

Course 3

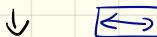
Deeplearning - Coursera Andrew NG

orthogonalization

ex) 회전각을 조정하는 것을 하나의 차원으로 생각하고 다른 차원을 속도조절로 생각
하나의 베른은 회전각만 영향을 주고 다른 베른은 가속페달이나 브레이크를
통해 속도를 조절하게 하는 것과 같다?

chain of assumption in ML

Fit training set well on cost function → bigger network



Adam 최적화



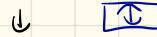
early stopping

: 딜리버리된 것

신경망에서 잘

사용하건 않는다.

Fit dev set well on cost function → regularization



Fit test



→ Bigger Devset



test set에서 인도하면 Dev로 돌아가야지

Performs well in real world

→ change dev set or
cost function.

Precision 95% = 95%의 학습률을 전자 고양이이다.

Recall 90% = 실제 고양이 사건 중에서 본류기기에 의해 정확히
맞파센트가 인식되었느냐.

정밀도와 재현율은 종종 트레이드오프 관계

정밀도와 재현율을 결합한 평가 기준을 새롭게 정의.

classifier	Precision	Recall	F1 score
A	95%	90%	92.4%
B	98%	85%	91.0%

=>

$$\frac{2}{(1/P)+(1/R)}$$

→ 평균

Example

Algorithm	US	China	India	Other
A	3%	9%	5%	9%
B	5%	6%	5%	10%

Average

6.1.

6.5%.

Another cat classification example
 Optimizing → Satisficing

classifier	Accuracy	Running time
A	90%	80ms
B	92%	95ms
C	95%	150ms

$$\text{Cost} = \text{accuracy} - 0.5 \times \text{running time}$$

maximize accuracy subject to
 running time \leq 100ms 꼭 그해야하는건 아니고
 이거면 좋다

N matrices | 1 optimizing
 N-1 satisfying

Region.

- US
- UK
- other Europe
- South America
- India
- China
- other Asia
- Australia

} Dev

} Test

=> 서로 다른 지역마다 때문이다.

좋은 성능을 내지 못함.

8개 지역을 전부 섞어서 개발세트와
 시험세트를 만들어내는것이 좋다.

Dev set & test set : same distribution.

70%		30%		100 1,000 10,000	
Train		Test			
60%		20%	30%	1% 1%	
Train	Dev	Test			
98%					

Test set: 최종 시스템 성능을 시험 세트를 이용하여 평가.
No test set might be okay.

Cat data set example

Metric: classification error

Algorithm A: 3% error → 절대 나오면 안되는 것도 나온다.

Algorithm B: 5% error → 그걸 받아야 가짜다.

$$\text{Error} = \frac{1}{m^{(i)}} \sum_{j=1}^{m^{(i)}} \mathbb{I}[y_{\text{pred}}^{(j)} \neq y^{(j)}]$$

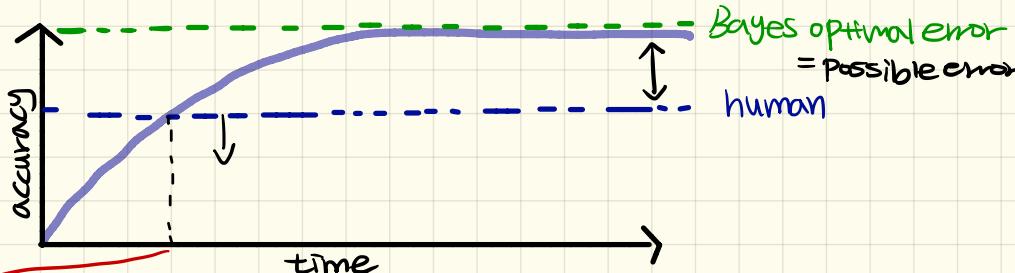
오류에 있는 수이 3%인지는 여전히 수는 인디케이터.

$$w^{(i)} = \begin{cases} 1 & x^{(i)} \text{ is non-porn} \\ 0 & x^{(i)} \text{ is porn} \end{cases}$$

1. 목표가 어디인지 정하는 것. 척도 정의

2. 척도에 대해 좋은 성능을 내는 방법을 찾는 것

내 척도와 Dev, test에서 좋은 결과를 내도
실제 사용자에게서 그리 좋지 않은 경우
척도, Dev, test를 바꾸는 것을 고려해봐야 한다.



