



Machine Learning

Application example: Photo OCR

Problem description and pipeline

The Photo OCR problem

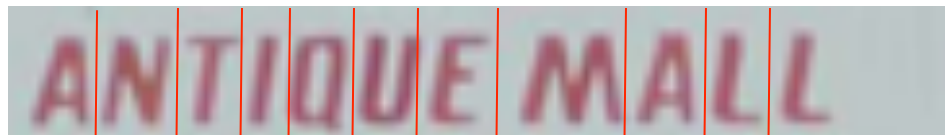


Photo OCR pipeline

→ 1. Text detection



→ 2. Character segmentation

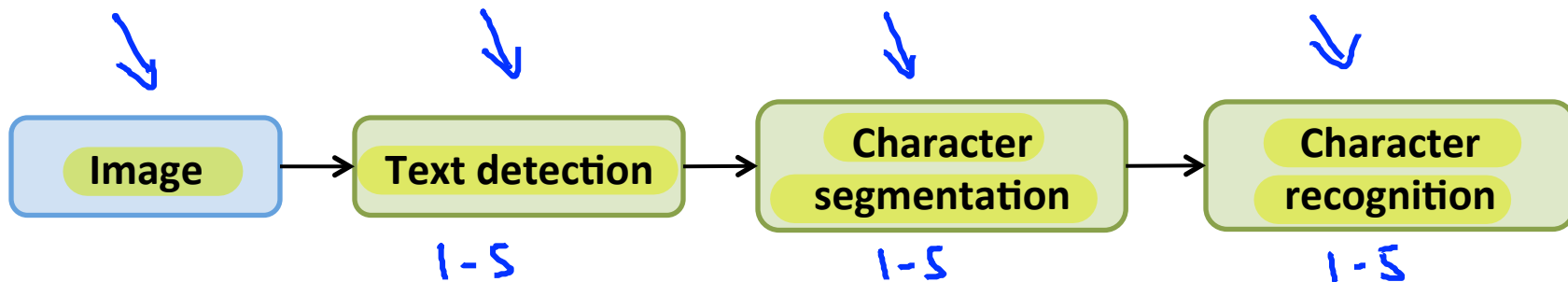


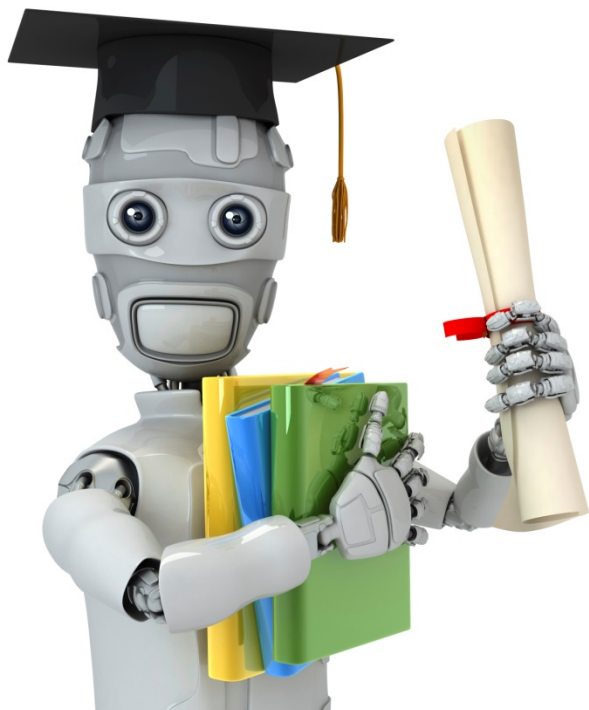
→ 3. Character classification



~~Cleaning~~ → ~~Cleaning~~

Photo OCR pipeline





Machine Learning

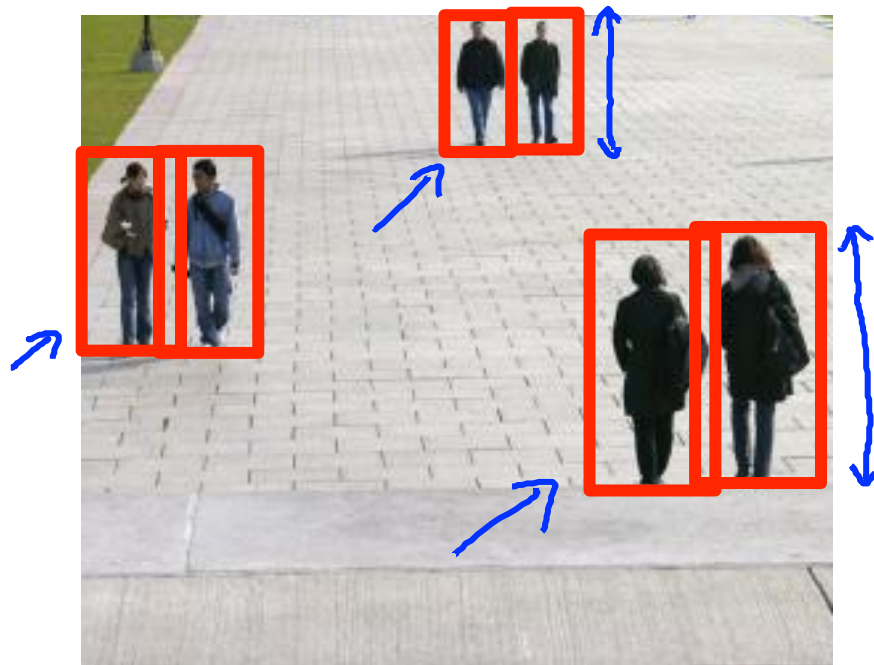
Application example:
Photo OCR

Sliding windows

Text detection



Pedestrian detection



Supervised learning for pedestrian detection

x = pixels in 82x36 image patches

1,000
10,000
...



Positive examples ($y = 1$)



Negative examples ($y = 0$)

