connections from input to output.
Skip Channels

Input

Output
Skip Channels based on Gating Function

Multi-stage Structure

Horse: 0.2

Multi-stage Structure

Horse: 0.6

Multi-stage Structure

Horse: 0.8

Skip Branches

Mixture of Experts (MoE)

Sample-wise Dynamic Neural Networks

Dynamic Architecture
- Dynamic Depth
  - Early Exiting
  - Layer Skipping
- Dynamic Width
  - Skip Neurons
  - Skip Channels
  - Skip Branches
- Dynamic Routing

Dynamically Activate network modules in classic architectures
Sample-wise Dynamic Neural Networks

**Dynamic Architecture**
- **Dynamic Depth**
- **Dynamic Width**
- **Dynamic Routing**

- Early Exiting
- Layer Skipping
- Skip Neurons
- Skip Channels
- Skip Branches
- Attention on Routing in a SuperNet with various inference paths
Dynamic Routing in SuperNets

Tree structure

Root (Input) -> Transformation -> Routing node -> Leaf node

Multi-scale structure

Depth

Scale

Sample-wise Dynamic Neural Networks

Dynamic Networks
  - Sample-wise Dynamic Networks
  - Spatial-wise Dynamic Networks
  - Temporal-wise Dynamic Networks

Dynamic Architecture
  - Dynamic Depth
  - Early Exiting
  - Layer Skipping

Dynamic Width
  - Skip Neurons
  - Skip Channels
  - Skip Branches

Dynamic Routing
  - Attention on Weights
  - Kernel Shape Adaptation

Dynamic Parameter
  - Parameter Adjustment
  - Parameter Prediction

Dynamic Features
  - Channel-wise
  - Spatial-wise

Dynamic Activation
Sample-wise Dynamic Neural Networks

- Sample-wise Dynamic Networks
  - Dynamic Architecture
    - Dynamic Parameter
    - Dynamic Features
      - Parameter Adjustment
      - Parameter Prediction
  - Dynamic Features
    - Attention on Weights
    - Kernel Shape Adaptation
    - Channel-wise
    - Spatial-wise
    - Dynamic Activation
Dynamic Parameter: Weight Adjustment

Regular convolution

Input Feature → Original Parameter → Output Feature
Dynamic Parameter: Weight Adjustment

Input Feature → Parameter Adjustment → Dynamic Parameter → Output Feature
Dynamic Parameter: Weight Ensemble

\[
\text{Input Feature} \rightarrow \alpha_1, \alpha_2, \alpha_3 \rightarrow \sum \rightarrow \text{Dynamic Parameter} \rightarrow \ast \rightarrow \text{Output Feature}
\]
Dynamic Parameter: Weight Prediction

Input Feature → Parameter Prediction → Dynamic Parameter → Output Feature
Kernel Shape Adaptation

Dynamic Parameter: Weight Adjustment

\[
\left( \sum_{n} \alpha_n W_n \right) \ast x = \sum_{n} \alpha_n (W_n \ast x)
\]

Channel-wise Attention

\[(x \ast W) \otimes \alpha = x \ast (W \otimes \alpha)\]

Dynamic Features  Dynamic Weights

Outline

1. Overview of CNN backbones
2. Architecture design for mobile CNNs
3. Dynamic CNNs for mobile applications
   A. Sample-wise Dynamic Networks
   B. Spatial-wise Dynamic Networks
   C. Temporal-wise Dynamic Networks
From *Sample Adaptive* to *Spatial Adaptive*
- **Sampling module**: generate a sampling indicator mask
- **Sparse Convolution**: compute features at sampled points
- **Interpolation module**: reconstruct entire feature map

---

Region-level Dynamic Network

Patch Selection \rightarrow (x, y, h, w) \rightarrow \text{Crop} \rightarrow \text{Network}
Human visual system processes information progressively.

