18 Photo OCR

Detect text in a Photo

steps to take

1. Text detection
2. Character segmentation
3. Character classification
   => spelling correction

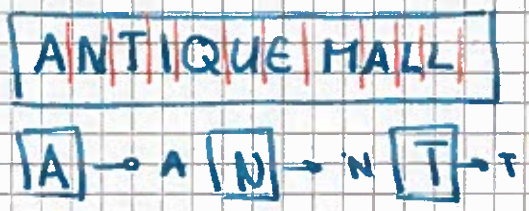


Photo OCR Pipeline

each can be a Machine learning component

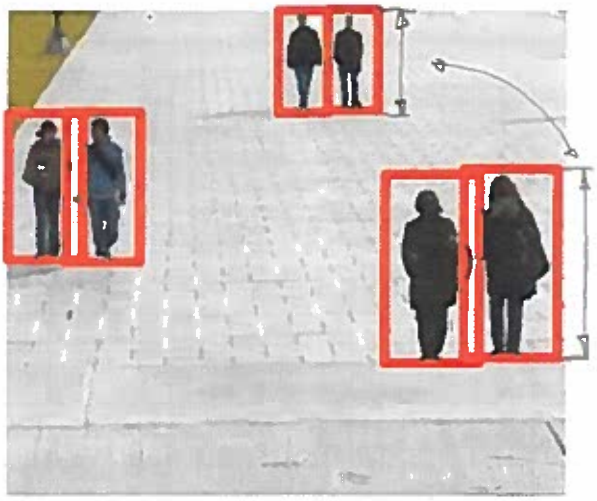
- Designing ~~piplin~~ pipeline
- Artificial data synthesizem

- what is the pipeline → how to break down the Pipeline into models

Text detection

diff. aspect ratio

Pedestrian detection

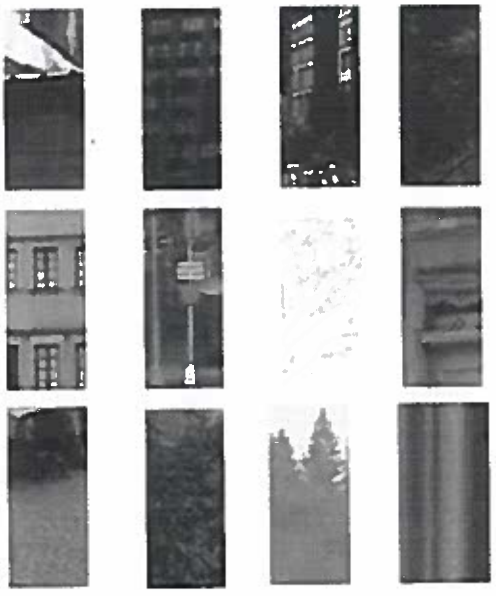


height diff.
but
aspect ratio
the same

try to find the pedestrians → aspect ratio (ratio btw. height and width)

Supervised learning for pedestrian detection

x = pixels in 82 x 36 image patches to train: 1000 - 10'000 images



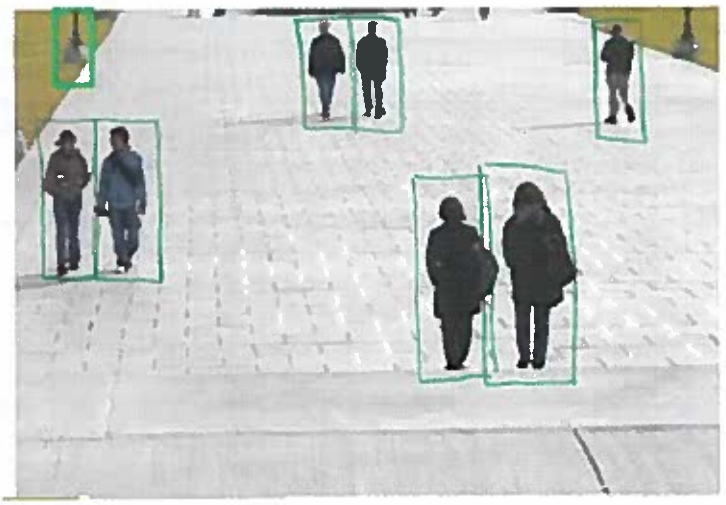
positive examples (y=1) negative examples (y=0)

train a NN to detect a pedestrian

Sliding Windows detection:

slide over and send patch to classifier →

- Step size (stride) 4px. Best is 1px but is comput. expensive



- Run image patch throug classifier to detect pedestrians

- start first with smaller image patches and run through image to detect smaller pedestrians.
- start then with a larger image patch to detect larger pedestrians (run through classifier) => take larges patch and resize to 82x36 px

Text detection:

positive examples ($y=1$) negative examples ($y=0$)

- Instead of detecting pedestrians we detect text, by sliding window approach...

