Hardy's Comments

April 12, 2023

These are the three papers:

- "Symbol detection in online handwritten graphics using Faster R-CNN", Frank D. Julca-Aguilar and Nina S. T. Hirata (2017) https://arxiv.org/pdf/1712.04833.pdf
- "Arrow R-CNN for handwritten diagram recognition", Bernhard Schäfer, Margret Keuper, Heiner Stuckenschmidt (2020) https://madoc.bib.uni-mannheim.de/58430/1/Sch%C3%A4fer2021_Article_ArrowR-CNNForHandwrittenDiagra.pdf
- "DrawnNet: Offline Hand-Drawn Diagram Recognition Based on Keypoint Prediction of Aggregating Geometric Characteristics", Jiaqi Fang, Zhen Feng and Bo Cai (2022) https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8947756/pdf/entropy-24-00425.pdf

I call the three paper (or approaches) Hirata, Arrow, and DrawNet. They should be read in order. The Hirata approach is the simplest.

There are two main datasets, the FC dataset (FC_A, FC_B) that Hirata referred to, and the DIDI dataset that Arrow referred to. FC is small, approximately 400 flowcharts. DIDI is big, over 30K diagrams.

Personally I think we should see how far we can go with just FC and Hirata. I think the accuracy is pretty good.

Arrow uses FC and DIDI. And they focus on arrow detection, and they *modify* Faster R-CNN (think of it as similar to YOLO) to add another detection for arrow position. I think DrawNet went even further. However, I think the incremental gain may not be that much. If possible we should not do something too fancy since we don't have that much time.

DIDI is actually a pretty clever dataset. Basically they create a collection of computer-drawn flowcharts that are randomly created, and have users draw over them. And they turn it around and treat the raw user drawing as input, and the computer-drawn flowcharts as the labels. The problem is that (1) the kind of flowcharts drawn by the users are limited by what the program generated in the first place (though this is not a big problem for us), and (2) sometimes users didn't follow instructions and as a result the drawings are very poor or even mistaken so some of the "labelings" were wrong (explained in the Arrow paper).

A few issues we need to deal with right away. I think these are natural *TASKS* that the group collectively can work on:

- Let's say we'll implement Hirata. That means we need to train YOLOv4 or Faster R-CNN. We need to go ahead and set up the flow and do that. I suggest everyone tries to do it. Anyway we'll have to install YOLOv4 etc for HW.

- Let's say we use the FC dataset. It took me a while but finally I have located it (I'll try to upload it, if not I can send to Roger and he uploads it).
- FC dataset contains actual digital drawings and presumably labelings? The files are of the format inkml which is an XML format capturing the strokes. There is a program called inkml2img that can convert that to PNG but it doesn't fully work for me (I think it drew everything including the bounding boxes – maybe it needs to be controlled in a certain way).
- So we also need to do transfer learning, starting with a pre-trained YOLO. There are tutorials about how to do that, for example this one: https://dontrepeatyourself.org/post/yolov4-custom-object-detection-with-opencv-and-python/ I am not sure what is the easiest way.
- After that, we need to figure out what to do with the arrows. How do we identify the arrows? Sure the Arrow paper and the DrawNet paper did that, but if possible we don't need to complicate things too much. How does Hirata do it?
- What about OCR? Actually this is easy. For now, we have not committed to it at all, but I think in most likelihood we'll do it because it is a simple call to PyTesseract.
- Once we have done all the classification so we have all the boxes and circles and arrows, what do we do? We need to turn it into a computer-generated drawing. I think I'm gonna take over this part. We'll just mostly draw the figure *in its original position* but clearly make it such that it is newly drawn. I need to figure out what to do with the arrows, because drawing rectangles, circles, diamonds, and texts are easy, but arrows (the lines) need to be straightened, etc. And also we need to make logical connections between the arrows and the blocks etc, or else the whole thing is not very compelling if the elements are not logically linked together. I will think about this. I haven't figured out. But I think we should get the first parts ready first.

[ ]: