

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

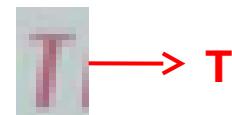
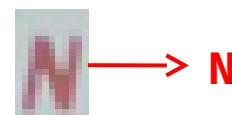
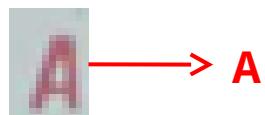
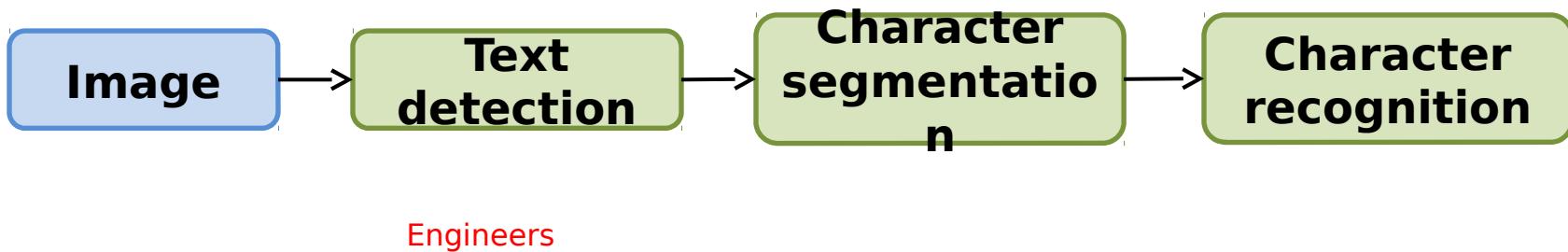
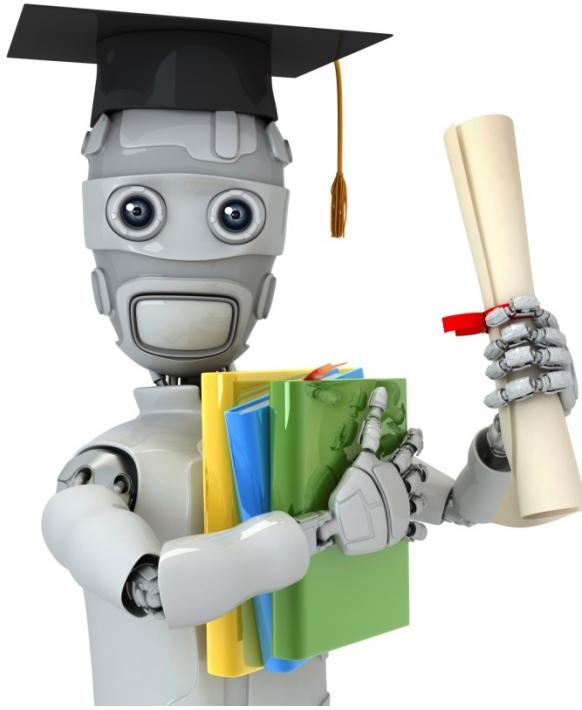