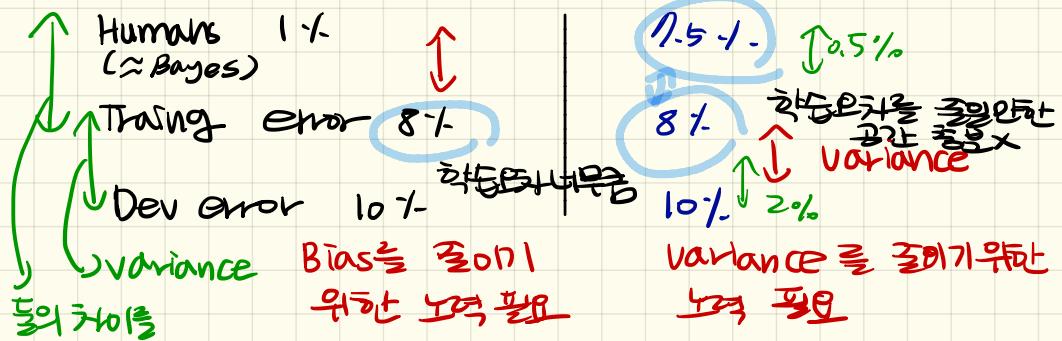
비이지안 추적모형: 가능한 최저의 오차값.
 $x \rightarrow y$ 로의 추적값 더 이상 좋은값 x

→ 사람 성능을 넘는 순간 속도가 감소.

① 사람 성능을 뛰어넘은 뒤에는 더 성능을 높일 만한
지수가 없을 수도 있다.

② 사람 성능을 뛰어넘기 전에 뛰어넘은 뒤 보다 더 쉽게
특정 방법을 이용해서 성능을 높일 수 있다.

Cat classification example



Avoidable bias : 더 이상 납득할 수 없는 편향, 체소문제가 있다.

Human-level error as a Proxy for Bayes error

- Typical human ... 3% error
- Typical doctor ... 1% error
- Experienced doctor ... 0.7% error
- Team of experienced doctors ... 0.5% error

What is "human-level" error? Bayes error $\leq 0.5\%$

Human (proxy for Bayes error) $\{ \begin{matrix} 1\% \\ 0.7\% \\ 0.5\% \end{matrix} \}$

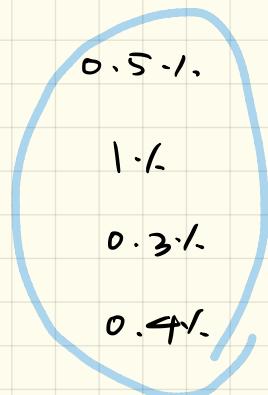
	Avoidable bias $4 \sim 4.5\%$	$0 \sim 0.5\%$	0.5%
Training Error	5%	1%	4%
Dev Error	6%	5%	0.1%
Bias	\uparrow	\uparrow	\uparrow

Team of humans 0.5%

One human 0.1%

Training error 0.6%

Dev error 0.8%



이런 경우 물줄여야
할지 쉽지 않다.

Surpasses Human-level

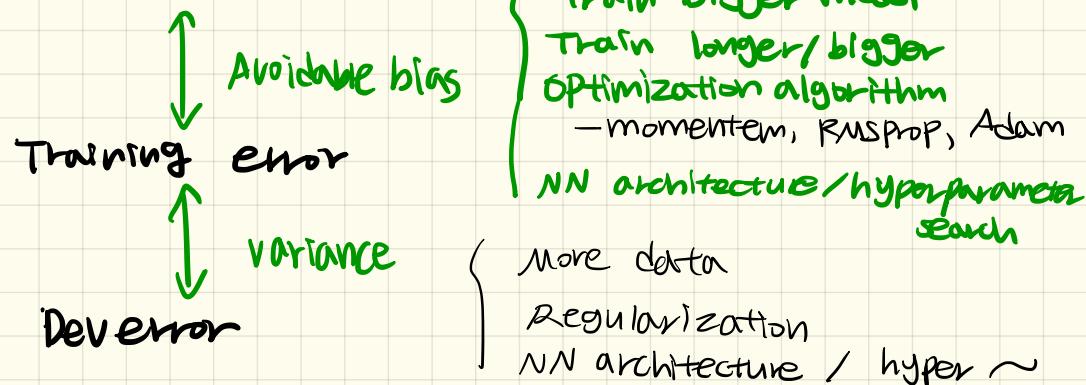
- online advertising
- product recommendations,
- logistics
- loan approval
- speech recognition
- some image recognition
- medical

The two fundamental assumptions of supervised learning

1. You can fit training set pretty well.
2. The training set performance generalize pretty well to the dev/test set.

Reducing bias and variance

Human-level



90% accuracy

10% error

오차분석을 통해 더 좋은 결과 도출 가능

- Get ~ 100 mislabeled dev set examples.
- Count up how many are dogs.
real

Ceiling : 최고로 좋은 경우.

