Structuring ML Projects

What/Why ML Strategy?

not good enough: data? training algorithm?
many ideas, how to pick:

Orthogonalization: Clear eyed

what to adjust to change *some thing 'x'*

steering wheel to direction
gas, break pedal to speed
a/c to change temp.
...

1- Fit training set well on cost func - (bigger network, change cost fnc: *adam*)
inbetween early stoping
2- Fit dev set " - (regularization, bigger train set)
3- Test set " - (bigger dev set)
4- Perform well in real world - (change dev set or cost function)

Single Number Eval. Metric:

First thing to do: setting up the eval. metric

Recall & Precision -> Accuracy? What is important, what matters (average?) ?

Satisfying and Optimizing Metric
Cost = Accuracy - 0.5 x runningTime
OR
max accuracy
subject to runningTime < 100 ms

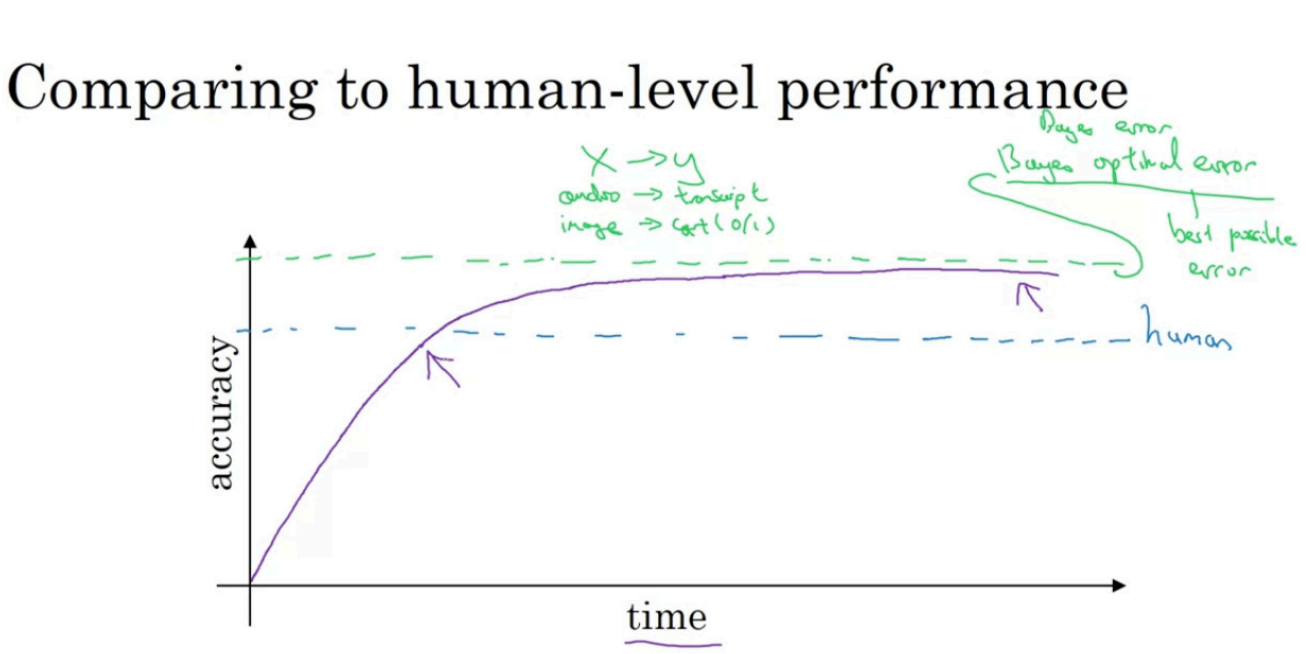
Train/Dev/Test Distributions

- should come from the same distribution
  Size ?
- previously fixed, 70-30 OR 60-20-20,
- now: 98-1-1, DL, Big Data, Data hunger
  When to Change Set Metrics?
- %3 error but lets through +18 content. %5 error may be better, changed eval.

