UNIVERSITY OF SOUTH ALABAMA

Deploying Edge AI

5-Week Master's Level Course
(5 hours per week: two 2.5-hour sessions)

This course introduces Master's students to the principles and practices of deploying advanced AI models at the edge. Over 5 intensive weeks, students will learn how to optimize, convert, and deploy popular generative AI, computer vision, and speech recognition models using ONNX and its CPUExecutionProvider. The course emphasizes practical approaches to delivering AI capabilities on resource-constrained devices without specialized hardware, focusing on strategies that leverage standard CPU capabilities found in most computing environments.

University of South Alabama

Contents

1 Week 1: Edge AI Fundamentals & ONNX Runtime

1.1 Session 1: Introduction to Edge AI & ONNX Ecosystem

- What is Edge AI and why it matters
- Challenges of deploying AI at the edge (compute, memory, power)
- Overview of edge deployment approaches
- Introduction to ONNX as a deployment solution
- ONNX Runtime and its execution providers
- The CPUExecutionProvider and its advantages
- Project kickoff: Select individual project from sample options (see Sample Projects)
- Setting up GitHub repository for course project (see GitHub Repository Requirements)

PROJECT MILESTONE - Phase 1 - Edge Deployment Planning & GitHub Setup: Identify an edge AI application scenario for your individual project from the sample project options. Define resource constraints, user requirements, and performance targets. Create a project plan that includes model selection, optimization approach, and deployment strategy. Initialize a GitHub repository with appropriate README, structure, and documentation templates as specified in the GitHub requirements.

1.2 Session 2: Converting & Preparing Models for Edge Deployment

- Popular model frameworks for edge-deployable AI (PyTorch, TensorFlow, JAX)
- Converting models from various frameworks to ONNX
- Navigating common conversion challenges for modern models
- Understanding ONNX model structure and operations
- Model profiling and performance benchmarking
- Initial resource utilization assessment
- Deployment environment considerations
- GitHub: Model conversion workflow documentation

PROJECT MILESTONE - Phase 1 - Model Conversion & Analysis: Select and convert two modern AI models to ONNX format. Perform basic profiling to identify resource requirements and potential bottlenecks. Create a baseline performance report documenting memory usage, inference time, and accuracy metrics. Commit all code, configuration files, and documentation to your GitHub repository with appropriate version control and documentation.

2 Week 2: Model Optimization for Edge Deployment

2.1 Session 1: Edge-Optimized Model Architecture

- Model architectures suitable for edge deployment
- Architectural modifications for edge efficiency
- Knowledge distillation techniques
- Pruning strategies for neural networks
- Quantization fundamentals (static vs. dynamic)
- Weight sharing and factorization
- Measuring accuracy impact of optimizations
- Optimization-accuracy tradeoff analysis
- GitHub: Documenting optimization experiments

PROJECT MILESTONE - Phase 2 - Model Architecture Optimization: Implement at least two optimization techniques (distillation, pruning, or architecture modification) on your selected models. Document the impact on model size, inference speed, and accuracy. Create a visualization comparing the tradeoffs between the original and optimized models. Update your GitHub repository with optimization code, results, and thorough documentation of your methodology.

2.2 Session 2: Advanced ONNX Optimization Techniques

- ONNX Runtime optimization levels
- Graph-level optimizations in ONNX
- Post-training quantization with ONNX
- Quantization-aware training approaches

- Mixed precision inference strategies
- Using ONNX Runtime profiling tools
- Operator fusion and custom operators
- Optimizing CPU memory access patterns
- GitHub: Creating optimization pipeline documentation

PROJECT MILESTONE - Phase 2 - ONNX-Specific Optimization: Apply ONNX-specific optimizations to your models, including quantization and graph optimizations. Create an optimization pipeline that can be applied to similar models in the future. Benchmark the fully optimized models on a standard laptop CPU and document the improvements over baseline. Update your GitHub repository with reproducible optimization pipeline code and benchmarking results.