Ideas for cat indicator

- Fix Pictures of dogs being recognized as cats.
- Fix great cats (lions, Panthers, etc...) being misrecognized.
- Improve performance on blurry images.

Image	Dog	Cat or Cats	Blurry	Comments
1	✓			Pitbull
2			✓	
3				
:		✓	✓	
	8%	43%	61%	이를 알 수 있다.

DL Algorithm are quite robust to random errors in the training set.

OVER all dev set error : 10%

Errors due incorrect labels : 0.6% ← 주는 부분 처리

Errors due to other causes : 9.4% 고지면 초기
하지만 표시 X

over all dev set error : 2%

errors due incorrect labels : 0.6% < 고치는게 좋다

Errors due to other causes : 1.4%

Correcting incorrect dev/test set ex.

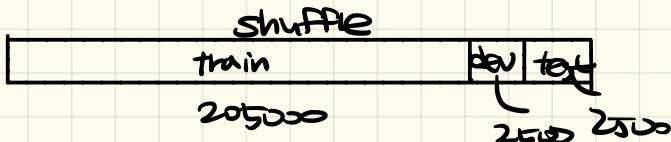
- 우선 개발세트와 시험세트가 다른 동일한 고정을 동시에 적용되라.
개발세트와 시험세트는 동일한 블포
- 알고리즘의 맛는 예시들을 확인하고 틀린부분이 있는지 확인한다.
- 개발/시험세트보다 큰 훈련세트를 고치는데 수고는 필요는 없다.

<Dev/Test Set 고치기>

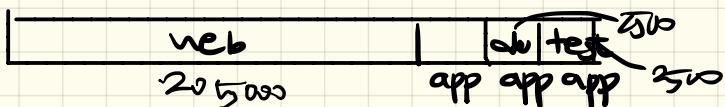
- ✓ set up dev/test set and metric
- ✓ build initial system quickly
- ✓ use Bias/Variance analysis & error analysis to prioritize next steps

web page PIC : 200000 app PIC : 10000

option 1:



option 2:



training

Purchased data

Smart speaker control

Voice keyboard

≈ 500000

Dev / test

Speech activated
headview mirror

≈ 2000

different



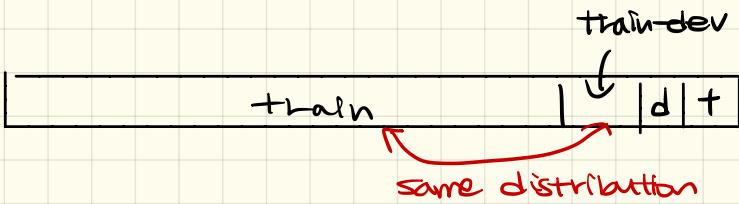
Cat classifier example

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

Training Error ... 10% \downarrow 9%

Dev Error ... 10%

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



Training error	1%	↓	variance on 문제 일반화하기 용도
Train-dev "	9%		
Dev error	10%		

"	1-10	↑ ↓
"	1.5%	
"	10%	

Data mismatch
Different dist

human error	0%	↓ ↓ ↓ ↓	Avoidable bias 10% 11% 12% 20% ↑ data mismatch
	10%		
	11%		
	12%		

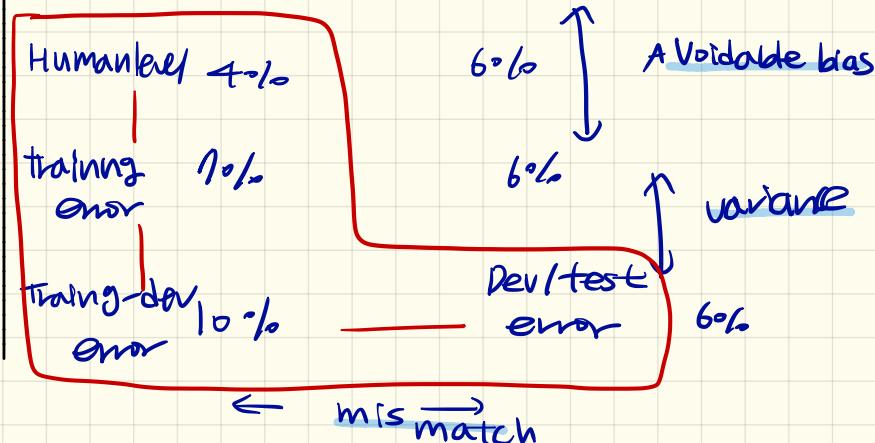
Avoidable bias
10%
11%
12%
20%
↑ data mismatch