Comparing ML to Human Level

- feasable to match human level
- more efficient to do something that is done in some way by humans

Bayes optimal error: best possible error
sometimes model performance surpasses human level though gets limited at this rate



Why?
You can get labeled data from humans
Get insights from the model output/performance
Better analysis of bias/variance

Avoidable Bias:

You want to do good at 'x', but not that good. You may match the optimal accuracy to human level performance.
Humans: %1, Training Error: %8, Dev Error: %10 - too much of a difference, model can learn more

For the case where human error is %7.5, you may stop tuning the model further since you do not want to pass this rate (probably causes overfitting)

Understanding Human Level Performance

Typical human, doctor, experienced doctor, team... How to define 'human level' error?
Define Bayes? -> team
Practical definition -> typical doctor

- Human Level (proxy for bayes)
  This diff. is avoidable bias
- Training Error
  This diff is Varince
- Dev Error

Surpassing Human Level Performance
Avoidable bias: Best performance by humans - Training Error

When model performance is better than humans?

Problems where ML surpasses human-level performance: ads, recommend.s, logistics, loans... -> Structured data.

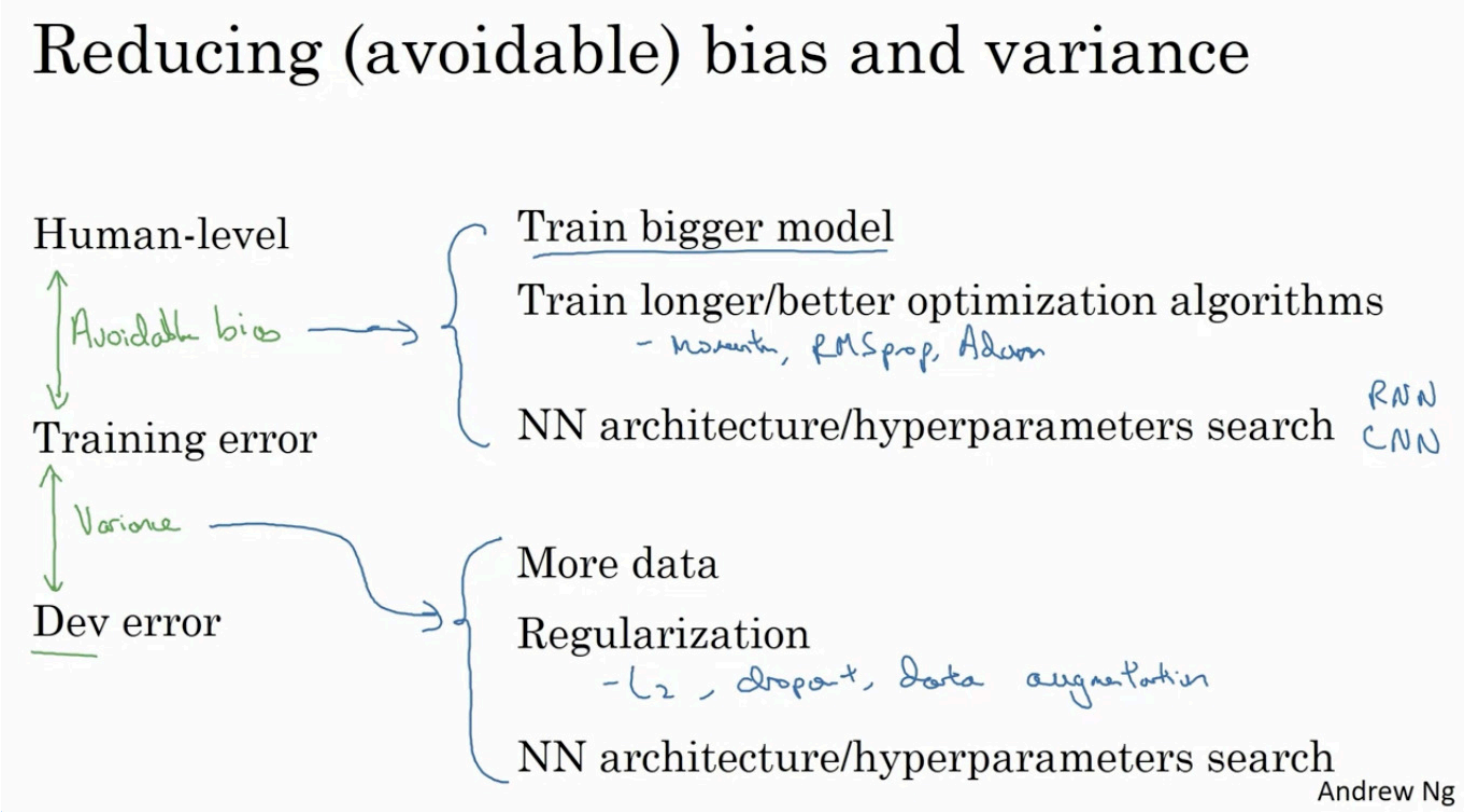
Speech recognition, image recognition, ECG, cancer detection...

