Bootstrap custom image classification using transfer learning

Wee Hyong Tok, Danielle Dean
Principal Data Scientist
Cloud AI, Microsoft

Credits: Ilia Karmanov, Anusua Trivedi, Vivek Gupta, Patrick Buehler
What are the common models?

**CNN**
Convolutional Neural Network

**RNN**
Recurrent Neural Network
How long does it take to train DNN models?

**Before...**
- **14 days**
- **10^{18}** single precision operations
- **ResNet-50 NVIDIA M40 GPU**

**April 2017**
- **1 hour**
- **ResNet-50 32 CPU**
- **256 Nvidia P100 GPUs**

**Nov**
- **15 minutes**
- **ResNet-50 1,024 P100 GPUs**
- **Facebook Preferred Network ChainerMN**
Overview

Use Cases

Image Classification using Transfer Learning

How to jumpstart
Computer Vision Patterns

**Image Classification**
- Is there a deer in the image?

**Object Detection**
- Where is the deer in the image?

**Image Segmentation**
- Where exactly is the deer? What pixels?

**Image Similarity**
- Which images are similar to the query image?
Use Cases

Aerial Use Classification  Lung Cancer Detection  **ESmart** – Connected Drone and Power line inspection  **Jabil** – Defect Inspection
What is common in all these use cases?
What is common in all these use cases?

Convolution Neural Network (CNN) + Transfer Learning
Example
Image Classification - Retail

Clothing texture dataset:
- Collected 1716 images from Bing, and annotated as one of 11 textures (striped, dotted, etc.)

Applying transfer learning to accurately classify clothing texture

https://github.com/Azure/MachineLearningSamples-ImageClassificationUsingCNTK
Example
Image Similarity - Retail

Clothing texture dataset:
- Collected 1716 images from Bing, and annotated as one of 11 textures (striped, dotted, etc.)

Applying transfer learning to accurately classify clothing texture

https://github.com/Azure/ImageSimilarityUsingCntk
Demo
Image Similarity - Retail
14,197,122 images
21841 synsets

Diverse images, Lots of labels!
How does CNN work?

- Deep neural networks with many hidden layers are more powerful than shallow networks.
- Deep networks with fully connected nodes are harder to train than shallow networks.
- Convolutional nets address and offers a more efficient way to train deep neural networks.
- Most deep neural networks are based on Convolution.
Convolutional Net - Basic Architecture
How does Convolution Work?

- The Convolution layer uses a small window to extract local features from the raw data:
- Each of the 5X5 input is connected to the hidden node in the convolution layer
- We slide the 5X5 window across the raw data to scan the whole image
How does Convolution Work?

- Each of the 24X24 hidden nodes in the Convolution layer shares the same weights and bias
- Each hidden layer in the convolution layer is a feature map that learns one feature of the image
- The shared 5X5 weights and bias are also called the kernel or filter
Why Convolution?

- The name convolution comes from the convolution operation

\[ a^1 = \sigma(b + w \ast a^0), \]

Where:
- \( a^1 \) represents the output activations from one feature map,
- \( a^0 \) are the inputs
- \( \ast \) is the convolution operation
How does Pooling Work?
A convolutional net can have multiple convolution and pooling layers feeding each other.

This allows the network to learn more abstract features of the raw data.
AlexNet, 8 layers (ILSVRC 2012)

VGG, 19 layers (ILSVRC 2014)

GoogleNet, 22 layers (ILSVRC 2014)

ILSVRC (ImageNet Large Scale Visual Recognition Challenge)
ResNet, 152 layers

Microsoft
Example – Visualizing the different layers

https://distill.pub/2017/feature-visualization/

Another fun site:
https://deepart.io/nips/submissions/random/
http://cs231n.stanford.edu/
Example – Visualizing the different layers

Olah, et al., "Feature Visualization", Distill, 2017
https://distill.pub/2017/feature-visualization/

