Course 3

STRUCTURING ML Project

Orthogonization

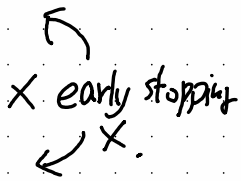
- many potential tools

Fit training set well on cost function
- bigger network
- adam
:

Fit dev set well on cost function
- Regularization
- bigger training set

Fit test set well on cost function
- Bigger dev set

perform well in real-world
- change dev set or cost function



Single Number Value Evaluation

- precision → recognized cat, % of them as cat
- recall → % of actual cat got recognized

Competing.. hard to evaluate

⇒ F1 score ~ average of P & R

$\frac{2}{\frac{1}{P}+\frac{1}{R}}$ "Harmonic Mean"

Satisficing & Optimizing Metric

e.g. maximizing accuracy ← optimizing

subject to running time < 100ms ← satisficing

e.g. wake word / trigger word

'hi siri'

maximize accuracy

s.t. ≤ 1 false positive every 24 hrs.

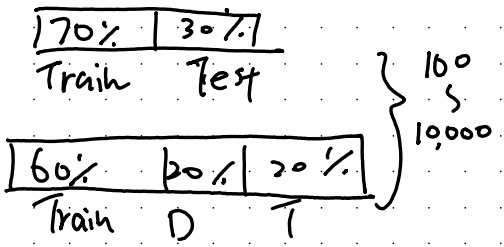
Train / Dev / Test set

Dev / Test should have same distribution

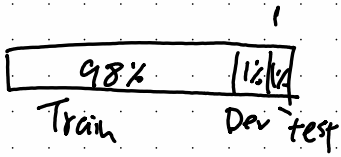
choose Dev/Test that reflect your future target

Size of dev/test sets

Old



1,000,000 data



size of test set ~ big enough to give high confidence on performance

- train | dev ← sometimes not recommanded
- train | test ✗ not good naming...

When to change Dev/Test and Metrics

real data or different from expectation

- metric + dev prefer A
- your user prefer B

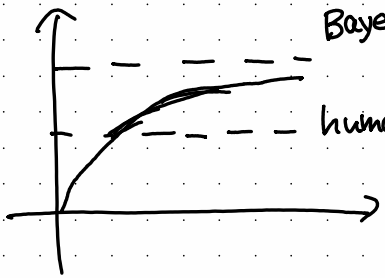
⇒ time to change

e.g. bigger penalty on undesired image

orthogonality

- place target (define metrics)
- shoot at it (change cost func... etc)

human-level performance



→ progress slows when surpass human

if ML is worse
- get more labels
- gain insight from manual analysis
- better analysis of bias / variance

avoidable bias

△ assume human-level error ≈ bayes error (esp. CV tasks)

△ human	1%	7.5%
training error	8%	8%
dev error	10%	10%
	↑ reduce bias	↑ reduce variance

↕ avoidable bias

define human-level error

human-level error as proxy to bayes error.

e.g. human
doctor
group of doctors ← closer to bayes error
⋮

theoretical upper bound

human-level error
↕ avoidable bias
training error
↕ variance
dev error

surpassing human-level error

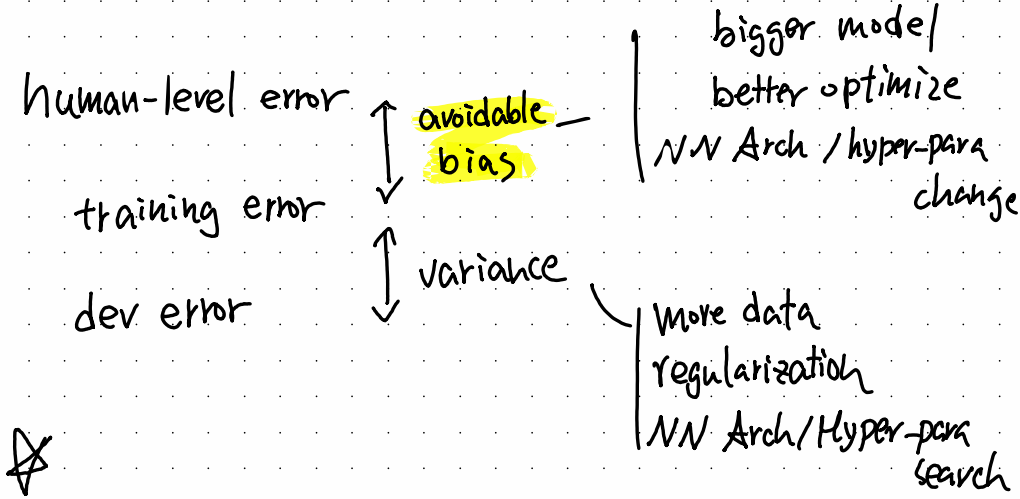
- online ad
- product recommand
- logistic
⋮

learned from structure data
not involve perception
lots of data

speech recog
medical data
⋮

fundamental assumption of supervised learning

- one can fit training set well
- training set generalizes to dev/test.



Error Ananlysis

cat vs dog. if find mislabelled ones are dog.

Analysis

1. get ~100 mislabeled dev set example
2. count how many are dogs
   - if low, probably not worth it.

△ build table for incorrectly labelled dev set.

Image	Dog	big cat	blurry .. ← in
1	✓		
⋮			
		✓	
n		-	

% ↑

based on percentage, decide what's next step.

Incorrect Labelled Data

- robust at "random" error in training data
- for dev/test, label accuracy might be less important. (based on the err analysis table)

correcting dev/test labels

- apply same process for both dev/test set (ensure same distri)
- consider examples your algorithm got 'right' as well as 'wrong'
- note training and dev/test might come from different distri

Build system quickly then iterate

- set dev/test set early
- build
- bias/var/error analysis
  - esp for new applications

Training / Dev from different distri

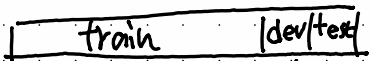
e.g.

cat from webpage ~200,000

web from mobile (blur etc) ~10,000 ← what we care

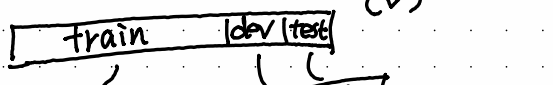
option:

1. 200k+10k = 210k (X)



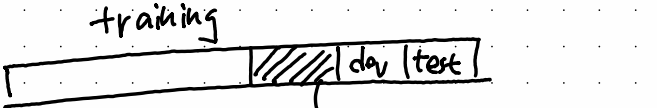
2,500 → $2500 \times \frac{10}{210} = 119$ mobile image.

2. (V)



200k + 5k
web mobile

Bias & variance with mismatched data dist.

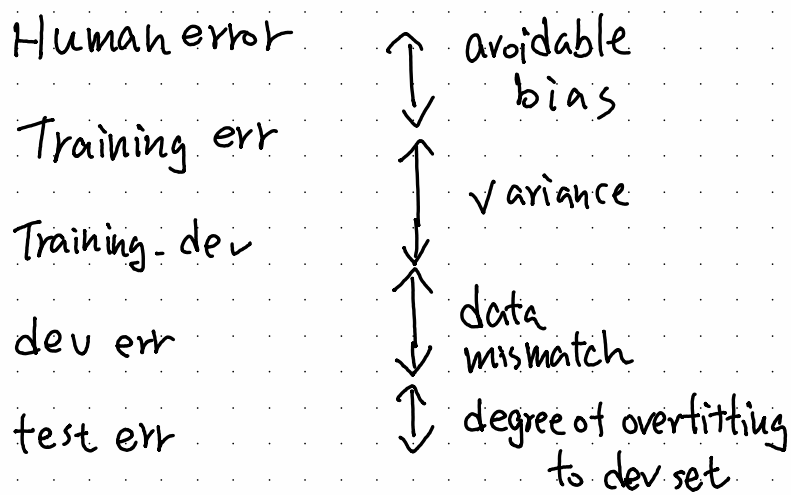


train on this

training-dev set

same dist as training set

human err	0%			
Train	1%	1%	10%	10%
train-dev	9%	1.5%	11%	11%
dev	10%	10%	12%	20%
	variance problem	data mismatch	bias problem	bias + data mismatch



Sometimes dev performs better than training

- e.g. dev/test are of different distribution
- additional label on the dev/training might better understand human performance

Addressing Data mismatching

- manual error analysis to understand the diff between training, dev/test.
- making training data more similar