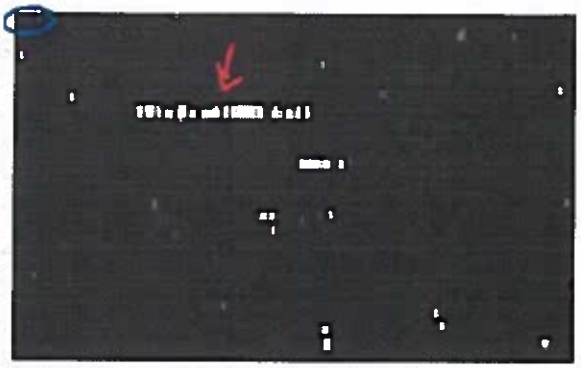
Text detection: → run sliding window patches

1. step

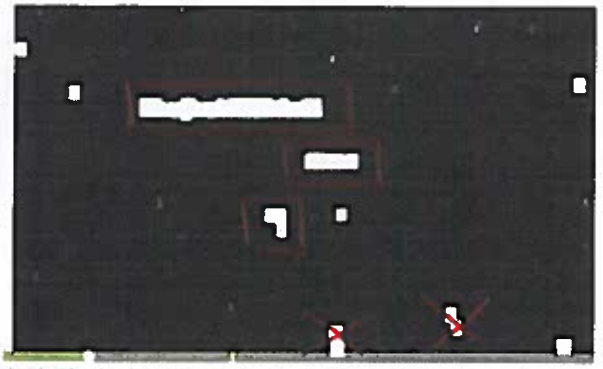
"expansion" operator

- white spots show areas where char. have been detected

2. step



3. step



- black → no text
- white → found text

aspect ratio incorrect

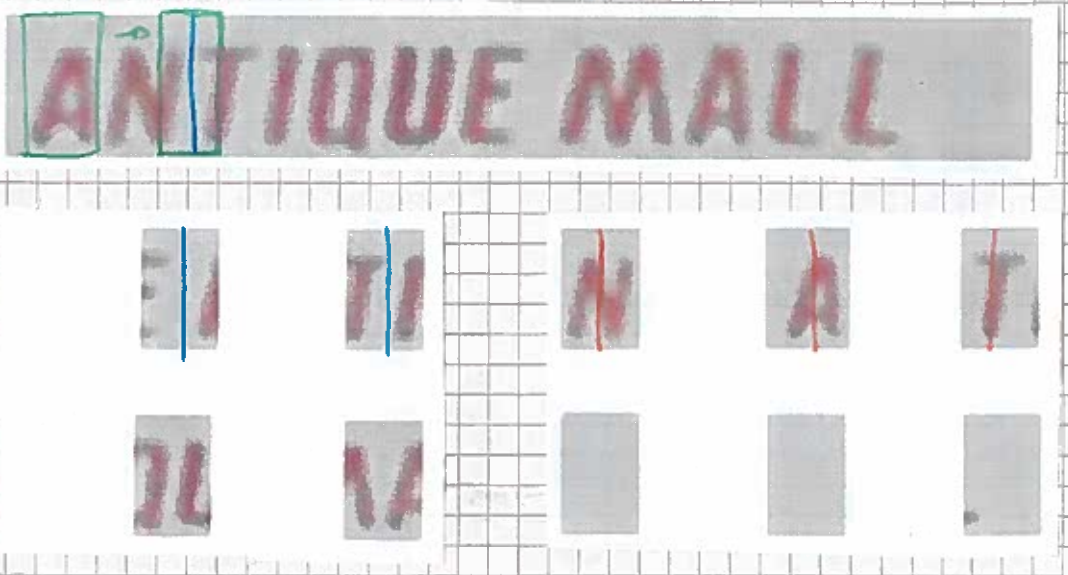
1. step => sliding window patches through image
2. step => mark found text as white spots (output of classifier)
3. step => take each of white blobs and expand that region by verifying if a pixel is white if yes expand by 5 to 10px.
4. step => connecting boxes with a certain "aspect ratio" mark these as text region
5. step => cut out the regions and use latter stages of pipeline to read the text.

- (light gray → lower probability of text detection
  white → high probability of text detection)

is it within some distance of the

-

1D Sliding window for character segmentation



positive examples ($y=1$) negative examples ($y=0$)

- detect if there is a split between two distinct characters (train a classifier e.g. NN. or other) to detect middel point..
- slide patch over the text as above to detect separation of characters (character segmentation.

