MATLAB을 활용한 컴퓨터 비전
(3차원 비전 및 기계학습)

Application Engineer
Caleb Kim
Contents

- Stereo and 3D Vision
- Machine Learning
- Deep Learning
Camera Calibration App

- Simplified workflow for estimating camera intrinsic and extrinsic parameters
- Removes the effects of lens distortion from an image
- Automatically detects checkerboard patterns
Evaluate Calibration Accuracy

Determine the accuracy of estimated camera parameters

- Plot re-projection errors as a bar graph or as a scatter plot
- Visualize the 3-D locations of the calibration patterns relative to the camera, or the cameras relative to the pattern.

» showReprojectionErrors(cameraParameters)
» showExtrinsics(cameraParameters)
Remove Lens Distortion From an Image

Removes radial and tangential distortion.

- Radial distortion ("barrel" or "pincushion") is caused by the curvature of the lens
- Tangential distortion is caused by misalignment between the lens and the sensor

\[ J = \text{undistortImage}(I, \text{cameraParameters}) \]
Measuring Planar Objects With a Calibrated Camera

Featured example: measure the diameter of a penny in millimeters.

- Undistort the image
- Detect the penny
- Project points from the image into the world
- Measure the diameter in millimeters

» MeasuringPlanarObjectsExample
Structure From Motion

Estimating 3-D structure of a scene from a set of 2D-images

- Match a set of points between the two images
- Estimate the fundamental Matrix
- Compute the motion of camera
- 3D reconstruction
- Detect an Object

» StructureFromMotionExample.m
Recovering Scene Depth with Stereo Cameras
Epipolar Geometry
Fundamental Matrix

\[ X_L^T F X_R = 0 \]
Stereo Camera Calibration

- Simplifies and automates calibration process
Stereo Vision Workflow

- Calibration - App (14b)
- Rectification - Codegen (15a)
- Disparity Estimation - Block matching, semi-global matching (14b) - Codegen (14b)
- 3-D Reconstruction - Codegen (15a)
Point Cloud Registration

- **Rigid registration**
  - `pcregigid`: Fundamental operation across point cloud applications
  - ‘Iterative Closest Point’ Algorithm
  - Comparable to state-of-the-art c++ package on academic benchmarks
  - 3-D Point Cloud Registration and Stitching Featured Example
Point Cloud Processing

- 3-D point cloud processing
  - File I/O, Viewers
  - Registration, denoising, downsampling, geometric transformation
Point Cloud Application – Robot Vision

- Robot Navigation

- Robot Perception
Point Cloud Application – Advanced Driver Assistance Systems (ADAS)

- Collision Detection

- Visual SLAM (Simultaneous localization and mapping) / Visual Odometry
Contents

- Stereo and 3D Vision
- Machine Learning
- Deep Learning
Today’s Objectives

Use examples to solve real-world problems to:

– See how MATLAB simplifies the machine learning workflow
– Quickly go from idea to prototype
– What’s new for machine learning, deep learning, image processing and computer vision
Agenda

- Introduction
  - Applications
  - Workflow
  - Common Challenges

- Demonstrations
  - Object recognition using live video
  - Deep learning for recognition
  - Training object detectors
  - Grouping or clustering images by visual similarity

- Conclusion
What Problems Can You Solve?

Object Detection

Object Recognition or Classification

Object Detection
Machine Learning

Machine learning uses **data** and produces a **program** to perform a **task**

**Task:** Image Category Recognition

- If `brightness > 0.5`
  - then ‘hat’
- If `edge_density < 4` and `major_axis > 5`
  - then “boat”
  - ...