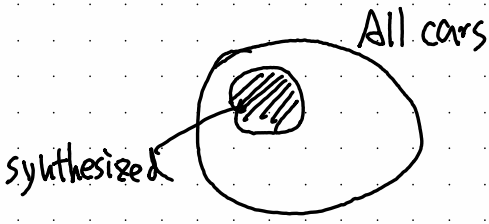
↓

Artificial Data Synthesis

e.g. speech + car noise = synthesised in car speech

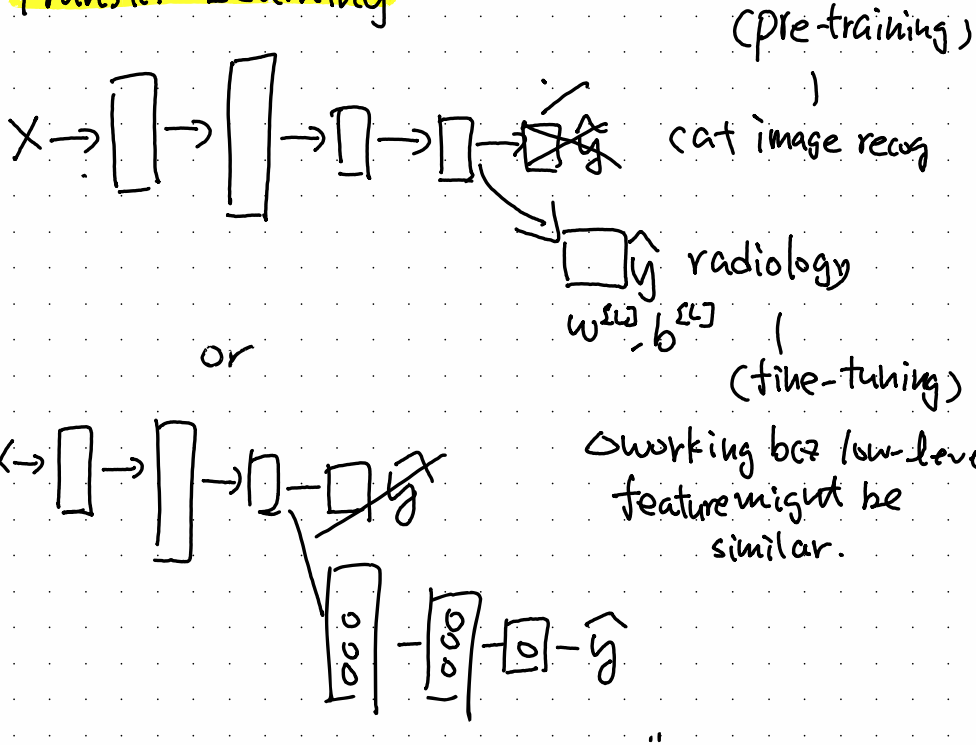
note: 10,000 hrs (speech), 1 hr (car noise), might be overfitting the 1hr car noise

e.g. car photo



" ↖ if using syn data, make sure it capture enough variety.

Transfer Learning



Transfer from A → B. useful when.

- task A & B have the same input
- much more data in A
- lower level feature of A is helpful for B

Multi-task Learning

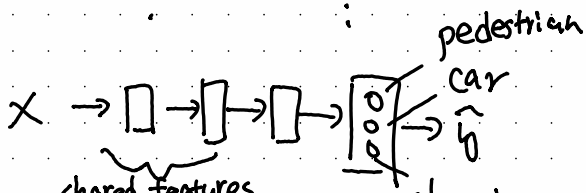
(less often used) except CV. Compared to TL

e.g. autonomous driving

△ one image with multiple labels

pedestrian
cars
stop sign
⋮

$$\begin{bmatrix}1\\0\\0\\\vdots\end{bmatrix}$$



$$\frac{1}{m}\sum_{i=1}^{m}\sum_{j=1}^{3} L(\hat{y}_j^{(i)}, y_j^{(i)})$$

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

△ might still work when some images only have partial labels

MAKE SENSE WHEN:

- tasks share similar low-level features
- amount of data for each feature are similar
- can train a big enough NN to do well on all tasks

End-to-End DL

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

audio ⟶ transcript

need a lot of data!

---

e.g.

Face recognition

image $\xrightarrow[\text{face detection}]{\text{tracking}}$ face $\xrightarrow{\text{ML}}$ recognition

& have lot data for each of 2 sub-tasks.
even more data for end-to-end data.

Machine Translation

✗ Eng → text analysis → ... → French

✓ Eng ⟶ French

Estimate child age

✓ image $\xrightarrow{①}$ bone $\xrightarrow{②}$ age

✗ image ⟶ age

wheather to use End-to-End DL

Pro:
- let the data speak
- less hand-design features

Con:
- data hunger
- exclude hand-designed features

△ hand-design feature helps when traing set is small

KEY Q:
sufficient data to learn a function maps X to Y?