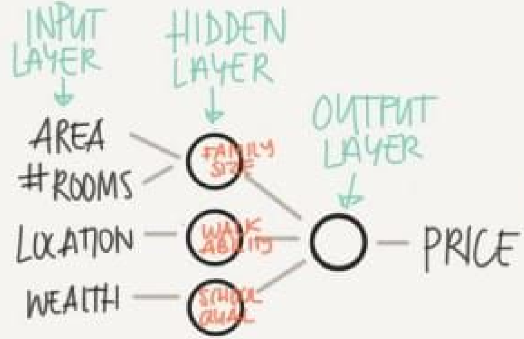
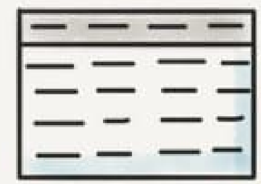
INTRO TO DEEP LEARNING



SUPERVISED LEARNING

INPUT: X	OUTPUT: y	NN TYPE
HOME FEATURES	PRICE	STANDARD NN
AD+USER INFO	WILL CLICK ON AD (0/1)	
IMAGE	OBJECT (1...1000)	CONV. NN (CNN)
AUDIO	TEXT TRANSCRIPT	RECURRENT NN (RNN)
ENGLISH	CHINESE	
IMAGE/RADAR	POS OF OTHER CARS	CUSTOM/HYBRID

NNs CAN DEAL WITH BOTH STRUCTURED & UNSTRUCTURED DATA



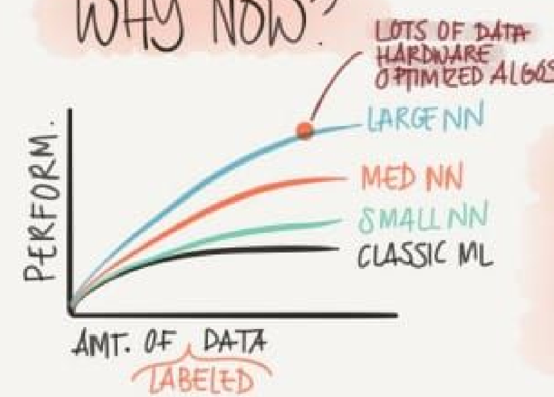
STRUCTURED

"THE QUICK BROWN FOX"

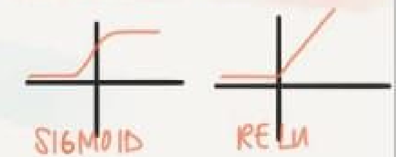
UNSTRUCTURED

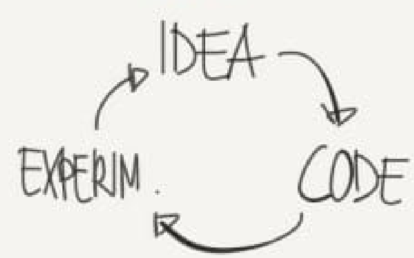
HUMANS ARE GOOD AT THIS

WHY NOW?



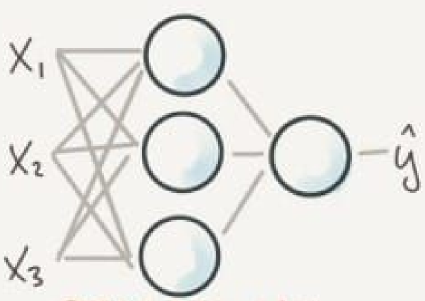
ONE OF THE BIG BREAKTHROUGHS HAS BEEN MOVING FROM SIGMOID TO RELU FOR FASTER GRADIENT DESCENT





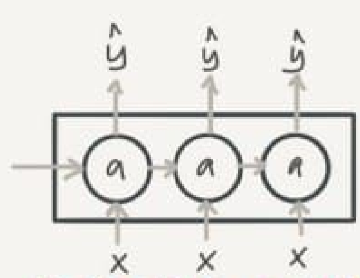
FASTER COMPUTATION IS IMPORTANT TO SPEED UP THE ITERATIVE PROCESS

NETWORK ARCHITECTURES



STANDARD NN

CONVOLUTIONAL NN



RECURRENT NN

NEURAL NETWORKS & DEEP LEARNING • COURSERA • ANDREW NG • WK 1

TRAIN vs DEV/TEST
MISMATCH

AVAILABLE DATA

200 k PRO CAT PICS FROM INTERNET

10 k BLURRY CAT PICS FROM APP — WHAT WE CARE ABT

HOW DO WE SPLIT → TRAIN/DEV/TEST?