Photo OCR pipeline





Machine Learning

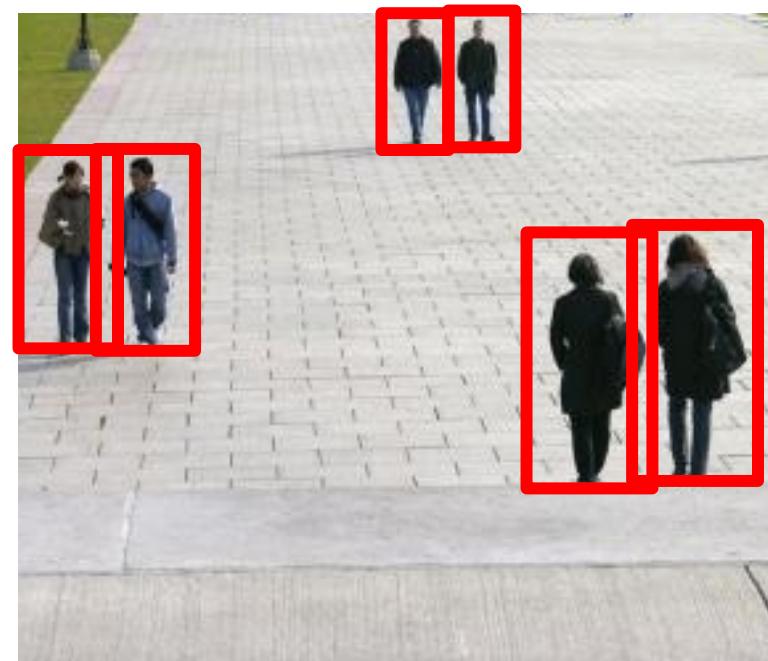
Application example: ~~Photo OCR~~ Sliding windows

Text detection



Aspect ratio of rectangles is different

Pedestrian detection



Aspect ratio of rectangles is same

Supervised learning for pedestrian detection

x = pixels in 82x36 image patches



Positive
examples

($y = 1$)



Negative
examples

($y = 0$)

