LG Advanced Data Scientists Program
Deep Learning

[1: Foundations of Deep Learning]

Prof. Sungroh Yoon

Electrical & Computer Engineering | Seoul National University

(last compiled at 20:54:00 on 2020/02/09)

Outline

References

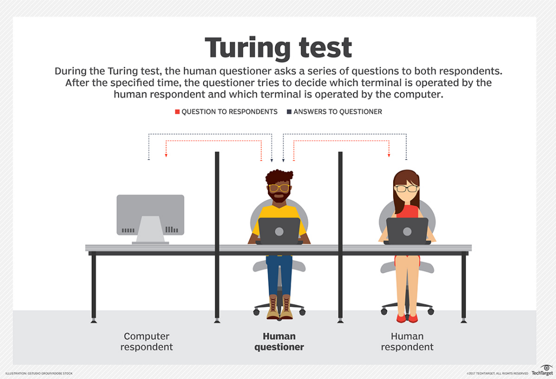
- *Deep Learning* by Goodfellow, Bengio and Courville ▸ Link
  - Chapters 1–5
- online resources:
  - *Deep Learning Specialization (coursera)* ▸ Link
  - *Stanford CS231n: CNN for Visual Recognition* ▸ Link

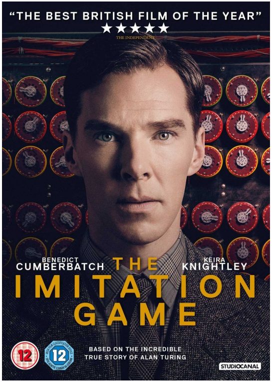
Outline

Artificial intelligence (AI)

- objective
  - to create a machine that can think and/or act like $\underbrace{\text{humans}}_{\substack{\uparrow \\ \text{think/act rationally}}}$
  - AI = computational ________
- rationality in engineering
  - refers to maximizing expected utility
- evaluation metric
  - human-level performance (suggested from day one)

- Turing test: the imitation game metric



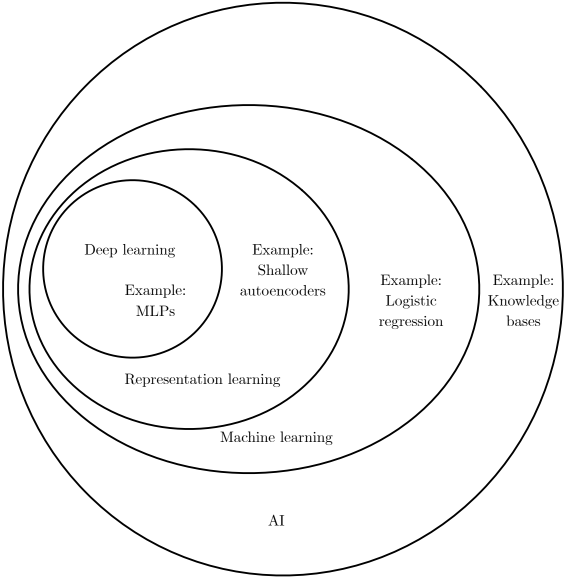


(source: http://searchenterpriseai.techtarget.com)

- Google Duplex ▸ Clip
  - human-level intelligence?

Comparison

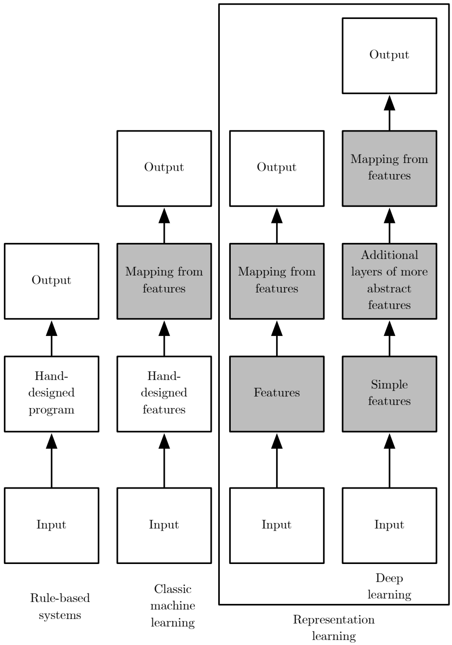
$\underbrace{\text{deep learning}}$
$\uparrow$
$\underbrace{\text{representation learning}}$
$\uparrow$
$\underbrace{\text{machine learning}}$
$\uparrow$
AI