- **model = `<Machine Learning Algorithm>(data, label)`**
Machine Learning Workflow Using Images

Training Data → Feature Extraction → Learning or Modelling → Training

Input Image → Feature Extraction → Classification

Classifier / Model → 'hat'
Viola Jones – Cascade Object Detectors

- Algorithm to detect people's faces, noses, eyes, mouth, or upper body.
- Ability to train custom classifiers using the Training Image Labeler
Cascade of Classifiers in CascadeObjectDetector

- Each stage of cascade is Gentle Adaboost, an ensemble of weak learners
- Each stage rejects negative samples using a weighted vote of these weak learners
- The samples not rejected are passed to the next stage
- Positive detection means the sample passed all stages of the cascade
Challenges: Machine Learning Workflow Using Images

- **Training Data**
- **Feature Extraction**
- **Learning or Modelling**

**Challenge 1**

**Challenge 2**

**Challenge 3**

Input Image → Feature Extraction → Classification

Classifier / Model → ‘hat’
Common Challenges for Machine Learning with Images

- **Challenge 1:** Handling large sets of images
- **Challenge 2:** How to extract discriminative information from images
- **Challenge 3:** How to model tasks or data using machine learning
Goal: Recognize/Classify Objects in Live Video

‘hat’ vs ‘mug’ vs ‘boat’

Known as object classification or recognition
What is Feature Extraction?

- Representations often **invariant to** changes in scale, rotation, illumination
- More compact than storing pixel data
- Feature selection based on nature of problem

**Feature Extraction**

**Sparse**

**Dense**

Bag of Words

SURF

HOG

Image Pixels
Perform **image processing**, **analysis**, and algorithm development

Image Processing Toolbox™ provides a comprehensive set of reference-standard algorithms, functions, and apps for **image processing**, **analysis**, visualization, and algorithm development. You can perform **image analysis**, **image segmentation**, **image enhancement**, noise reduction, geometric transformations, and **image registration**. Many toolbox functions support multicore processors, GPUs, and C-code generation.

**Image Processing** Toolbox supports a diverse set of **image** types, including high dynamic range, gigapixel resolution, embedded ICC profile, and tomographic. Visualization functions and apps let you explore **images** and videos, examine a region of **pixels**, adjust color and contrast, create contours or histograms, and manipulate regions of interest (ROIs). The toolbox supports workflows for **processing**, displaying, and navigating large **images**.

**Bag of Words**

<table>
<thead>
<tr>
<th>Image Processing Toolbox</th>
<th>Class / Label</th>
</tr>
</thead>
</table>

**Training Data**

<table>
<thead>
<tr>
<th>Vocabulary / Bag of Words</th>
<th>Class / Label</th>
</tr>
</thead>
</table>
Bag of “Visual Words” (features)

Class / Label

Training Data

Vocabulary / Bag of Words

‘mugs’
Image Classification with Bag of Words

Training Data

Bag = Visual Vocabulary

Input Image

Classifier

‘hat’
Many Options for Features and Machine Learning

Feature Extraction
- BRISK, FREAK, SURF
- Histogram of Oriented Gradients (HoG)
- Using box filters (integral images)
- Bag of visual words
- Color-based features
- Frequency-domain features

Machine Learning
- SVM
- Decision trees
- AdaBoost
- Bagged trees
- k-NN
- Discriminant analysis
- Bayes classifiers

Bottom Line: Many permutations and combinations to fit the needs of your problem
Challenges: Machine Learning Workflow Using Images

Training Data → Feature Extraction → Learning or Modelling

Challenge 1

Challenge 2

Challenge 3

Input Image → Feature Extraction → Classification

Classifier / Model → ‘hat’
Common Challenges for Machine Learning with Images

- **Challenge 1:** Handling large sets of images

- **Challenge 2:** How to extract discriminative information from images

- **Challenge 3:** How to model problem using machine learning techniques

- Easy to handle large sets of images
  - `imageSet`

- **Bag of words for feature extraction**
  - More available in Computer Vision System Toolbox
Examples of Object Recognition/Classification

- Automatic scene categorization
- Biometrics
  - Face recognition
  - IRIS recognition
  - Fingerprint recognition
- Part recognition for factory automation
- Autonomous robotics
Contents

- Stereo and 3D Vision
- Machine Learning
- Deep Learning
Deep Learning is Ubiquitous

Computer Vision
- Pedestrian and traffic sign detection
- Landmark identification
- Scene recognition
- Medical diagnosis and drug discovery

Text and Signal Processing
- Speech Recognition
- Speech & Text Translation

Robotics & Controls

and many more…
What is Deep Learning?

Deep learning performs **end-end learning** by learning **features, representations and tasks** directly from **images, text and sound**.
Demo: Live Object Recognition with Webcam
Why is Deep Learning so Popular?