Photo OCR Pipeline:

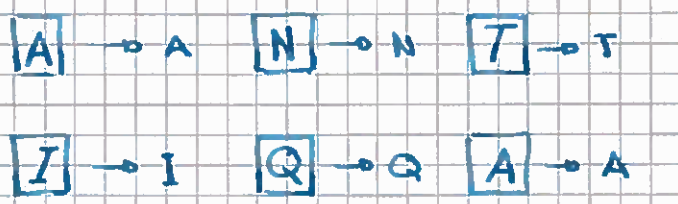
1. Text detection

2. Character segmentation

3. Character classification

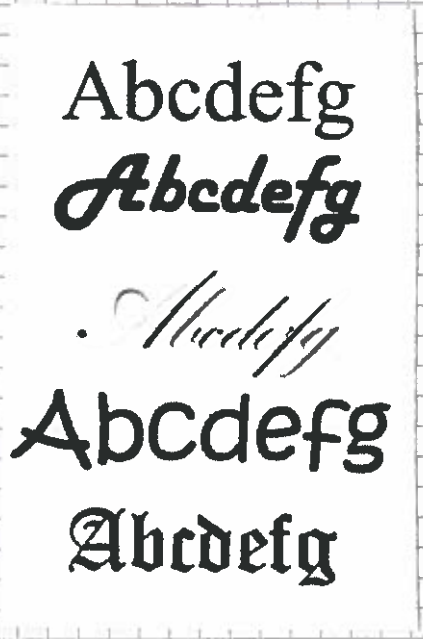
Character recognition

get a low bias Model and get lots of data!



Character recognition pipeline

Artificial data synthesis for Photo OCR



Real data

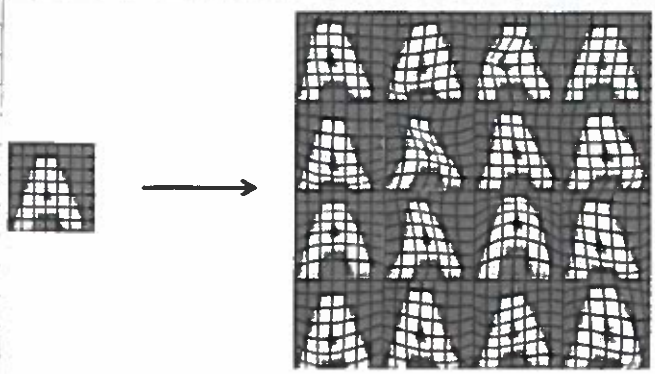
- Use a font library to create data. use characters and place on different backgrounds.

- random background
- blurring
- distortion
-

Real data

Synthetic data

Synthesizing data by introducing distortions



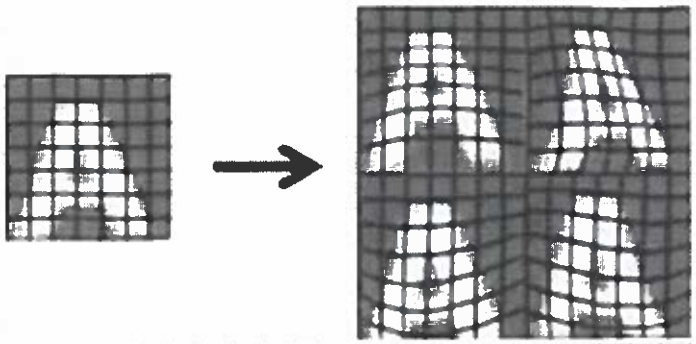
- Introduce artificial warping (distortions)
-

Speech recognition:

- Original audio
- Audio on bad cell phone connection → add distortion
- Noisy background - Crowd → add distortion
- Noisy background - Machinery → add distortion

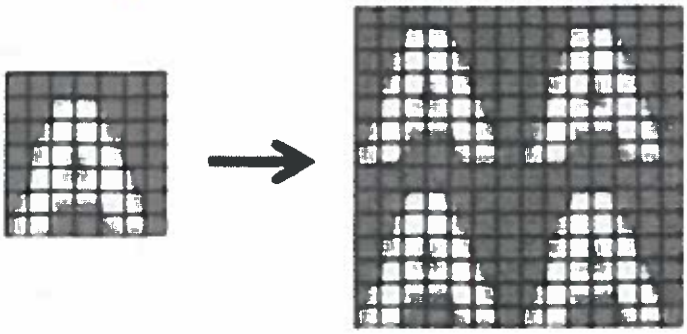
⇒ multiply the original data by synthesizing data with distortion.