Sliding window detection

step-size / stride



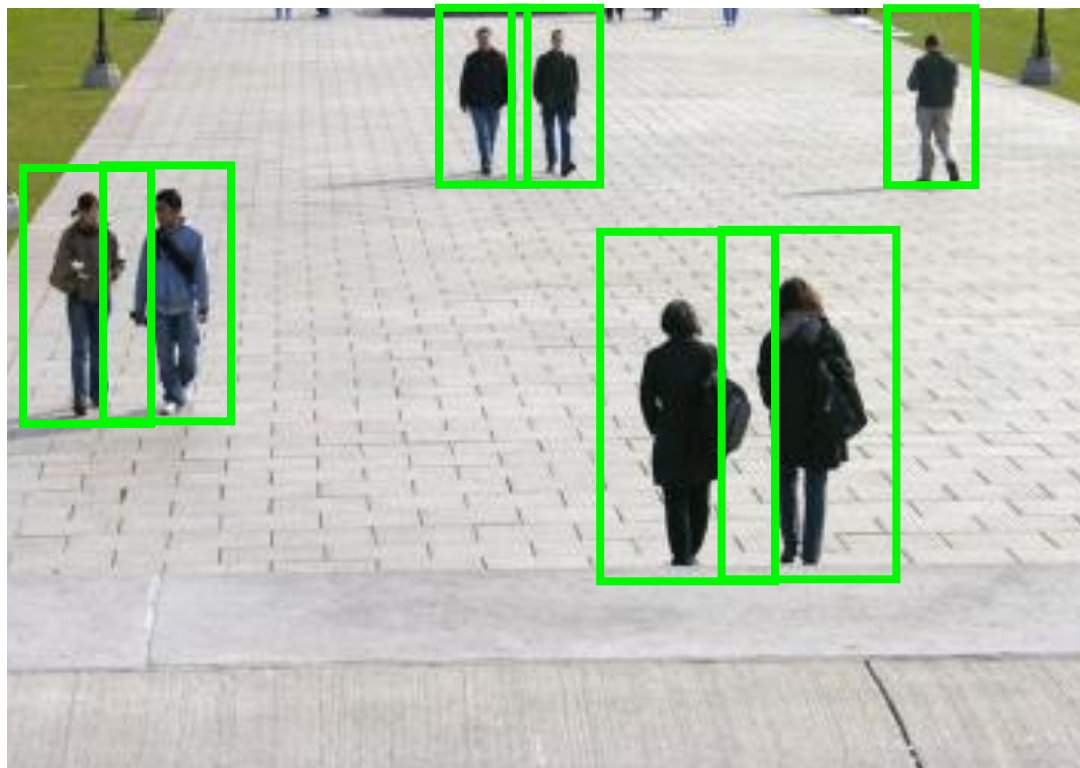
Sliding window detection



Sliding window detection



Sliding window detection



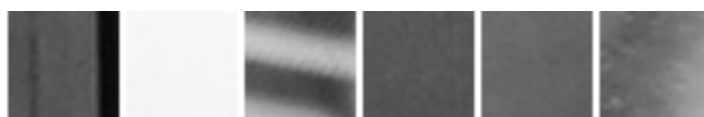
Text detection



Text detection



