Error Analysis

e.g. get ~100 mislabeled dev set examples from cat detector

how many are dog pictures? → 5 × work on dogs
→ 50 ✓ work on dogs

Image	Dog	Gray Cat	Blurry	Incorrectly Labels	Comments
1	✓				Pitbull
2			✓		
3		✓	✓		Rainy Day
⋮				✓	
% Total	8%	43%	61%	6%	

- Overall Dev Set Error : 10% 2%
- Errors due to incorrect labels : 0.6% 0.6% fix label errors

Correct dev/test sets together to make sure they continue to come from the same distribution

Train and dev/set data may come from slightly different distribution

↳ robust to label errors

Guideline: build first system quickly and then iterate

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

Mismatched Training and Dev/Test Data

e.g. Cats from web: 200,000
Cats from phone: 10,000

✗ Option 1: Shuffle together

210k → 205k Train, 2.5k Dev, 2.5k Test

Option 2: better

200k web + 5k app : Train
2.5k app : Dev
2.5k app : Test

Different

Human Level	4%	Avoidable Bias	4%
Train Set Error	7%	Variance	7%
Train-Dev Set Error	8%	Data Mismatch	10%
Dev Error	12%	Degree of Overfitting to Dev Set	6%
Test Error	13%		6%

same dist. not used for training

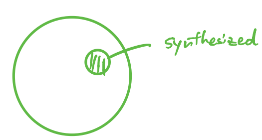
Dev/Test easier

e.g.

	General Speech Recognition	Rear View Speech Data
Human Level	"Human Level" 4%	
Error on eg. trained on	"Training Error" 7%	
Error on eg. not trained on	"Training-Dev Error" 10%	"Dev/Test Error" 6%

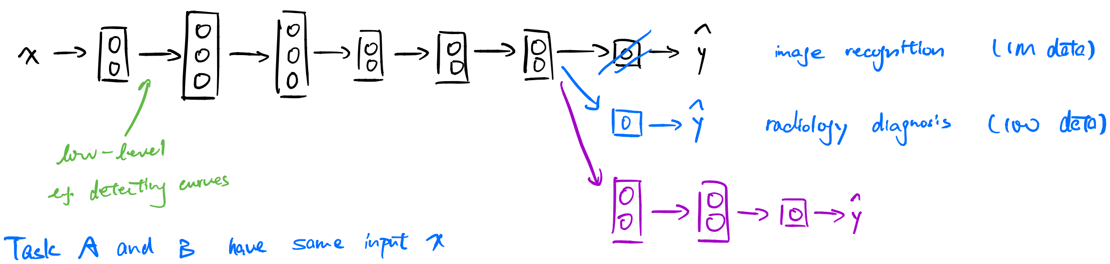
Address Data Mismatch

- Carry out error analysis to understand diff btw train & dev/test sets
- Make training data more similar, or collect more data similar
  - ↳ artificial data analysis
    - sentence + car noise ⇒ synthesized in-car audio
    - sentence: 10,000 hr
    - car noise: 1 hr (may overfit)



Learn from multiple tasks

Transfer Learning



1) Task A and B have same input x
2) You have a lot more data for Task A
3) Low level features from A could be useful for Task B

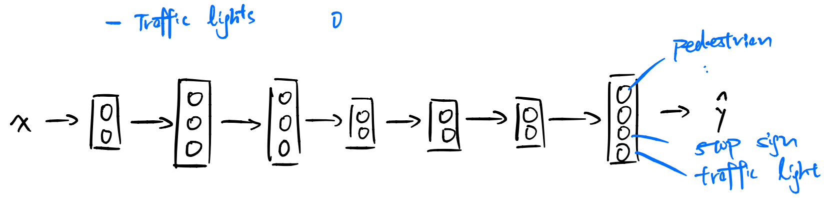
Multi-task Learning

e.g. autonomous driving

	$y^{(i)}$
– Pedestrians	0
– Cars	1
– Stop signs	1
– Traffic lights	0

$\Rightarrow Y = [y^{(1)}, y^{(2)} \cdots, y^{(m)}]$

$(4 \times m)$



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

w/ missing skip in sum

usual logistic loss

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

Unlike softmax
- one image can have multiple labels

When to use multi-task learning ?

- 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 NN to do well on all tasks

$\left\{ \begin{array}{l} A_1 : 1000 \\ A_2 : 1000 \\ \vdots \quad\quad \vdots \\ \boxed{A_{100}} : 1000 \end{array} \right\}$ 99,000 helps

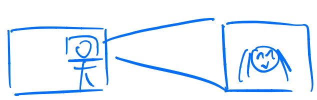
End-to-End Deep Learning

e.g. Speech recognition

audio $\xrightarrow{\text{MFCC}}$ feature $\xrightarrow{\text{ML}}$ phonemes $\rightarrow$ words $\rightarrow$ transcript (y)

audio $\longrightarrow$ transcript (need more data)

e.g. Face recognition at entrance



$\longrightarrow$ Identity

more data for both tasks !

Whether to use end-to-end DL ?

Pros: Let the data speak
Less hand-designing of components needed

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