



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.

↗ 5%  
5/100

10%  
↙  
95%

"ceiling"

↗ 50%  
50/100

10%  
↙  
5%

# 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	<u>8%</u>	<u>43%</u>	<u>61%</u>	<u>12%</u>	



deeplearning.ai

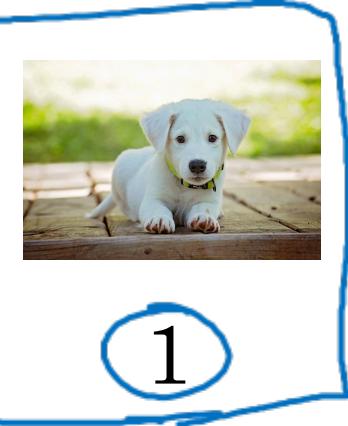
# Error Analysis

---

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	<u>8%</u>	<u>43%</u>	<u>61%</u>	<u>6%</u>	

Overall dev set error ..... 100% ← 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.

# 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 right as well as ones it got wrong.  
*(81%)*      *(19%)*
- Train and dev/test data may now come from slightly different distributions.



deeplearning.ai

# Error Analysis

---

Build your first system  
quickly, then iterate

# Speech recognition example



- • Noisy background
  - • Café noise
  - • Car noise
  - • Accent
  - • Far from
  - • Young
  - • Stutter
  - • ...
- 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

Data from webpages



care about this

Data from mobile app



$\rightarrow \approx 200,000$

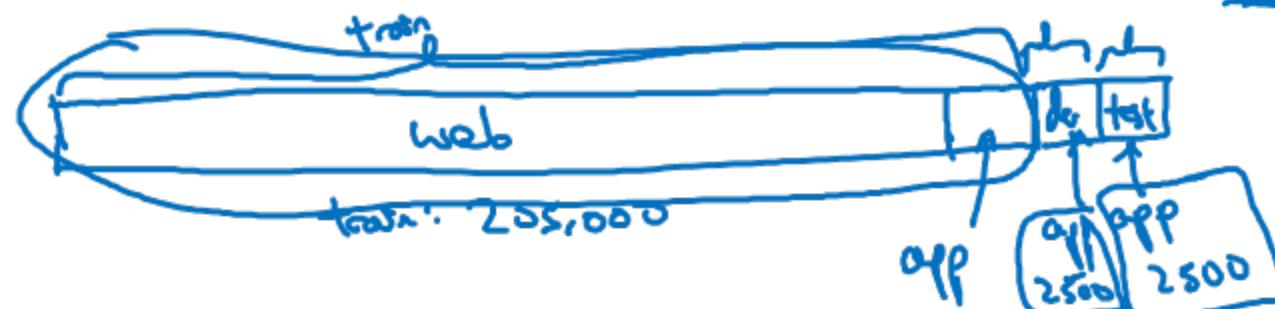
$210,000$   
(shuttle)

$\rightarrow \approx 10,000$

X Option 1:



Option 2:



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



2381 - web  
119 - mobile app



# Speech recognition example

Speech activated rearview mirror



## Training

- { Purchased data       $\downarrow \downarrow$   
 $x, y$
- Smart speaker control
- Voice keyboard

