AI Summer Camp

From Neural Networks to Intelligent Machines

Course Overview

This one-week intensive camp introduces students to the exciting intersection of artificial intelligence and robotics. Students will learn how neural networks power computer vision systems and how these systems can be deployed on edge devices to control robots.

By the end of the week, students will have trained their own AI models, optimized them for real-world deployment, and seen their code come to life on actual robotic hardware.

Format: Hybrid (2 in-person days + 3 remote days)
Duration: 5 days, 6 hours per day (3hr lecture + 3hr lab)
Instructor: Dr. Bo Sheng, Professor of Computer Science, UMass Boston
Lab Instructor: Graduate student assistant

Prerequisites:

- Basic Python programming (variables, loops, functions, lists)
- Comfortable using a laptop and installing software
- No prior machine learning or robotics experience required

What to Bring:

- Laptop (Windows, Mac, or Linux)
- Curiosity and willingness to experiment

Learning Outcomes

By the end of this camp, students will be able to:

1. Explain how neural networks learn to recognize patterns in data
2. Train convolutional neural networks (CNNs) for image classification and object detection
3. Optimize AI models for deployment on resource-constrained edge devices
4. Understand how computer vision enables robotic perception and control
5. Complete a hands-on project applying AI to a real-world application

---

Schedule Overview

Day	Mode	Theme	Lecture	Lab
Monday	**In-Person**	Welcome & Hardware Setup	3 hr	3 hr
Tuesday	Remote	Neural Network Foundations	3 hr	3 hr
Wednesday	Remote	Computer Vision & Object Detection	3 hr	3 hr
Thursday	Remote	Edge Deployment & Project Prep	3 hr	3 hr
Friday	**In-Person**	Hardware Integration & Showcase	—	6 hr

Hardware Availability:

- **Monday & Friday (in-person):** Raspberry Pi, robotic arm, cameras available
- **Tuesday–Thursday (remote):** Students use laptops + remote GPU server only; laptop webcams for testing detection

---

Detailed Daily Schedule

Day 1 (Monday) — IN-PERSON

Welcome, Hardware Orientation & Setup

Lecture Session (3 hours) — Dr. Sheng

The Big Picture (1.5 hours)

- What is AI? What is robotics? How do they connect?
- Live demo: Watch a vision-guided robotic arm sort objects
- Overview of the week: from training a neural network to controlling a robot
- Introduction to the final project options

Hardware Orientation (1.5 hours)

- Tour of equipment: Raspberry Pi, robotic arms, cameras
- Robotic arm architecture: joints, gripper, camera mount
- The "observation pose" concept for eye-in-hand vision
- Live demonstration: manual arm control via Python

Lab Session (3 hours) — Graduate TA

Environment Setup (1.5 hours)

- Set up laptop: SSH client, VS Code with Remote-SSH extension
- Connect to remote GPU server, verify Jupyter access
- Run test scripts to confirm Python/PyTorch environment
- Brief intro to command line basics (if needed)

Hands-On Hardware & Data Collection (1.5 hours)

- Students take turns controlling the robotic arm
- Capture images of colored blocks using robot camera
- Create personal dataset (~30–50 images per student)
- Transfer images to GPU server for later training

Deliverables:

- Working connection to GPU server
- Personal image dataset uploaded and organized
- Familiarity with robotic arm operation

Day 2 (Tuesday) — REMOTE

Neural Network Foundations

Lecture Session (3 hours) — Dr. Sheng

Introduction to Machine Learning (1 hour)

- What is machine learning? Supervised vs. unsupervised learning
- Training data, labels, and the learning process
- Classification vs. regression problems

Neural Network Fundamentals (1.5 hours)

- Anatomy of a neural network: neurons, layers, weights, biases
- Activation functions: why non-linearity matters
- Forward propagation: how inputs become outputs
- Training basics: loss functions, gradient descent, backpropagation (intuition-level)

Practical Considerations (30 min)

- Overfitting and validation: why we split data

- Hyperparameters: learning rate, batch size, epochs
- When to stop training

Lab Session (3 hours) — Graduate TA

Guided Tutorial (1 hour)

- Walk through a complete PyTorch training pipeline
- Tensors, datasets, dataloaders, and the training loop
- Understanding the provided base code

Hands-On: Build a Digit Classifier (2 hours)

- Load and explore the MNIST dataset
- Define a feedforward neural network architecture
- Train the model, monitor loss and accuracy
- Experiment with different architectures (layers, neurons, activations)
- Goal: achieve >95% test accuracy
- *Base code provided:* Data loading, training loop template