Positive examples ($y = 1$)

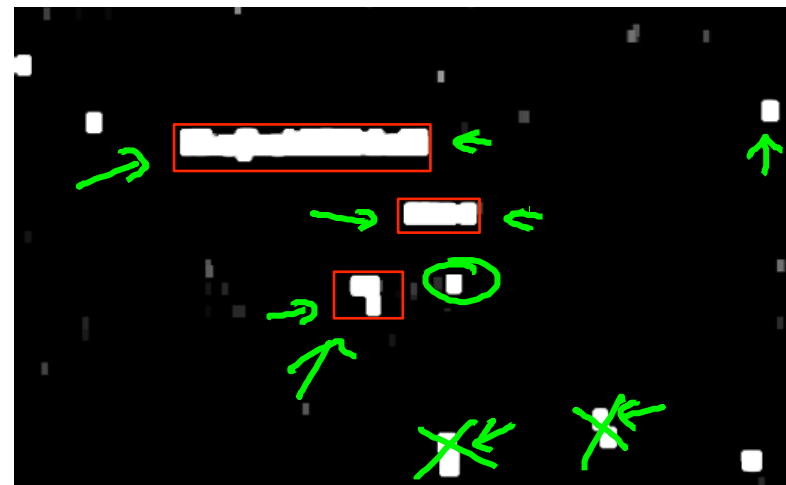
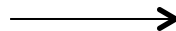


Negative examples ($y = 0$)

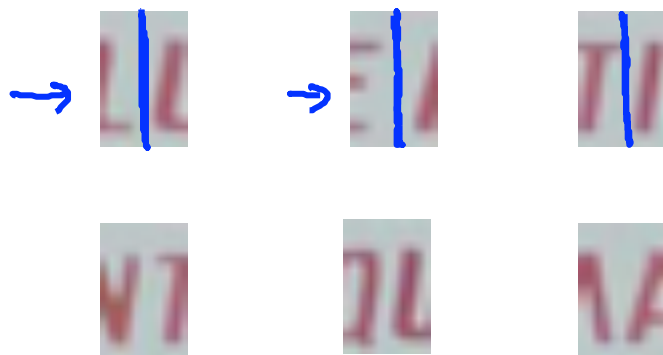
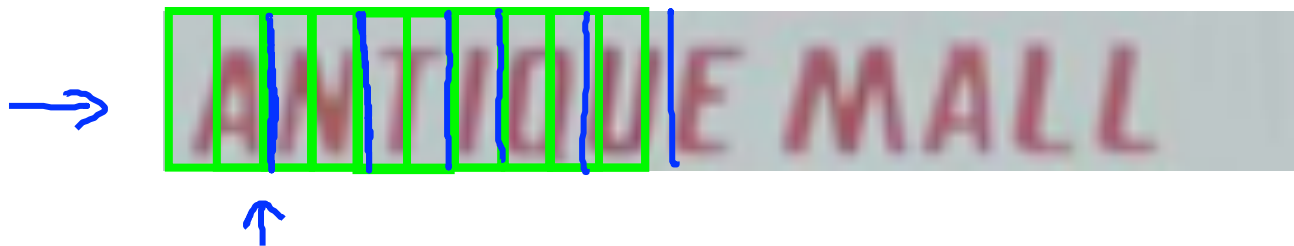
Text detection