...  
500,000 utterances

## Dev/test

Speech activated  
rearview mirror }

$\rightarrow 20,000$

↓  
10K SK SK D T  
train 500K

↓  
510K  
10K mirr  
D T  
SK SK



deeplearning.ai

# Mismatched training and dev/test data

---

## Bias and Variance with mismatched data distributions

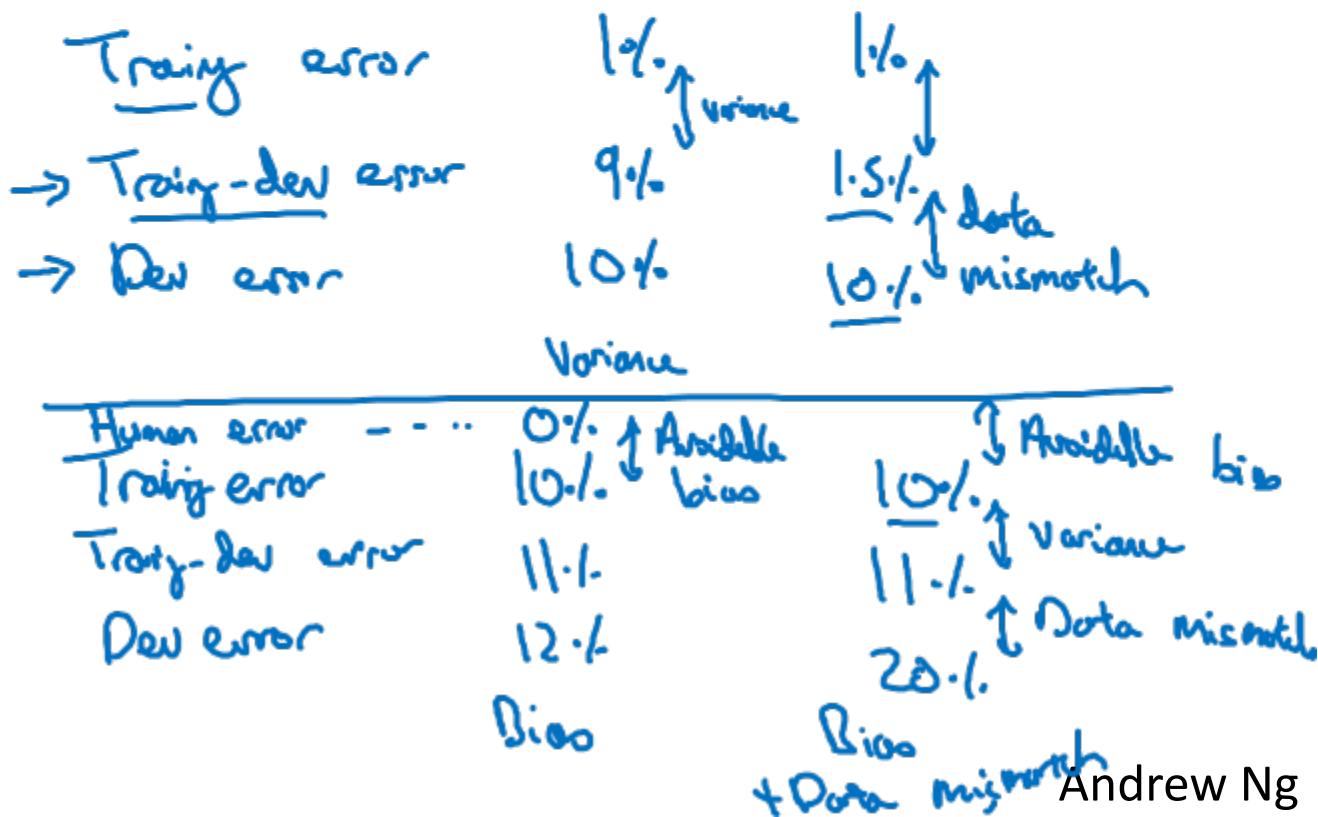
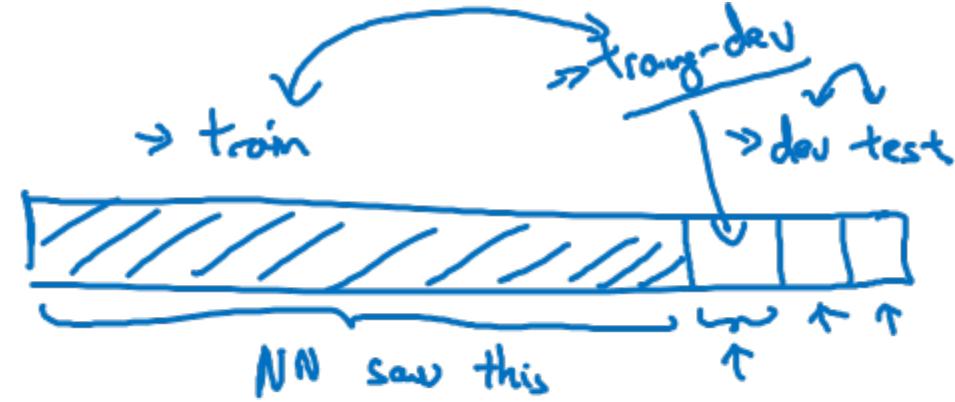
# Cat classifier example

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

Training error ..... 1%  $\downarrow$  9%

Dev error ..... 10%

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



# Bias/variance on mismatched training and dev/test sets

Human level

Training set error

Training - dev set error

→ Dev error

→ Test error

4% ↑ avoidable bias

7% ↑ variance

10%

12% } data mismatch

12% ↑ degree of similarity  
to dev set.

