Application example: Photo OCR (Photo Optical Character Recognition)

Problem description and pipeline

⇒ focus on how to get a computer to read the text to the purest in images we take.

Photo OCR pipeline

1. Text detection ⇒ detect out where the text is and draw a rectangle around the text
2. Character segmentation ⇒ In the rectangle, segment out each character of the words
3. Character classification ⇒ foreach character, classify out what alphabet it is.

Complex machine learning problem break down into many modules

Image ⟶ Text detection ⟶ Character segmentation ⟶ Character recognition

Machine learning pipeline ⇒ A system with many stages/components, several of which may use machine learning.

Application example: Photo OCR

Sliding windows

⇒ Starting by setting a window of pixels in 82×36 image patches

- Using supervised learning, feed the pedestrian detection classifier with photo with pedestrian (y=1) and photo without pedestrian (y=0)
- Now using the sliding window to slide the image row by row to check ~~#~~ whether the pedestrian is in the image or not.
- The distance between 1 window and the next window is determined by the slide parameter (step-size/stride)
- Now see is the algorithm can detect out the pedestrian in the image.

⇑
This same goes to Text detection

Text detection

- After doing sliding window, the output will be an image with colour in between black and white
- The highest probability bracket with words will be white colour, and then grey and least probability will be black
- Then, using "expansion" operator to expand the white region in the image

⇒ Draw out rectangle in those white region which the height to width ratio is more look like a rectangular

* To check how many times you need to run your patches can use

$$\left(\frac{\text{height} * \text{width}}{(\text{step-size})^2}\right) \text{of image}$$

Character segmentation

1D sliding window

⇒ Using supervised learning to train a classifier by feeding the classifier with image that got space (split) between 2 character as (y=1) and the image with only 1 character as (y=0)

Character classification

⇒ using supervised learning classifier

Application example: Photo OCR

Getting lots of data: Artificial data synthesis

Synthetic data ⇒ Data where the image of character is randomly chosen from a font and put on a random background, may be edited with blurry effect or brighter / darker image
which look similar to real data
⇒ we can build new data from scratch

Introduce artificial distortion into dataset to generate more data

Discussion on getting more data

1. Make sure you have a low bias classifier before expending the effort. (Plot learning curves). E.g. keep increasing the number of features / number of hidden units in neural network until you have a low bias classifier.
2. Good question to ask —— "How much (time) work would it be to get 10x as much data as we currently have?"
   - Artificial data synthesis
   - Collect / label it yourself → # hours?
   - "Crowd source" (E.g. Amazon Mechanical Turk) ⇒ ask for data labelling

   m=1,000
   10sec/example
   how about m=10,000?
   } calculation

Application example: Photo OCR

Ceiling analysis: What part of the pipeline to work on next

Estimating the errors due to each component (ceiling analysis)

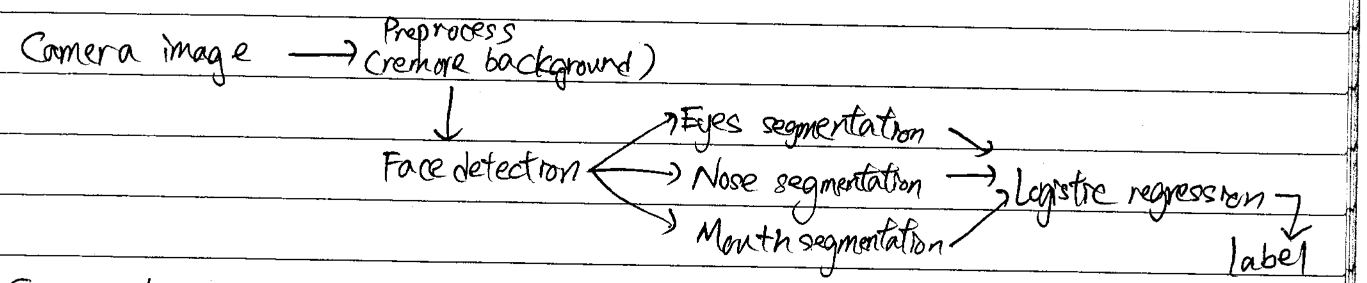
Image → Text detection → Character segmentation → character recognition

What part of the pipeline should you spend the most time trying to improve?

	Component	Accuracy	
	Overall system	72%	↓ +17%
(Manually put in correct output and let it be 100% accuracy for this component) → then, check overall accuracy again	Text detection	89%	↓ +1%
same ⇒	Character segmentation	90%	↓ +10%
same ⇒	Character recognition	100%	

Text detection worth spending more time with compared to another two components.

Ceiling analysis example
Face recognition from images (Artificial example)



Component	Accuracy	
Overall system	85%	↓ 0.1%
Preprocess (remove background)	85.1%	↓ 5.9% → Most worth
Face detection	91%	↓ 4%
Eyes segmentation	95%	↓ 1%
Nose segmentation	96%	↓ 1%
Mouth segmentation	97%	↓ 3%
Logistic regression	100%	

Quiz

Suppose you perform ceiling analysis on a pipelined machine learning system, and when we plug in the ground-truth labels for one of the components, the performance of the overall system improves very little. This probably means:

- It is probably not worth dedicating engineering resources to improving that component of the system.
- If that component is a classifier training using gradient descent, it is probably not worth running gradient descent for 10x as long to see if it converges to better classifier parameters.