Sliding window detection



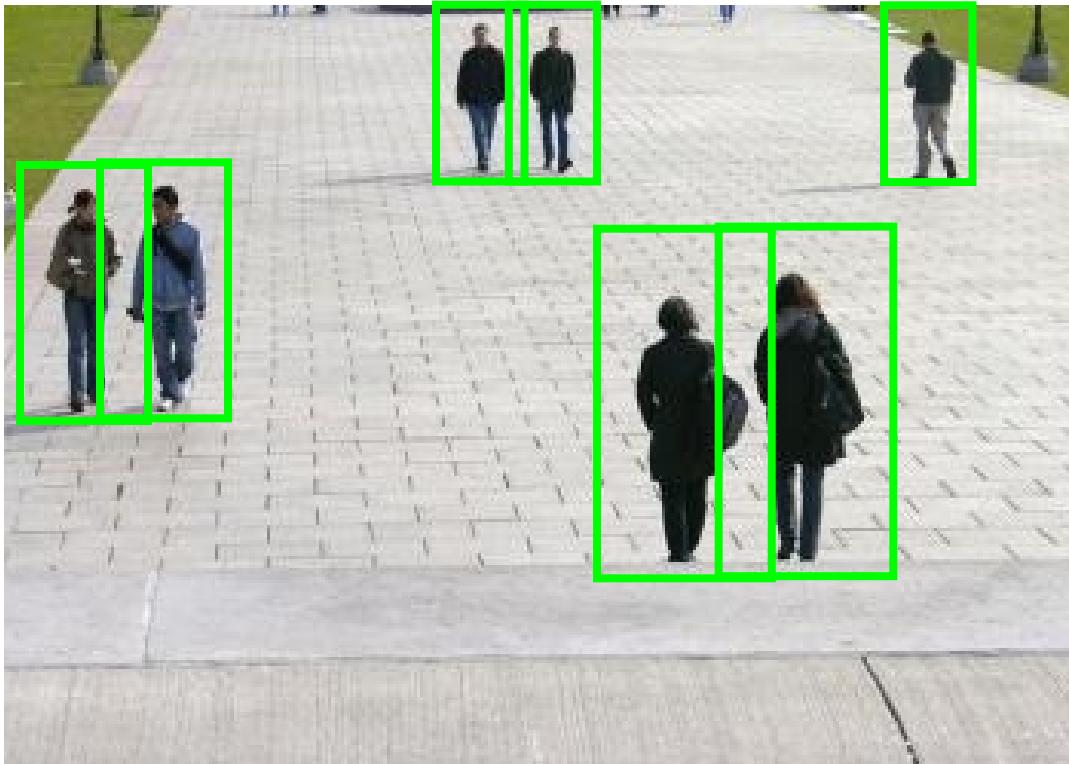
Sliding window detection



Sliding window detection



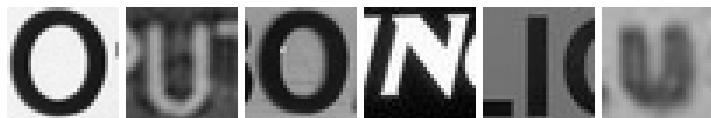
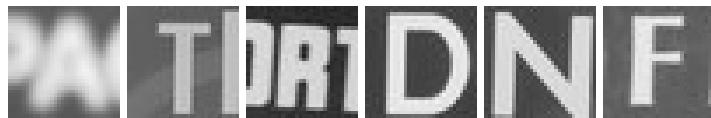
Sliding window detection



Text detection



Text detection



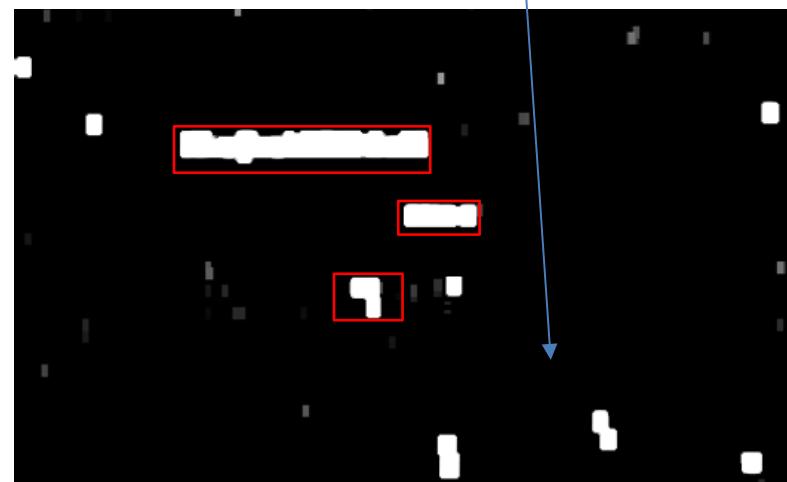
Positive examples $(y = 1)$



Negative examples $(y = 0)$

Text detection

We can discard boxes who's height and width almost same as text should have more width as compared to the height. But it might lead to skipping some text.



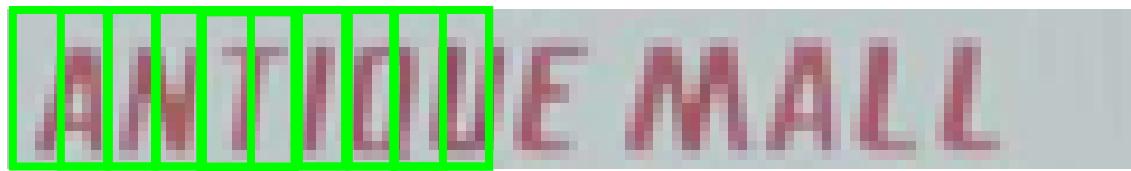
Suppose you are running a text detector using 20x20 image patches. You run the classifier on a 200x200 image and when using sliding window, you “step” the detector by 4 pixels each time. (For this problem assume you apply the algorithm at only one scale.) About how many times will you end up running your classifier on a single image? (Pick the closest answer.)

- About 100 times.
- About 400 times.
- About 2,500 times.

Correct Response

- About 40,000 times.