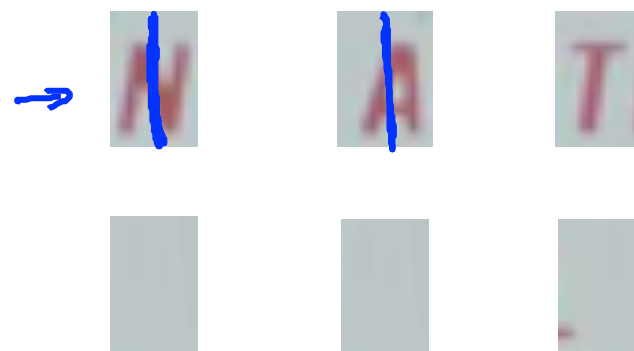
"expansion"



1D Sliding window for character segmentation



Positive examples ($y = 1$)



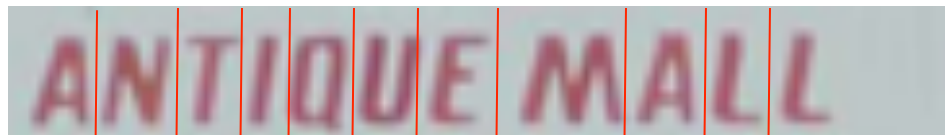
Negative examples ($y = 0$)

Photo OCR pipeline

→ 1. Text detection

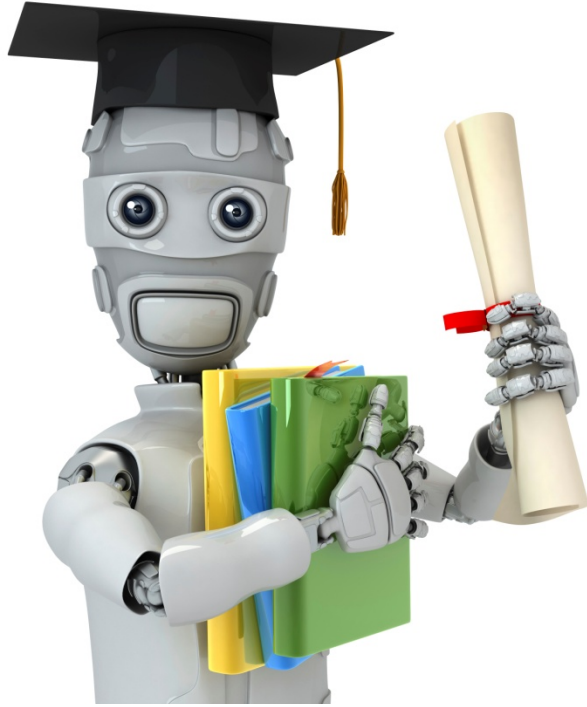


→ 2. Character segmentation



→ 3. Character classification



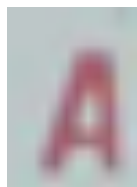


Machine Learning

Application example: Photo OCR

Getting lots of data: Artificial data synthesis

Character recognition



→ **A**



→ **N**



→ **T**



→ **I**



→ **Q**



→ **A**

Artificial data synthesis for photo OCR



Real data

Abcdefg
Abcdefg
Abcdefg
Abcdefg
Abcdefg
Abcdefg

Artificial data synthesis for photo OCR

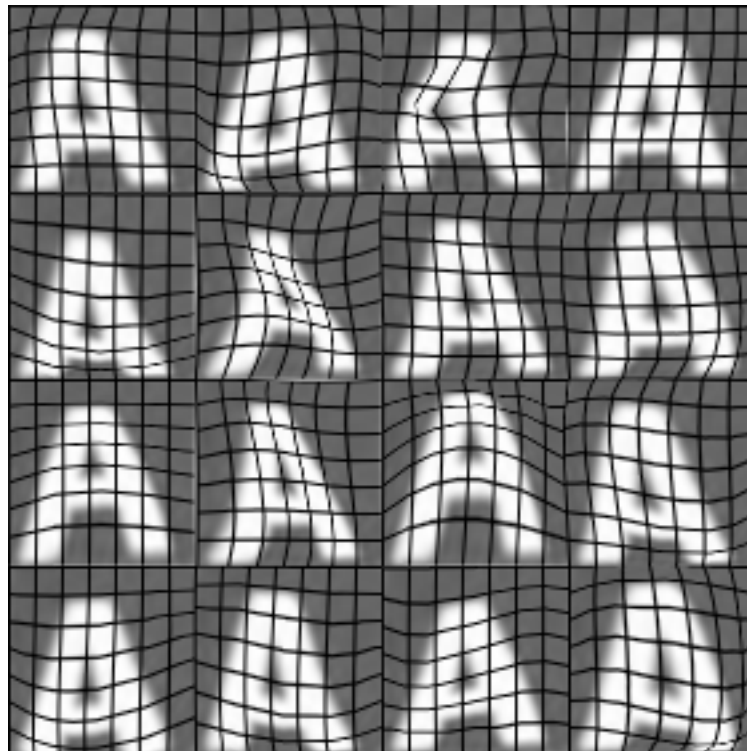
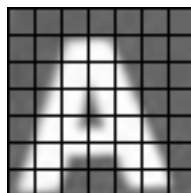