Human level	4%	↑	avoidable
Training set	7%		
Train-dev	10%	↑	variance
Dev error	12%		
Test error	12%	↑	data mismatch degree of overfit to dev set
	12%		

General speech recognition

~ minor
Speech data

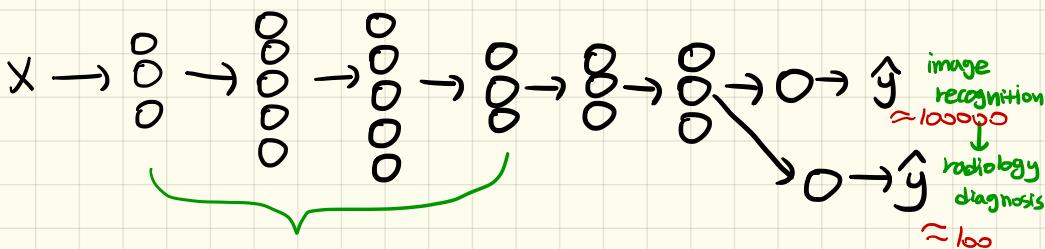
Human
Eharon
train
Error = 11
example
Not train

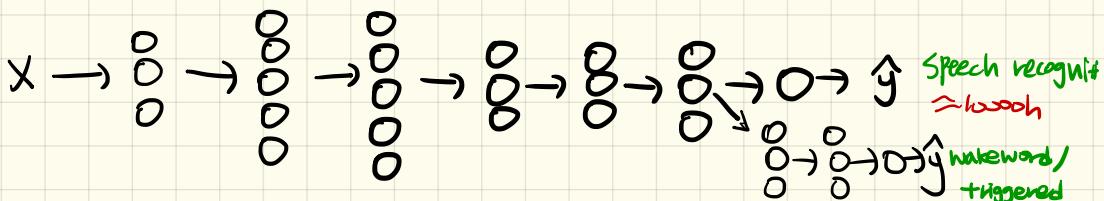


Addressing data mismatch

- Carry out manual error analysis to try to understand difference between training and dev/test sets
- Make training data more similar or collect more data similar to dev/test sets.

Transfer learning



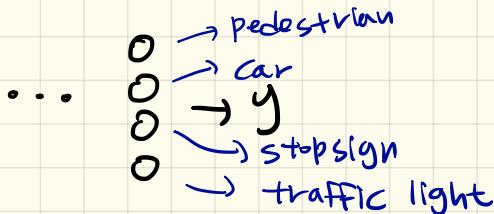


큰 데이터 \rightarrow 작은 데이터 전이.
(\Leftarrow)

트제가 되는 것 아니지만 유의미한 결과 예상 어려움.

Transfer from A \rightarrow B

- Task A and B have 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



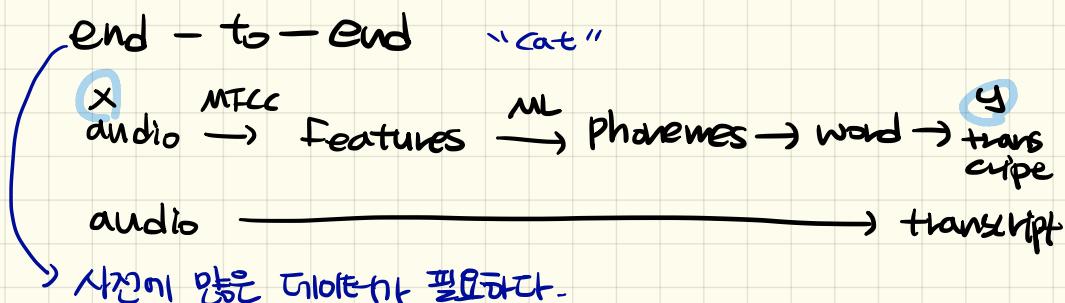
: 한 이미지가 여러 물체를 가지는 경우.

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

Sum only over
d's with o/
label

When multi-task learning make sense

- 낮은 수준의 특성을 공유할 때 이미이 되는 작업들을 학습하는 경우.
- 각 작업이 비슷한 양의 데이터를 가지는 것 (엄격하게 지켜져야 할 부분은 아니다)
- 모든 작업이 대해 충분히 큰 신경망을 훈련할 수 있어야 한다.



<두 가지 알고리즘이 더 좋은 이유>

1. 각각의 모델이 더 간단해질 때
2. 많은 자료들이 각각의 작업에 적용될 때

Pros

- Let the data speak $x \rightarrow y$
- 단순화

cons.

- 많은 데이터 양이 필요 (x, y)
- 유용하게 쓰일 것들을 배제한다.