deeplearning.ai

Error Analysis

Carrying out error analysis

Look at dev examples to evaluate ideas

90% accuracy

→ 10% error

Should you try to make your cat classifier do better on dogs?

Error analysis: → 5-10 min

- Get ~100 mislabeled dev set examples.
- Count up how many are dogs.

"ceiling"

→ 5% 10% → 9.5%

5/100

→ 50% 10% → 5%

50/100

Andrew Ng

Evaluate multiple ideas in parallel

Ideas for cat detection:

- Fix pictures of dogs being recognized as cats
- Fix great cats (lions, panthers, etc..) being misrecognized
- Improve performance on blurry images

Image	Dog	Great Cats	Blurry	Instagram	Comments
1	✓			✓	Pitbull
2			✓	✓	
3		✓	✓		Rainy day at zoo
⋮	⋮	⋮	⋮		
% of total	8%	43%	61%	12%	

Andrew Ng

deeplearning.ai

Error Analysis

Cleaning up Incorrectly labeled data

Incorrectly labeled examples

x

y 1 0 1 1 0 1 1

Training set.

DL algorithms are quite robust to random errors in the training set.

Systematic errors

Error analysis

Image	Dog	Great Cat	Blurry	Incorrectly labeled	Comments
…					
98				✓	Labeler missed cat in background
99		✓			
100				✓	Drawing of a cat; Not a real cat.
% of total	8%	43%	61%	6%	

Overall dev set error .... 10% | 2%

Errors due incorrect labels .... 0.6% | 0.6%

Errors due to other causes .... 9.4% | 1.4%

2.1% 1.9%

Goal of dev set is to help you select between two classifiers A & B.

Andrew Ng

Correcting incorrect dev/test set examples

- Apply same process to your dev and test sets to make sure they continue to come from the same distribution

- Consider examining examples your algorithm got
98% right as well as ones it got wrong. 2%

- Train and $\boxed{\text{dev/test}}$ data may now come from slightly different distributions.

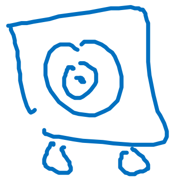
Andrew Ng

deeplearning.ai

Error Analysis

Build your first system quickly, then iterate

Speech recognition example



- Noisy background
  - Café noise
  - Car noise
- Accent
- Far fro
- Young
- Stutte
- …

Guideline:

Build your first system quickly, then iterate

- Set up dev/test set and metric
- Build initial system quickly
- Use Bias/Variance analysis & Error analysis to prioritize next steps.

deeplearning.ai

Mismatched training and dev/test data

Training and testing on different distributions

Cat app example

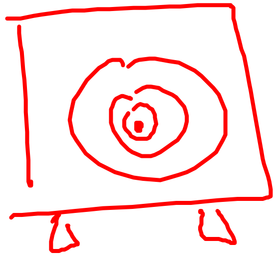
care about this

Data from webpages

Data from mobile app

→ ≈ 200,000 → 210,000 ← → ≈ 10,000

↓ shuffle



~~Option 1:~~

train	dev	test

205,000 | 2,500 | 2,500

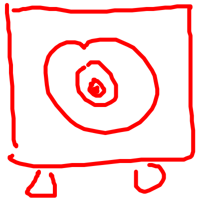
$\frac{200K}{210K}$

2381 – web
119 – mobile app

Option 2:

