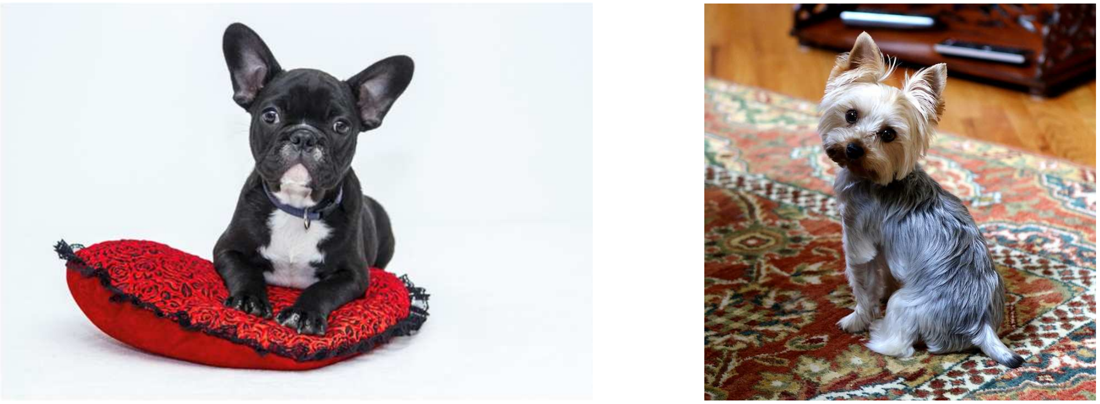
deeplearning.ai

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%

5/100

10%

9.5%

→ 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	8%	43%	61%	12%	

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.

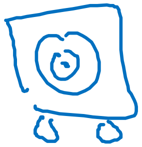
Correcting incorrect dev/test set examples

- 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 dev/test data may now come from slightly different distributions.

Speech recognition example



- Noisy background
    - Café noise
    - Car noise
- Accented speech
- Far from microphone
- Young children's speech
- Stuttering   uh, ah, um, ...
- …

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

Andrew Ng

deeplearning.ai

Mismatched training and dev/test data

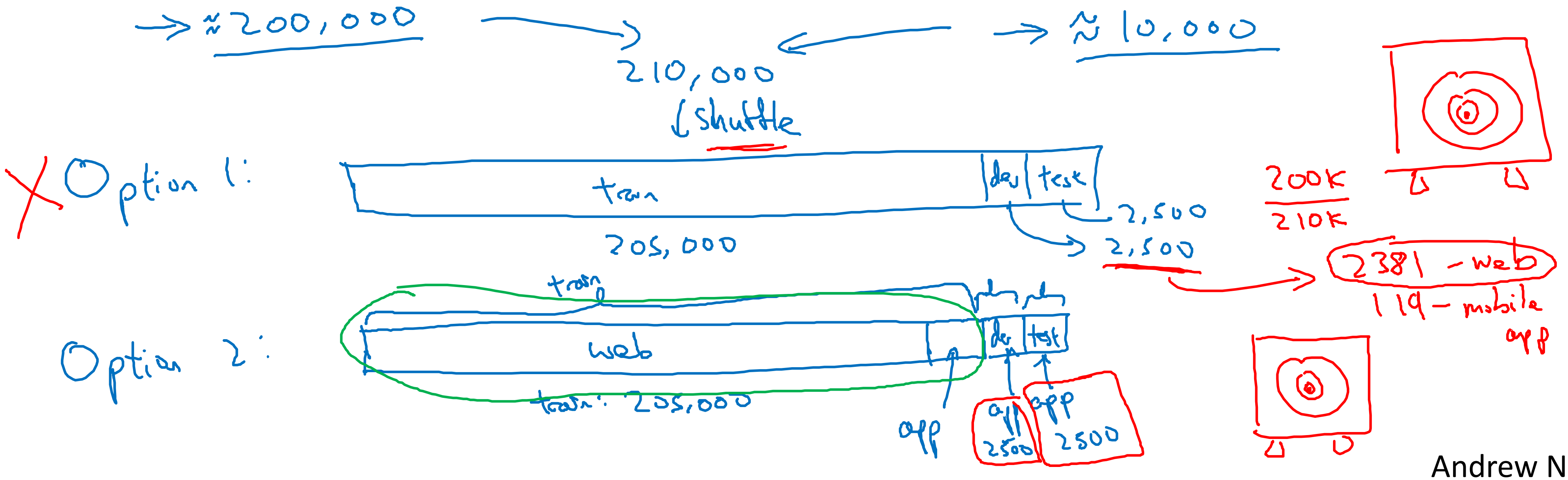
Training and testing on different distributions

Cat app example

care about this

Data from webpages

Data from mobile app



Speech recognition example

Speech activated rearview mirror

Training

Purchased data x, y

Smart speaker control

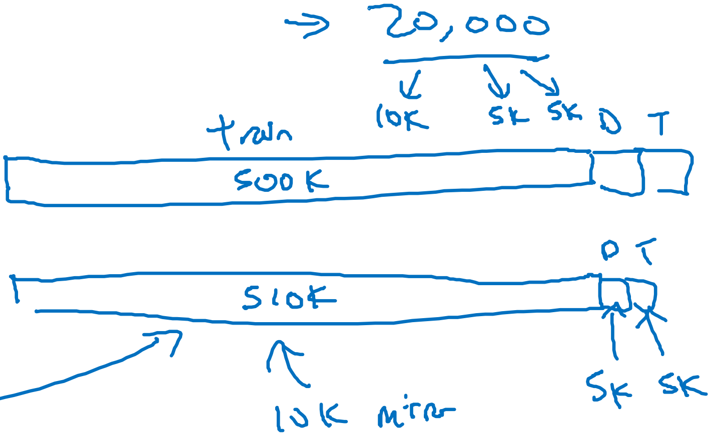
Voice keyboard

…

500,000 utterances

Dev/test

Speech activated rearview mirror



Andrew Ng

deeplearning.ai

Mismatched training and dev/test data

Bias and Variance with mismatched data distributions

Cat classifier example

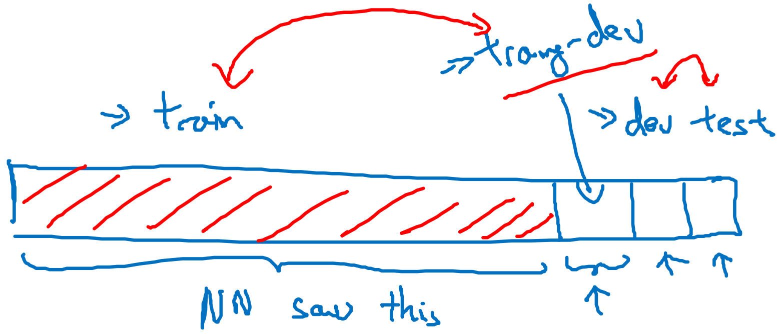
Assume humans get ≈ 0% error.

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			

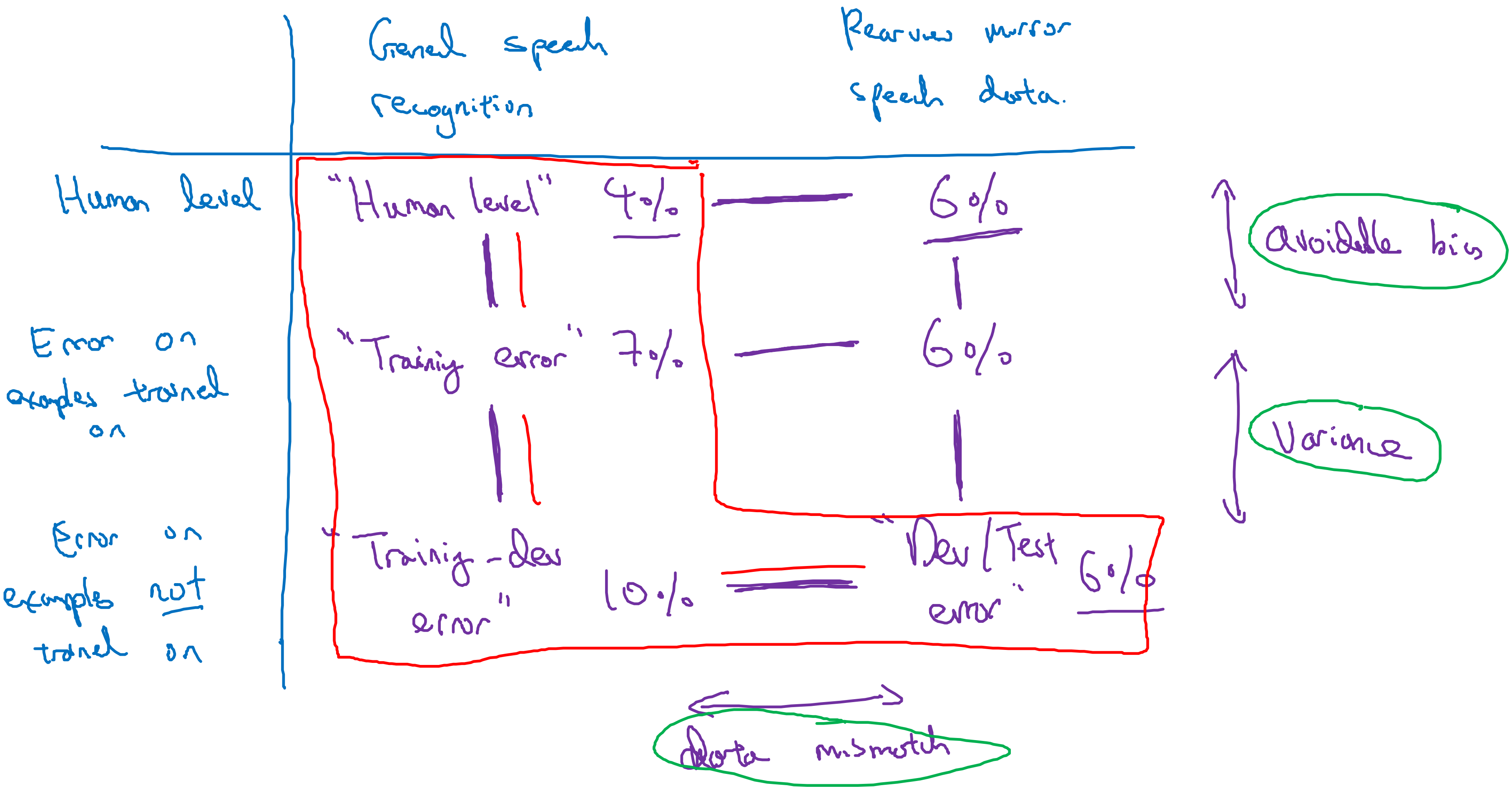
Human error	0%	Avoidable bias		Avoidable bias
Training error	10%		10%	Variance
Train-dev error	11%		11%	Data mismatch
Dev error	12%		20%	
	Bias		Bias + Data mismatch	

Bias/variance on mismatched training and dev/test sets

Human level	4%		4%
		avoidable bias	
Training set error	7%		7%
		variance	
Training-dev set error	10%		10%
		data mismatch	
→ Dev error	12%		6%
		degree of overfitting to dev set.	
→ Test error	12%		6%

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

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

Andrew Ng

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

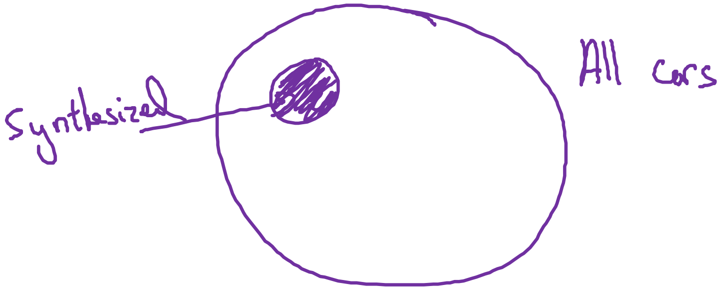
10,000 hours

Synthesized in-car audio

Artificial data synthesis

Car recognition:

$\approx \underline{20}$ cars



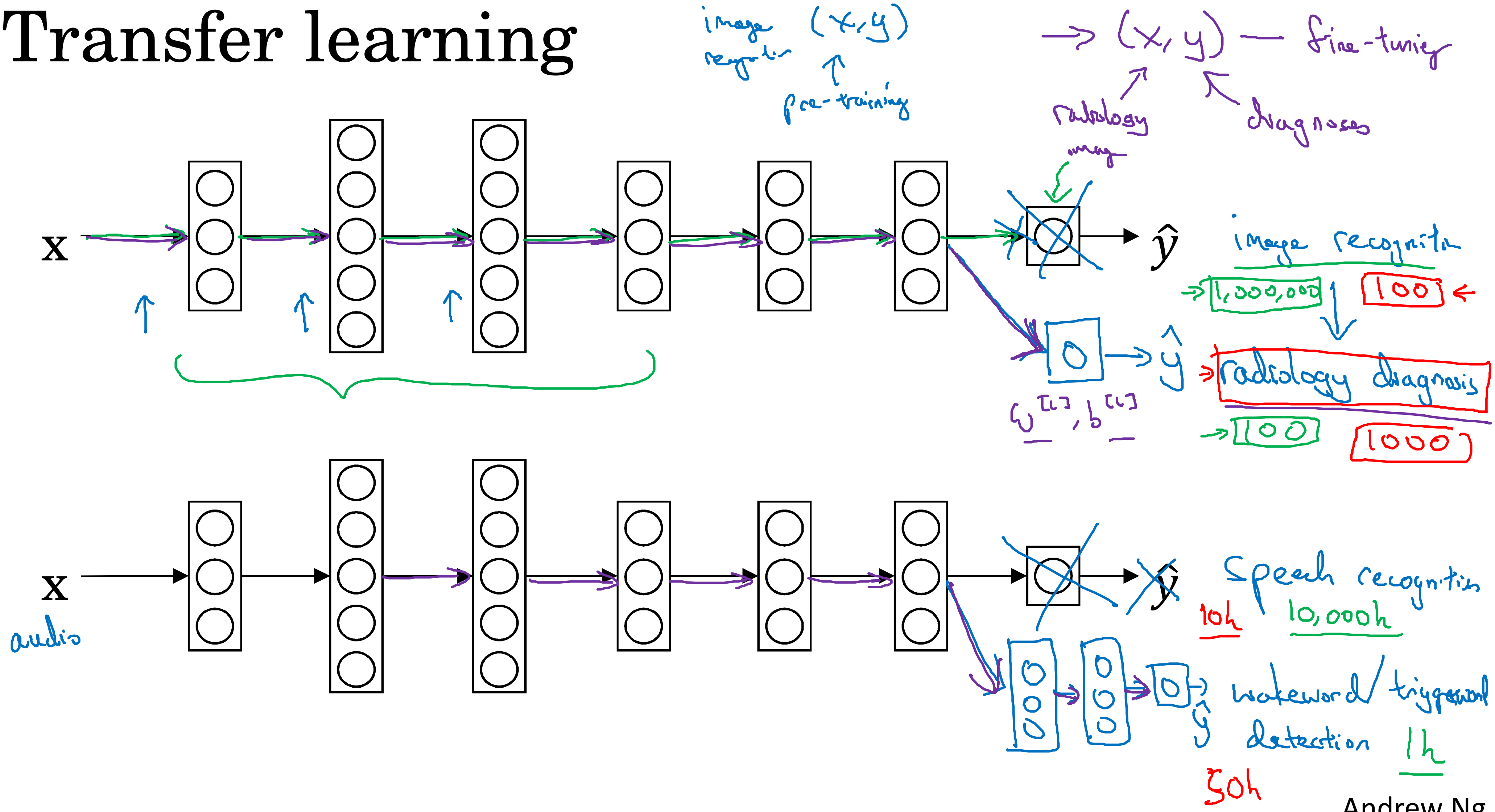
Andrew Ng

deeplearning.ai

Learning from multiple tasks

Transfer learning

Transfer learning

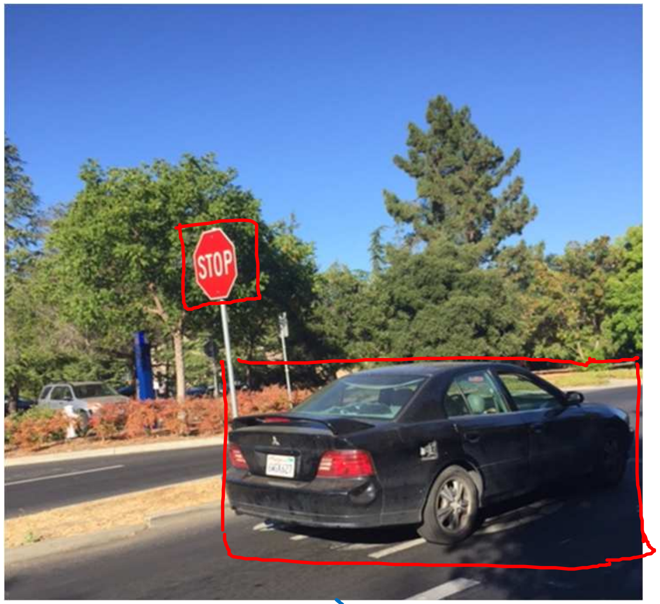


When transfer learning makes sense

Transfer from A → 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.

Simplified autonomous driving example



$x^{(i)}$

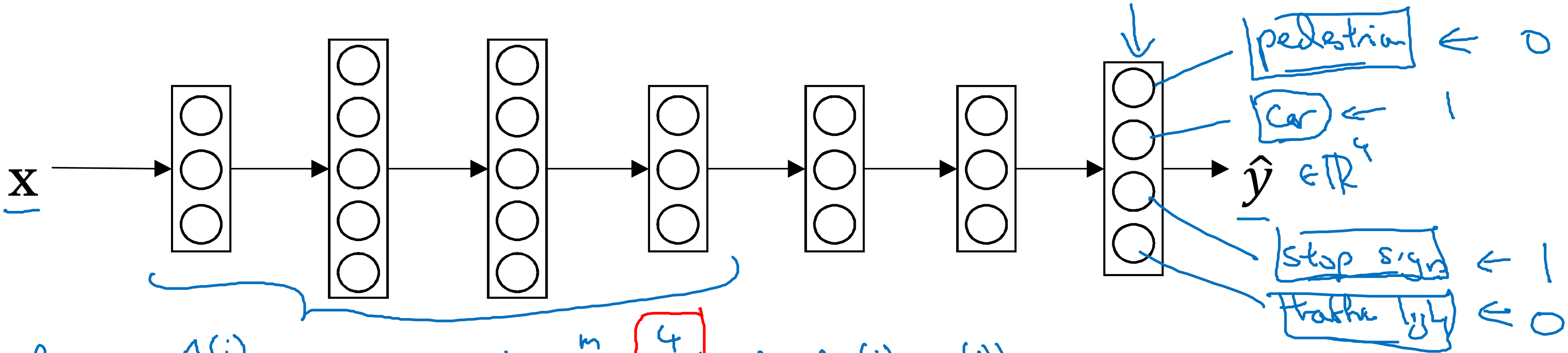
$y^{(i)}$ (4,1)

Pedestrians	0
Cars	1
Stop signs	1
Traffic lights	0
⋮	⋮

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

(4, m)

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 value 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:

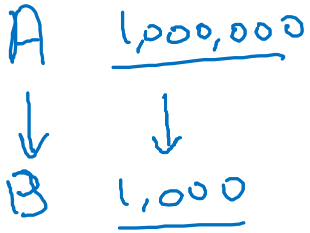
One image can have multiple labels

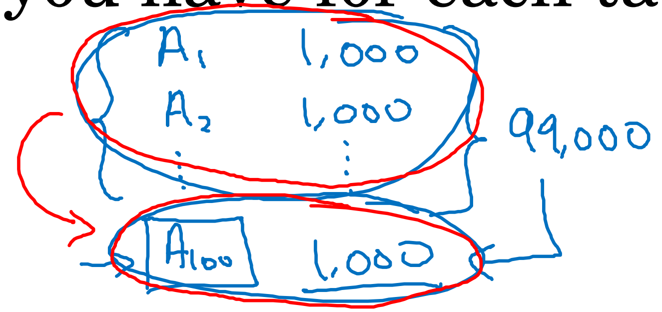
Multi-task learning ←

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

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.





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

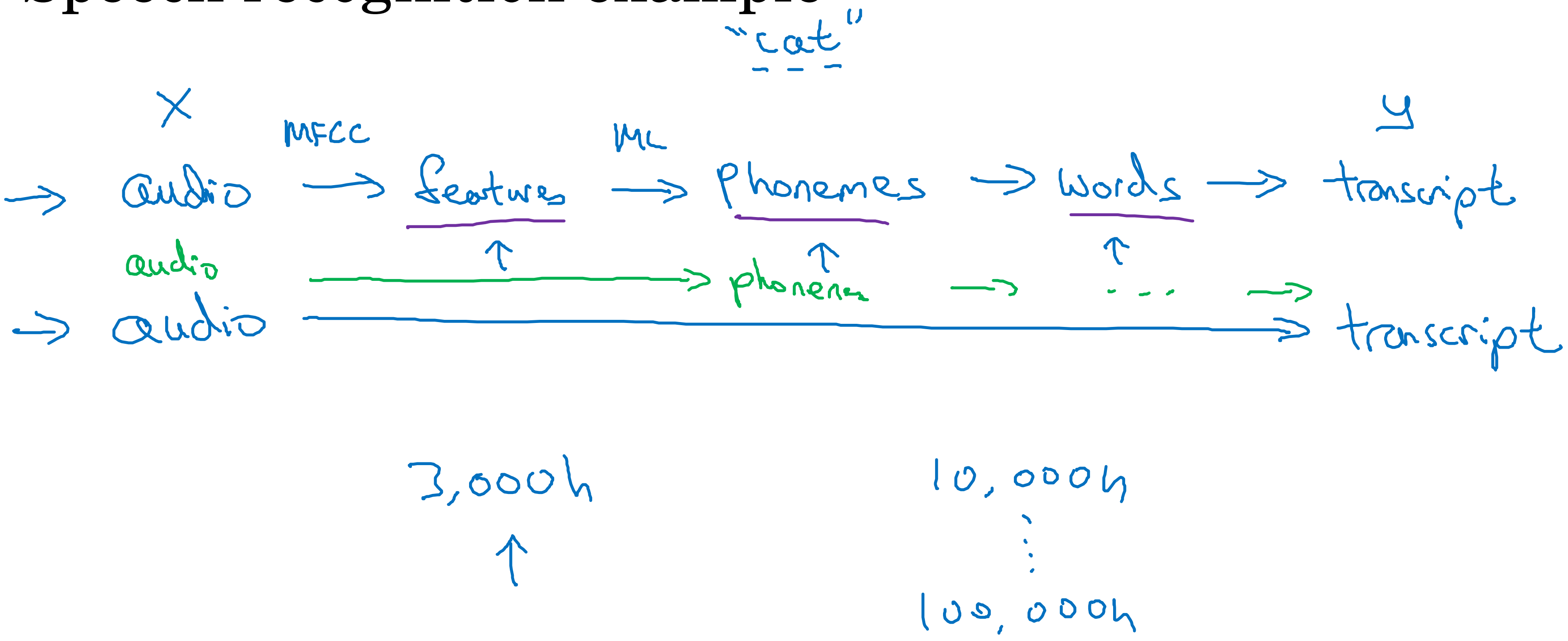
deeplearning.ai

End-to-end deep learning

What is end-to-end deep learning

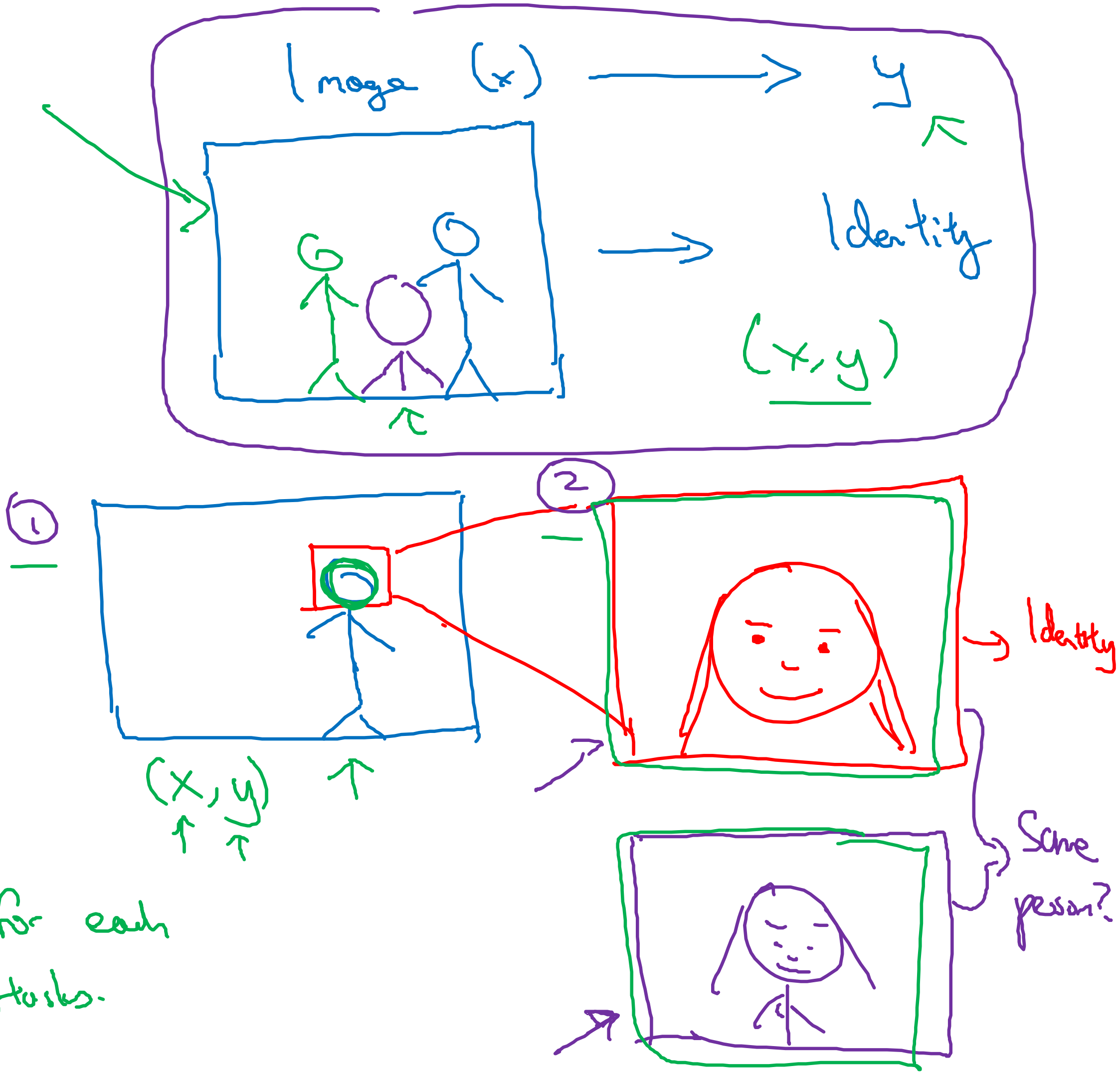
What is end-to-end learning?

Speech recognition example



Face recognition

[Image courtesy of Baidu]



Have data for each of 2 subtasks.

Andrew Ng

More examples

Machine translation

(x, y)

English ↗ ↖ French

English → text analysis → . . . → French

English → French

Estimating child's age:

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

Image → age ←

Pros and cons of end-to-end deep learning

Pros:

- Let the data speak $x \rightarrow y$
  → "phonemes"
  c a t
- Less hand-designing of components needed

$x \dashrightarrow y$

Cons:

input end

output end

- May need large amount of data $x \rightarrow y$ (x, y)
- Excludes potentially useful hand-designed components

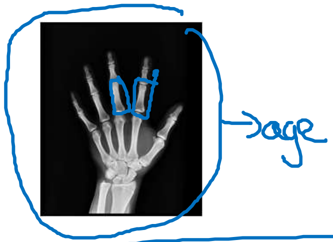
Data.

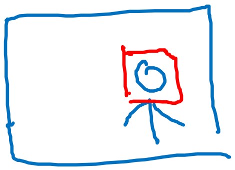
Hand-design.

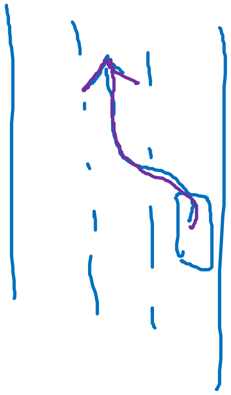
Applying end-to-end deep learning

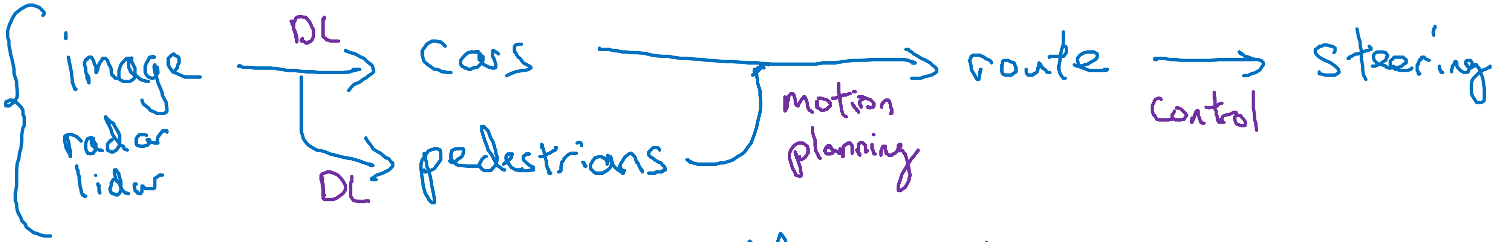
Key question: Do you have sufficient data to learn a function of the complexity needed to map x to y?

X → y









- Use DL to learn individual components
- Carefully choose X → Y depending what tasks you can get data for.

→ image ——→ steering

Andrew Ng