Deep learning

- hierarchical representation learning
  - implementation: neural nets
  - fueled by big data
  - workhorse: GPU
- each layer in neural nets
  - ______ representation
- main applications
  - tasks humans can do well

(shaded boxes: components that are able to learn from data)



Status quo

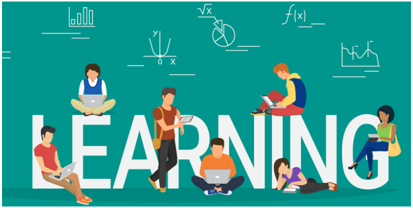
- subhuman performance
  - general intelligence
  - domains with small/pricey data, expensive human experts (*e.g.* medical)
- human-level performance
  - some perception tasks: visual/speech recognition
- superhuman performance
  - domains with ________ big data (*e.g.* recommendation, online AD)
  - some perception tasks (*e.g.* massive surveillance), game play

Outline

Machine learning

- learning from ____
- what do we mean by learning?
  - Mitchell (1997):

    "A computer program is said to learn from experience E
    with respect to some class of tasks T and performance measure P,
    if its performance at tasks in T,
    as measured by P,
    improves with experience E."
- common types:
  - supervised
  - unsupervised
  - reinforcement
  - many more



Tasks in ML

- described in terms of how to process an **example**
- an "example":
  - a collection of **features** quantitatively measured from object/event
  - represented as a vector $\boldsymbol{x} \in \mathbb{R}^n$ (each entry x_i : a feature)

  e.g. features of an image: pixels values

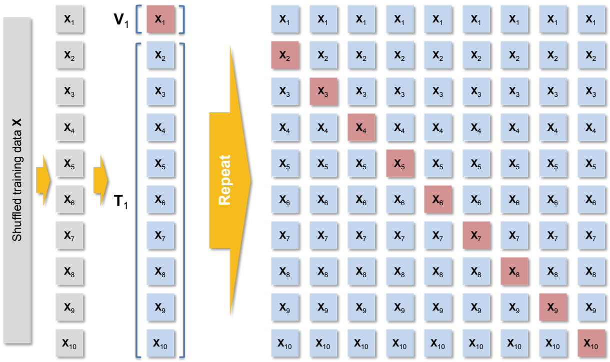
- common ML tasks:

  T1. classification
  T2. classification with missing inputs
  T3. regression
  T4. transcription
  T5. machine translation
  T6. structured output
  T7. anomaly detection
  T8. synthesis and sampling
  T9. imputation of missing values
  T10. denoising
  T11. density/pmf estimation

Data set

- a collection of examples
  - training set: for fitting
  - validation set ("dev set"): for model selection
  - test set: for ______

10-fold
cross-validation:



Performance measure

- specific to task T

  e.g. classification: accuracy, error rate E $\leftarrow$ we focus on this for a while

  density estimation: average log-probability the model assigns to examples

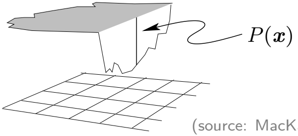
- evaluated using data sets
  - training/dev/test sets $\Rightarrow$ $\mathrm{E}_{\mathrm{train}}$, $\mathrm{E}_{\mathrm{dev}}$, $\mathrm{E}_{\mathrm{test}}$

- often challenging to choose
  1. difficult to decide what to measure

     e.g. penalize frequent mid-sized mistakes or rare large mistakes?
  2. know ideal measure but measurement is _________

     e.g. density estimation

a lake whose depth at $\boldsymbol{x} = (x, y)$ is $P(\boldsymbol{x})$



(source: MacKay)

Central challenge in ML

- $\boxed{}$
  - ability to perform well on previously unobserved examples
- generalization error $\mathrm{E}_{\mathrm{gen}}$
  - expected error on a new example $\Rightarrow$ implausible to calculate
- training error $\mathrm{E}_{\mathrm{train}}$
  - measured on a training set $\Rightarrow$ bad proxy for $\mathrm{E}_{\mathrm{gen}}$
- test error $\mathrm{E}_{\mathrm{test}}$
  - measured on a test set (not used in training) $\Rightarrow$ better proxy for $\mathrm{E}_{\mathrm{gen}}$