web	dev	test

train: 205,000
app | app 2500 | app 2500



Andrew Ng

Speech recognition example

Speech activated rearview mirror

Training

Purchased data x, y

Smart speaker control

Voice keyboard

…

500,000 utterances

Dev/test

Speech activated rearview mirror

→ 20,000

10K 5K 5K

train D T

500K

D T

510K

10K mirror

5K 5K

deeplearning.ai

Mismatched training and dev/test data

Bias and Variance with mismatched data distributions

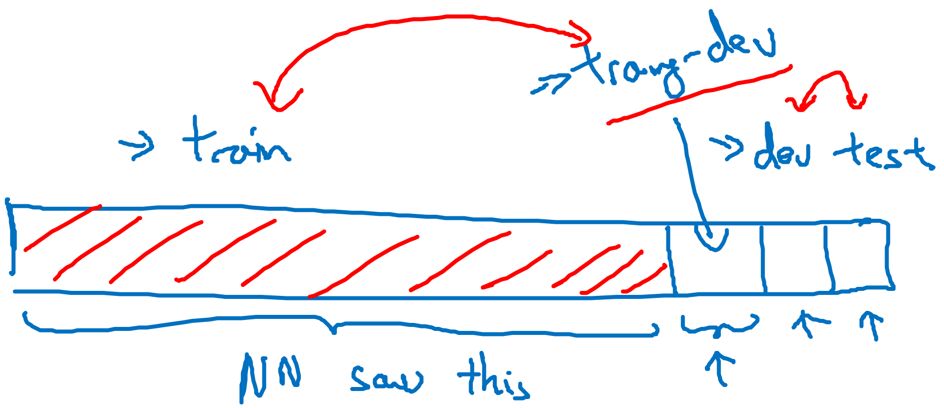
Cat classifier example

Assume humans get $\approx 0\%$ error.

Training error 1%

Dev error 10%

9%



Training-dev set: Same distribution as training set, but not used for training

Training error	1%	variance	1%	
→ Train-dev error	9%		1.5%	data mismatch
→ Dev error	10%		10%	
	Variance			

Human error ...	0%	Avoidable bias		Avoidable bias
Training error	10%		10%	variance
Train-dev error	11%		11%	Data mismatch
Dev error	12%		20%	
	Bias		Bias + Data mismatch	

Bias/variance on mismatched training and dev/test sets

Human level	4%	4%
Training set error	7%	7%
Training-dev set error	10%	10%
→ Dev error	12%	6%
→ Test error	12%	6%

Human level ↔ Training set error: avoidable bias

Training set error ↔ Training-dev set error: variance

Training-dev set error ↔ Dev error: data mismatch

Dev error ↔ Test error: degree of overfitting to dev set.

More general formulation

Rearview mirror

	General speech recognition	Rearview mirror speech data
Human level	"Human level" 4%	6%
Error on examples trained on	"Training error" 7%	6%
Error on example not trained on	"Training-dev error" 10%	"Dev/Test error" 6%

Avoidable bias

Variance

data mismatch

deeplearning.ai

Mismatched training and dev/test data

Addressing data mismatch

Addressing data mismatch

- Carry out manual error analysis to try to understand difference between training and dev/test sets

E.g. noisy – car noise   street numbers

- Make training data more similar; or collect more data similar to dev/test sets

E.g. Simulate noisy in-car data

Artificial data synthesis

+

=

"The quick brown fox jumps over the lazy dog."