OPTION 1: SHUFFLE ALL

205 k (TRAIN)	D	T

2.5 k

PROBLEM: DEV/TEST IS NOW MOSTLY WEB IMG (NOT REPRES. OF END SCENARIO)

SOLUTION: LET DEV/TEST COME FROM APP. THEN SHUFFLE 5k OF APP PICS w WEB FOR TRAIN

205 k	2.5	2.5
WEB+APP	APP	APP

BIAS & VARIANCE w MISMATCHED TRAIN/DEV

HUMANS	~0%
TRAIN	1%
DEV ERR	10%

IS THIS DIFF DUE TO THE MODEL NOT GENERALIZING OR IS DEV DATA MUCH HARDER

A: CREATE A TRAIN-DEV SET THAT WE DON'T TRAIN ON

TRAIN	TD	D	T

	A	B	C	D
TRAIN	1%	1%	10%	10%
TRAIN-DEV	9%	1.5%	11%	11%
DEV	10%	10%	12%	20%
	VARIANCE	TRAIN/DEV MISMATCH	BIAS	BIAS + DATA MISMATCH

ADDRESSING DATA MISMATCH

EX. CAR GPS · TRAINING DATA IS 10.000 H OF GENERAL SPEECH DATA

1. CARRY OUT MANUAL ERROR ANALYSIS TO UNDERSTAND THE DIFFERENCE (EX NOISE, STREET NUMBERS)
2. TRY TO MAKE TRAIN MORE SIMILAR TO DEV OR GATHER MORE DEV-LIKE TRAIN-DATA

AUDIO W CAR NOISE

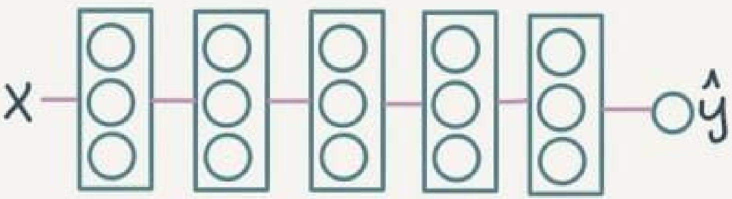
NOTE: BE CAREFUL · IF YOU ONLY HAVE 1 HR OF CAR NOISE & APPLY IT TO 10K HR SPEECH YOU MAY OVERFIT TO THE CAR NOISE

STRUCTURING ML PROJECTS · COURSERA

EXTENDED LEARNING

TRANSFER LEARNING

PROBLEM: YOU WANT TO CLASSIFY SOME MEDICAL IMG. YOU HAVE AN NN THAT CLASSIFIES CATS



NN TO CLASSIFY CATS/DOGS/CARS

$\boxed{\text{OPTION 1}}$: YOU ONLY HAVE A FEW RADIOLOGY IMAGES

SOLUTION: INIT W. WEIGHTS FROM CAT NN

ONLY RETRAIN LAST LAYER(S) ON RADIOLOGY IMAGES

$\boxed{\text{OPTION 2}}$ YOU HAVE LOTS OF RADIOLOGY IMG.

SOLUTION: INIT WITH WEIGHTS FROM CAT NN

RETRAIN ALL LAYERS

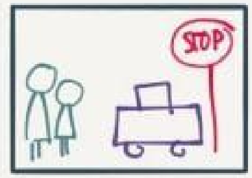
THIS IS MICROSOFT CUSTOM VISION

MULTI TASK LEARNING

TRAINING ON MULT. TASKS AT ONCE

DETECT
- CAR
- STOP SIGN
- PEDESTR.
- TRAFFIC LIGHT

$\hat{y} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \end{bmatrix}$



UNLIKE SOFMAX · MANY THINGS CAN BE TRUE

COST: $J(w,b) = \frac{1}{m}\sum_{i=1}^{m}\sum_{j=1}^{4}\mathcal{L}(\hat{y}_j^{(i)}, y_j^{(i)})$

SUMMING OVER ALL OUTP OPTIONS

WE COULD HAVE JUST TRAINED 4 NNS INSTEAD BUT... MT LEARNING MAKES SENSE WHEN

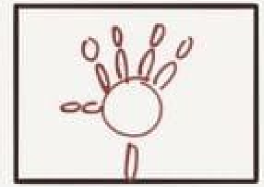
A. THE LEARNING DATA YOU HAVE FOR THE DIFF TASKS IS QUITE SIMILAR – & THE AMOUNTS (EG. 1K CARS, 1K STOP SIGNS)

