Yojun Moon (문요준)

yojun.moon@postech.ac.kr · github.com/yojunmn · linkedin.com/in/yojun-moon · yojun1010.github.io

Education

POSTECH (Pohang University of Science and Technology) — Mar 2021 - Present
B.S. in Computer Science and Engineering

Work Experience

Samsung Fire & Marine Insurance — Jul 2026 - Aug 2026
AI Engineer Intern, Enterprise Tech Part
Built two computer vision modules for insurance claim data, each delivered as an end-to-end pipeline with a web service for internal use by claims adjusters and underwriters.
- Delivered a module that scans every vehicle license plate in premium footage, reaching 99.6% recall and 0.997% of an FLOPs image reduction via
- Delivered a module that detects and classifies in-car service covering upload, processing, and download.
- Detection of damaged parts in car images
- Built a web service that estimates the repair cost in several of domain insurance claim forms, and fine-tuned YOLO-11-OPB-S/X models for detecting damaged parts.
- Built a web service that estimates the repair cost in several of domain insurance claim forms, and fine-tuned YOLO-11-OPB-S/X models for detecting damaged parts.
- Tech Stack: PyTorch, Ultralytics YOLO, OpenCV, FFmpeg, FastAPI

Hyundai Motor Group Robotics — POSTECH Future Public Safety Engineering Institute — Nov 2025 - Present
Research System Developer
- Data Collection — ERMR2E: an open research platform in service at a shared FastAPI backend, deployed in production at www.hyundai-robotics.com
- Data Collection — ERMR2E: an open research platform in service at a shared FastAPI backend, deployed in production at www.hyundai-robotics.com
- Built a web-based data collection platform for robot manipulation research, with a FastAPI backend and a React + TypeScript + Vite frontend. Implemented real-time video streaming, episode recording, and dataset export, and deployed it to production at www.hyundai-robotics.com for data collection.
- Built a web-based data collection platform for robot manipulation research, with a FastAPI backend and a React + TypeScript + Vite frontend. Implemented real-time video streaming, episode recording, and dataset export, and deployed it to production at www.hyundai-robotics.com for data collection.
- Tech Stack: FastAPI, React, TypeScript, Vite, MongoDB, Docker

POSTECH HAKI Lab — Jan 2026 - Feb 2026
Research Assistant, Robot Learning
- Developed the data collection module for a VLM-based robot arm control system, enabling efficient task action from demonstration videos.
- Skills: Python, PyTorch, Vision Language Models

Startup Team (Supported by a POSTECH CSE Alumnus) — Feb 2025 - Jul 2025
Backend Developer Intern
- Developed a backend for a virtual networking service and supported data-driven product decisions.
- Key Contributions:
  - Developed the WAI backend with NestJS.
  - Implemented a Google OAuth 2.0 authentication system.
  - Completed the CEO's Python scripts for a variety of entrepreneurship training at UC Berkeley INT. Tech Bootcamp.
- Skills: NestJS, TypeScript, Python, MongoDB, Docker

Projects

ROSE (POSTECH Computer Scientist Search) — Jan 2025 - Mar 2026
Web service where the POSTECH CSE department's faculty members can search their expertise, developed on contribution.
- Developed the backend with NestJS.
- Parsed DBLP XML dumps and stored normalized publication records in MongoDB.
- Skills: NestJS, TypeScript, MongoDB, Docker
- Developed the backend with NestJS.
- Parsed DBLP XML dumps and stored normalized publication records in MongoDB.
- Skills: NestJS, TypeScript

Activities

POSTECH Broadcasting Station (PBS) — Mar 2023 - Dec 2025
Video Editor, Art Director, Executive Committee Member
- Filmed, edited, and directed broadcast content for approximately 10 episodes.
- Led a team of 5 video editors and managed the production workflow.

Programming Tutor — Jan 2020 - Present
Teaching Python and C++ to middle and high school students.

Kumon Math Instructor — Jan 2019 - Dec 2020
Taught elementary school students math using Kumon's self-learning curriculum.

Skills

Programming Languages: Python, C, C++, TypeScript, JavaScript

Full-Stack Web Learning: React, Vite, Ultralytics YOLO, OpenCV, ALSA, LLM APIs

Backend: FastAPI, NestJS, MongoDB

Tools / Miscellaneous: Docker, Git, Linux, Raspberry Pi, FFmpeg

---

1. Samsung Fire & Marine Insurance — Computer Vision Modules

Jul 2026 - Aug 2026 · AI Engineering Internship, AI Development Part

GitHub: Private · samsung-fire-ai-intern

Automatic License Plate Masking

GitHub: masking_plate.pth

Built a web service that automatically locates license plates in dashcam videos. The pipeline uses YOLO-11n to detect vehicles, crops and enlarges the vehicle regions, and passes them to a separate license plate detector. The model then creates and writes the masked video, downloadable as-is or after downscaling. When processed processing overhead through batched GPU inference, frame decoding, and parallel encoding of output segments.

The detector achieved 99.6% recall and approximately 0.997 on the image test set. The video pipeline processes footage at roughly 1.5× real-time on a single GPU, including upload, inference, and encoding. Built as a FastAPI service, the frontend supports upload, progress tracking, and download.