10,000 hours

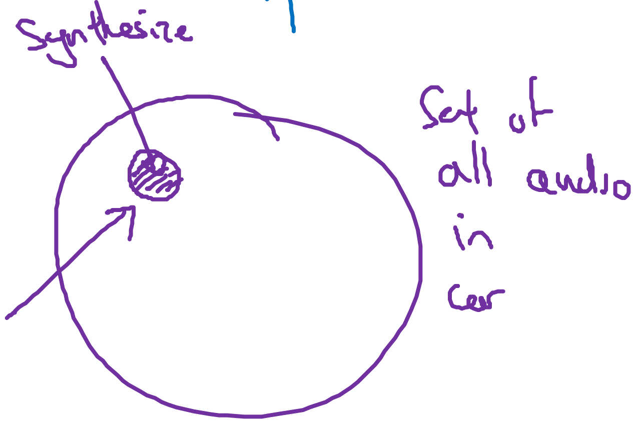
Car noise

1 hour of car noise

Overfit to 1 hour of car noise

10,000 hours

Synthesized in-car audio

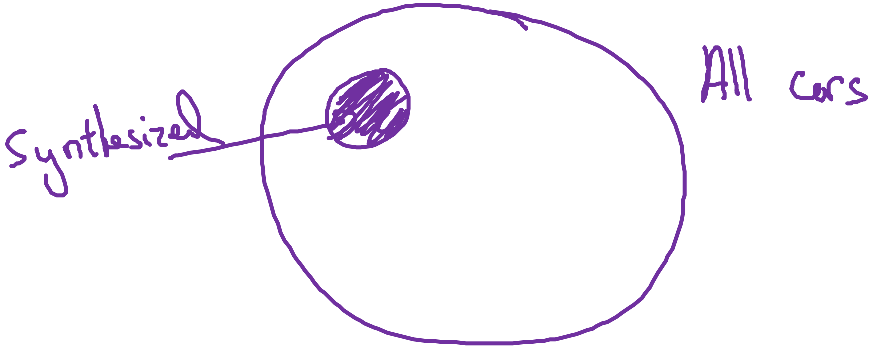


Andrew Ng

Artificial data synthesis

Car recognition:

$\approx$ 20 cars



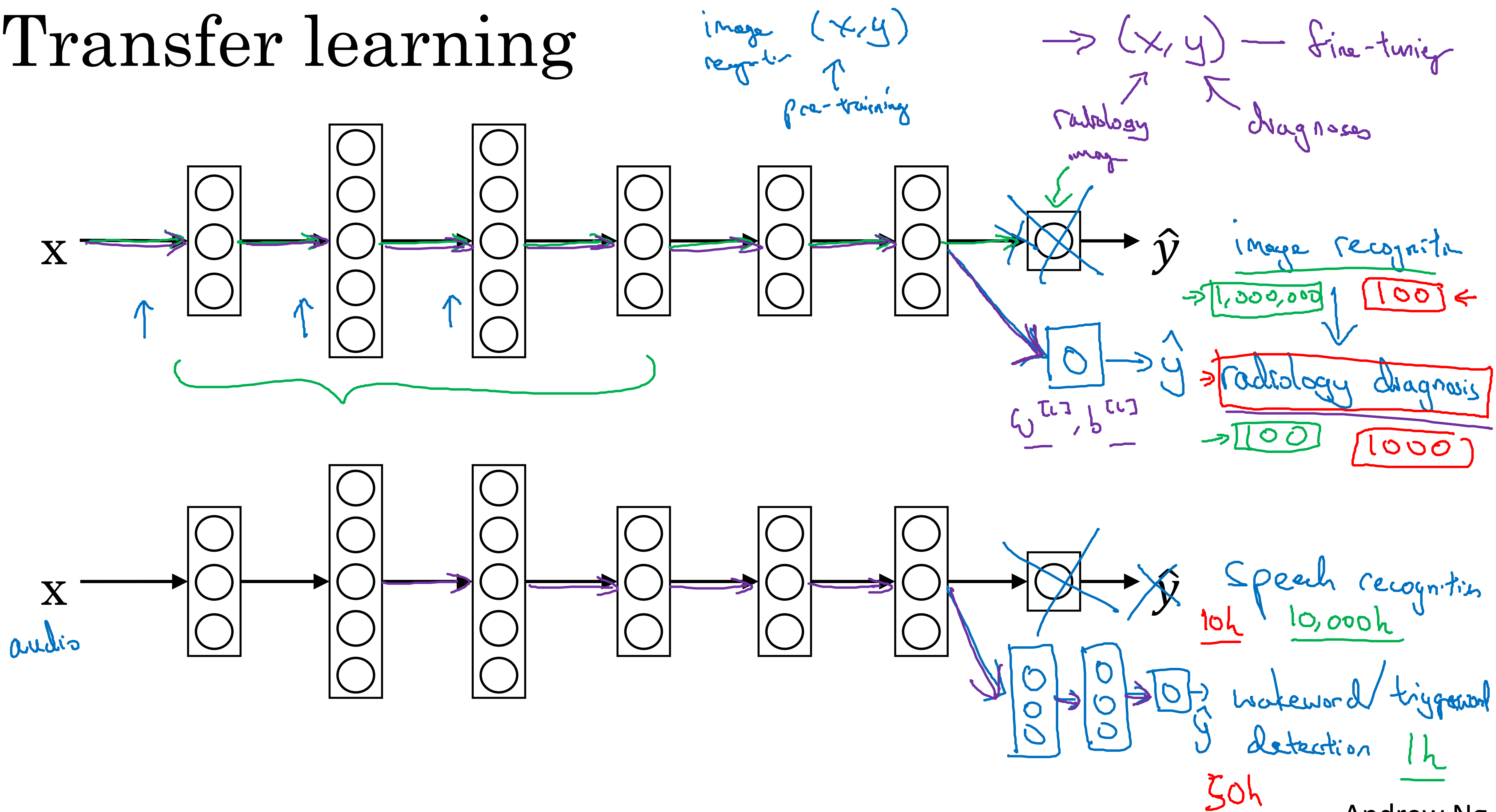
Andrew Ng

deeplearning.ai

Learning from multiple tasks

Transfer learning

Transfer learning



Andrew Ng

When transfer learning makes sense

Transfer from A $\rightarrow$ B

- Task A and B have the same input x.
- You have a lot more data for Task A than Task B.
- Low level features from A could be helpful for learning B.

deeplearning.ai

Learning from multiple tasks

Multi-task learning

Simplified autonomous driving example

$x^{(i)}$

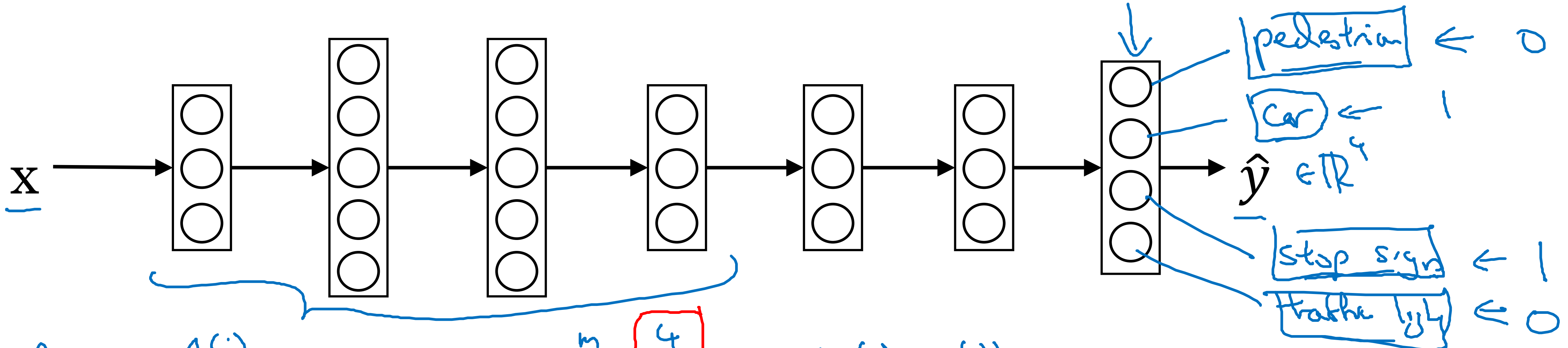
$$y^{(i)} = \begin{array}{l} \text{Pedestrians} \\ \text{Cars} \\ \text{Stop signs} \\ \text{Traffic lights} \\ \vdots \end{array} \left.\begin{array}{c} 0 \\ 1 \\ 1 \\ 0 \\ \vdots \end{array}\right\} \quad (4,1)$$

$$Y = \begin{bmatrix} | & | & | & & | \\ y^{(1)} & y^{(2)} & y^{(3)}, & \cdots, & y^{(m)} \\ | & | & | & & | \end{bmatrix}$$

$(4, m)$

Andrew Ng

Neural network architecture



Loss: $\underset{(4,1)}{\hat{y}^{(i)}}$ $\rightarrow \frac{1}{m}\sum_{i=1}^{m}\boxed{\sum_{j=1}^{4}}\mathcal{L}(\hat{y}_j^{(i)}, y_j^{(i)})$

Sum only over values of j with 0/1 label.