B. THE SUM OF THE DATA ALLOWS YOU TO TRAIN A BIG ENOUGH NN TO DO WELL ON ALL TASKS

IN REALITY TRANSFER LEARNING IS USED MORE OFTEN

END-TO-END LEARNING

FROM X-RAY OF CHILDS HAND TELL ME THE AGE OF THE CHILD



TYPICAL SOLN:

1. LOCATE BONES TO FIND LENGTHS USING ML
2. TRAIN MODEL TO PREDICT AGE BASED ON BONELENGTH

END-TO-END

RADIOLOGY IMG $\longrightarrow$ CHILD AGE

PROS:
- LET'S THE DATA SPEAK (MAYBE IT FINDS RELATIONS WE'RE UNAWARE OF)
- LESS HAND-DESIGNING OF COMPONENTS NEEDED

CONS:
- NEEDS LARGE AMTS OF LABLED DATA ($X \rightarrow Y$)
- EXCLUDES POTENTIALLY USEFUL HAND-MADE COMPONENTS

STRUCTURING ML PROJECTS · COURSERA

TESS NOTES

ERROR
ANALYSIS

YOU HAVE 10% ERRORS, SOME ARE DOGS MIS-CLASSIFIED AS CATS. SHOULD YOU TRAIN ON MORE DOG PICS?

1. PICK 100 MIS-LABLED
2. COUNT ERROR REASONS

	DOG	BLURRY	INSTA FILTER	BIG CAT	...
1	1		1		
2				1	
3		1			
...					
100			1		
	5	...			

5% OF ALL ERRORS

FOCUSING ON DOGS - THE BEST WE CAN HOPE FOR IS 9.5% ERROR

STRUCTURING ML PROJECTS · COURSERA

YOU FIND SOME INCORR. LABLED DATA IN THE DEV SET. SHOULD YOU FIX IT?

DL ALGORITHMS ARE PRETTY ROBUST TO RANDOM ERRORS. BUT NOT TO SYSTEMATIC ERR.

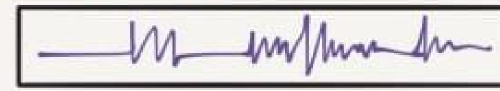
(EX. ALL WHITE CATS INCORR LABLED AS MICE)

ADD EXTRA COL. IN ERROR ANALYSIS AND USE SAME CRITERIA

NOTE IF YOU FIX DEV YOU SHOULD FIX TEST AS WELL.

FOR NEW PROJ. BUILD 1st SYSTEM QUICK & ITERATE

EX: SPEECH RECOGNITION



WHAT SHOULD YOU FOCUS ON?

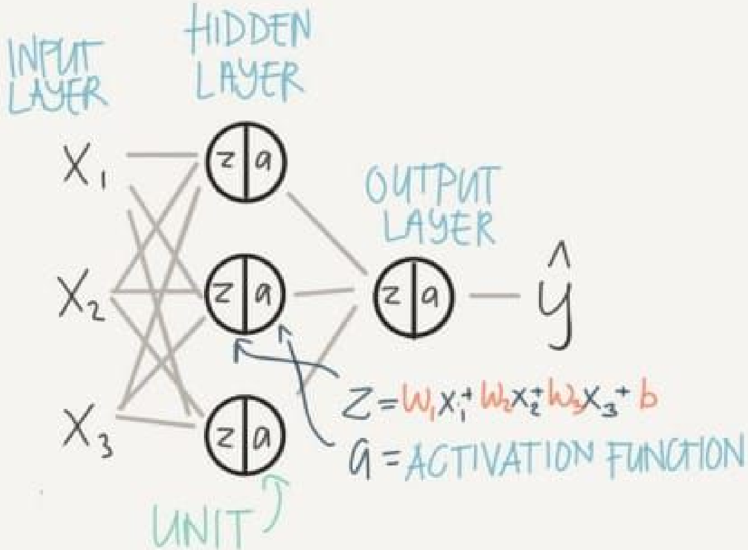
NOISE
ACCENTS
FAR FROM MIKE

1. START QUICKLY
   DEV/TEST METRICS
2. GET TRAIN-SET
3. TRAIN
4. BIAS/VARIANCE ANAL
5. ERROR ANALYSIS
6. PRIORITIZE NEXT STEP

NEURAL NETWORKS & DEEP LEARNING • COURSERA • ANDREW NG • WK 3

SHALLOW NEURAL NETS

2 LAYER NEURAL NET



ACTIVATION FUNCTIONS

SIGMOID σ: BINARY CLASSIFIER - ONLY USED FOR OUTPUT LAYER

SLOW GRAD DESCENT SINCE SLOPE IS SMALL FOR LARGE/SMALL VAL

TANH: NORMALIZED ⇒ GRADIENT DESCENT IS FASTER

RELU: DEFAULT CHOICE FOR ACTIVATION

SLOPE = 1/0

LEAKY RELU: AVOIDS UNDEF SLOPE AT Ø BUT RARELY USED IN PRACTICE

WHY ACTIVATION FUNCTIONS?

