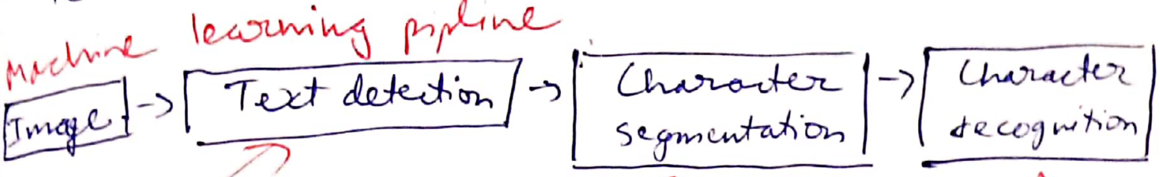
Week 11

Photo OCR (optical character recognition)

This is a problem of recognising text in images

Machine learning pipline

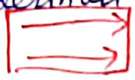


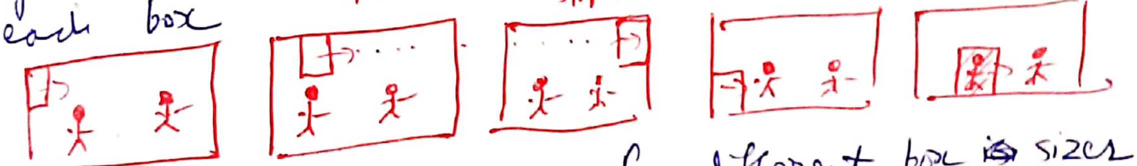
Detecting where the text is in the image & marking it with a bounding box.

Splitting the bounding box into more boxes where each box contains a character

Performing character recognition on each box

Sliding window - if we have an image & lets say we need to do, consider finding pedestrians in an image as an example we create a box (ex: 10px × 10px), slide it along the image (shifting by example 1-4 px) ~~step stride~~ step size/stride, and run pedestrian recognition on each box





We can repeat this process ~~for~~ for different box ~~is~~ sizes

Getting lods of data & artificial data

Artificial data synthesis - we have fonts available, so we can paste characters on different backgrounds for photo OCR.

to create ~~an~~ artificial data we can even take an existing training example & create distortions on it to generate new data.

It doesn't help to add meaningless/random noice (like randomly changing pixel intensity), but it must be realistic (like bad cellphone connection noise)

Discussion on getting more data

1. Make sure we have a high variance classifier
2. Ask "How much work would it be to get 10 times as much data as we currently have". If its less work, then do it.

- → Artificial Data Synthesis
- → Collect/label it yourself
- → "Crowd Source" (Eg: Amazon Mechanical Turk)

Pay ppl to label data for you

Ceiling Analysis (consider ML pipeline as example)

It helps in choose what part of the pipeline should one spend the most time trying to improve

1. First find overall system accuracy
2. For each part of the pipeline provide ground truth labels. Exa: For text detection, you manually mark where the text in the image is. So that part has 100% accuracy.
   Intuition: If we were to spend time on one part and improve its efficiency to 100%, how would that affect our overall model's accuracy?
5. Calculate how it affects overall model accuracy

Component	Accuracy	
Overall System	72%	
Text detection	89%	17%
Character segmentation	90%	1%
Character recognition	100%	10%

Improving character segmentation is a waste of time

Hence it makes sense to improve the text detection