Glance and Focus Network (GFNet)

**Glance**

Input: Image $\mathbf{x}$

- **Resize**
  - GFNet $p_1$
  - $\max_j p_{1j} > \eta_1$?
  - Yes: Output: $p_1$
  - No: Crop
- **Crop**
  - GFNet $p_2$
  - $\max_j p_{2j} > \eta_2$?
  - Yes: Output: $p_2$
  - No: Crop
- ... (repeated)
- **Crop**
  - GFNet $p_T$
  - ... (repeated)
  - Output: $p_T$

**Focus**
Network Architecture

Input: Image $X$

ConvNets

Recurrent Networks

Feature Maps

Feature Vectors

Cropping & Resizing
Training - Stage I

Minimize Average Cross-entropy Loss

\[ \frac{1}{T} \sum_{t=1}^{T} L_{CE}(p_t, y) \]

Input: Image \( X \)

Resize

Patch Proposal Network

Random Location

Crop

\( h^\pi_1 \)

\( X_1 \)

Global Encoder

\( f_g \)

Global Pooling

\( e_1 \)

Classifier

\( f_c \)

Feature Maps

Feature Vectors

Cropping & Resizing

Random Location

Minimize Average Cross-entropy Loss

\[ \frac{1}{T} \sum_{t=1}^{T} L_{CE}(p_t, y) \]

\( p_1 \)

\( p_2 \)

\( p_3 \)

ConvNets

Recurrent Networks

Local Encoder

\( f_1 \)

\( e_2 \)

\( h^c_1 \)

\( X_2 \)

\( h_1^c \)

\( h_2^c \)

\( h_3^c \)

\( p_1 \)

\( p_2 \)

\( p_3 \)

Crop

\( h^\pi_2 \)

\( h^\pi_3 \)

\( X_3 \)

\( h_2^c \)

\( h_3^c \)
Training - Stage II

Minimize Average Cross-entropy Loss

\[ \frac{1}{T} \sum_{t=1}^{T} L_{CE}(p_t, y) \]

Input: Image \( X \)

ConvNets

Recurrent Networks

Feature Maps

Feature Vectors

Cropping & Resizing

Resize

Global Encoder

Global Pooling

Classifier

\( f_g \)

\( e_1 \)

\( h^c_1 \)

\( p_1 \)

\( f_c \)

\( e_2 \)

\( h^c_2 \)

\( p_2 \)

\( f_c \)

\( e_3 \)

\( h^c_3 \)

\( p_3 \)

\( x_1 \)

\( x_2 \)

\( x_3 \)

\( \pi \)

\( \pi \)

\( \pi \)
Minimize Average Cross-entropy Loss

\[
\max \mathbb{E} \left[ \sum_{t=2}^{T} \gamma^{t-2} r_t \right]
\]

Training - Stage II

Input: Image \( X \)

Patch Proposal Network

\[ \pi \]

Resize

Crop

Global Encoder

\[ f_g \]

\[ e_1 \]

Global Pooling

\[ e_1 \]

Local Encoder

\[ f_1 \]

\[ e_2 \]

Classifier

\[ f_c \]

\[ p_1 \]

\[ p_2 \]

\[ p_3 \]

Training - Stage II

Maximize the Discounted Rewards

\[ \max \pi \mathbb{E} \left[ \sum_{t=2}^{T} \gamma^{t-2} r_t \right] \]

ConvNets

Recurrent Networks

Feature Maps

Feature Vectors

Cropping & Resizing
Results (FLOPs)

(a) MobileNet-V3

(b) RegNet-Y
Results (FLOPs)

(c) EfficientNet
Results (FLOPs)

(d) ResNet
(e) DenseNet
Results (iPhone XS Max)

(a) MobileNet-V3

(b) ResNet
Results (Visualization)

- **698 - palace**: 1: 90.2%
- **186 - Norwich terrier**: 1: 65.6%
- **21 - kite**: 1: 5.4%, 2: 98.3%
- **683 - oboe**: 1: 9.3%, 2: 63.8%
- **701 - parachute**: 1: 99.9%
- **865 - toyshop**: 1: 99.8%
- **707 - pay-phone**: 1: 23.3%
- **269 - timber wolf**: 1: 9.4%, 2: 44.5%
## Results (Visualization)

<table>
<thead>
<tr>
<th>Image ID</th>
<th>Object</th>
<th>Top 1</th>
<th>Top 2</th>
<th>Top 3</th>
<th>Top 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>696</td>
<td>paintbrush</td>
<td>8.4%</td>
<td>15.5%</td>
<td>38.0%</td>
<td></td>
</tr>
<tr>
<td>143</td>
<td>oystercatcher</td>
<td>11.2%</td>
<td>14.0%</td>
<td>62.6%</td>
<td></td>
</tr>
<tr>
<td>514</td>
<td>cowboy boot</td>
<td>2.0%</td>
<td>14.6%</td>
<td>17.1%</td>
<td>55.4%</td>
</tr>
<tr>
<td>655</td>
<td>miniskirt</td>
<td>5.3%</td>
<td>13.3%</td>
<td>22.8%</td>
<td>29.8%</td>
</tr>
<tr>
<td>279</td>
<td>Arctic fox</td>
<td>5.1%</td>
<td>33.0%</td>
<td>70.3%</td>
<td></td>
</tr>
<tr>
<td>570</td>
<td>gasmask</td>
<td>5.3%</td>
<td>21.4%</td>
<td>48.1%</td>
<td></td>
</tr>
<tr>
<td>187</td>
<td>Yorkshire terrier</td>
<td>10.3%</td>
<td>31.8%</td>
<td>54.7%</td>
<td></td>
</tr>
<tr>
<td>752</td>
<td>racket</td>
<td>22.8%</td>
<td>30.3%</td>
<td>39.2%</td>
<td>53.2%</td>
</tr>
</tbody>
</table>
Low-resolution representations are sufficient to recognize “easy” samples.
Resolution Adaptation

- **Easy samples** (e.g. images containing large objects):  
  ![Easy samples diagram]

- **Hard samples** (e.g. images containing tiny objects):  
  ![Hard samples diagram]

---

Resolution Adaptive Network

(Prediction Confidence < $\varepsilon_1$)
(Prediction Confidence > $\varepsilon_2$)

Confusing
Exit

Threshold ($\varepsilon_i$) controlling exiting of a sample.

Predict label: Owl

Inference Stop

Resolution Adaptive Network

Results: Budgeted Batch Classification

Budgeted batch classification on CIFAR-10

Budgeted batch classification on CIFAR-100

Budgeted batch classification on ImageNet

~1.2%

~1.0%
Images with tiny objects can be hard samples.

Images with multiple objects can be hard samples.

Images with objects w/o representative characteristics can be hard samples.
1. Overview of CNN backbones
2. Architecture design for mobile CNNs
3. Dynamic CNNs for mobile applications
   A. Sample-wise Dynamic Networks
   B. Spatial-wise Dynamic Networks
   C. Temporal-wise Dynamic Networks
Spatially & Temporally Adaptive Inference for Videos

Dynamic Networks

- Instance-wise Dynamic Networks
- Spatial-wise Dynamic Networks
- Temporal-wise Dynamic Networks
A small portion of frames have sufficient task-relevant information!
Adaptive Focus for Efficient Video Recognition

(a) Input Video (label: diving)

(b) Temporal-based Methods (existing works)

(c) AdaFocus (ours)

(d) AdaFocus+
Offline Video Recognition on ActivityNet
RNN-based Approaches

(a) Skip update of hidden state.

(b) Partial update of hidden state.

(c) Hierarchical RNN architecture.

(d) Temporal dynamic jumping.

Advantages of Dynamic Neural Networks

- Efficiency
- Representation Power
- Adaptiveness
- Compatibility
- Generality
- Interpretability
Challenges in Dynamic Neural Networks

<table>
<thead>
<tr>
<th>Theories</th>
<th>Architecture Design</th>
<th>Applicability on more diverse tasks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gap between theoretical &amp; practical efficiency</td>
<td>Robustness</td>
<td>Interpretability</td>
</tr>
</tbody>
</table>
Hierarchy of dynamic networks

Thank you!