ML Strategy (1)

ML Strategy: Motivation

Ideas:

- Collect more data
- Collect more diverse training set
- Train algorithm longer with gradient descent
- Try Adam instead of gradient descent
- Try bigger network
- Try smaller network
- Try dropout
- add L_2 regularization
- Network architecture
- Activation functions
- # hidden units
- ...

직교화(Orthogonalization)

어떤 파라미터를 조정했을 때 어떤 효과를 가져오는지

ex)자동차: 핸들은 좌우만, 엑셀은 전후방 가속만, 브레이크는 전후방 감속만 조정

Chain of assumpstions in ML

1. Fit training set well on cost function
   → bigger network, Adam, ...
2. Fit dev set well on cost function
   → regularization bigger training set
3. Fit test set well on cost function
   → bigger dev set
4. Performs well in real world
   → change dev set or cost function

※ early stoppping을 1번과 2번 단계에서 사용할 수는 있지만 ng교수는 early stopping을 별로 사용하지 않는다. training set에 얼마나 잘 맞추냐가 관건이기 때문에 너무 일찍 정지하면 training set을 잘 못 맞출 수 있다. dev set에 사용하면 동시에 2개의 요소가 영향을 받기 때문에 덜 직교화될 수 있다.

Setting up goal

Single number evaluation metric

		실제 정답	
		positive	negative
실험 결과	positive	true positive	false positive
	negative	false negative	true negative

1. Precision: $\frac{true\,positive}{true\,positive + false\,postivie}$

2. Recall: $\frac{true\,positive}{true\,positive + false\,negativie}$

3. F1 score: Average of precision and recall $\frac{2}{\frac{1}{precision} + \frac{1}{recall}}$

classifer에 따라 recall가 혹은 precision가 좋을 수 있다. 여러 아이디어와 여러 하이퍼 파라미터를 시도하면서 2가지 classifier만 시도할 것이 아니라 약 12가지의 다양한 classifier를 테스트하면서 가장 좋은 것을 고르는 것이 좋다. 12가지 중에서 가장 좋은 것을 선택하는 방법은 평균을 내보는 것이다.

2가지 metric으로는 classifier를 고르는게 쉽지 않을 수 있다. 따라서 ng 교수의 추천은 precision과 recall을 결합시킨 새로운 metric인 F1 score이다.

Satisficing and optimizing metrics

– Optimizing metric

– Satisficing metric

ex) maximizing accuracy(Optimizing metric)

subject to running_time ≤ 100ms(Satisficing metric)

⇒ N metrics: 1 optimizinig & N–1 Satisficing metrics

– Wake wordsTrigger words

ex) Hi Bixby, Hey Clova, Hey Kakao, Ok Google

Train/Dev/Test distiribution

Dev: development set, hold out cross validation set

받아온 데이터에 따라 다른 분포를 가질 수 있으므로

Randomly shuffle into dev/test set

⇒ Choose a dev set and test set to reflect data you expect to get in the

future and consider important to do well on.

→ det set과 test set은 same distribution

Size of dev and test sets

data가 10,000개 이하로 적을 때

70%	30%
Train set	Test set

or

60%	20%	20%
Train set	Dev set	Test set

data가 10,000개를 넘어 많을때(big data)

98%	1%	1%
Train set	Dev set	Test set

Size of dev set:

Set your dev set to be big enough to detect differences in algorithm/models you're trying out

Size of test set:

Set your test set to be big enough to give high confidence in the overall performance of your system.

When to change dev/test sets and metrics

준비했던 metric이 실제 선호를 반영하지 못하면 새로운 평가 metric을 도입해야 한다.

ex) metric: classification error

algorithm A: 3% error but not filter pornographic

algotithm B: 5% error

⇒ metric & dev: algo. A BUT users(reality): algo. B

⇒ change metric:

$$error = \frac{1}{m_{dev}} \sum_{i=1}^{m_{dev}} I\left\{ y_{pred}^{(i)} \neq y^{(i)} \right\}$$

$$\rightarrow error = \frac{1}{\sum_i w^{(i)}} \sum_{i=1}^{m_{dev}} w^{(i)} I\left\{ y_{pred}^{(i)} \neq y^{(i)} \right\}, \quad w^{(i)} = \begin{cases} 1 & \text{if } x^{(i)} is \mid porn \\ 10 & \text{if } x^{(i)} is\, porn \end{cases}$$

$I\{\ \}$: indicator. 안에 들어있는 내용 중 조건이 사실인 것의 개수

Orthogonalization for cat picture: anti-porn

1. 상황에 맞는 metric 설정

   So far we've only discussed how to define a metric to evaluate classifiers.

2. 1번에서 정한 metric에 적합하게 model을 고치기

   Worry separately about how to do well on this metric.

가중 평균을 이용한 metric이었으면 모델의 cost J도 이와 적합하게 고치기

$$J=\frac{1}{m}\sum_{i=1}^{m}L(\hat{y}^{(i)},y^{(i)}) \rightarrow J=\frac{1}{\sum w^{(i)}}\sum_{i=1}^{m}w^{(i)}L(\hat{y}^{(i)},y^{(i)})$$

dev/test set의 데이터와 user의 데이터의 quality 차이가 있을 수 있다.

학습시킬 때는 high quality images지만 실제 user images는 low quality일 수 있다.

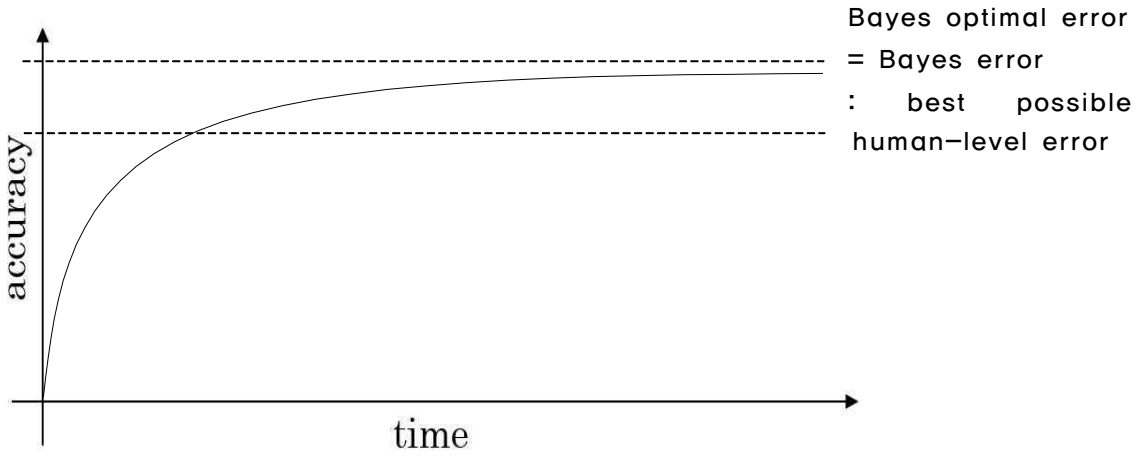
ng 교수가 추천하는 것은 metric과 dev set을 완벽히 정하기 어려우니까 팀의 반복 수행을 위해 빨리 한가지로 세팅하는 것이다. 나중에 프로젝트 진행 중 안 좋은 것으로 판명되거나 더 좋은 아이디어가 있으면 그때 바꾸면 된다.

권장하지 않는 것은 metric이든 dev set이든 오래 만들거나 세팅하는 것이다. 이러면 속도가 느려지고 팀의 생산성에 문제가 생겨 결과적으로 알고리즘 개선에 제한이 생긴다.

Comparing to human-level performance

머신러닝과 인간 레벨을 비교하는 이유

1. DL의 발전으로 ML 알고리즘이 더 잘 작동하여 인간 레벨과 비교했을 때 견줄 만한 정도로 실효성이 입증되었다.
2. ML 시스템을 디자인하고 수행절차를 설립하는 과정이 인간이 할 수 있는 영역에 많이 효율적으로 발전했다.



Bayes error에 도달 혹은 넘을 수 없는 2가지 이유

1. human-level 이전까지는 가파르게 발전하지만, human-level 넘으면 더 발전할 수 있는 부분이 제한적이다.
2. human-level 이전에서 발전이 더디면 여러 가지 방법으로 성능을 발달시킬 수 있지만 이런 방법은 human-level을 넘으면 사용하기 어렵다.

human-level 이전에 발전이 더딜 때 할 수 있는 방법

1. Get labeled data from human
2. Fain insight from manual error analysis:
   Why did a person get this right?
3. Better analysis of bias/variance

Avoidable bias

human-level error ≈ Bayes error	1%	↕ 7%	7.5%	↕ 0.5%	↕ Avoidable bias
Training error	8%	↕ 2%	8%	↕ 2%	↕ Variance
Dev error	10%		10%		
	Focus on bias		Focus on variance		

Human-level error as a proxy for Bayes error.

Understanding human-level performance

human-level error as a proxy for bayes error

ex) medical image classification

Suppose:

(a) Typical humans	3 % error
(b) Typical doctor	1 % error
(c) Expperienced doctor	0.7 % error
(d) Team of experienced doctors	0.5 % error

What is "human-level" error?

human-level error를 proxy for bayes error와 같은 수준으로 보면 (d)

즉, bayes error ≤ (d) 0.5%

논문이나 시스템 도입에 따라 human-level error의 정의가 다를 수 있으나 (b) 보다는 좋아야 할 것이다.

bias와 variance가 둘다 높을 수 있다. 이것이 바로 인간레벨을 도달하기 위해 bias와

variance를 제거하기 굉장히 어렵다. 잘하면 잘할수록 머신러닝 모델 발전을 이루가 점진적으로 어려워진다.

Summary

0%
↕ Bias
Training error

Human-level error ≈ Bayes error
↕ Avoidable bias
Training error
↕ Variance
Dev error

Surpassing human-level performance

if Training error < human-level error

overfitting? or bayes error≠human-level error?

⇒ need more data. 결국 비교할 대상, 자료가 부족해서 확신할 수 없다. bias에 문제가 있을 수도, variance에 문제가 있을 수도 어느 것을 보정해야 할지도 확신할 수 없다.

Improving your model performance

the two fundamental assumptions of supervised learning

1. you can fit the training set pretty well.
   간단히 말해서 낮은 avoidable bias를 획득할 수 있다.
2. The training set performance generalizes pretty well to the dev/test set.
   dev set과 test set에도 잘 작동되는 경우가 많다. 편차가 나쁘지 않다는 것을 말한다.

Reducing (avoidable) bias and variance

Human-level error ≈ Bayes error		– Train bigger model – Train longer/better optimization algorithms – NN architecture/hyperparameters search
↕ Avoidable bias	⇒	
Training error		– More data – Regularization – NN architecture/hyperparameters seaech
↕ Variance	⇒	
Dev error		