Real data



Synthetic data

Synthesizing data by introducing distortions



Synthesizing data by introducing distortions: Speech recognition



Original audio: 



Audio on bad cellphone connection



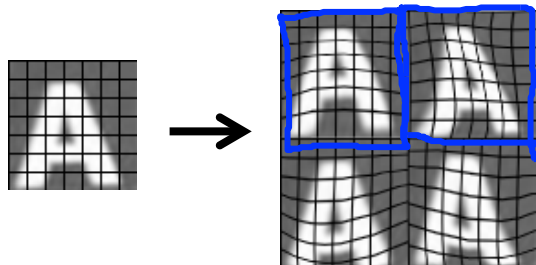
Noisy background: Crowd



Noisy background: Machinery

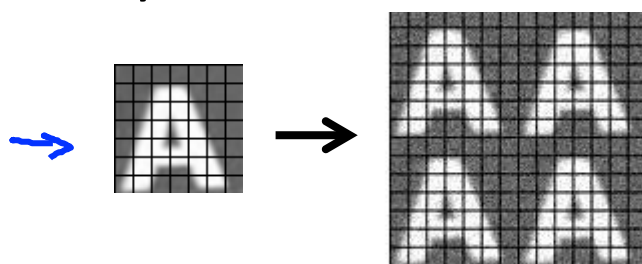
Synthesizing data by introducing distortions

- Distortion introduced should be representation of the type of noise/distortions in the test set.



- Audio:
Background noise,
bad cellphone connection

- Usually does not help to add purely random/meaningless noise to your data.



- x_i = intensity (brightness) of pixel i
- $x_i \leftarrow x_i + \text{random noise}$

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. “How much work would it be to get 10x as much data as we currently have?”

- Artificial data synthesis
- Collect/label it yourself
- “Crowd source” (E.g. Amazon Mechanical Turk)

→ #hours? $n = 1,000$
→ 10 secs/example $n = 10,000$ ←

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. “How much work would it be to get 10x as much data as we currently have?”
 - Artificial data synthesis
 - Collect/label it yourself
 - “Crowd source” (E.g. Amazon Mechanical Turk)

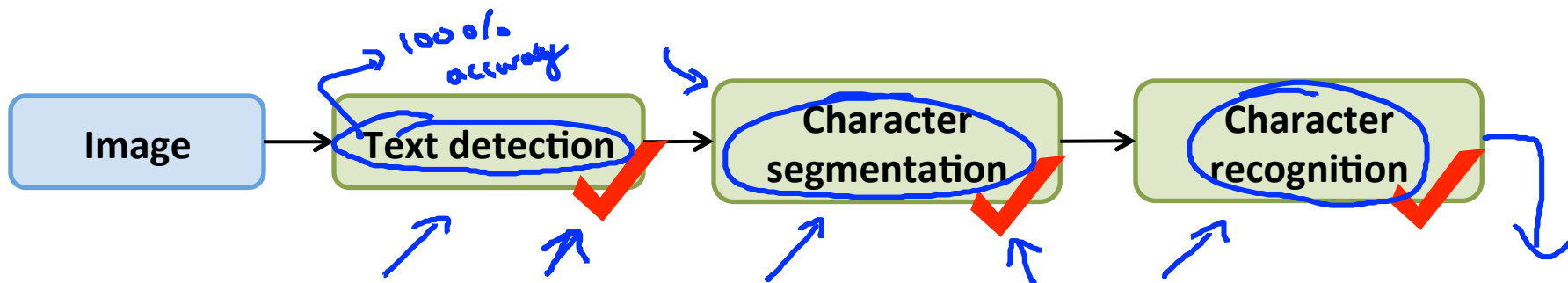


Machine Learning

Application example: Photo OCR

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

Estimating the errors due to each component (ceiling analysis)