3 Week 3: Deploying Generative AI at the Edge

3.1 Session 1: Edge-Optimized Text Generation Models

- Overview of generative text models deployable at the edge
- Efficient transformers: techniques and architectures
- Optimizing LLMs for edge deployment (quantization, pruning, distillation)
- ONNX conversion for generative text models
- KV cache optimization for inference
- Efficient prompt and response handling
- Batching strategies for text generation
- Streaming text outputs on limited hardware
- GitHub: Documenting model optimization workflows

PROJECT MILESTONE - Phase 3 - Edge Text Generation: Deploy a compact LLM using ONNX Runtime's CPUExecutionProvider. Implement optimized inference with proper KV cache management and response streaming. Benchmark the model's performance on a standard laptop, measuring tokens per second, memory usage, and latency. Update your GitHub repository with edge-optimized inference code, benchmarking utilities, and detailed performance analysis.

3.2 Session 2: Edge-Optimized Image Generation Models

- Edge-deployable image generation models (lightweight diffusion models)
- Converting generative image models to ONNX
- Progressive generation techniques to reduce memory usage
- Low-rank adaptations for improved efficiency
- Scheduler optimizations for diffusion models
- Memory-efficient attention mechanisms
- On-device caching strategies
- Measuring and improving generation quality
- GitHub: Organizing and documenting model artifacts

PROJECT MILESTONE - Phase 3 - Edge Image Generation: Deploy a lightweight image generation model using ONNX Runtime. Implement memory-efficient generation strategies to enable running on limited hardware. Create a demonstration application that allows users to generate images with various prompts and settings while monitoring resource usage. Update your GitHub repository with the complete edge-optimized image generation pipeline and demonstration application.

4 Week 4: Computer Vision at the Edge

4.1 Session 1: Optimizing Computer Vision Models

- Modern efficient vision architectures (MobileNetV3, EfficientNet, etc.)
- Selecting appropriate CV models for edge deployment
- Converting vision models from PyTorch/TensorFlow to ONNX
- Efficient image preprocessing pipelines
- Memory-efficient inference techniques for vision models
- Optimizing complex vision tasks (detection, segmentation)
- Vision Transformers at the edge
- Profiling and benchmarking vision models

- GitHub: Organizing vision model repository structure

PROJECT MILESTONE - Phase 4 - Edge Computer Vision Pipeline: Deploy an advanced computer vision model (object detection or segmentation) optimized for edge deployment. Create an efficient end-to-end pipeline from image capture to result visualization that can run on standard laptop hardware. Benchmark different optimization techniques and document the performance gains. Update your GitHub repository with the complete computer vision pipeline and comprehensive documentation of your optimization techniques.

4.2 Session 2: Real-time Computer Vision Applications

- Designing real-time vision applications for edge devices
- Frame processing optimization strategies
- Camera input handling and preprocessing optimization
- Batching and parallel inference techniques
- Tracking and temporal consistency in edge vision
- Post-processing optimizations
- Hardware-aware scheduling for vision tasks
- Building responsive UIs for edge vision applications
- GitHub: Creating deployment documentation

PROJECT MILESTONE - Phase 4 - Real-time Vision Application: Create a real-time computer vision application using a webcam input that can process frames efficiently on CPU. Implement memory-efficient tracking of objects across frames and optimize the end-to-end pipeline to achieve maximum possible FPS on standard laptop hardware. Document the optimization strategies used and their impact. Update your GitHub repository with the complete real-time vision application and detailed performance analysis.

5 Week 5: Speech Processing & Multimodal Edge AI

5.1 Session 1: Edge-Optimized Speech Models

- Efficient speech recognition architectures
- Converting speech models to ONNX format

- Optimizing audio preprocessing for edge devices
- Streaming speech recognition approaches
- Memory-efficient speech processing
- Continuous listening optimization
- Wake word detection at the edge
- Measuring and improving speech recognition accuracy
- GitHub: Organizing speech model component documentation

PROJECT MILESTONE - Phase 5 - Edge Speech Recognition: Deploy an optimized speech recognition model using ONNX Runtime's CPUExecutionProvider. Implement efficient audio processing and streaming recognition. Create a demonstration that can transcribe speech in real-time on standard laptop hardware while monitoring resource usage. Update your GitHub repository with the complete speech recognition component and integration instructions.