1D Sliding window for character segmentation



Positive
examples

$$(y = 1)$$

Negative
examples

$$(y = 0)$$

Photo OCR pipeline

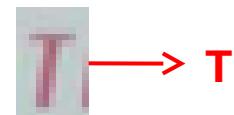
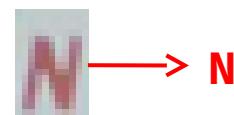
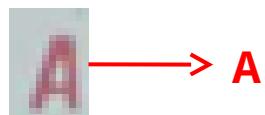
1. Text detection

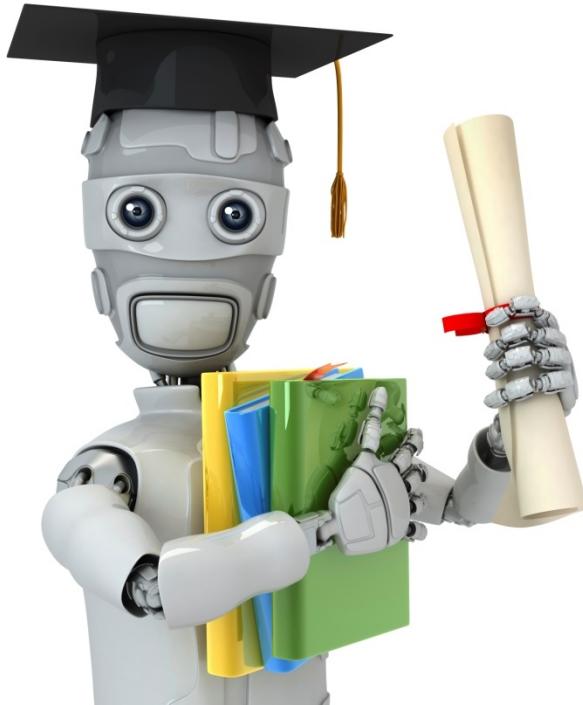


2. Character segmentation



3. Character classification





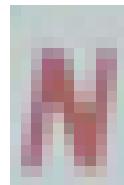
Machine Learning

Application
example:
Photo OCR
Getting lots of
data: Artificial
data synthesis

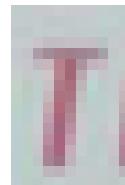
Character recognition



→ A



→ N



→ T



→ I



→ Q



→ A

Artificial data synthesis for photo OCR



Real data

Abcdefg
Abcdefg
Abcdefg
Abcdefg
Abcdefg
Abcdefg
Abcdefg

Artificial data synthesis for photo OCR

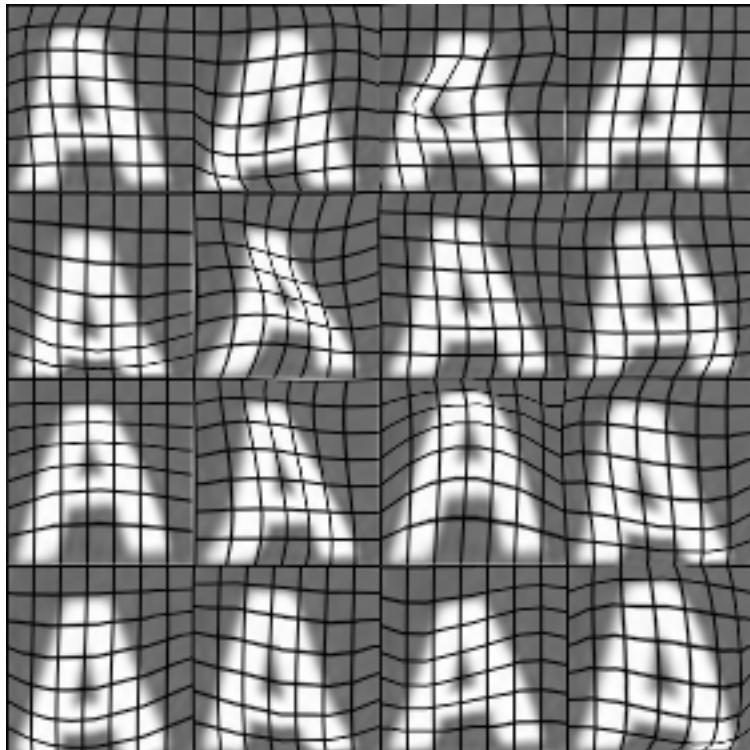
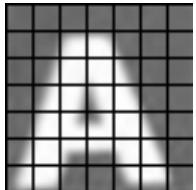