Distortion introduced should be a representation of the type of noise/distortions in the test sets.



Audio:
- Background noise, bad cell phone connection

Usually does not apply to add purely random/meaningless noise to your data



x_i = intensity (brightness) of pixel i

$x_i \leftarrow x_i$ + random noise

⇒ look for meaningful distortions
gaussian noise pixel → meaning less

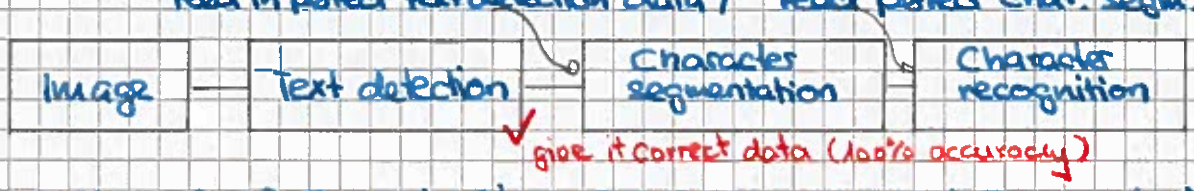
Discussion on getting more data

1. Make sure you have a low bias classifier before expanding 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. => plot learning curves

2. "How much work would it be to get 10x as much data as we currently have?"
   - Artificial data synthesis
   - Collect/label it yourself => how many hours to label and collect.
     ↳ 10 sec/example => 10'000 sec => ~ 3h
     ↳ m = 1000 => 3h
     m = 10'000 => 30h
   - Crowd source (E.g. Amazon mechanical turk system)

   - sanity check with learning curves
   - how long for creating data

Estimating the errors due to each component (ceiling analysis)



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

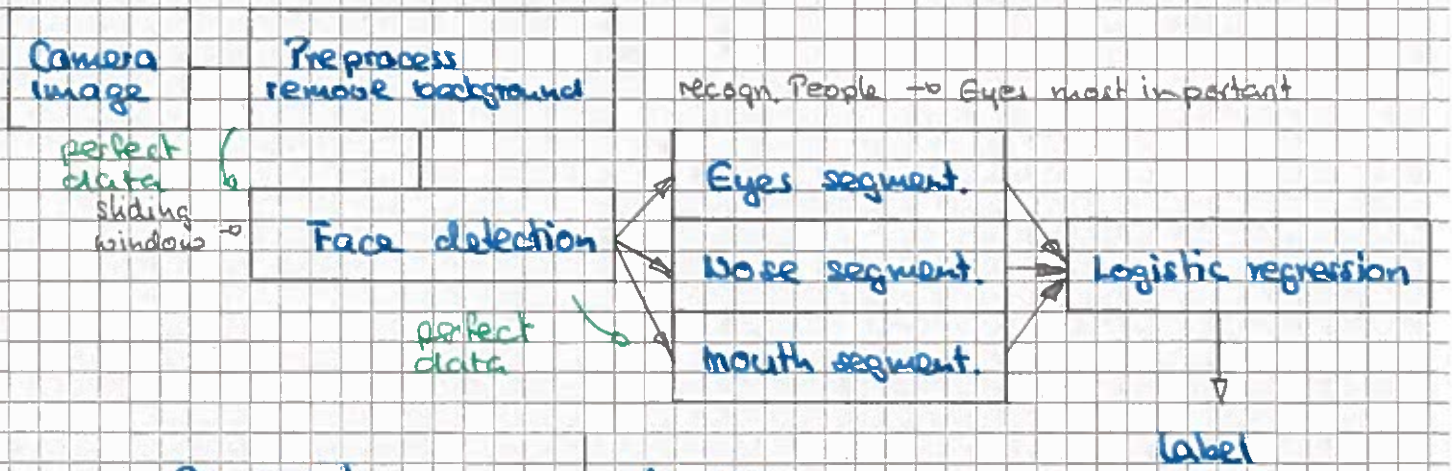
Single row number accuracy

Component	Accuracy (final output)	
overall system	72%	
Text detection	89%	+17% ← with perfect Text detection rate is.
Char. segmentation	90%	+1%
Char. recognition	100%	+10%

place resources on problem resolving.

Another ceiling analysis example:

Face recognition from images (Artificial example)



Component	Accuracy	
Overall System	85%	
Preprocess (remove backg.)	85.1%	0.1%
Face detection	91%	5.9%
Eyes	95%	4.0%
Nose	96%	1.0%
mouth	97%	1.0%
Logistic regression	100%	3.0%