Technologies: PyTorch, PyTorch, Ultralytics YOLO, OpenCV, FFmpeg, FastAPI

Generative AI Document Forgery Detection

Detection Service · forge_pipe.pkl · Dataset Viewer · dochforge.kaya.kr

Built both anomaly detection trained on genuine documents and supervised classifiers trained on synthetic forgeries. The anomaly detection approach models the distribution of genuine documents and flags deviations, while the supervised classifiers learn explicit boundaries between genuine and forged documents. The system supports 5 forgery types: copy-move, splicing, inpainting, text replacement, and whole-region generation.

The detection achieved 99.2% recall and approximately 0.99 F1 on the image test set. The video pipeline processes footage at roughly 1.5× real-time on a single GPU, including upload, inference, and encoding. Built as a FastAPI service, the frontend supports upload, progress tracking, and download.

Technologies: PyTorch, PyTorch, TensorFlow, OpenCV, scikit-learn, FastAPI

---

2. PCSS — Faculty Candidate Search

Aug 2026 - Present · Commissioned by the POSTECH CSE Department

Website: pcss.postech.ac.kr

I developed PCSS to reduce the time professors spend manually searching conference papers for Korean researchers. I parsed DBLP XML data, stored publication and author records in MongoDB, and used an LLM API to classify papers by research area and Korean author.

I built a search conference and year range to view conference trends and publication statistics. It saves that previously required hours of manual browsing became a query that returns results from the prepared database in seconds. I also implemented monthly data updates and email notifications for new papers from selected conferences.

Technologies: Python, MongoDB, LLM APIs

3. Commeet — Meeting Scheduling and Record Management

Aug 2026 - Present · Service for the POSTECH CSE Department

Website: commeet.postech.ac.kr

Commeet calculates meeting schedules from participants' availability. I modeled scheduling as a constrained optimization problem and implemented a solver that finds the schedule as many participants as possible while respecting the number of rooms.

I also added an advisory tool for frequent invitees and invite for professors to manage consultation records. Both features are backed by MongoDB and integrated into the existing scheduling and organizing student records.

Technologies: FastAPI, MongoDB, Motor, JinjaX, JavaScript

4. Korean National Police University — Research Software

Nov 2025 - Present · Big Data Research Lab

Research Systems · Video AI Academic · C# · GitHub: github.com/yojun0705/Hwa

I started by building a web crawler and continued developing software for the lab's research needs. I made data collection and analysis tools for police research, and I also made a video AI program for academic use. I worked with a team of 3 students and 1 professor, and I was responsible for designing the application architecture to be easy to maintain, could collect data, and receive results through a browser.

Technologies: Python, TensorRT, MongoDB, PyQt, vLLM, YOLO

Web Crawler

Website · crawler.kopo.kr

I built the crawling data from Naver News, Blog, Cafe, and YouTube. I implemented asynchronous task queues and proxy IP rotation, with a web interface for monitoring collection jobs and reviewing the results.

Statistical Analysis

Website · statistics.kopo.kr

I built the modeling and analysis tools, covering statistical analyses, and visualizing results. Researchers can select variables and analysis settings, then review tables and charts together.

Network Analysis

Website · network.kopo.kr

I built the word-based co-occurrence networks from text. It combines network graphs with analysis controls so researchers can explore connections between words and entities.

Keyword Signal Analysis

Website · keyword.kopo.kr

I built the keyword time-series methodology as software to support research analysis. The tool calculates indicators from keyword frequencies, document counts, and time windows, then classifies keywords. Researchers can adjust the settings, compare signal distributions and changes, and export results.

Research Data Manager — MANAGER

Website · manager.kopo.kr

I developed a desktop tool that brings together data collection, analysis, and research resource management. It includes database browsing, collection jobs, analysis functions, and user management in one environment. I also added the recovery functionality it was applications.

AI Complaint Drafting

Website · complaint.kopo.kr

I developed a service that uses an LLM to draft a criminal complaint from the user's description of an incident. I organized the input to the model and placed the generated content into a document template, with previews and downloads. I contributed to the software development for the research and received a Letter of Appreciation from the Commissioner General of the Korean National Police.

5. LecAI — Study Assistant for Slides, Recordings, and Board Notes

Mar 2026 - Jun 2026 · GitHub · github.com/yojun0705/LecAI

I built LecAI for my own studies. It analyzes lecture materials and generates explanations for each slide. Learning progress and notes are stored in a local Whisper model, and processed locally, so they can be used offline.

Also, I wrote a shell script that computes quoted passages against the transcript.

I used Claude Code for most of the implementation. I designed the architecture, implemented the LaTeX, and edits the prompts. I used Claude Code to write the tests, but I reviewed them to update expectations, and found many bugs.

Technologies: Python, LLM APIs, VLM, Whisper, FFmpeg, LaTeX

6. Eng2Mouth — AI Phone-Call English Practice

Website · eng.yojun.dev · GitHub · github.com/yojun0705/eng2mouth

I built Eng2Mouth to practice spoken English in a real phone call. Users choose a conversation scenario, and the AI responds in real time. The system also tracks grammar mistakes and vocabulary, so I can see my progress.

The main challenge was latency. I used Twilio Media Streams to connect the phone call to the server, and I streamed the audio to Deepgram for speech recognition and to ElevenLabs for text-to-speech. This reduced the delay to about 1 second.

I used Claude Code to write the code, but I reviewed the architecture and fixed many bugs. I also wrote tests to prevent regressions.

Technologies: Twilio Media Streams, Deepgram, ElevenLabs, Claude API, Python, FastAPI, WebSocket, SQLite