Real data



Synthetic data

Synthesizing data by introducing distortions



Synthesizing data by introducing distortions: Speech recognition



Original audio:



Audio on bad cellphone connection



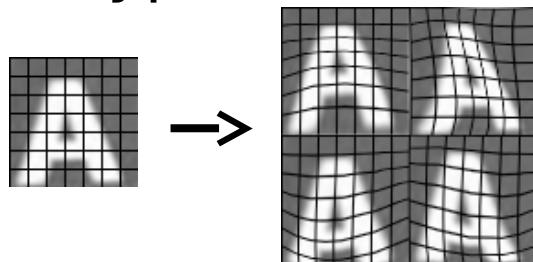
Noisy background: Crowd



Noisy background: Machinery

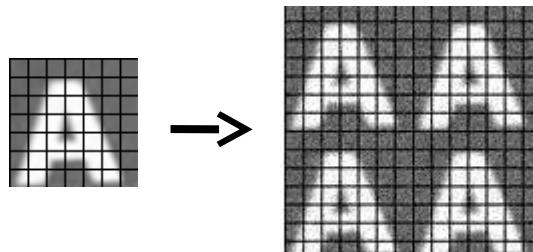
Synthesizing data by introducing distortions

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



Audio:
Background noise,
bad cellphone

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



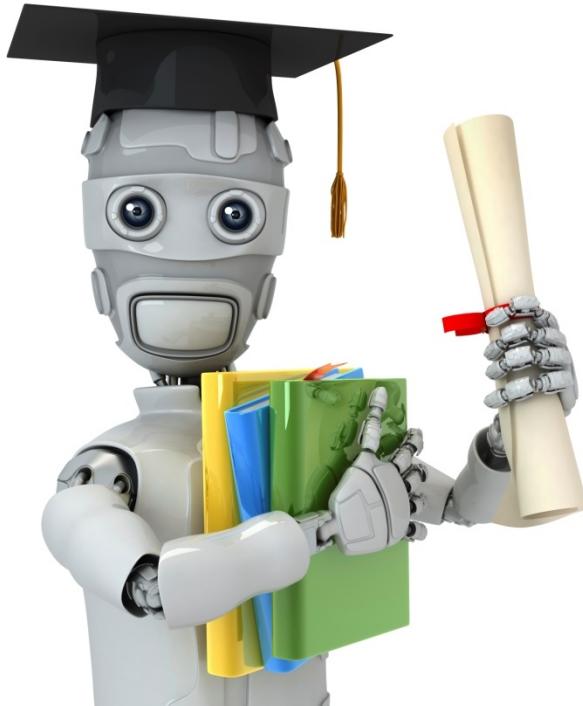
x_i = intensity (brightness) i
 $x_i \leftarrow \text{original pixel} + \text{random noise}$

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)