Another fun site:
https://deepart.io/nips/submissions/random/
http://cs231n.stanford.edu/
Demo
Visualizing different layers of the CNN

https://github.com/ilkarman/Blog/tree/master/Visuals
What is Transfer Learning?
## Types of Transfer Learning

<table>
<thead>
<tr>
<th>Type</th>
<th>How to Initialize Featurization Layers</th>
<th>Output Layer Initialization</th>
<th>How is Transfer Learning used?</th>
<th>How to Train?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard DNN</td>
<td>Random</td>
<td>Random</td>
<td>None</td>
<td>Train featurization and output jointly</td>
</tr>
<tr>
<td>Headless DNN</td>
<td>Learn using another task</td>
<td>Separate ML algorithm</td>
<td>Use the features learned on a related task</td>
<td>Use the features to train a separate classifier</td>
</tr>
<tr>
<td>Fine Tune DNN</td>
<td>Learn using another task</td>
<td>Random</td>
<td>Use and fine tune features learned on a related task</td>
<td>Train featurization and output jointly with a small learning rate</td>
</tr>
<tr>
<td>Multi-Task DNN</td>
<td>Random</td>
<td>Random</td>
<td>Learned features need to solve many related tasks</td>
<td>Share a featurization network across both tasks. Train all networks jointly with a loss function (sum of individual task loss function)</td>
</tr>
</tbody>
</table>
Using a Pre-Trained CNN as a Featurizer

Outputs of penultimate layer of ImageNet Trained CNN provide excellent general purpose image features
Using a pre-trained DNN, an accurate model can be achieved with thousands (or less) of labeled examples instead of millions.
<table>
<thead>
<tr>
<th>Method</th>
<th>Input Image Size</th>
<th>Area Under Curve</th>
<th>Classification Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNN featurization</td>
<td>224x224 pixels</td>
<td>0.59</td>
<td>69.0%</td>
</tr>
<tr>
<td>Fine-tuning (full CNN)</td>
<td>224x224 pixels</td>
<td>0.76</td>
<td>77.4%</td>
</tr>
<tr>
<td>Fine-tuning (full CNN)</td>
<td>896x886 pixels</td>
<td>0.83</td>
<td>88.2%</td>
</tr>
</tbody>
</table>
How do you get started with transfer learning?
Pre-trained Models

CNTK Pre-trained Image Models

This page contains pre-trained image models either converted from other available models includes:

- AlexNet
- GoogLeNet
- ResNet
- VGG

Related Models:

http://bit.ly/2jf97NE
http://bit.ly/1KlVMf0

TensorFlow Official Models

The TensorFlow official models are a collection of example models that use TensorFlow's API.

Current models are compatible with TensorFlow 1.4. If you are on an earlier version:

Model Zoo

Check out the model zoo documentation for details.

To acquire a model:

1. Download the model's gist by ./scripts/download_model_from_gist.sh <gist_id> <archive> to load the model's metadata, architecture, solver configuration, and so on. <archive> is optional and defaults to caffe/models.
2. Download the model's weights by ./scripts/download_model_binary.py <model_dir> where <model_dir> is the gist directory from the first step.

or visit the model zoo documentation for complete instructions.

Table of Contents

- Berkeley-trained models
- Network in Network model
- Models from the BMVC 2014 paper “Return of the Devil in the Details: Driving Deep into Convolutional Nets”
- Models used by the VGG team in ILSVRC-2014
- Places-365 model from MIT.
Visual Intelligence Made Easy

Easily customize your own state-of-the-art computer vision models that fit perfectly with your unique use case. Just bring a few examples of labeled images and let Custom Vision do the hard work.

Upload Images
Bring your own labeled images, or use Custom Vision to quickly add tags to any unlabeled images.

Train
Use your labeled images to teach Custom Vision the concepts you care about.

Evaluate
Use simple REST API calls to quickly tag images with your new custom computer vision model.
Demo
Customize a CNN for your data
Summary

Use Cases

Image Classification using Transfer Learning

How to jumpstart