EX. WITH NO ACTIVATION - $a = Z$

$$a^{[1]} = Z^{[1]} = W^{[1]}X + b^{[1]} \quad \text{LAYER 1}$$
$$a^{[2]} = Z^{[2]} = W^{[2]}a^{[1]} + b^{[2]} \quad \text{LAYER 2}$$

PLUG IN $a^{[1]}$

$$a^{[2]} = W^{[2]}(W^{[1]}X + b^{[1]}) + b^{[2]}$$
$$= \underbrace{W^{[2]}W^{[1]}}_{W'}X + \underbrace{W^{[2]}b^{[1]} + b^{[2]}}_{b'}$$

$W' \; X \; + \; b'$ ← LINEAR FUNCTION

WE COULD JUST AS WELL HAVE SKIPPED THE WHOLE NEURAL NET & USED LIN. REGR.

INITIALIZING W+b

WHAT IF: INIT TO Ø

THIS WILL CAUSE ALL THE UNITS TO BE THE SAME AND LEARN EXACTLY THE SAME FEATURES

SOLUTION: RANDOM INIT

BUT ALSO WANT THEM SMALL SO RAND * 0.01 ← HYPERPARAM

@TessFerrandez

OPTIMIZING
TRAINING

NORMALIZING INPUTS

STEP1: CENTER AROUND Ø,Ø

STEP2: SCALE SO VARIANCE IS SAME (X −1 → 1

TIP
USE SAME AVG/VAR TO NORMALIZE DEV/TEST

WHY DO WE DO THIS?

IF WE NORMALIZE, WE CAN USE A MUCH LARGER LEARNING RATE α

DEALING WITH VANISHING/EXPLODING GRADIENTS

EX: DEEP NN (L LAYERS)

$$\hat{y} = \underbrace{W^{[L-1]} W^{[L-2]} \cdots W^{[1]}}\, x + b$$

IF $W = \begin{bmatrix} 0.5 & 0 \\ 0 & 0.5 \end{bmatrix} \Rightarrow 0.5^{L-1} \Rightarrow$ VANISHING

OR $W = \begin{bmatrix} 1.5 & 0 \\ 0 & 1.5 \end{bmatrix} \Rightarrow 1.5^{L-1} \Rightarrow$ EXPLODING

IN BOTH CASES GRADIENT DESCENT TAKES A VERY LONG TIME

PARTIAL SOLUTION: CHOOSE INITIAL VALUES CAREFULLY

$W^{[l]} = rand * \sqrt{\frac{2}{n^{[l-1]}}}$ (FOR RELU)

#inputs

XAVIER $\sqrt{\frac{1}{n^{[l-1]}}}$ (FOR TANH)

SETS THE VARIANCE

GRADIENT CHECKING

IF YOUR COST DOES NOT DECREASE ON EACH ITER YOU MAY HAVE A BACKPROP BUG.

GRADIENT CHECKING APPROXIMATES THE GRADIENTS SO YOU CAN VERIFY CALC.

NOTE ONLY USE WHEN DEBUGGING SINCE IT'S SLOW

IMPROVING DEEP NEURAL NETS • COURSERA

@TESSFERRANDEZ

STRUCTURING
YOUR ML PROJECTS

SETTING YOUR GOAL

★ A GOAL SHOULD BE A SINGLE #

	PRECISION	RECALL
A	95%	90%
B	98%	85%

IS A OR B BEST?

	PRECISION	RECALL	F1
A	95%	90%	92.4%
B	98%	85%	91%

A IS BEST

F1 = HARMONIC MEAN BETW. RECALL & PRECISION

★ DEFINE OPTIMIZING VS SATISFICING METRICS

	ACCURACY	RUNTIME
A	90%	80ms
B	92%	95ms
C	~~95%~~	~~1500 ms~~

MAXIMIZE ACC. GIVEN TIME ≤ 100ms

ACCURACY = OPTIMIZING

RUNTIME = SATISFICING

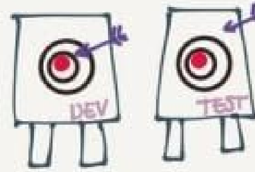
STRUCTURING ML PROJECTS • COURSERA

SELECTING YOUR DEV/TEST SETS

DATA

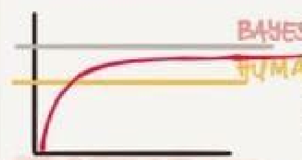
US
UK
~~EUROPE~~
S.AM
INDIA
CHINA
AUST.