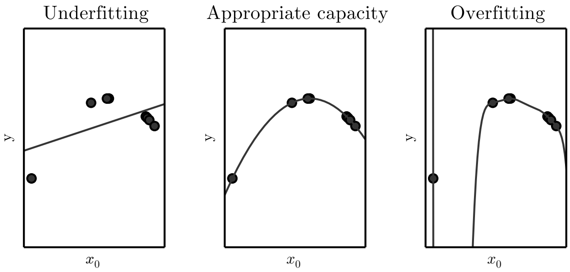
Two specific objectives

- objective: $\boxed{\mathrm{E}_{\mathrm{gen}} = 0}$ in theory or $\boxed{\mathrm{E}_{\mathrm{test}} \simeq 0}$ in practice
- split into two objectives:
  1. $\mathrm{E}_{\mathrm{test}} \simeq \mathrm{E}_{\mathrm{train}}$
  2. $\mathrm{E}_{\mathrm{train}} \simeq 0$
- objective 1: make $\mathrm{E}_{\mathrm{test}} \simeq \mathrm{E}_{\mathrm{train}}$
  - failure: ________ $\rightarrow$ high variance
  - cure: regularization, more data
- objective 2: make $\mathrm{E}_{\mathrm{train}} \simeq 0$
  - failure: underfitting $\rightarrow$ high bias
  - cure: optimization, more complex model

Capacity of a model

- the ability of the $\underbrace{\text{model}}_{\substack{\uparrow \\ \text{representation (+ learning algorithm)}}}$ to fit various functions

- altering capacity controls over/underfitting
  - example (truth: quadratic; fit: linear, quadratic, degree-9)



Choosing a model (conventional advice)

- Occam's razor (a principle of parsimony)
  - among competing hypotheses, choose the "_______" one
- why? **VC generalization bound**: for any $\epsilon > 0$ and $N > 0$

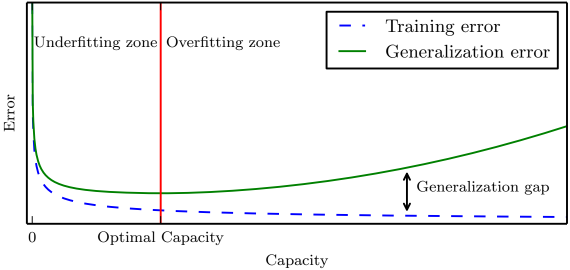
$$\mathbb{P}[\underbrace{\overbrace{|\mathrm{E}_{\text{train}}(f) - \mathrm{E}_{\text{test}}(f)|}^{\text{generalization gap}} > \epsilon}_{\text{bad event}}] \leq \underbrace{4 \cdot (2N)^{\overbrace{\color{red}{d_{\text{VC}}}}^{\text{capacity}}} \cdot e^{-\frac{1}{8}\epsilon^2 N}}_{\text{VC bound}}$$

  - N : # of training examples
  - f : a model (d_{VC} : its *VC dimension*, a measure of model capacity)
- in words: discrepancy between $\mathrm{E}_{\text{train}}$ and E_{test}
  - grows as model capacity grows (but $\underbrace{\text{shrinks as } N \text{ increases}}_{\uparrow \ \text{power of big data}}$)

A tradeoff: the main challenge in ML

- approximation-generalization tradeoff or bias-variance tradeoff

$$\underbrace{\mathrm{E}_{\mathrm{test}} \simeq \overbrace{\mathrm{E}_{\mathrm{train}}}^{\text{complex model is better}}}_{\text{simple model is better}} \simeq 0$$



- in theory: choose simpler functions
  - better generalization (smaller gap between training/test error)
- in practice: must still choose a sufficiently complex hypothesis
  - to achieve low training error

Two major weapons to fight the tradeoff

- **optimization**: ___ reduction (better approximation)
  - finds model parameters that minimize error