4%

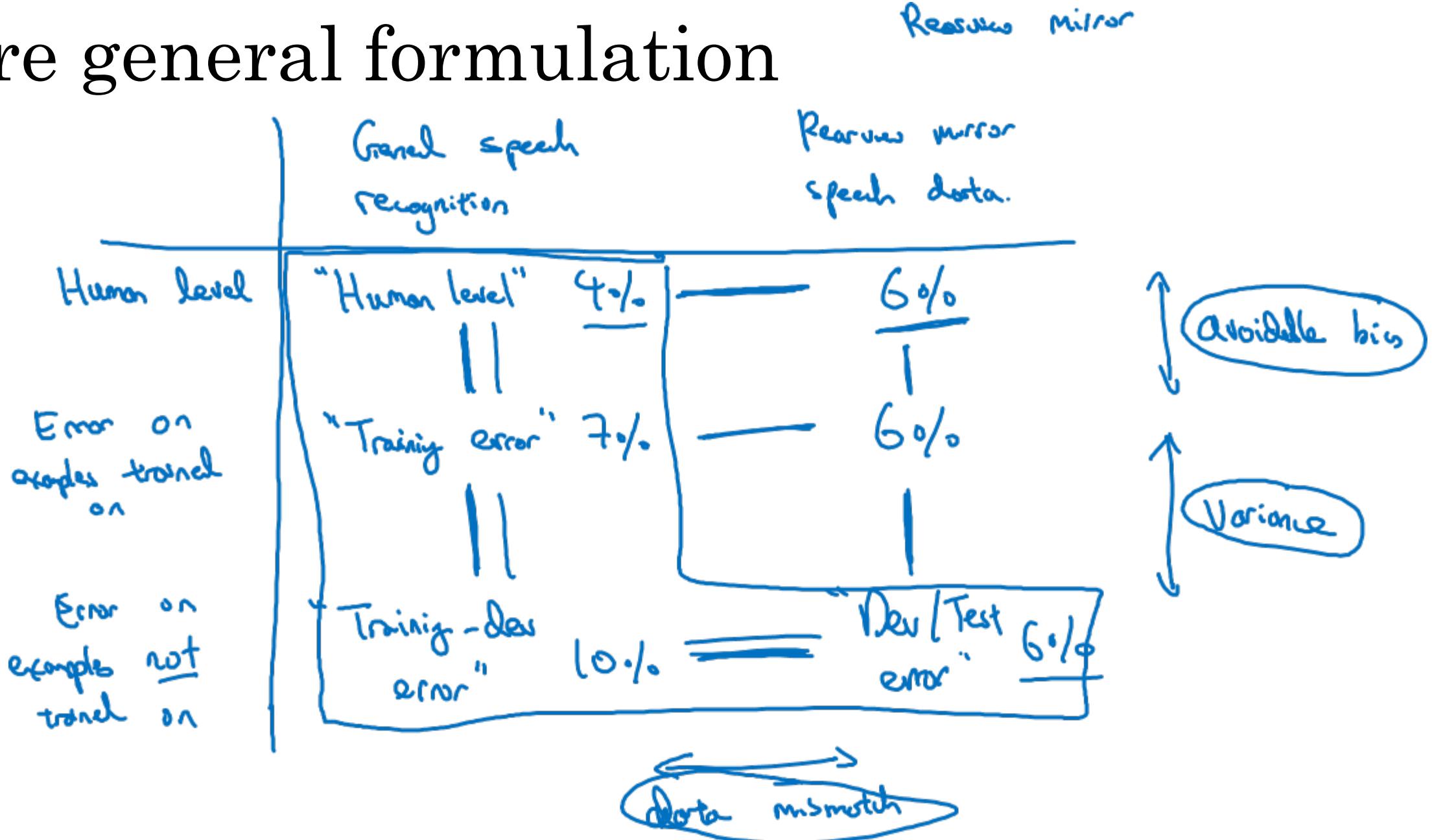
7% }

10%

6% }

6% }

# More general formulation





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

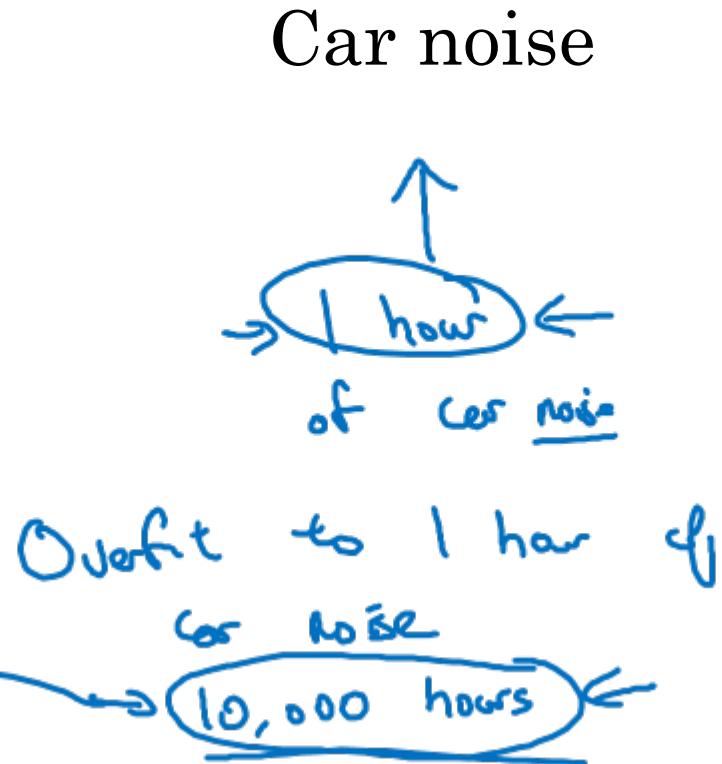
# Artificial data synthesis

+

=

“The quick brown  fox jumps  
over the lazy dog.”

  
10,000 hours



Synthesized  
in-car audio

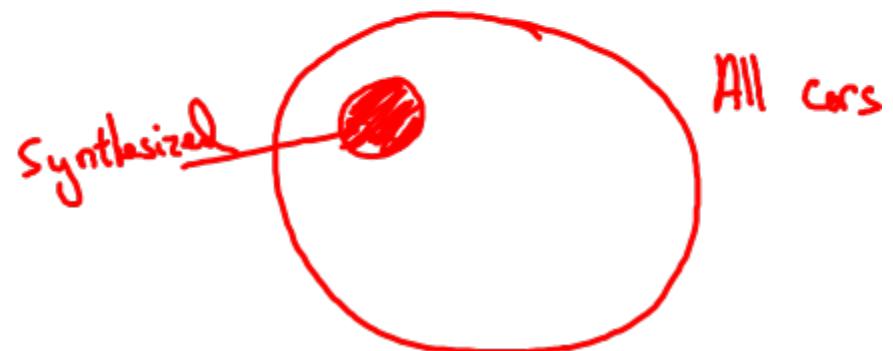


# Artificial data synthesis

Car recognition:



$N \approx 20$  cars





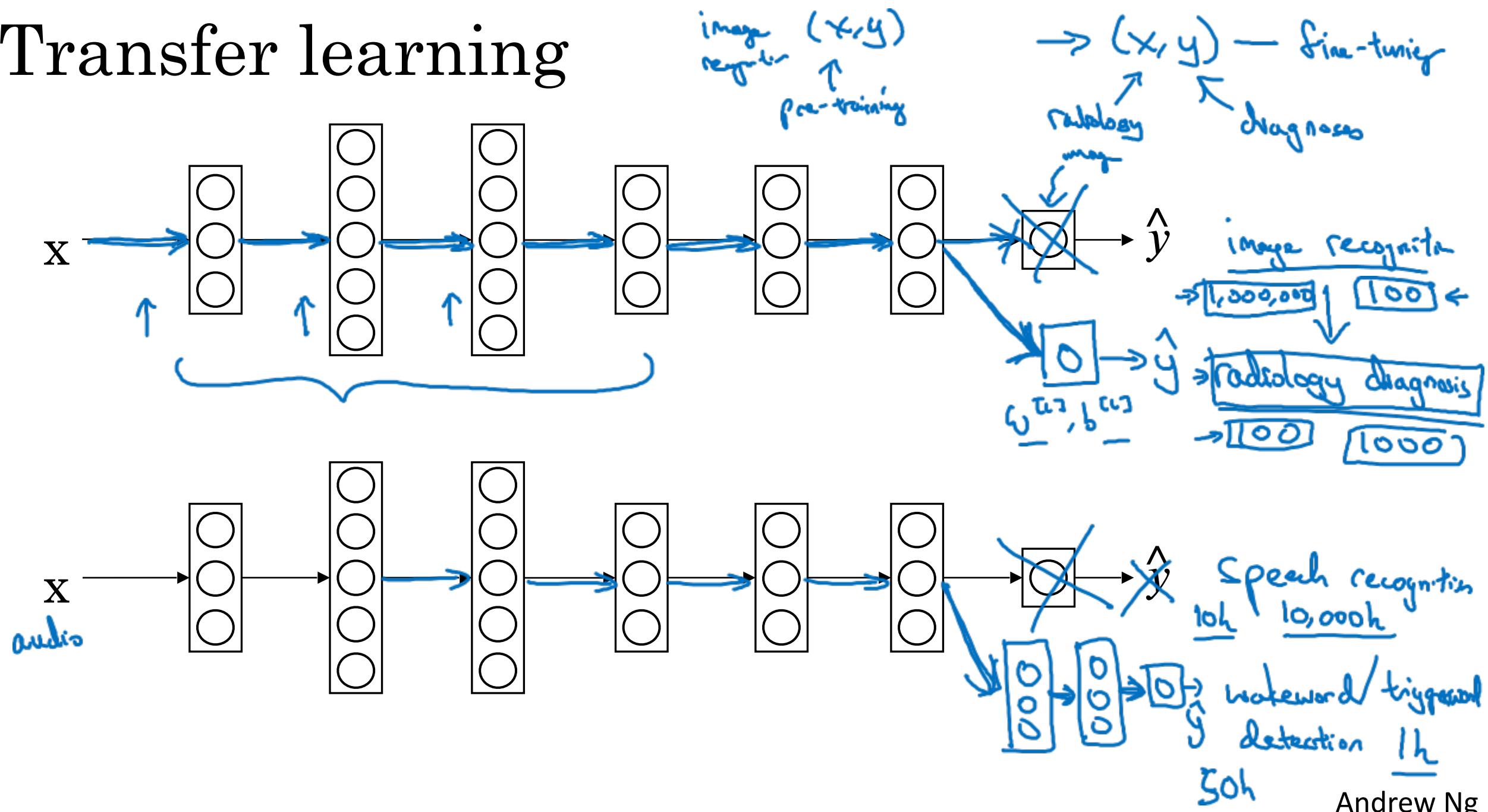
deeplearning.ai

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



$x^{(i)}$

Pedestrians

Cars

Stop signs

