Piyush Hingu

94089-26001 — CS22M1107@iittp.ac.in — linkedin.com/Piyush-Hingu — github.com/Piyush-Hingu

Summary — Creative and results-driven software engineer with expertise in designing and optimizing software applications. Proficient in full-stack development, system architecture, and high-performance computing. Skilled in developing scalable applications using C++, Python, and JavaScript. Experienced in working with ML models, databases, and cloud-based deployments. Passionate about writing clean, maintainable code and contributing to high-impact projects.

Skills

Languages: C++, C, CUDA, Python, Java
OS: Debian, Ubuntu, macOS, Windows
Compiler: Lex, Yacc, Bison, LLVM
ML Compilers: CTranslate2, Glow, TVM, OpenXLA
Machine Learning: ML models (LeNet, AlexNet, ZFNet, etc.), Pruning, Quantization, Data Preprocessing
Backend: Flask, FastAPI, Express.js, Django, Node.js
S.E. Concepts: Agile, Waterfall, DevOps, Microservices, REST APIs, CI/CD
Testing: Git, LaTeX, VS Code, Jest, gtest, PyTest
Technologies: React.js, Node.js, Figma

Experience

Zentreelabs Pvt. Ltd. **2024 – Present**
Research and Development Engineer (Machine Learning)

Model Optimization:

- Trained and optimized AlexNet and LeNet-5 for deployment on resource-constrained devices using pruning (weight, neuron, structured, global) and quantization (Post-Training, QAT).

Hardware Deployment

- Evaluated ML compilers (TVM, OpenXLA, Glow, CTranslate2, LLVM) and optimized models for hardware-specific performance.

LLVM Debugging Optimization

- nalyzed LLVM IR, AST transformations, memory management, and optimizations (-O1 to -Ofast, Polly).

Accomplishments

- **Pruning Improvements on AlexNet:**
  - Loaded the pretrained AlexNet model (Accuracy: 36.5%, Inference Time: 26.17 seconds, Model Size: 61.10 MB).
  - After pruning and fine-tuning for 3 epochs, achieved improved accuracy of 48.94% with an inference time of 37.82 seconds and the same model size.
- **Quantization Results on AlexNet:**
  - Initial pretrained AlexNet model: Accuracy: 36.12%, Inference Time: 39.93 seconds, Model Size: 61.10 MB.
  - After Quantization: Accuracy: 36.53%, Inference Time: 39.39 seconds, Model Size: 2.47 MB (reduction from 61.1 MB to 2.47 MB).
  - With Quantization-Aware Training (QAT): Accuracy improved to 36.25% with inference time reduced to 7.34 seconds.
- Reduced AlexNet model size significantly (61.1 MB to 2.47 MB) and optimized inference time for deployment on low-resource devices.
- TVM Optimization for LLVM Target: Accuracy: 73.26%, Inference Time: 3.20 seconds, Model Size: 99.13 MB.
- Accelerated model inference and maintained comparable accuracy through hardware-specific optimizations and compiler integrations.

Education

Gujarat University **2017 - 2021**
Bachelor of Technology in Computer Science and Engineering
CGPA: 8.75/10

Indian Institute of Technology Tirupati **2022 - 2024**
Master of Technology in Computer Science and Engineering
CGPA: 8.53/10

Certifications

- NLP - Introduction to Natural Language Processing

Projects

eTagger

- Developed a Chrome extension that automatically labels GitHub issues in Android project repositories based on energy consumption patterns
- Built the backend with Node.js Express.js to process issue descriptions and classify them using NLP models.
- Designed the frontend using React.js, ensuring an intuitive UI for issue tracking.
- Integrated with GitHub API to fetch and update issue labels dynamically.
- **Key Impact**: Helped developers identify and reduce energy-intensive code patterns, improving app efficiency.
- **Tools Technologies:** React.js, Node.js, Express.js, MongoDB, Chrome Extension APIs, GitHub API

MathML-to-Voice - Accessibility Tool for Visually Impaired Users

- Developed a compiler-based tool that converts MathML equations into natural speech for visually impaired users.
- Implemented Lex/Yacc for parsing MathML and generating speech-friendly representations.
- Built a web-based UI using JavaScript, HTML, and CSS, allowing users to input and hear equations in real-time.
- Optimized text-to-speech (TTS) synthesis for clear and accurate pronunciation of mathematical symbols.
- **Key Impact:** Enabled over 100+ tested equations to be converted into accessible voice format.
- **Tools Technologies:** C++, Lex/Yacc, JavaScript, HTML, CSS, Text-to-Speech (TTS) APIs

Machine Learning

- Making an ML model as small as possible for deployment into edge devices by using pruning and quantization while maintaining accuracy.
- Using different types of pruning methods like weight, neuron, structure, global, and fine-grained pruning to make the ML model run faster.
- By applying quantization methods like post-training, quantization-aware training, and uniform quantization to minimize weight size.