  e.g. stochastic gradient descent

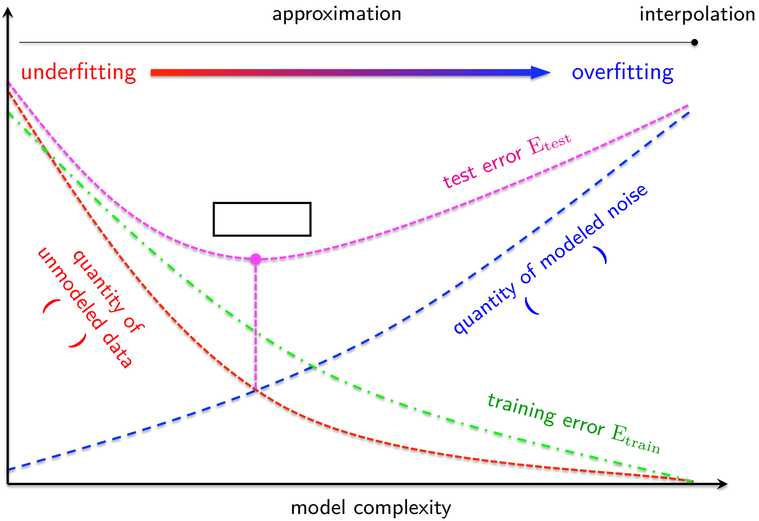
- **regularization**: _______ reduction (better generalization)
  - constrains model capacity by reflecting prior knowledge

  e.g. dropout, weight decay

Choosing a model (modern advice)

- $\boxed{\text{complex model + effective} \qquad\qquad\qquad \text{+ big data}}$
- complex model
  - higher chance of fitting data $\rightarrow \mathrm{E}_{\text{train}} \simeq 0$
- regularization + big data
  - reduces generalization gap $\rightarrow \mathrm{E}_{\text{test}} \simeq \mathrm{E}_{\text{train}}$

Big picture


approximation
interpolation
underfitting
overfitting
test error E_{test}
quantity of unmodeled data
quantity of modeled noise
training error E_{train}
model complexity

Outline

Linear models

- basis for more sophisticated models
- has many advantages $\rightarrow$ worth trying first
  - simplicity: easy to implement, test, and interpret
  - generalization: higher chance of $\mathrm{E}_{\text{test}} \simeq \mathrm{E}_{\text{train}}$ than complex models
  - extension: nonlinear transform, kernel trick, neural nets
- can solve three important problems
  1. classification
  2. regression
  3. probability estimation (*aka* _______ regression)
  - come with different but related algorithms

Example: credit card application

- given:
  - applicant information $\longrightarrow$
- decide:
  - approve a credit card or not?

feature	value
age	23 years
gender	female
annual salary	\$30,000
years in residence	1 year
years in job	1 year
current debt	\$15,000
...	...

Formalization

- let $\mathcal{X} = \mathbb{R}^d$ be the input space
  - $\mathbb{R}^d$: the d-dimensional Euclidean space
  - input vector $\mathbf{x} \in \mathcal{X}$: $\mathbf{x} = (x_1, x_2, \ldots, x_d)$
- let $\mathcal{Y} = \{+1, -1\}$ be the output space
  - denotes a ______________ decision
- in our credit example
  - coordinates of input $\mathbf{x}$:
    salary, debt, and other fields in a credit card application
  - binary output y: approved or denined

component	symbol	credit approval metaphor
input	$\mathbf{x}$	customer application
output	y	approve or deny
target function	$f: \mathcal{X} \to \mathcal{Y}$	ideal approval formula
data	$(\mathbf{x}^{(1)}, y^{(1)}), \ldots, (\mathbf{x}^{(N)}, y^{(N)})$	historical records
hypothesis	$g: \mathcal{X} \to \mathcal{Y}$	formula to be used

- f: unknown target function
- $\mathcal{X}$: input space (set of all possible inputs $\mathbf{x}$)
- $\mathcal{Y}$: output space (set of all possible outputs)
- N: the number of input-output examples (*i.e.* training examples)
- $\mathbb{X} \triangleq \{(\mathbf{x}^{(1)}, y^{(1)}), \ldots, (\mathbf{x}^{(N)}, y^{(N)})\}$: data set where $y^{(n)} = f(\mathbf{x}^{(n)})$

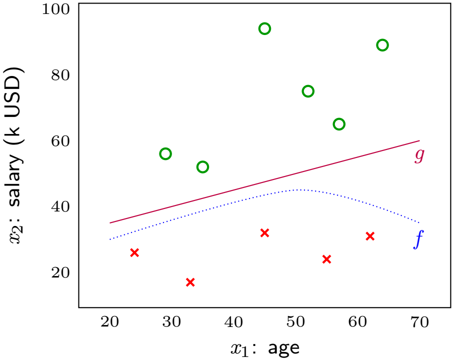
Example

- $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ where x_1: age and x_2: annual salary in USD