- **Results:** Achieved substantially better results on ImageNet large scale recognition challenge
  - 95% + accuracy on ImageNet 1000 class challenge

- **Computing Power:** GPU’s and advances to processor technologies have enabled us to train networks on massive sets of data.

- **Data:** Availability of storage and access to large sets of labeled data
  - E.g. ImageNet, PASCAL VoC, Kaggle

<table>
<thead>
<tr>
<th>Year</th>
<th>Error Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pre-2012 (traditional computer vision and machine learning techniques)</td>
<td>&gt; 25%</td>
</tr>
<tr>
<td>2012 (Deep Learning)</td>
<td>~ 15%</td>
</tr>
<tr>
<td>2015 (Deep Learning)</td>
<td>&lt;5 %</td>
</tr>
</tbody>
</table>
Two Approaches for Deep Learning

1. Train a Deep Neural Network from Scratch

   **Convolutional Neural Network (CNN)**

   - **Learned features**
   - **[95% 3% 2%]**

   - **Car ✓**
   - **Truck ✗**
   - **Bicycle ✗**

   **Lots of data**

2. Fine-tune a pre-trained model (transfer learning)

   **Pre-trained CNN**

   **Fine-tune network weights**

   **Car ✓**
   **Truck ✗**

   **Medium amounts of data**
Two Deep Learning Approaches

**Approach 1: Train a Deep Neural Network from Scratch**

- **Training data**: 1000s to millions of labeled images
- **Computation**: Compute intensive (requires GPU)
- **Training Time**: Days to Weeks for real problems
- **Model accuracy**: High (can over fit to small datasets)

**Convolutional Neural Network (CNN)**

- **Learned features**
  - Car: 95%
  - Truck: 3%
  - Bicycle: 2%

**Recommended only when:**
- **Training data**: 1000s to millions of labeled images
- **Computation**: Compute intensive (requires GPU)
- **Training Time**: Days to Weeks for real problems
- **Model accuracy**: High (can over fit to small datasets)
Two Deep Learning Approaches

Approach 2: Fine-tune a pre-trained model (transfer learning)

CNN trained on massive sets of data
- Learned robust representations of images from larger data set
- Can be fine-tuned for use with new data or task with small – medium size datasets

Recommended when:

<table>
<thead>
<tr>
<th>Training data</th>
<th>100s to 1000s of labeled images (small)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computation</td>
<td>Moderate computation (GPU optional)</td>
</tr>
<tr>
<td>Training Time</td>
<td>Seconds to minutes</td>
</tr>
<tr>
<td>Model accuracy</td>
<td>Good, depends on the pre-trained CNN model</td>
</tr>
</tbody>
</table>
Convolutional Neural Networks

- Train “deep” neural networks on structured data (e.g. images, signals, text)
- Implements Feature Learning: Eliminates need for “hand crafted” features
- Trained using GPUs for performance
Convolutional Neural Networks

Every feature map output is the result of applying a filter to the image.
The new feature map is the next input.
Convolution Layer

- Core building block of a CNN
- Convolve the filters sliding them across the input, computing the dot product

Intuition: learn filters that activate when they “see” some specific feature
Convolution Layer – Choosing Hyperparameters

- Number of filters, \( K \)
- Filter size, \( F \)
- Stride, \( S \)
- Zero padding, \( P \)

\[
W_2 = \frac{(W_1 - F + 2P)}{S} + 1 \\
H_2 = \frac{(H_1 - F + 2P)}{S} + 1 \\
D_2 = K
\]
Rectified Linear Unit (ReLU) Layer

- Frequently used in combination with Convolution layers
- Do not add complexity to the network
- Most popular choice: $f(x) = \max(0, x)$, activation is thresholded at 0
Pooling Layer

- Perform a **downsampling** operation across the spatial dimensions
- Goal: progressively decrease the size of the layers
- Max pooling and average pooling methods
- Popular choice: Max pooling with 2x2 filters, Stride = 2
## Challenges using Deep Learning for Computer Vision