Deliverables:

- Working MNIST classifier with >95% accuracy
- Experiment log documenting different architectures tried

Day 3 (Wednesday) — REMOTE

Computer Vision & Object Detection

Lecture Session (3 hours) — Dr. Sheng

How Computers See (1 hour)

- Images as data: pixels, channels, resolution
- Color spaces: RGB, grayscale
- Image preprocessing: resizing, normalization, augmentation

Convolutional Neural Networks (1.5 hours)

- Limitations of feedforward networks for images
- Convolution operation: filters, feature extraction

- Pooling layers: downsampling and translation invariance
- CNN architecture: stacking conv, pool, and fully connected layers
- Visualizing what CNNs learn: feature maps

From Classification to Detection (30 min)

- Classification vs. localization vs. detection
- Bounding boxes and confidence scores
- Overview of detection architectures: YOLO, SSD, MobileNet
- Transfer learning: leveraging pre-trained models

Lab Session (3 hours) — Graduate TA

Guided: Pre-trained Object Detection (45 min)

- Load and run YOLOv5-nano / MobileNet-SSD on sample images
- Understand model outputs: boxes, classes, confidence
- Visualize detections with bounding boxes

Dataset Preparation (45 min)

- Use provided labeling tool to annotate your block images from Monday
- Label classes: red block, blue block, green block, etc.
- Data augmentation: flips, rotations, brightness variations

Hands-On: Train Your Custom Detector (1.5 hours)

- Fine-tune pre-trained model on your labeled dataset
- Monitor training metrics: loss, mAP
- Evaluate on held-out test images
- Test detection using your **laptop webcam** (live inference preview)

Deliverables:

- Labeled dataset (50+ annotated images)
- Custom object detector trained on your data
- Live webcam detection demo working on laptop

Note: Raspberry Pi deployment will happen on Friday during in-person session.

Day 4 (Thursday) — REMOTE

Edge Deployment & Project Work

Lecture Session (3 hours) — Dr. Sheng

Why Edge AI? (45 min)

- Cloud vs. edge inference: latency, privacy, cost, reliability
- Edge devices: Raspberry Pi, Jetson, mobile phones
- Constraints: limited memory, no GPU, power consumption

Model Optimization Pipeline (1.5 hours)

- Model formats: PyTorch → ONNX → TensorRT/OpenVINO
- Quantization: float32 → int8, accuracy vs. speed tradeoff
- Model pruning and architecture choices for edge
- ONNX Runtime: cross-platform inference engine
- Benchmarking: measuring FPS, latency, memory usage

From Vision to Robotics (45 min)

- Connecting perception to action
- Coordinate systems: pixel space → world space
- The observation pose workflow for eye-in-hand cameras
- Homography transforms for monocular vision
- Preview of Friday's hardware integration

Lab Session (3 hours) — Graduate TA

Guided: Model Export and Optimization (1 hour)

- Export trained PyTorch model to ONNX format
- Apply int8 quantization
- Compare model sizes: original vs. optimized
- Benchmark inference speed on **laptop CPU** (simulating edge constraints)

Project Work Session (2 hours)

- Select project track (see Project Options)
- Review project requirements and stretch goals

- Begin implementation **in preparation for Friday's hardware session**
- For Track A (robotics): write control logic, test with mock/simulated inputs
- For Track B (edge): prepare deployment scripts, test ONNX inference on laptop
- For Track C (custom): finalize dataset and train custom model
- Instructor and TA available for Q&A

Deliverables:

- Optimized ONNX model ready for deployment
- Project track selected
- Project code ready to deploy on hardware Friday

Note: All hardware integration (Raspberry Pi, robotic arm) happens Friday.

Day 5 (Friday) — IN-PERSON

Hardware Integration & Showcase

Morning: Integration Session (3 hours) — Dr. Sheng + Graduate TA

Hardware Setup (30 min)

- Reconnect to Raspberry Pi and robotic arm
- Transfer optimized models to Pi
- Verify camera and arm connectivity

Hands-On Integration (2.5 hours)

- Deploy your trained model to Raspberry Pi
- Test real-time detection with live camera feed
- For Track A: Connect vision output to arm control, implement pick-and-place
- For Track B: Optimize Pi pipeline, add features (counting, alerts, UI)
- For Track C: Demo custom application on Pi or laptop
- Instructors circulate to help debug and refine

Afternoon: Showcase & Wrap-Up (3 hours)

Final Polish (1 hour)

- Complete project implementation

- Test and debug final demo
- Prepare 5-minute presentation

Project Showcase (1.5 hours)

- Each student/team presents their project
- Live demonstration
- Explain: What did you build? What challenges did you face? What did you learn?
- Q&A from peers and instructors

Closing Session (30 min)

- Recap of the week's journey
- Where to go from here: resources for continued learning
- Certificates of completion
- Group photo

Deliverables:

- Working project demonstration
- Completed presentation
- Certificate of completion

Project Options

Students will choose one of the following project tracks on Day 4. All tracks use the same core skills (CNNs, object detection, edge deployment) but apply them differently.