- $N = 11$, $d = 2$, $\mathcal{X} = \mathbb{R}^2$, and $\mathcal{Y} = \{\text{approve}, \text{deny}\}$

- data set $\mathcal{D}$:

n	x_1	x_2	y
1	29	56k	approve
2	64	89k	approve
3	33	17k	deny
4	45	94k	approve
5	24	26k	deny
6	55	24k	deny
7	35	52k	approve
8	57	65k	approve
9	45	32k	deny
10	52	75k	approve
11	62	31k	deny



Decision making

- to make a decision
  - weighted coordinates are combined to form a 'credit score'
  - the resulting score is then compared to a ________
- in our credit card approval example
  - for input $\mathbf{x} = (x_1, \ldots, x_d)$, 'attributes of an applicant':

$$\_\_\_\_\_\_ \text{ the application if } \sum_{i=1}^{d} w_i x_i > \text{threshold}$$

$$\_\_\_\_ \text{ the application if } \sum_{i=1}^{d} w_i x_i < \text{threshold}$$

The perceptron

- this linear formula can be written more compactly:

$$g(\mathbf{x}) = \operatorname{sign}\left(\left(\sum_{i=1}^{d} w_i x_i\right) - \text{threshold}\right) \tag{1}$$

$$= \operatorname{sign}\left(\left(\sum_{i=1}^{d} w_i x_i\right) + b\right) \tag{2}$$

where b is called the ___ and $\operatorname{sign}(z)$[1] $= \begin{cases} +1 & \text{if } z > 0 \\ -1 & \text{if } z < 0 \end{cases}$

- this model: called the **perceptron**
  - a simple linear classifier

[1] value of $\operatorname{sign}(z)$ when $z = 0$ is a simple technicality we can ignore

- different parameters $\boldsymbol{\theta} = (\underbrace{w_1, w_2, \ldots, w_d}_{\text{weights}}, \underbrace{b}_{\text{bias}})$
    - yield different hyperplanes $w_1 x_1 + w_2 x_2 + \cdots w_d x_d + b = 0$

- for simplification
    - treat bias b as a weight $w_0 \equiv b$
    - introduce an artificial coordinate ______

- with this convention, $\mathbf{w}^\top \mathbf{x} = \sum_{i=0}^{d} w_i x_i$
    - this gives the perceptron in vector form:

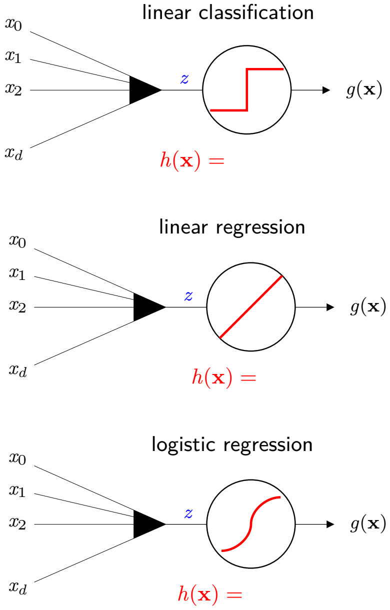
$$\boxed{g(\mathbf{x}) = \mathrm{sign}(\mathbf{w}^\top \mathbf{x})} \tag{3}$$

- $\mathbf{w}^\top \mathbf{x}$: called signal

Linear models

- based on "signal" z:

$$z = \sum_{i=0}^{d} w_i x_i$$



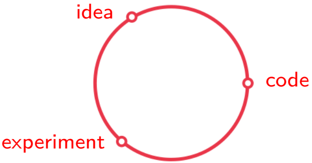
Comparison

	linear classification	linear regression	logistic regression
$\mathcal{Y}$	$\{-1, +1\}$	$\mathbb{R}$	$\{-1, +1\}$
$\hat{y} = g(\mathbf{x})$	$\mathrm{sign}(\mathbf{w}^\top \mathbf{x})$	$\mathbf{w}^\top \mathbf{x}$	$\theta^\star(\mathbf{w}^\top \mathbf{x})$
$\mathrm{e}(\hat{y}, y)$	0-1 loss $[\![\hat{y} \neq y]\!]$	squared error $(\hat{y} - y)^2$	cross-entropy error $[\![y{=}+1]\!] \ln \frac{1}{\hat{y}}$ $+[\![y{=}-1]\!] \ln \frac{1}{1-\hat{y}}$
$\mathrm{E}_{\mathrm{train}}(h)$	$\frac{1}{N} \sum_{n=1}^{N} [\![h(\mathbf{x}^{(n)}) \neq y^{(n)}]\!]$	$\frac{1}{N} \sum_{n=1}^{N} (h(\mathbf{x}^{(n)}) - y^{(n)})^2$	$\frac{1}{N} \sum_{n=1}^{N} \ln\left(1 + e^{-y^{(n)} \mathbf{w}^\top \mathbf{x}^{(n)}}\right)$
training	combinatorial optimization (NP-hard)	set $\nabla \mathrm{E}_{\mathrm{in}}(\mathbf{w}) = 0$ (closed-form solution exists)	set $\nabla \mathrm{E}_{\mathrm{in}}(\mathbf{w}) = 0$ iterative optimization (*e.g.* gradient descent)

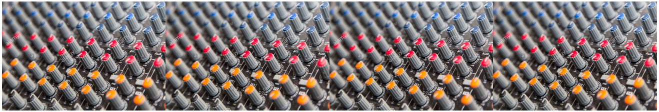
$\star$ logistic sigmoid $\theta(z) = 1/(1 + e^{-z})$

Outline

Motivation



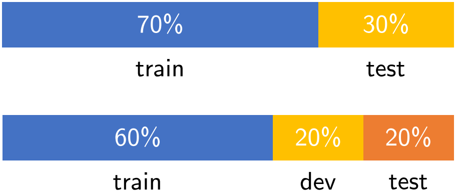
- deep learning
  - highly _______ process
- many knobs to tweak
  - data, metric, optimizer, regularizer, hyperparameters/architecture, ...



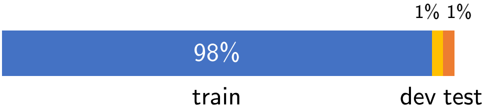
- how to accelerate this iterative process?
  - before autoML comes on earth

Data breakdown

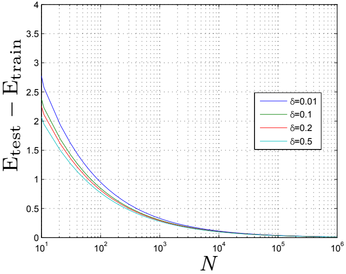
- small data ($n \approx$ 100-10,000):

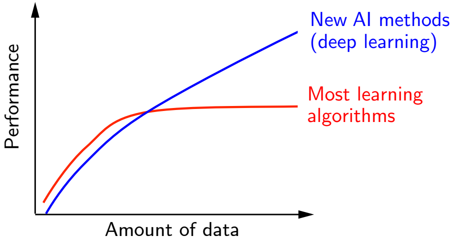


- big data ($n \approx$ 1,000,000):



Power of big data





- as $N \to \infty$
  - $\mathrm{E}_{\mathrm{test}} - \mathrm{E}_{\mathrm{train}} \to 0$ regardless of model/statistical confidence[2]
  - performance generally improves

[2] δ in left plot

How much data?

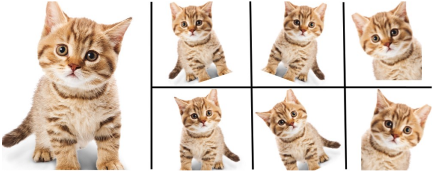
- highly dependent on ______ problems
- a rough rule of thumb (Goodfellow et al., 2016):
  - 5000 labeled examples per category
    - to achieve acceptable performance by supervised deep learning
  - at least 10 million labeled examples
    - to match/exceed human performance
- active research areas
  - pre-training and/or transfer learning
  - un/semi-supervised learning to use unlabeled data

When you do not have enough data

1. data augmentation
   - rotation, noise, translation

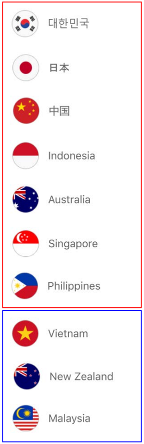
2. ________
   - AlphaGo Zero

3. generation
   - generative adversarial net (GAN)



Match data distributions

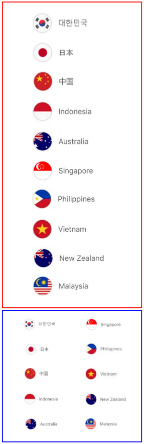
- dev distr $\neq$ test distr



dev set

test set

- dev distr $\approx$ test distr (better)



dev set

test set

Orthogonalization

(source: Porsche)