<table>
<thead>
<tr>
<th>Steps</th>
<th>Challenge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Importing Data</td>
<td>Managing large sets of labeled images</td>
</tr>
<tr>
<td>Preprocessing</td>
<td>Resizing, Data augmentation</td>
</tr>
<tr>
<td>Choosing an architecture</td>
<td>Background in neural networks (deep learning)</td>
</tr>
<tr>
<td>Training and Classification</td>
<td>Computation intensive task (requires GPU)</td>
</tr>
<tr>
<td>Iterative design</td>
<td></td>
</tr>
</tbody>
</table>
Demo: Classifying the CIFAR-10 dataset

Objective: Train a Convolutional Neural Network to classify the CIFAR-10 dataset

Data:

<table>
<thead>
<tr>
<th>Input Data</th>
<th>Thousands of images of 10 different Classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Response</td>
<td>AIRPLANE, AUTOMOBILE, BIRD, CAT, DEER, DOG, FROG, HORSE, SHIP, TRUCK</td>
</tr>
</tbody>
</table>

Approach:

- Import the data
- Define an architecture
- Train and test the CNN

Demo: Classifying the CIFAR-10 dataset

%% Download the CIFAR-10 dataset
if ~exist('cifar-10-batches-mat','dir')
cifar10Dataset = 'cifar-10-matlab';
disp('Downloading 174MB CIFAR-10 dataset...');
webread(fullfile(cifar10Dataset,' tar.gz'));

>>
## Addressing Challenges in Deep Learning for Computer Vision

<table>
<thead>
<tr>
<th>Challenge</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Managing large sets of labeled images</td>
<td><code>imageSet</code> or <code>imageDataStore</code> to handle large sets of images</td>
</tr>
<tr>
<td>Resizing, Data augmentation</td>
<td><code>imresize</code>, <code>imcrop</code>, <code>imadjust</code>, <code>imageInputLayer</code>, etc.</td>
</tr>
<tr>
<td>Background in neural networks (deep learning)</td>
<td>Intuitive interfaces, well-documented architectures and examples</td>
</tr>
<tr>
<td>Computation intensive task (requires GPU)</td>
<td>Training supported on GPUs, No GPU expertise is required</td>
</tr>
<tr>
<td></td>
<td>Automate. Offload computations to a cluster and test multiple architectures</td>
</tr>
</tbody>
</table>
Demo

*Fine-tune a pre-trained model (transfer learning)*

Pre-trained CNN
(AlexNet – 1000 Classes)

New Data

Car

SUV

New Task – 2 Class Classification
Demo

Fine-tune a pre-trained model (transfer learning)
## Addressing Challenges in Deep Learning for Computer Vision

<table>
<thead>
<tr>
<th>Challenge</th>
<th>Solution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Managing large sets of labeled images</td>
<td><code>imageSet</code> or <code>imageDataStore</code> to handle large sets of images</td>
</tr>
<tr>
<td>Resizing, Data augmentation</td>
<td><code>imresize</code>, <code>imcrop</code>, <code>imadjust</code>, <code>imageInputLayer</code>, etc.</td>
</tr>
<tr>
<td>Background in neural networks (deep learning)</td>
<td>Intuitive interfaces, well-documented architectures and examples</td>
</tr>
<tr>
<td>Computation intensive task (requires GPU)</td>
<td>Training supported on GPUs</td>
</tr>
<tr>
<td></td>
<td>No GPU expertise is required</td>
</tr>
<tr>
<td></td>
<td>Automate. Offload computations to a cluster and test multiple architectures</td>
</tr>
</tbody>
</table>
Key Takeaways

▪ Consider Deep Learning when:
  – Accuracy of traditional classifiers is not sufficient
    ▪ *ImageNet classification problem*
  – You have a pre-trained network that can be fine-tuned
  – Too many image categories (100s – 1000s or more)
    ▪ *Face recognition*

**MATLAB for Deep Learning and Computer Vision**

Email us:
## Challenges using Deep Learning for Computer Vision

<table>
<thead>
<tr>
<th>Steps</th>
<th>Challenge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Importing Data</td>
<td>Managing large sets of images</td>
</tr>
<tr>
<td>Training &amp; Classification</td>
<td>Compute intensive • Requires GPU</td>
</tr>
<tr>
<td>Learning Curve</td>
<td>Most engineers didn't study deep learning</td>
</tr>
</tbody>
</table>
Thank You!

Questions?