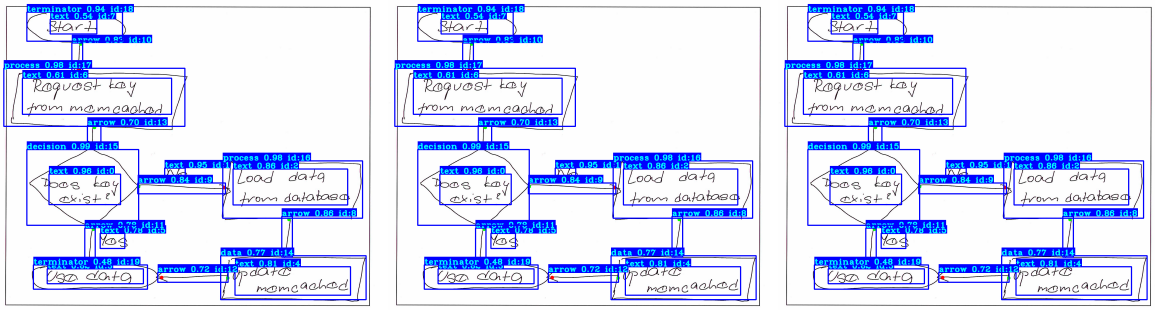
CMPE-258 Executive Summary: Hand-drawn Flowchart Recognition



In this project, we developed a program to recognize hand-drawn flowcharts. and render them in productivity applications including word processors and presentation softwares.

Many softwares support the creation of flowcharts through providing drawing primatives such as circles, blocks, and arrows, they are often awkward to use. Many people avoid flowcharts due to such difficulty, despite flowcharts are versatile and are often the best and most succinct waay to summarize concepts such as workflow and relationships. We sought to offer a solution to this problem.

We developed a deep learning model that takes take an image of a flowchart, and extract the blocks, text, arrows, and their relative positioning. Thereafter, the flowchart can be recreated in graphical applications. Our approach was built on YOLO-V4 trained on objects specific to flowchart detection, including blocks (processes), parallelograms (data), diamonds (decisions), circles (I/O), arrows, and lines. We also extract text elements and converted to text using Tesseract OCR. The following are technical details of our implementation.

- Our YOLO was trained on a dataset with 600 annotated flowchart images, with custom XML preprocessing.
- Our model was trained to detect seven custom classes (`text`, `arrow`, `data`, `decision`, `process`, `terminator`, and `connection`), starting with pre-trained YOLO weights. Training took a total of six hours with GPU, achieving an MAP of 90.365%.
- During post-processing, we compile a list of detected objects with a rich set of attributes, including position, size, object class, and confidence.
- We implemented a custom algorithm to detect arrow direction, using OPENCV. We also established the logical connections between directed edges and flowchart elements.
- Text elements were identified and converted to actual text using Google's TESSERACT-OCR engine.
- We then follow the logical order of arrows to render the flowchart elements in order.
- Finally, to demonstrate using SCHEMADRAW as the drawing application.
- We utilized the following tools in the project: PYTHON 3, OPENCV 2, PYTORCH 2.0, scripts from PyLessons, Google Colab (faster GPU-based training, YOLO-V4, Google TESSERACT, and SCHEMADRAW.

In conclusion, we have successfully implemented a YOLO-V4 based program to recognize hand-drawn flowcharts, and demonstrated the effectiveness of our method.

Name	Major & ID	Main Responsibility	C	T&V	P	ES	Overall
Mu Chen	SE, 014725425	R&D, YOLO, SCHEMADRAW	40%	40%	10%	10%	25%
Roger Kuo	SE, 013784706	R&D, YOLO, arrow detection	40%	40%	10%	10%	25%
Hardy Leung*	AI, 016711877	Proposal, Summary, Coordinator	10%	10%	10%	70%	25%
Jasmine Wang	AI, 002805362	testing, presentation	10%	10%	70%	10%	25%

Table 1: C = coding, TV = testing and verification, P = PPT, ES = executive summary. $\star$ Project Coordinator.