Usual logistic loss

$-y_j^{(i)}\log \hat{y}_j^{(i)} - (1-y_j^{(i)})\log(1-\hat{y}_j^{(i)})$

Unlike softmax regression:

One image can have multiple labels

Multi-task learning ←

$$Y = \begin{bmatrix} 1 & \boxed{1} & \cdots & \boxed{0} & \cdots & ? & \cdots \\ 0 & \boxed{1} & \cdots & \boxed{1} & \cdots & 1 & \cdots \\ \boxed{?} & ? & \cdots & \boxed{1} & \cdots & ? & \cdots \\ ? & ? & \cdots & \boxed{0} & \cdots & ? & \cdots \end{bmatrix} \leftarrow$$

Andrew Ng

When multi-task learning makes sense

- Training on a set of tasks that could benefit from having shared lower-level features.
- Usually: Amount of data you have for each task is quite similar.

A $\underline{1{,}000{,}000}$

$\downarrow$ $\downarrow$

B $\underline{1{,}000}$

A_1 1,000

A_2 1,000

$\vdots$ $\vdots$

99,000

$\boxed{A_{100}}$ $\underline{1{,}000}$

- Can train a big enough neural network to do well on all the tasks.

deeplearning.ai

End-to-end deep learning

What is end-to-end deep learning

What is end-to-end learning?

Speech recognition example

"cat"

x → y

→ audio $\xrightarrow{\text{MFCC}}$ features $\xrightarrow{\text{ML}}$ Phonemes → words → transcript