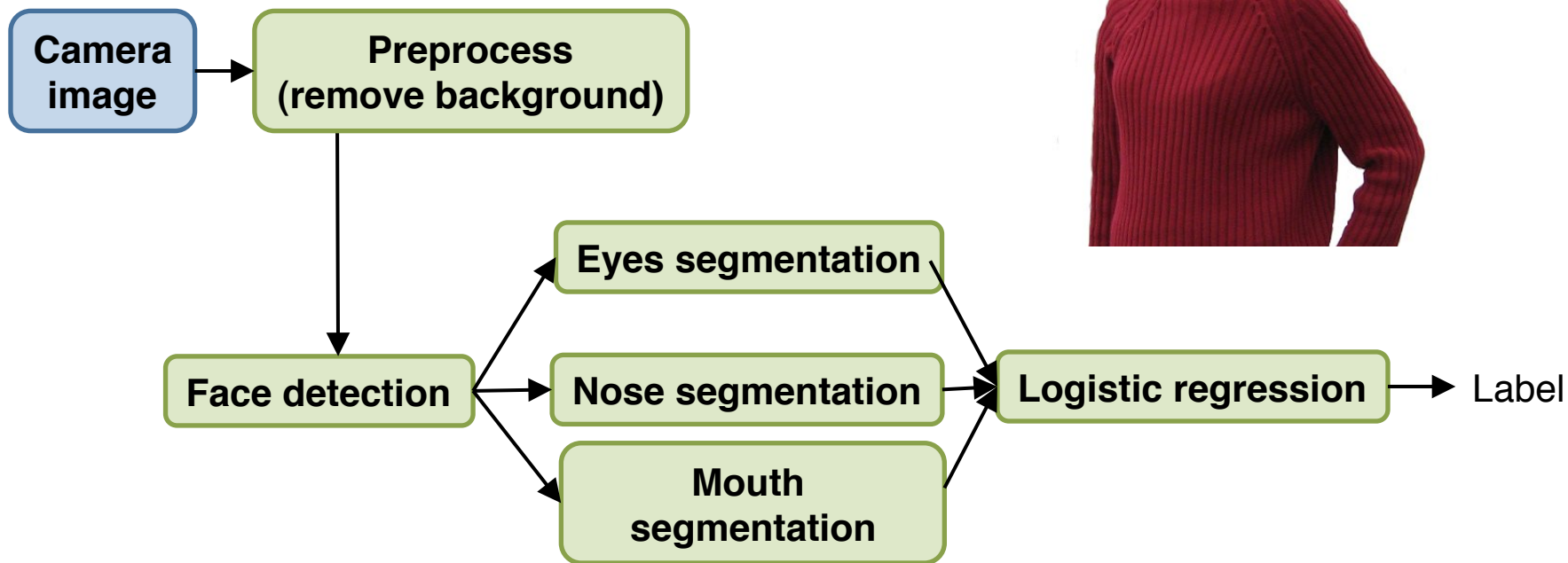
What part of the pipeline should you spend the most time trying to improve?

Component	Accuracy
Overall system	72%
→ Text detection	89%
Character segmentation	<u>90%</u>
Character recognition	100%

Handwritten annotations on the table: Blue arrows point to the 'Accuracy' header and the 'Overall system' row. Blue arrows also point to the 'Text detection' and 'Character segmentation' rows. Blue arrows point to the accuracy values 72%, 89%, 90%, and 100%. Blue arrows point to the differences between the overall system accuracy and the component accuracies: 17% (from 72% to 89%), 1% (from 89% to 90%), and 10% (from 90% to 100%).

Another ceiling analysis example

Face recognition from images
(Artificial example)



Another ceiling analysis example