Discussion on getting more data

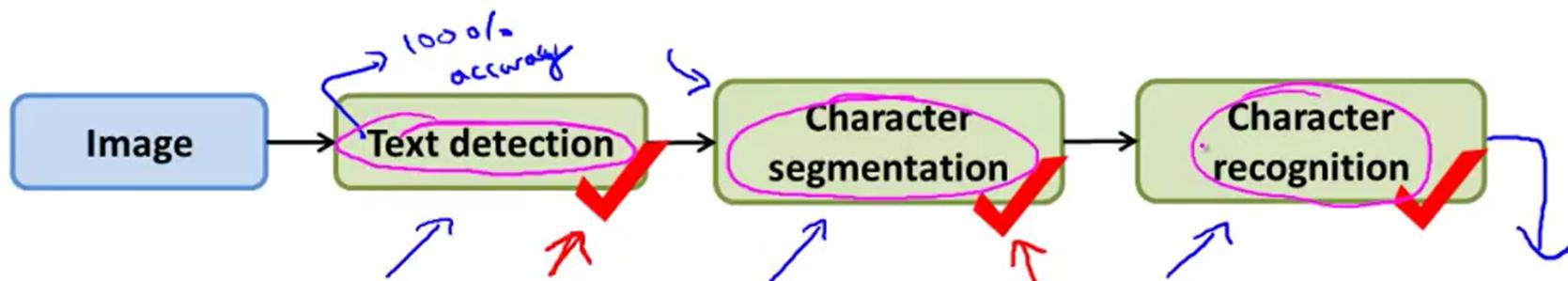
1. Make sure you have a low bias classifier before expending the effort. (Plot learning curves). E.g. keep increasing the number of features/number of hidden units in neural network until you have a low bias classifier.
2. “How much work would it be to get 10x as much data as we currently have?”
 - Artificial 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?

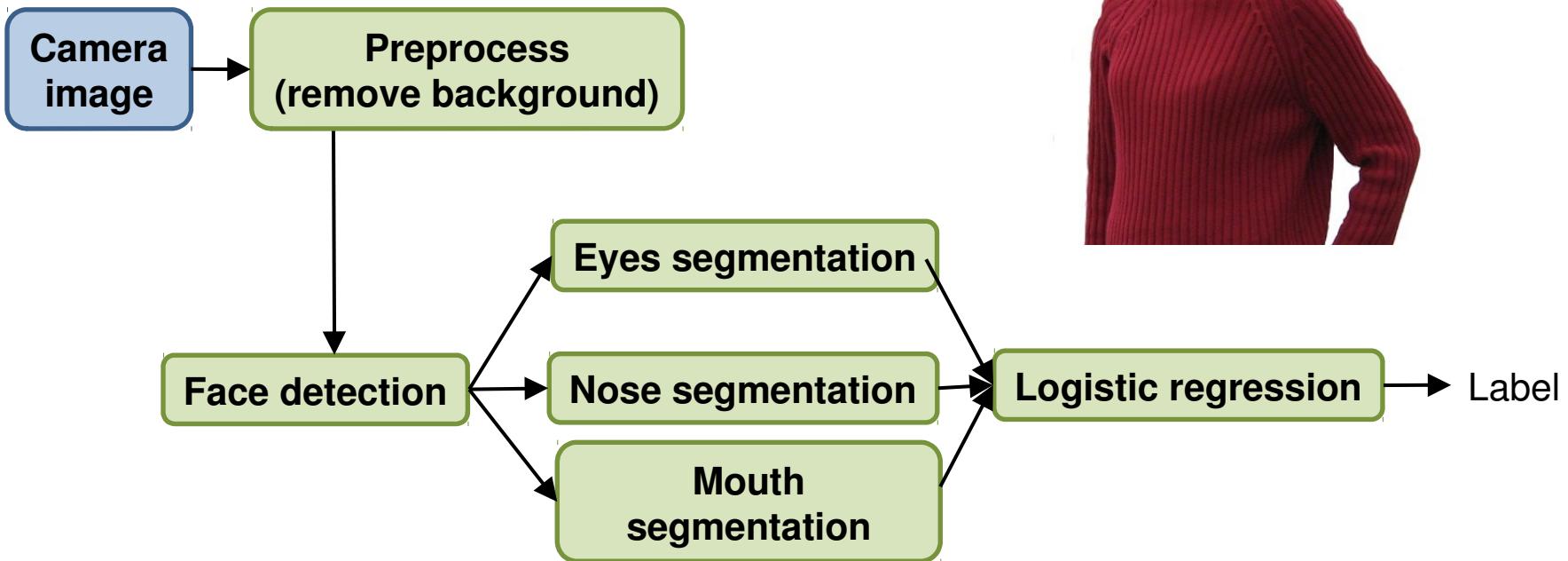
Component	Accuracy
Overall system	72%
→ Text detection	89%
Character segmentation	90%
Character recognition	100%