Important: During remote days (Tue–Thu), students prepare their code and models. All hardware testing (Raspberry Pi, robotic arm) happens on Friday.

Track A: Vision-Guided Robotic Arm (Hardware Focus)

Build a system where the robotic arm detects and picks up objects based on visual input.

During remote days:

- Train object detection model
- Write arm control logic with mock/simulated inputs
- Prepare coordinate mapping code

On Friday:

- Deploy model to Pi, connect to arm
- Calibrate camera-to-robot coordinates
- Test and demo pick-and-place

Requirements:

- Detect colored blocks using your trained model
- Map camera coordinates to robot coordinates
- Implement pick-and-place for at least one object type

Stretch goals:

- Sort multiple objects by color
- Handle objects in random positions

Best for: Students excited about robotics and hardware integration

---

Track B: Real-Time Detection System (Edge AI Focus)

Build an optimized object detection system running on Raspberry Pi.

During remote days:

- Train and optimize detection model
- Export to ONNX, apply quantization
- Test inference on laptop CPU

On Friday:

- Deploy to Raspberry Pi
- Optimize for real-time performance
- Add features and polish demo

Requirements:

- Deploy custom-trained model to Raspberry Pi
- Achieve real-time performance (10+ FPS)
- Display detection results with bounding boxes

Stretch goals:

- Add object counting or tracking
- Create a simple alert system (e.g., beep when object detected)
- Build a web interface showing live detection

Best for: Students interested in edge computing and optimization

---

Track C: Custom Detection Application (Creative Focus)

Apply object detection to a problem of your choice.

Example ideas:

- Face mask detector
- Pet detector (dog vs. cat vs. other)
- Card game assistant (recognize playing cards)
- Plant health monitor
- Personal item organizer

During remote days:

- Collect and label your own dataset
- Train and optimize custom detector
- Test on laptop webcam

On Friday:

- Deploy to Pi or demo on laptop
- Polish and present application

Requirements:

- Collect and label your own dataset (50+ images)
- Train and optimize a custom detector
- Demo working inference

Stretch goals:

- Build a simple UI or application around your detector

Best for: Students who want creative freedom and have a specific interest

---

Technical Requirements

What We Provide:

- Access to remote GPU server for model training
- Raspberry Pi 4 (available during in-person days)
- Robotic arm access (Hiwonder ArmPi Ultra)
- Base code repository with templates and utilities
- Pre-configured software environments

What Students Need:

- Laptop with:
    - Webcam (for testing detection locally)
    - Ability to install VS Code and SSH client
    - Stable internet connection for remote days
    - At least 8GB RAM recommended

Software (free, installed during Day 1):

- VS Code with Remote-SSH extension
- Python 3.9+
- Web browser for Jupyter notebooks

Assessment

This is a learning-focused camp, not a graded course. However, we track progress through daily checkpoints:

Day	Checkpoint
Mon	Environment working, sample data collected
Tue	MNIST classifier >95% accuracy
Wed	Custom detector working on webcam
Thu	Model exported to ONNX, project started
Fri	Project demo completed

All students who complete the week and present their project will receive a certificate of completion.

About the Instructor

Dr. Bo Sheng is a professor of computer science at UMass Boston. His research interests include edge computing, mobile systems, and cybersecurity. He directs the newly funded Networked AI Lab, which focuses on robotics and collaborative AI.

FAQ

Q: I don't have much programming experience. Can I still attend?
A: You should be comfortable with basic Python (loops, functions, lists). If you've completed an intro programming course or online Python tutorial, you're ready.

Q: Do I need a powerful laptop with a GPU?
A: No. Heavy computation happens on our remote GPU server. Any laptop that can run VS Code and a web browser is sufficient.

Q: What if I can't attend one of the remote days?
A: Recordings and materials will be available, but live attendance is strongly encouraged for the lab sessions where you can get help.

Q: Can I keep working on my project after the camp?
A: Yes! You'll have all your code and trained models. We'll provide resources for continued learning.