Improving Model Performance

Two fundamental assumptions of supervised learning:
You can fit the training set pretty well
The training set performance generalizes pretty well to the dev/test set

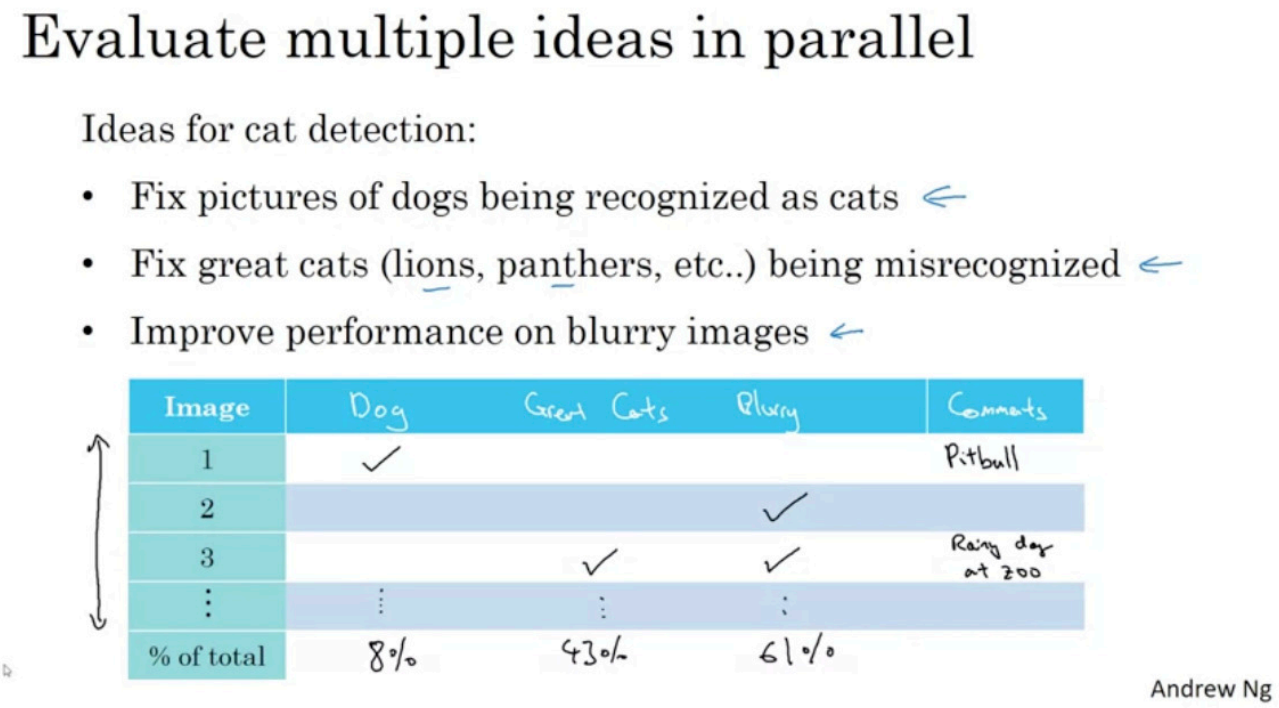
Reducing (avoidable) bias and variance

Check the accuracy differences in between different cases.



Carrying Out Error Analysis

Does *x* worth working on to drive down the error rate?



Cleaning Up Incorrect Labels from Data

If errors are reasonably **random**, then there is not much problem.
DL algorithms are quite robust to random errors in the training set.
they dgaf

- If there is a systematic error, it may cause problems.

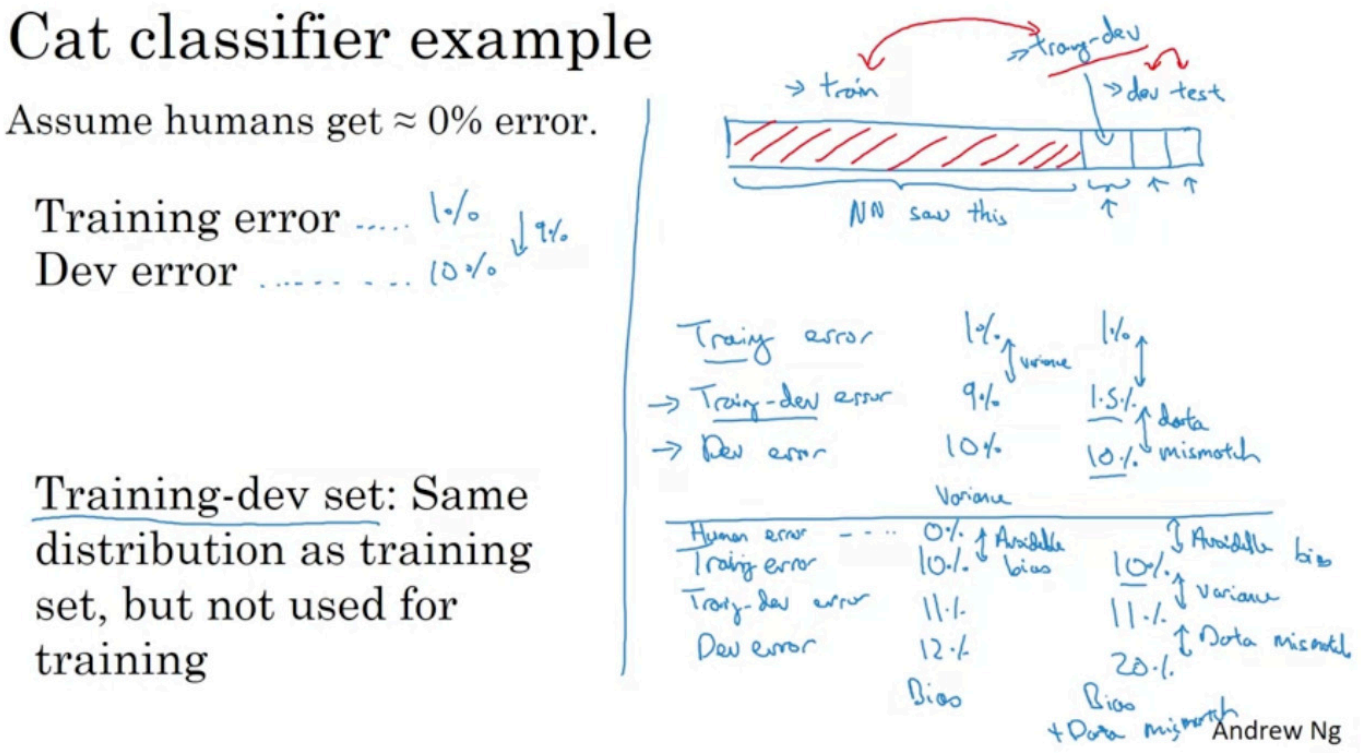
Correcting incorrect set examples
whatever you do, do it to all sets. apply same process
Consider examining examples both your algorithm got right as well as ones it got wrong.

Build your first system quickly, then iterate

DL Era: Find whatever data you can, shove it to the model. Result: different distributions for train/dev/test

1st Case: Only difference: trained explicitly on train, but not on train-dev. Variance problem - Not generalizing well.
2nd Case: Data mismatch problem. Was not trained explicitly on train-dev - different distribution.
3rd Case: Avoidable bias. Variance between train-train-dev variance is low. Also data mismatch

problem bcs of the difference between train-dev.



Metric	General Speech Recognition	Resource-Mismatched Speech Data	Gap
Human Level	4%	6%	Avoidable Bias
Error on Examples Trained On	7%	6%	Variance
Error on Examples Not Trained On	10%	6%	Data Mismatch

Key Takeaways:
• **Avoidable Bias**: Difference between “Human Level” and “Training Error.”
• **Variance**: Difference between “Training Error” and “Training-Dev Error.”
• **Data Mismatch**: Difference between “Training-Dev Error” and “Dev/Test Error.”

Addressing Data Mismatch

no complete systematic soln.
manual error analysis. make training data more similar, collect more data.

artificial data synthesis -> add noise etc.
10k hours of data + 1 hour of car noise = - 20k hours of data
problem: DL may overfit to the same car noise, may also need to change it.

Think about your source. Example: Video game. There are numerous cars, but only 20 unique designs that is populated.

The quick brown fox jumps over the lazy dog.

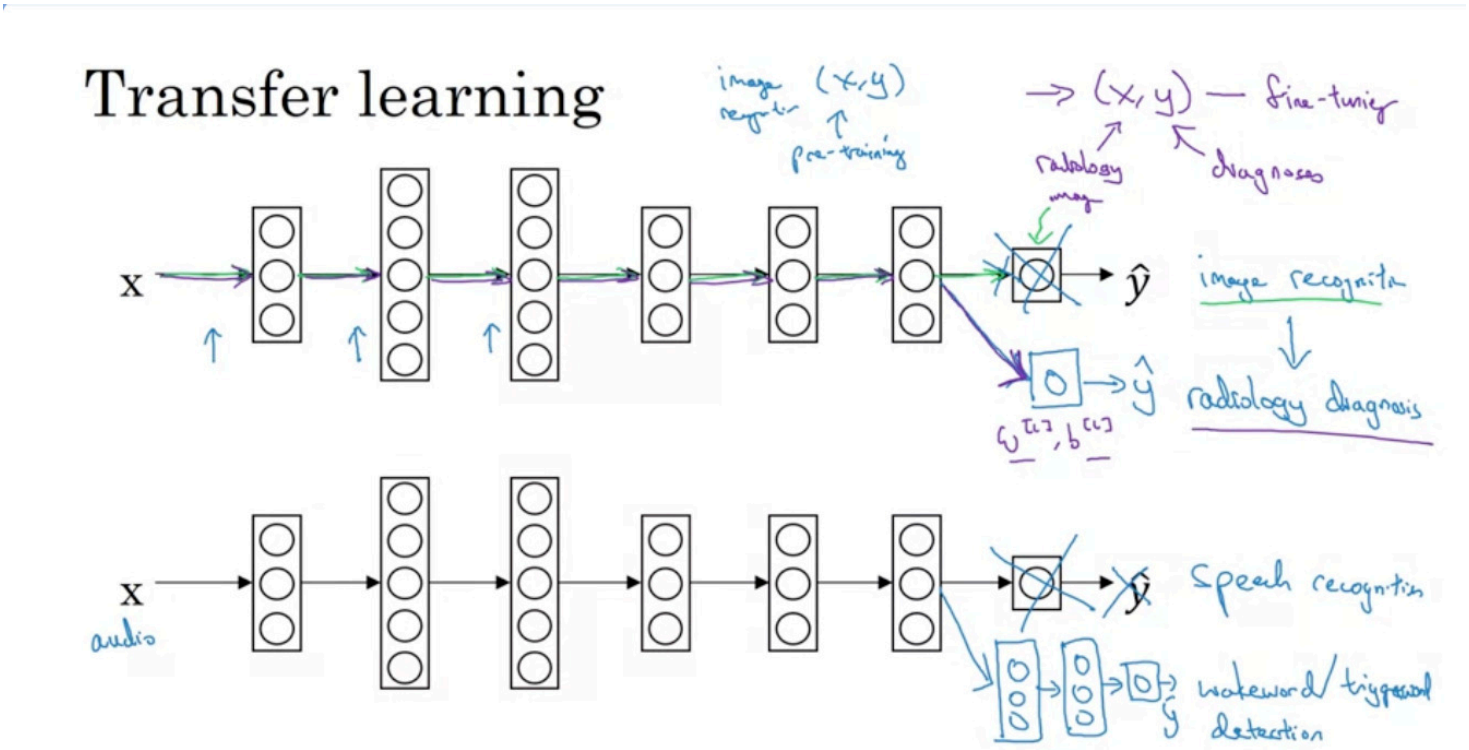
Transfer Learning.

Apply one tasks knowledge to another task.

Image recognition -> radiology diagnosis
Initialize last layers weights, re-train the model.

Small data: freeze rest, only train last layer
Enough data: retrain all parameters in the NN

(pretraining, finetuning)



you may also create more outputs

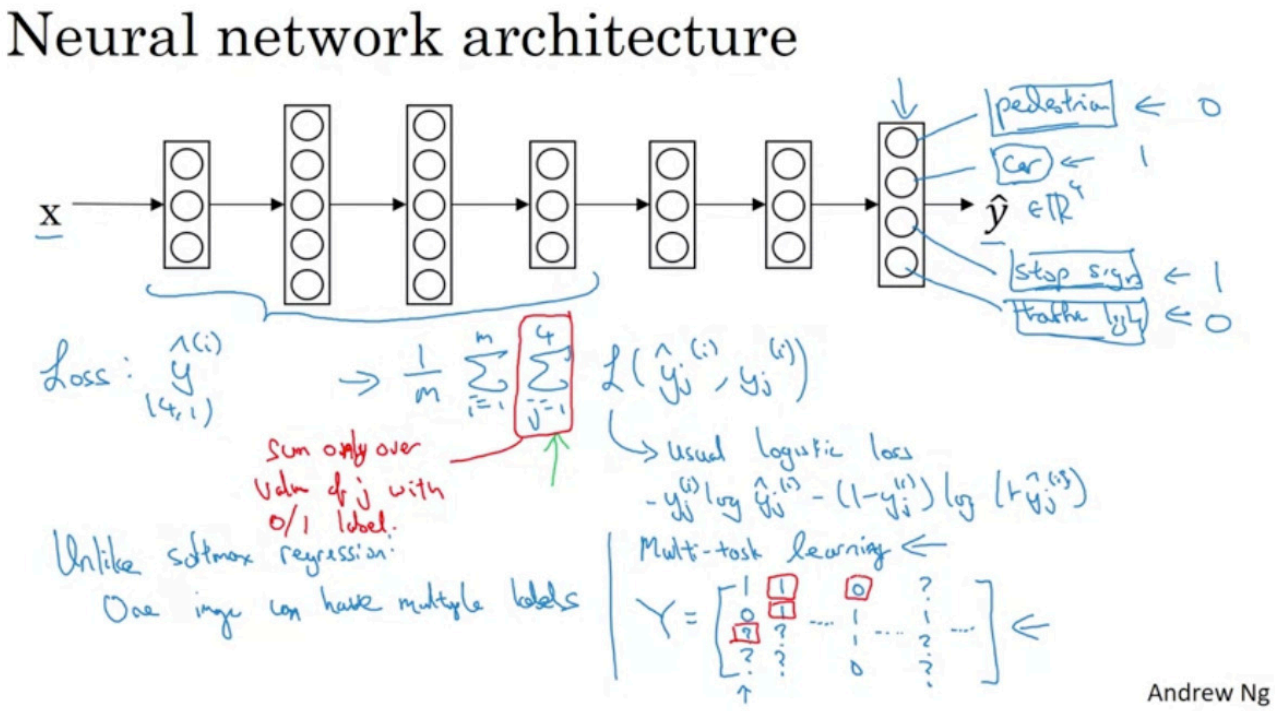
When does it make sense?

- Lot of data for the previous task, but less data for the new task.
- Same input
- Low level features can be helpful - logic & instinct

Multi-Task Learning

Teach several things at once to the model.
Loss: 4 dimensional, a little complex, may check it in detail.
Unlike softmax regression, one image can have multiple labels (YOLO image - CryoET)

May perform better then 4 separate NNs



Some images can include only some labels - you can use more data

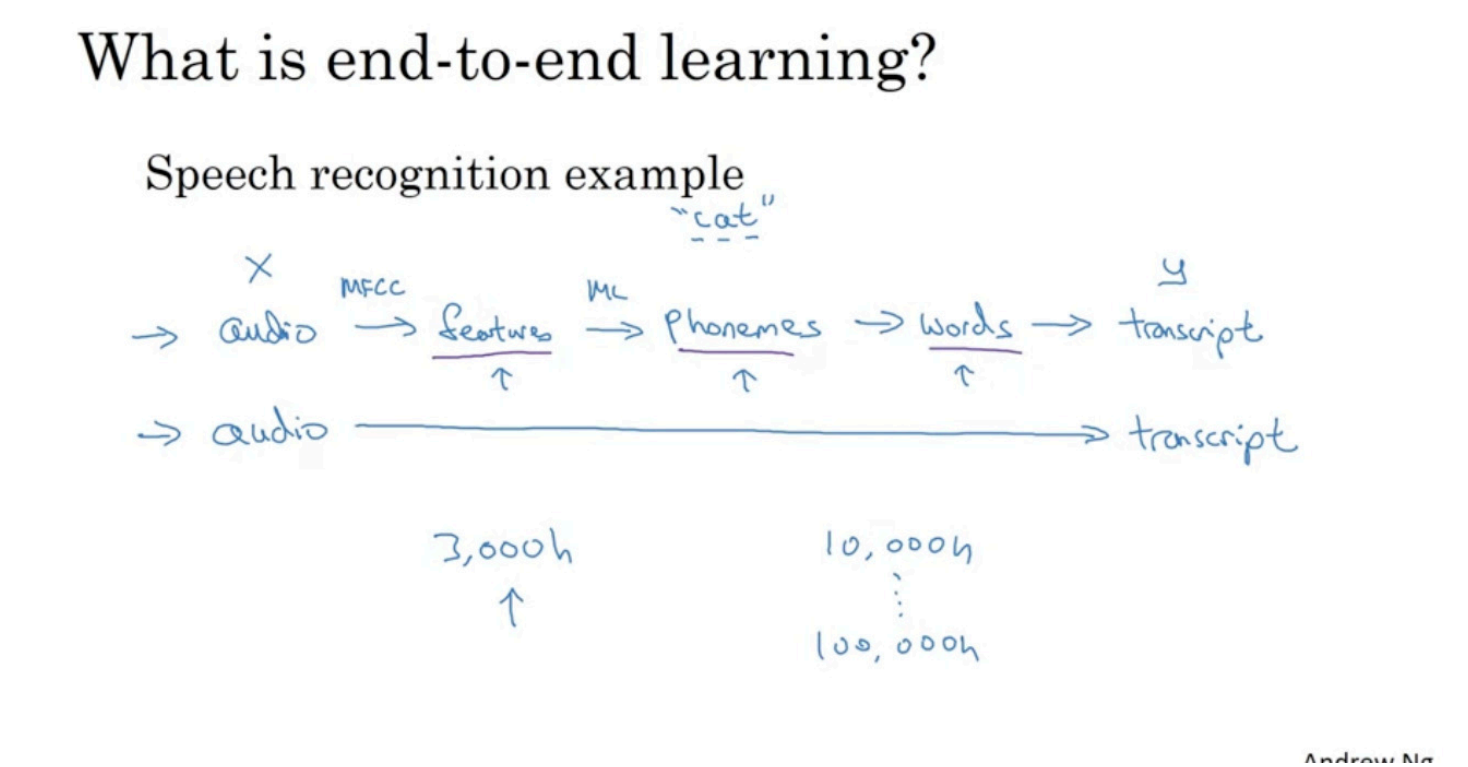
When multi-task learning makes sense?

- Shared low level features
- Amount of data you have for each task is quite similar
- Can train a big enough NN to do well on all the tasks

End to End DL

Some applications need multiple stages of processing.
Convert this to a single DL architecture

Example



Face Recognition

Best approach is not directly using the original image, but first cropping with centering the target, then feeding this processed data to the network.

Instead of single step, 2 simple step approach.

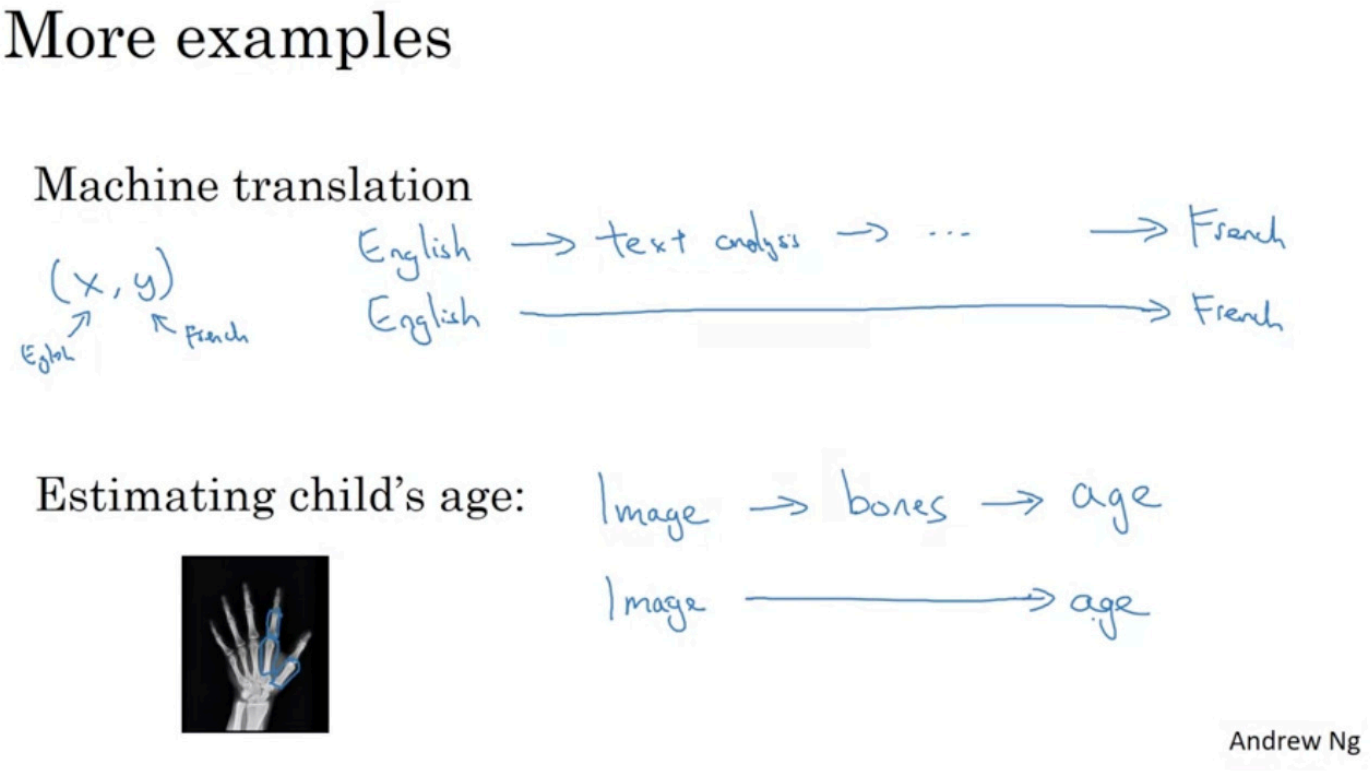
How it works: Takes 2 input images, same person or not? 10k employees, compare it with 10k employees.

Why it is better?

A lot of data with face and coordinate of the face.

100s of m.s of images of cropped images for task 2.

task 2 requires less data to learn.



2nd example, Task 2 does not work well today, bcs not enough data (may be outdated info)

Whether to use End to End DL

Pros and Cons of End-to-End Deep Learning

Pros:
• **Let the data speak**:
• The model directly maps inputs (**X**) to outputs (**Y**).
• Example: "cat" → **phonemes**.
• **Less hand-designing of components needed**:
• Intermediate steps like feature extraction or manual engineering are minimized.

Cons:
• **May need a large amount of data**:
• End-to-end models often require vast datasets to generalize effectively.
• **Excludes potentially useful hand-designed components**:
• Hand-crafted elements (like structured linguistic rules) may provide additional value but are bypassed in favor of raw data-driven methods.

Additional Notes:
• Input → **X** → Output → **Y**: A straightforward input-to-output transformation.
• Some systems bypass intermediate steps such as phoneme-level processing (e.g., breaking "cat" into **c-a-t** phonemes).
• While data is crucial, ignoring hand-designed components can lead to suboptimal outcomes in resource-limited scenarios.

1- Some task are better done without DL
2- DO YOU HAVE ENOUGH DATA?

Carefully choose x->y mapping depends what tasks you can get data for.

image -> steering

Credit: Deeplearning.AI