Traffic lights

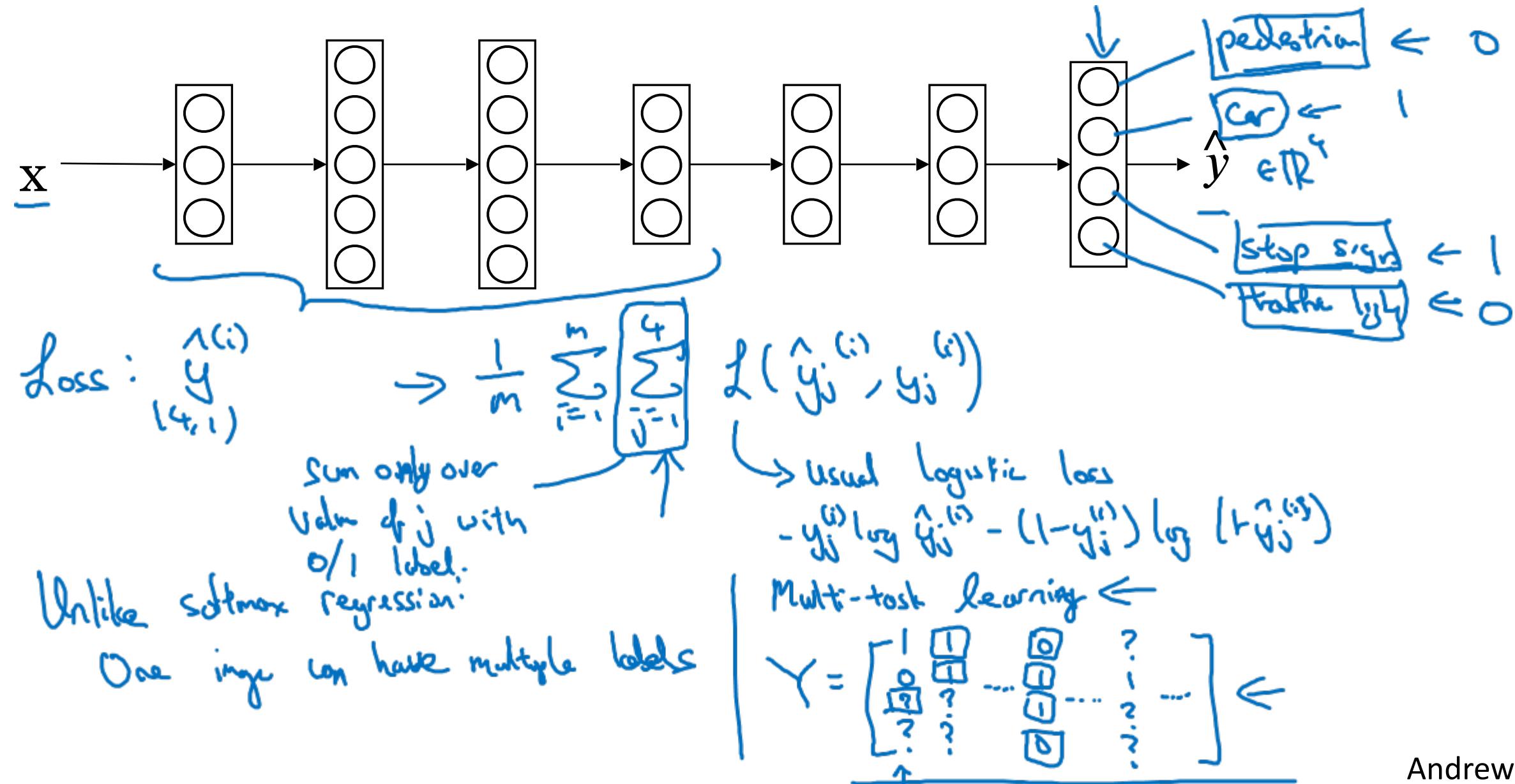
⋮

$y^{(i)}$	(4,1)
0	
1	
1	
0	
⋮	

$$Y = \begin{bmatrix} y_1^{(1)} & y_1^{(2)} & y_1^{(3)}, \dots, y_1^{(m)} \\ y_2^{(1)} & y_2^{(2)} & y_2^{(3)}, \dots, y_2^{(m)} \\ y_3^{(1)} & y_3^{(2)} & y_3^{(3)}, \dots, y_3^{(m)} \\ \vdots & \vdots & \vdots \end{bmatrix}$$

(4, m)

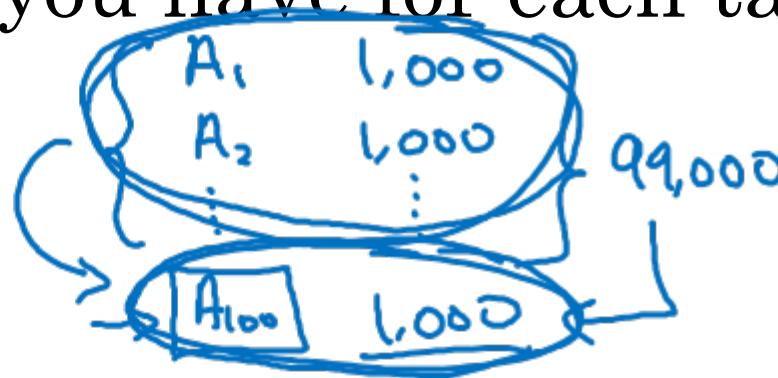
# Neural network architecture



# 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.

$$\begin{array}{ll} A & \underline{1,000,000} \\ \downarrow & \downarrow \\ B & \underline{1,000} \end{array}$$



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



deeplearning.ai

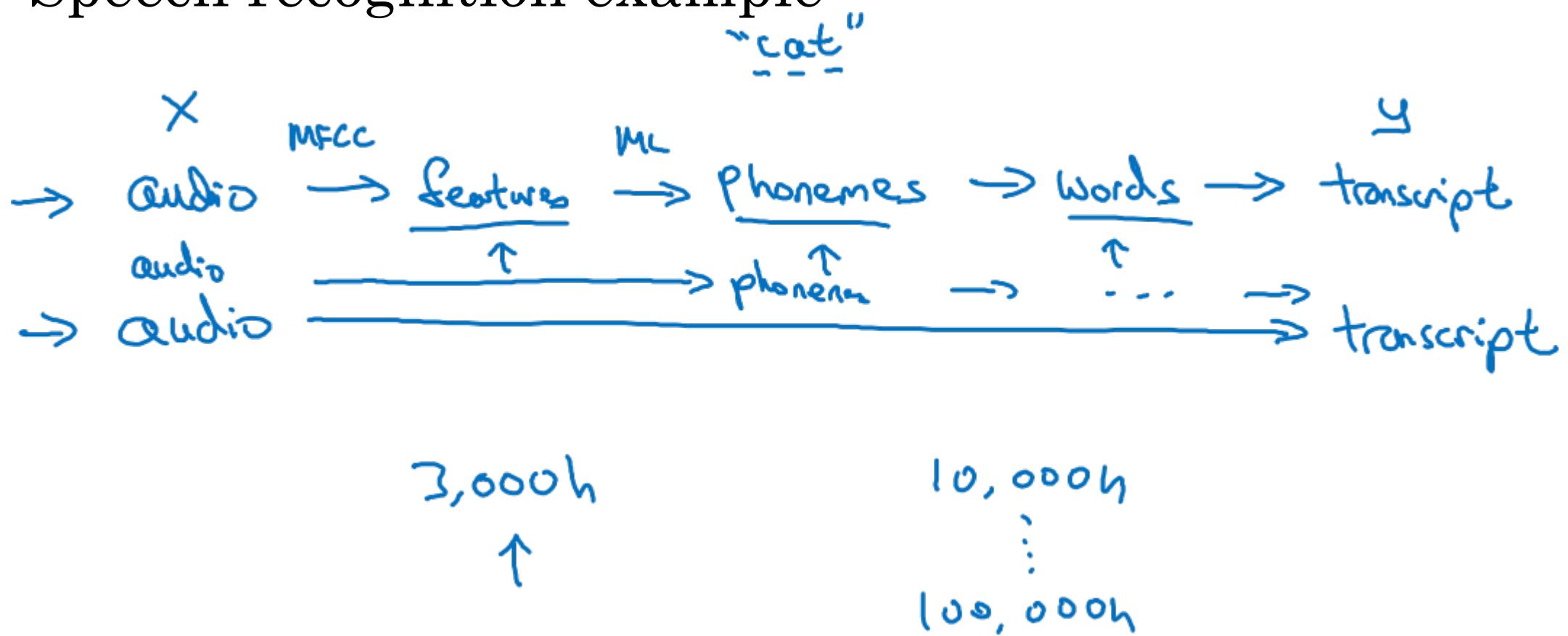
End-to-end deep  
learning

---

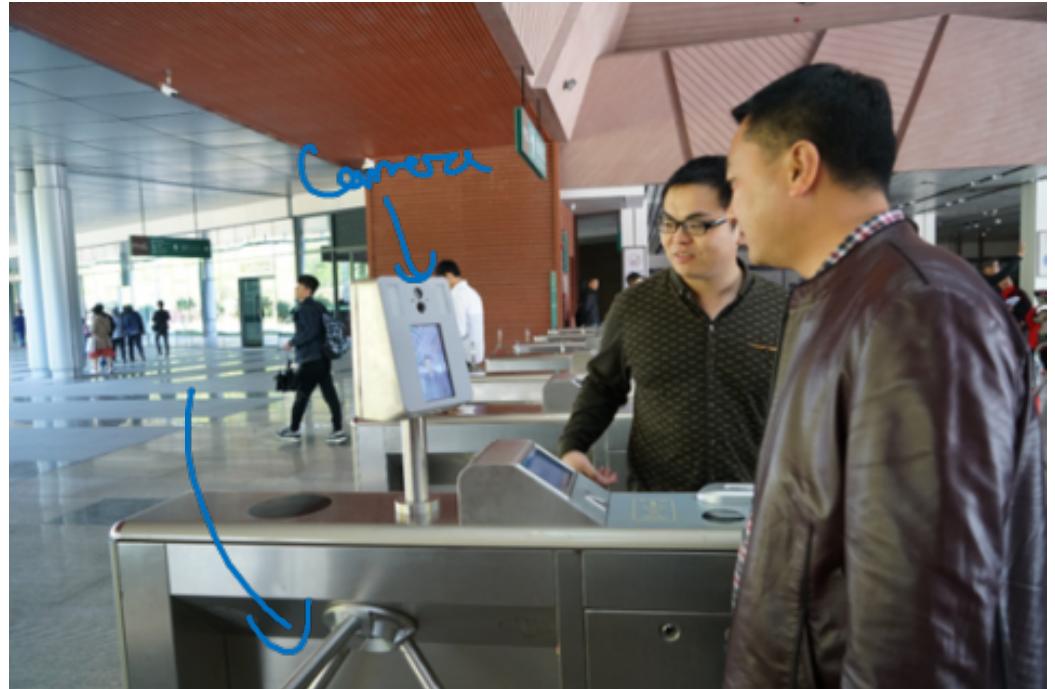
What is  
end-to-end  
deep learning

# What is end-to-end learning?

# Speech recognition example

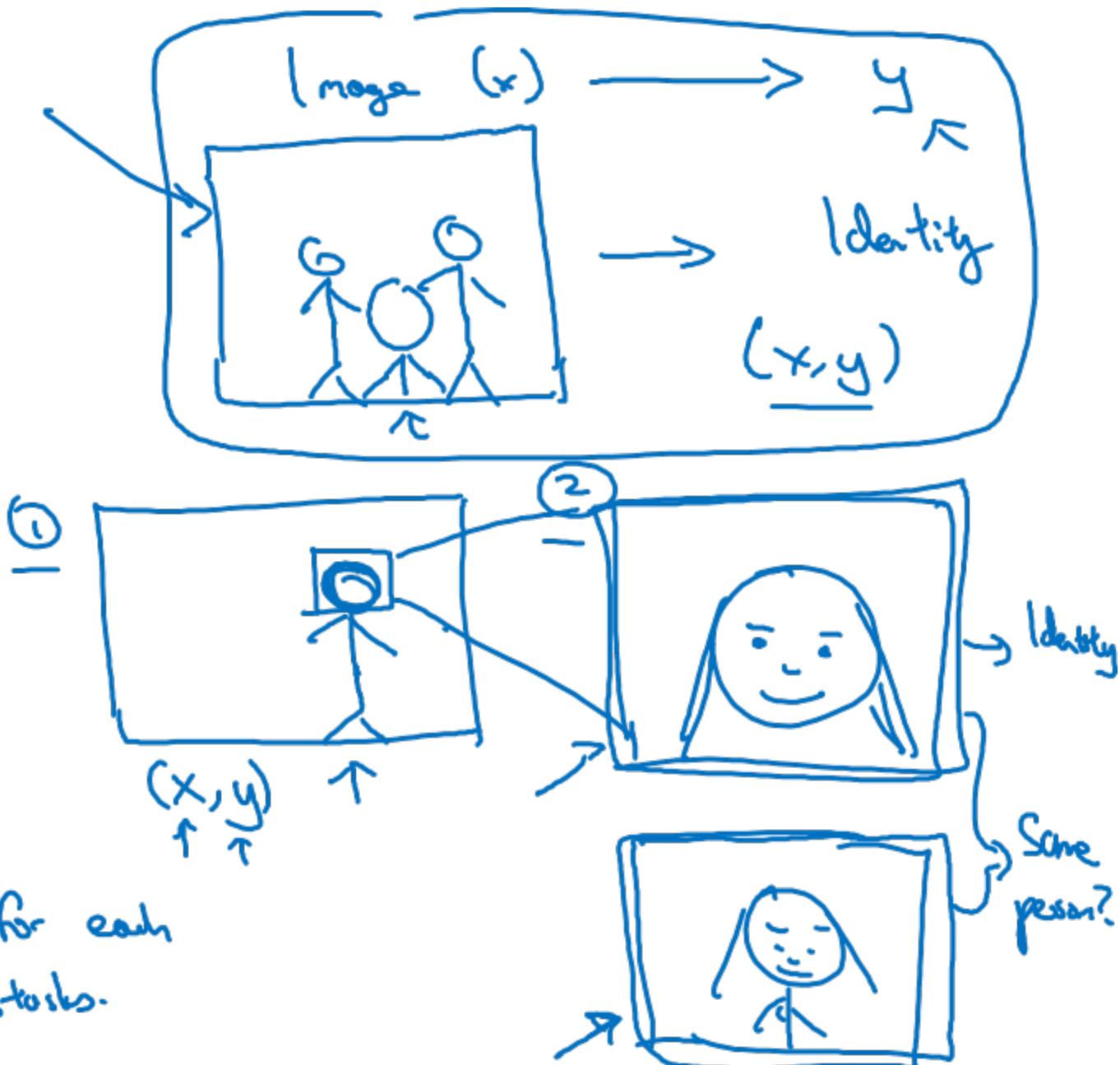


# Face recognition



[Image courtesy of Baidu]

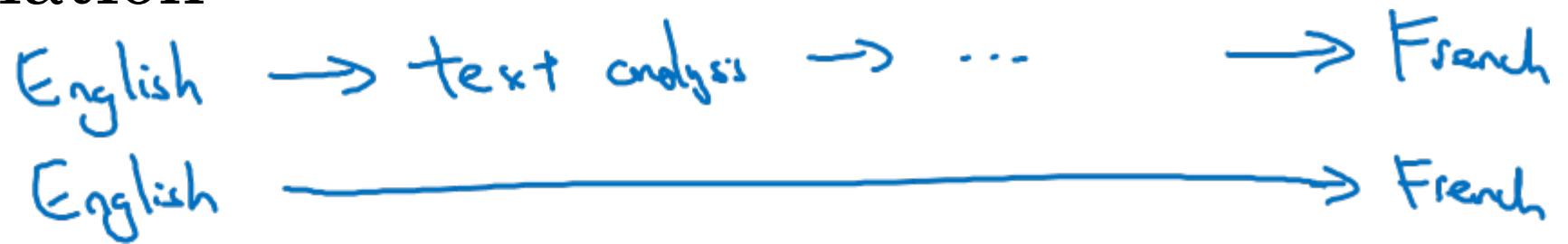
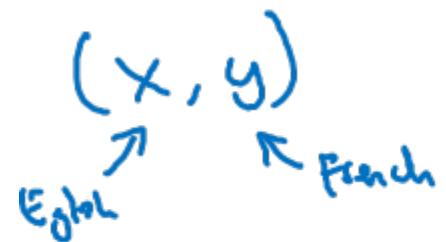
Have data for each  
of 2 sub-tasks.



Andrew Ng

# More examples

## Machine translation



Estimating child's age:





deeplearning.ai

# End-to-end deep learning

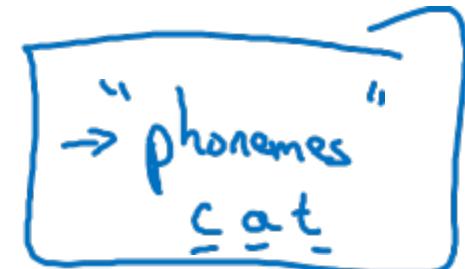