Annotations in blue:

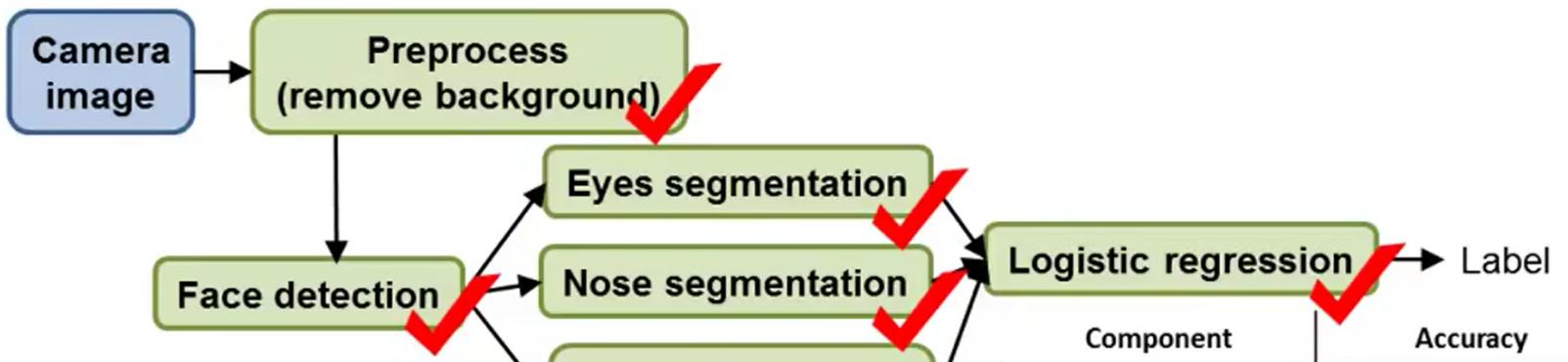
- A blue arrow points from the 'Overall system' row to the 'Text detection' row.
- A blue arrow points from the 'Text detection' row to the 'Character segmentation' row.
- A blue arrow points from the 'Character segmentation' row to the 'Character recognition' row.
- A blue arrow points from the 'Character recognition' row back to the 'Overall system' row.
- Handwritten blue text '17%' is written vertically next to the arrows between 'Text detection', 'Character segmentation', and 'Character recognition'.
- Handwritten blue text '1%' is written vertically next to the arrow between 'Character segmentation' and 'Character recognition'.
- Handwritten blue text '10%' is written vertically next to the arrow between 'Character recognition' and 'Overall system'.

Another ceiling analysis example

Face recognition from images
(Artificial example)



Another ceiling analysis example



Component	Accuracy
Overall system	85%
Preprocess (remove background)	85.1%
→ Face detection	91%
→ Eyes segmentation	95%
Nose segmentation	96%
Mouth segmentation	97%
→ Logistic regression	100%

Andrew Ng

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: (check all that apply)

- We should dedicate significant effort to collecting more data for that component.

Correct Response

- It is probably not worth dedicating engineering resources to improving that component of the system.

Correct Response

- 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.

Correct Response

- Choosing more features for that component may help (reducing bias), and reducing the number of features for that component (reducing variance) is unlikely to do so.

Correct Response

Summary: Main topics

→ Supervised Learning

$(x^{(i)}, y^{(i)})$

- Linear regression, logistic regression, neural networks, SVMs

→ Unsupervised Learning

$x^{(i)}$

- K-means, PCA, Anomaly detection

→ Special applications/special topics

- Recommender systems, large scale machine learning.

→ Advice on building a machine learning system

- Bias/variance, regularization; deciding what to work on next: evaluation of learning algorithms, learning curves, error analysis, ceiling analysis.