5.2 Session 2: Multimodal Integration & Final Projects

- Combining text, vision, and speech models efficiently
- Resource-aware model orchestration
- Designing multimodal applications for edge deployment
- Creating responsive user experiences
- Deployment patterns for production use
- Packaging edge AI applications
- Final project integration and testing
- Performance optimization and troubleshooting
- GitHub: Finalizing project documentation and preparing for presentation

PROJECT MILESTONE - Phase 5 - Complete Multimodal Edge AI System & Demo: Integrate optimized text, vision, and speech models into a cohesive edge AI application that can run efficiently on standard laptop hardware. Create a compelling 10-minute demonstration that showcases your system handling complex multimodal tasks while maintaining responsive performance. Present performance metrics comparing your edge-optimized solution against non-optimized baseline models. Finalize your GitHub repository with comprehensive documentation, installation instructions, and performance analysis.

PROJECT MILESTONE - Final Delivery: Deliver a compelling ”Edge AI Product Demo” that demonstrates your system’s capabilities, highlights the technical optimizations achieved, and presents a deployment strategy for resource-constrained environments. The demonstration should include live performance on standard CPU hardware and quantitative comparisons showing the improvements over non-optimized approaches. Your final GitHub repository should serve as a complete reference implementation that others could use to reproduce your work.

6 Individual Project Components

Throughout the 5-week intensive course, each student will build a multimodal edge AI application with the following components:

THE EDGE AI DEPLOYMENT CHALLENGE

- **Edge Deployment Excellence:** Create an AI application that runs efficiently on standard CPU hardware without requiring specialized accelerators
- **Advanced Model Optimization:** Demonstrate significant size and speed improvements through quantization, pruning, and architecture optimization
- **Generative AI Capability:** Implement edge-optimized text or image generation that maintains quality while reducing resource requirements
- **Computer Vision Performance:** Deploy advanced vision capabilities (detection, segmentation) with real-time performance on standard hardware
- **Speech Processing:** Enable efficient speech recognition that works reliably in various environments
- **Resource-Aware Design:** Create intelligent scheduling and resource management to maximize performance on limited hardware
- **Compelling User Experience:** Design interfaces that maintain responsiveness despite edge constraints
- **Deployment Versatility:** Ensure the solution can run consistently across various CPU-based environments
- **Practical Application:** Develop a solution for a real-world use case where edge deployment provides significant advantages
- **GitHub Excellence:** Maintain a well-structured, thoroughly documented GitHub repository that demonstrates professional software development practices

Each session will include 75 minutes of instruction followed by 75 minutes of guided project work to ensure continuous progress toward the final deliverable.

7 Sample Projects

Choose one of the following sample projects to implement throughout the course. Each project has been designed to incorporate all key learning objectives across the 5 weeks.

7.1 Sample Project 1: Edge AI Assistant

Concept: A desktop-based AI assistant that runs entirely on CPU without cloud dependencies.

Components:

- **Text Generation:** Deploy a quantized small language model (e.g., Phi-2, Gemma 2B) for contextual responses
- **Computer Vision:** Implement efficient object detection and document scanning capabilities
- **Speech Recognition:** Enable voice commands and dictation with a lightweight ASR model
- **UI:** Create a responsive interface showing real-time resource usage and model performance

Learning Focus:

- Memory-efficient LLM inference with optimized KV-caching
- Resource-aware scheduling between multiple AI capabilities
- Progressive optimizations through pruning, quantization, and operator fusion

7.2 Sample Project 2: Portable Document Intelligence

Concept: A document analysis system that can run offline on a standard laptop for field use.
Components:

- **Computer Vision:** Implement document detection, OCR, and layout analysis
- **Text Understanding:** Deploy a lightweight model for information extraction and summarization
- **Image Generation:** Add capabilities to visualize extracted information and generate annotations
- **Search:** Implement efficient on-device retrieval of document information