---

Whether to use  
end-to-end learning

# Pros and cons of end-to-end deep learning

Pros:

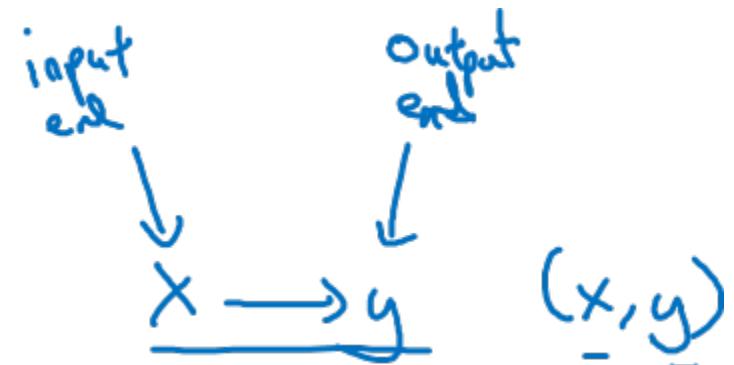
- Let the data speak  $x \rightarrow y$
- Less hand-designing of components needed



Cons:

- May need large amount of data
- Excludes potentially useful hand-designed components

$$x \dashrightarrow \dashrightarrow \rightarrow y$$



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?

