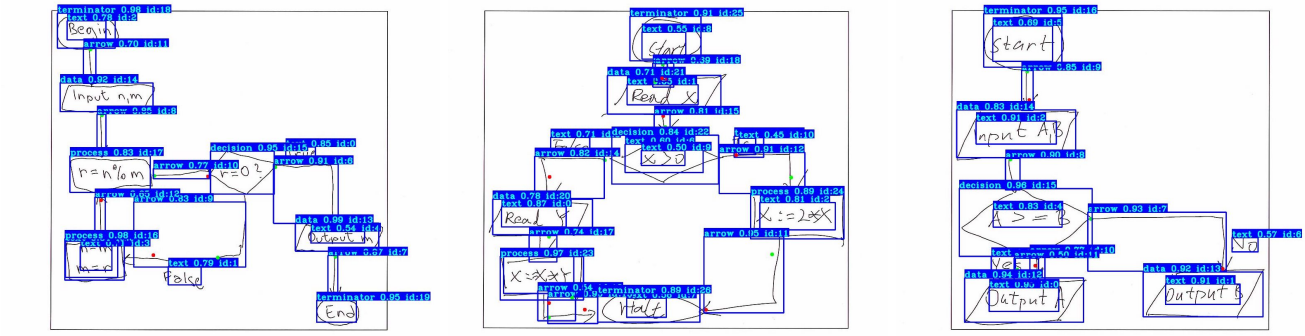
CMPE-258: Hand-drawn Flowchart Recognition



Overview – Flowcharts are incredibly versatile and flexible tools, often the most succinct way to summarize or visualize concepts such as workflow and relationships. While all productivity softwares support the creation of flowcharts through drawing primitives such as circles, blocks, and arrows, the creation process is often awkward and unwieldy. Many people avoid flowcharts due to such difficulty, as well as the inevitable discrepancy between what was intended and how it turns out. As such, the objective of our project is to apply deep-learning techniques to develop a hand-drawn flowchart recognition system that accurately translates user's intent to actual rendering of flowcharts in productivity applications.

Technical Details – The key ingredient of our flowchart recognition system is the YOLO-V4 engine custom trained to detect flowchart objects. We then analyze these elements and extract connectivity and symbolic relationships between them, and perform OCR to convert handwritings to actual English words. The result is a symbolic flowchart that fully captures the positional, relational, and textual intents, which we can then render in a variety of different ways. In the following, we describe our algorithm in more technical details.

- Our model is a custom YOLO-V4 deep neural network trained to detect seven custom classes, each corresponding to a key flowchart concept: `text`, `arrow`, `data` (parallelograms), `decision` (diamonds), `process` (straight or rounded rectangles), `terminator`, and `connection` (input/output).
- We used a training dataset with 600+ high-quality annotated flowchart images, with custom scripts to proprocess XML. The training was done on Google Colab with GPU, taking 6+ hours starting with pre-trained YOLO weights. Our model achieved 90%+ mAP, with 96%+ accuracy on all flowchart elements except arrows and texts.
- We wrote post-processing code to collect the list of detected objects with a rich set of positional and logical attributes. Handwritings are identified and converted to actual text using Google's TESSERACT-OCR engine.
- We wrote a highly accurate custom algorithm to detect arrow direction using OPENCV, which allows us to establish the full logical relationship between flowchart elements. Together with the shapes and text elements, we now have a full symbol representation of the flowchart.
- For demonstration purpose, we recreate the flowchart in SCHEMDRAW as intended by the user, doing so by following the logical order of arrows. Our project can be easily extended to output many other formats including SVG, various Python drawing formats (e.g. Turtle and iPyCanvas), Asymptote for LaTeX, Graphviz, and so on.

Software and Tools – We utilized the following tools in the project: Python 3.8, OPENCV 2.0, Pytorch 2.0, YOLO scripts from PyLessons, Google Colab (for fast GPU-based training), YOLO-V4, Google TESSERACT, and SCHEMDRAW.

Challenges and Lessons – We were pleased with what we have achieved in building the complex flowchart detection system, given the time constraint. While not production-ready, we have made significant inroad with the completion of a well-trained YOLO-V4 based detection system, a battery of downstream algorithms (arrow detection, flow-following) to analyze and re-render the flowcharts. That said, we do acknowledge multiple areas of improvement: (1) we would benefit from more diverse and high-quality training sets from multiple sources, (2) high mAP on block elements was undone by relatively low text and arrow detection accuracy in YOLO, (3) arrowhead detection was difficult, and prone to failure in noisy or busy environment, (4) TESSERACT needs to be retrained for hand-writings to attain acceptable accuracy, (5) we were severely limited by the inflexibility of SCHEMDRAW, and could have chosen a more robust alternative.

Contributions

Name	Major, SJSU ID	Main Responsibility	C	T&V	P	ES	Overall
Mu Chen	SE, 014725425	R&D, YOLO, SCHEMDRAW	54	49	18	9	32
Roger Kuo	SE, 013784706	R&D, YOLO, Arrow Detection	42	43	36	5	32
Hardy Leung*	AI, 016711877	Proposal, Summary, Coordination	4	5	32	84	31
Jasmine Wang	AI, 002805362	Testing, Presentation	0	3	14	2	5

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