audio → phonemes → ... →

→ audio → transcript

3,000h

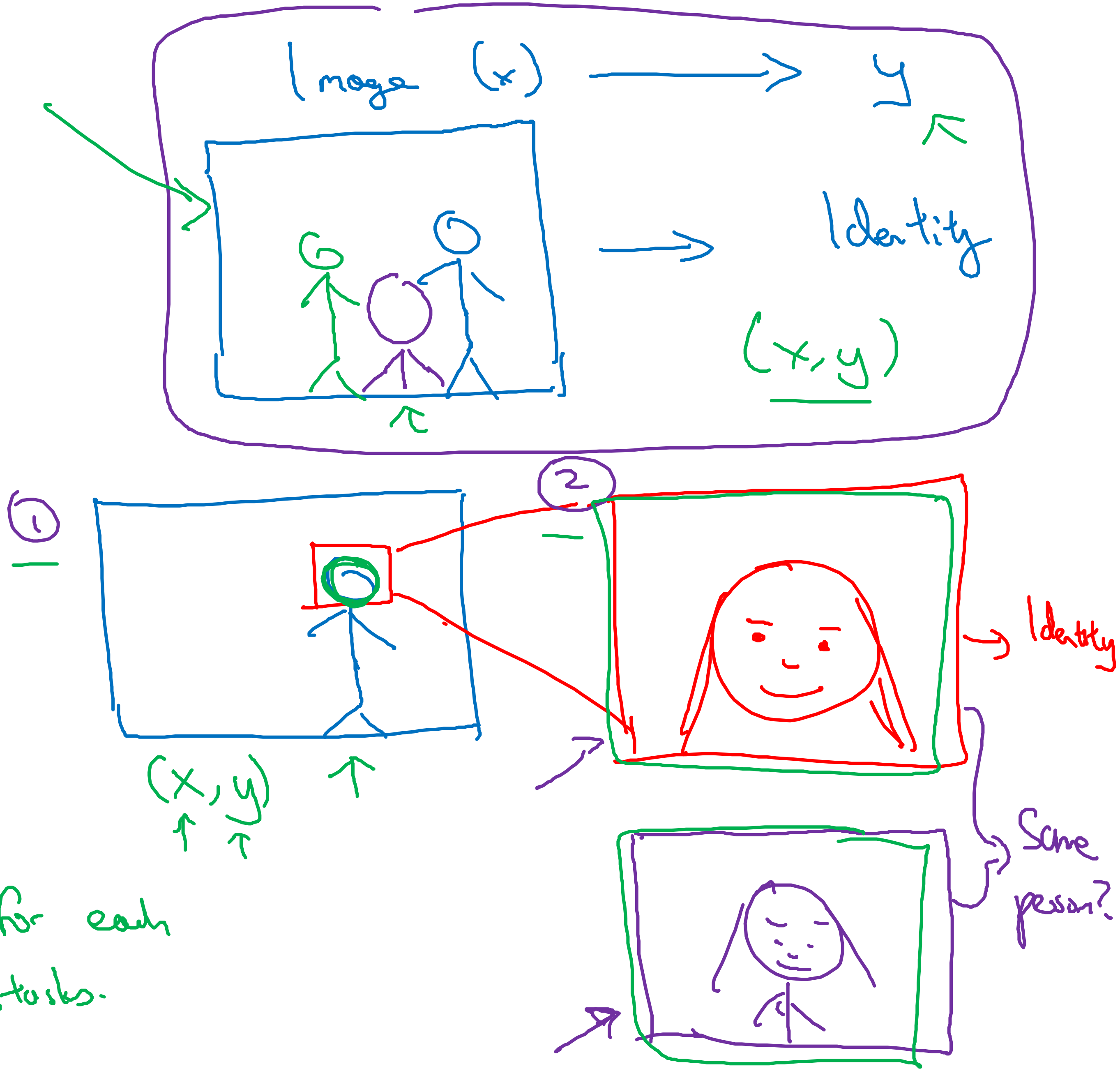
10,000h

⋮

100,000h

Face recognition

[Image courtesy of Baidu]



Have data for each of 2 subtasks.

More examples

Machine translation

(x, y)

Eglsh French

English → text analysis → ... → French

English → French

Estimating child's age:

Image $\xrightarrow{1}$ bones $\xrightarrow{2}$ age

Image → age ←

deeplearning.ai

End-to-end deep learning

Whether to use end-to-end learning

Pros and cons of end-to-end deep learning

Pros:

- Let the data speak

  $X \longrightarrow y$

  → "phonemes"
  c a t

- Less hand-designing of components needed

$X - - - - \dashrightarrow y$

Cons:

input end

output end

- May need large amount of data

  $\underline{X \longrightarrow y}$ (x, y)

- Excludes potentially useful hand-designed components

  Data.

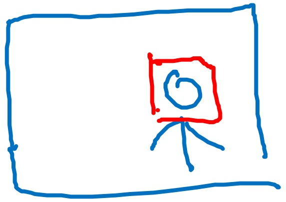
  Hand-design.

Applying end-to-end deep learning

Key question: Do you have sufficient data to learn a function of the complexity needed to map x to y?

$x \to y$

 → age



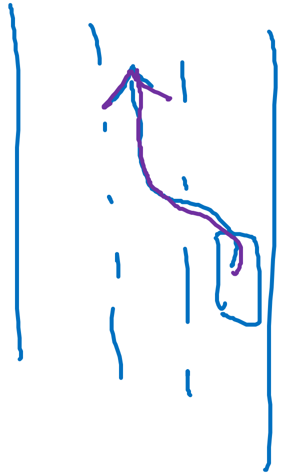


image $\xrightarrow{DL}$ cars → route $\xrightarrow[\text{control}]{}$ steering

radar

lidar

$\xrightarrow{DL}$ pedestrians

motion planning

• Use DL to learn individual components

• Carefully choose $X \to Y$ depending on what tasks you can get data for.

→ image → steering