Learning Focus:

- Optimizing complex vision pipelines for OCR and document understanding
- Memory-efficient text analysis and extraction pipelines
- Integrating multiple optimized models into a cohesive application

7.3 Sample Project 3: Edge Creative Studio

Concept: A content creation tool with on-device generative AI capabilities.
Components:

- **Image Generation:** Implement a lightweight diffusion model for image creation
- **Text Generation:** Add a small LLM for creative writing assistance and prompt refinement
- **Style Transfer:** Deploy efficient vision models for real-time style manipulation
- **Voice Conversion:** Enable basic speech-to-speech transformation effects

Learning Focus:

- Memory-efficient diffusion model deployment
- Progressive generation techniques for image synthesis on limited hardware
- Optimizing real-time creative effects through model pruning and quantization

7.4 Sample Project 4: Smart Monitoring System

Concept: A privacy-preserving monitoring system that performs all processing on-device.
Components:

- **Computer Vision:** Implement person detection, activity recognition, and unusual event detection
- **Audio Analysis:** Deploy models for sound event classification and anomaly detection
- **Text Generation:** Add capabilities to generate incident reports and alerts
- **Time Series Analysis:** Implement efficient pattern recognition for sensor data

Learning Focus:

- Real-time vision optimization for continuous monitoring
- Memory-efficient multi-modal anomaly detection
- Resource-aware scheduling to maintain 24/7 operation on standard hardware

7.5 Sample Project 5: Field Research Assistant

Concept: An offline-capable field research tool for scientific data collection and analysis.
Components:

- **Computer Vision:** Implement species/object identification and counting
- **Image Enhancement:** Deploy lightweight models for improving field photography
- **Speech-to-Text:** Enable voice note transcription for hands-free documentation
- **Text Generation:** Add capabilities for generating field reports and data summaries

Learning Focus:

- Optimizing classification and detection models for field conditions
- Memory-efficient transcription with limited resources
- Creating battery-aware AI processing pipelines

8 GitHub Repository Requirements

Each student will maintain a GitHub repository throughout the course that will serve as both a development workspace and final portfolio piece. Your repository should follow these guidelines:

GITHUB REPOSITORY STRUCTURE

- **Project Organization:**
  - `/models` - Contains model conversion and optimization code
  - `/benchmark` - Includes performance testing and comparison utilities
  - `/app` - Houses the application code and UI components
  - `/notebooks` - Jupyter notebooks documenting experimentation and analysis
  - `/docs` - Comprehensive documentation including setup and usage instructions
- **Required Documentation:**
  - Detailed README with project overview, architecture diagram, and quick-start guide
  - Model optimization documentation detailing techniques and results
  - Performance benchmarking reports with comparative analysis
  - Weekly progress reports documenting challenges and solutions
  - Final presentation slides summarizing achievements and optimizations
- **Best Practices:**
  - Maintain clean commit history with descriptive messages
  - Use GitHub issues to track feature implementation and bug fixes
  - Create a project board to organize and track progress
  - Include requirements.txt or environment.yml for reproducibility
  - Add clear installation and setup instructions
  - Document all optimization techniques with before/after metrics

9 Assessment and Deliverables

EVALUATION FRAMEWORK

- **Weekly Milestone Achievements (30%):** Meeting or exceeding the weekly project milestones
- **Technical Optimization Quality (25%):** Effectiveness of model optimization techniques, measured by quantitative improvements
- **GitHub Repository Quality (15%):** Repository organization, documentation quality, and adherence to best practices
- **System Performance (10%):** Responsiveness and efficiency of the solution on standard CPU hardware
- **Final Demo & Benchmarks (15%):** Compelling demonstration and quantitative performance analysis
- **Documentation & Reproducibility (5%):** Comprehensive documentation enabling others to deploy similar edge AI solutions

10 Prerequisites

REQUIRED BACKGROUND

Students should have:

- Master's level understanding of deep learning principles and neural network architectures
- Strong programming experience in Python
- Experience with at least one deep learning framework (PyTorch, TensorFlow, JAX)
- Familiarity with model training and evaluation
- Understanding of performance profiling and optimization concepts
- Previous exposure to computer vision, NLP, or speech processing fundamentals
- Experience with Python package management and virtual environments
- Basic Git/GitHub knowledge (branching, committing, pull requests)

11 Resources & Support

PROJECT RESOURCES

- **ONNX Core Documentation:** github.com/onnx/onnx
- **ONNX Runtime Python API:** onnxruntime.ai/docs/api/python/
- **ONNX Runtime Optimization Documentation:** onnxruntime.ai/docs/performance/
- **Hugging Face Optimum:** huggingface.co/docs/optimum/
- **TensorFlow Lite Converter:** Documentation for TensorFlow to ONNX conversion
- **PyTorch ONNX Export:** Documentation for PyTorch to ONNX conversion
- **Quantization Tools:** Resources for both static and dynamic quantization
- **Model Profiling Tools:** ONNX Runtime profiling and benchmarking tools
- **Edge AI Benchmark Suite:** Standard benchmarks for evaluating edge AI performance
- **Conda/Pip Environment Files:** Pre-configured Python environments for edge AI development
- **Jupyter Notebooks:** Weekly starter templates and optimization examples
- **GitHub Templates:** Project structure templates and documentation examples
- **Sample Project Starter Repositories:** Basic code structure for each sample project
- **Technical Office Hours:** Instructor-led optimization and debugging sessions

12 Individual Project Development Journey

FROM MODEL TO DEPLOYABLE EDGE AI

- **Week 1: Analysis Phase** (See Week 1)
  - Understand edge deployment requirements and constraints
  - Master ONNX fundamentals and CPUExecutionProvider
  - Establish model selection and conversion workflow
  - Create performance profiling methodology
  - Set up GitHub repository structure and documentation templates
- **Week 2: Optimization Phase** (See Week 2)
  - Apply model architecture modifications
  - Implement quantization and pruning techniques
  - Optimize neural network operations
  - Establish optimization benchmarks
  - Document optimization approaches and results in GitHub
- **Week 3: Generative Capabilities** (See Week 3)
  - Deploy efficient text or image generation models
  - Optimize memory usage for generative tasks
  - Implement progressive generation techniques
  - Create responsive generative applications
  - Update GitHub with generative model optimizations
- **Week 4: Visual Intelligence** (See Week 4)
  - Deploy efficient computer vision models
  - Optimize real-time processing pipelines
  - Implement memory-efficient tracking
  - Create responsive vision applications
  - Document vision optimization techniques in GitHub
- **Week 5: Multimodal Integration** (See Week 5)
  - Add optimized speech processing
  - Integrate multiple modalities efficiently
  - Fine-tune system performance
  - Prepare final demonstrations and benchmarks
  - Finalize GitHub repository with comprehensive documentation

13 Application Domains

EDGE AI APPLICATION DOMAINS

Students may choose to specialize in one of these high-impact domains for edge AI deployment:

- **Healthcare Edge AI:** Portable medical diagnostics, patient monitoring systems, and on-device health analysis
- **Field-Deployable AI:** Remote or offline AI solutions for agricultural, environmental, or disaster response scenarios
- **Smart Manufacturing:** On-device quality control, equipment monitoring, and offline process optimization
- **Edge-Based Security:** Privacy-preserving surveillance, authentication, and threat detection without cloud dependence
- **Remote Collaboration:** Edge-enhanced communication tools for low-connectivity environments
- **Mobile Creativity:** On-device generative AI for content creation without reliance on cloud services
- **Accessible AI:** Low-resource AI solutions for developing regions or legacy hardware
- **Custom Domain:** Students may propose their own edge deployment focus with instructor approval

14 Contact Information

INSTRUCTOR CONTACT & OFFICE HOURS

Course Instructor:	Dr. Ali Haidous
Email:	ali.haidous@gmail.com
Office Hours:	By Appointment
Course Website:	edge-ai

For technical assistance with course projects, edge AI implementation questions, or additional resources, please visit the course GitHub repository or contact the instructor via email.