- 😟 to open window, press 0.3 of $\boxed{\text{bttn 1}}$ + 0.2 of $\boxed{\text{bttn 2}}$ + 0.5 of $\boxed{\text{bttn 3}}$
- 😀 just press bttn $\boxed{\text{open}}$
  - ________ knobs → more effective control

- orthogonalization in training ML models

desired task	**if you fail, try the following:** (orthogonal knob)	(less orthogonal knob)
fit train set well	bigger network better optimizer	early stopping
fit dev set well	regularization bigger training set	
fit test set well	bigger dev set	
perform well in real world	change dev set change cost function	

- early stopping (terminating training prematurely)
  - affects both training and validation performance $\Rightarrow$ less orthogonal
  - sometimes not recommended in deep learning training

Choosing a metric

- using a _____ real number evaluation metric
  - clear objective $\Rightarrow$ can speed up the iterative process
- optimizing and satisficing metrics
  - M metrics $\Rightarrow$ 1 optimizing metric $+$ $(M-1)$ satisficing metrics
  - example

classifier	accuracy	runtime
A	90%	80ms
B	92%	95ms
C	95%	1,500ms

$$\begin{aligned} \text{maximize} \quad & \overbrace{\text{accuracy}}^{\text{optimizing metric}} \\ \text{s.t.} \quad & \underbrace{\text{runtime}}_{\text{satisficing metric}} \le 100\text{ms} \end{aligned}$$

$\Rightarrow$ optimal: B

Setting (and adjusting) a target

- learning target: set by a metric + dev/test sets
  - bullets: shot by training sets

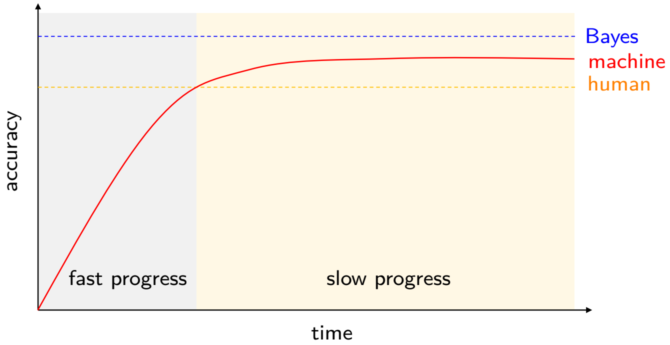
- change your metric and/or dev/test sets
  - if you experience bad ____________

    (*i.e.* have low test error but cannot handle new inputs well)

Referencing human-level performance

- _____ error (irreducible error): lowest possible error
- human error
  - often close to Bayes error (especially for natural perception tasks)

  $\Rightarrow$ used as a proxy for Bayes error $\Rightarrow$ target for ML



- when ML performance $<$ human performance: tools exist
  - more labeled data from humans
  - manual error analysis (why did humans get things right?)
  - better bias-variance analysis
- when ML performance $>$ human performance:
  - the above tools no longer useful
  - more difficult to improve machine learning

Bias-variance analysis

human ($\simeq$ Bayes):	1%	8.5%	
			bias
training error:	9%	9%	
			variance
dev error:	10%	10%	
	$\Downarrow$	$\Downarrow$	
	reduce **bias**	reduce **variance**	

- reducing ___
  - more complex model, longer training, better optimization
  - better hyperparameter/architecture
- reducing _______
  - more data, regularization
  - better hyperparameter/architecture

Outline

Summary

- deep learning: hierarchical representation learning
  - driving forces: big data, parallel hw (GPU), advanced algorithms
- machine learning: learn from data to achieve generalization
  - objectives: making $\mathrm{E}_{\mathrm{test}} \simeq \mathrm{E}_{\mathrm{train}}$ + making $\mathrm{E}_{\mathrm{train}} \simeq 0$
  - challenge: approximation-generalization or bias-variance tradeoff
  - weapons: big data, optimization, regularization
  - example: linear models for classification/regression/prob estimation
- data sets: train/dev/test
  - breakdown in big data era: train/dev/test $\simeq$ 98%/1%/1%
  - handling data scarcity: data augmentation, simulation, generation
- machine learning strategy: needed to accelerate iterative process
  - orthogonalization, optimizing/satisficing metrics, bias-variance analysis