OPTION 1:
DEV = UK, US, EUR
TEST = REST



IF DEV & TEST ARE DIFF & WE OPTIMIZE FOR DEV WE WILL MISS THE TEST-TARGET

HUMAN LEVEL PERF



MEDICAL IMG CLASS

TYPICAL HUMAN	3%
TYPICAL DOCTOR	1%
EXPERIENCED DR.	0.7%
TEAM OF EXP DRs.	0.5%

HUMAN LEV PERF (PROXY FOR BAYES)

WHY DOES ACC SLOW DOWN WHEN WE SURPASS HUMAN LEVEL PERF?

1. OFTEN CLOSE TO BAYES
2. A HUMAN CAN NO LONGER HELP IMPROVE (INSIGHTS)
3. DIFFICULT TO ANALYSE BIAS/VARIANCE

CAT CLASSIFICATION

	A	B
HUMAN	1%	7.5%
TRAIN ERR	8%	8%
DEV ERR	10%	10%
	FOCUS ON BIAS	FOCUS ON VARIANCE

BLURRY

AVOIDABLE BIAS

VARIANCE

HUMAN
| AVOIDABLE BIAS
TRAIN
| VARIANCE
DEV

AVOIDABLE BIAS:
- TRAIN BIGGER NETW.
- TRAIN LONGER / BETTER OPT. ALGOS (RMSProp, ADAM)
- CHANGE NN ARCH OR HYPERPARAMS

VARIANCE:
- MORE DATA (TRAIN)
- REGULARIZATION
- NN ARCHITECTURE

	A	B
HUMAN	0.5	0.5
TRAIN ERR	0.6	0.3
DEV ERR	0.8	0.4
AVOID. BIAS	0.1	?

AVOIDABLE BIAS

VARIANCE

DON'T KNOW IF WE OVERFIT OR IF WE'RE CLOSE TO BAYES

OPTIONS TO PROCEED ARE UNCLEAR

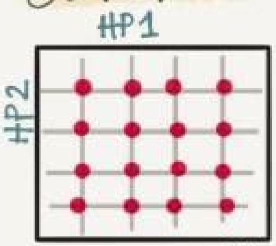
HYPERPARAM TUNING

WHICH HYPERPARAMS ARE MOST IMPORTANT?

- α LEARNING RATE
- # HIDDEN UNITS
- MINIBATCH SIZE
- β MOMENTUM TURN = 0.9
- # LAYERS
- LEARNING RATE DECAY
- $\beta_1 = 0.9$ $\beta_2 = 0.999$ $\varepsilon = 10^{-8}$ (ADAM)

TESTING VALUES

CLASSIC ML

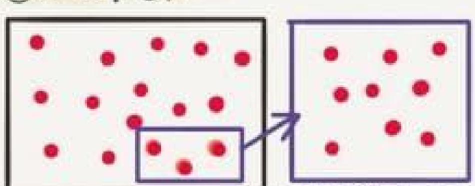


GRID SEARCH

PROBLEM: ONE ITERATION TAKES A LONG TIME & IN 16 GO'S WE HAVE ONLY TRIED 4 α - BUT 4 DIFF ε

NOT AS IMPORTANT

SOLUTION



RANDOM SEARCH + COARSE → DENSE

USE AN APPROPRIATE SCALE

HIDDEN UNITS

50 60 70 80 90 100

UNIFORMLY RANDOM

α LEARNING RATE

0.0001 0.001 0.01 0.1 1

$r = -4 \cdot \text{rand } [-4, 0]$

$\alpha = 10^r$

β EXP WEIGHT AVE

0.9 0.99 0.999

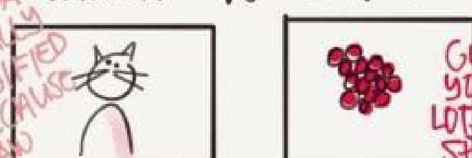
$r = -3 \cdot \text{rand } [-3, -1]$

$\beta = (1 - 10^r)$

TIP

RE-EVALUATE YOUR HYP. PARAMS EVERY FEW MONTHS

PANDA vs CAVIAR



BABYSIT ONE MODEL & TUNE

SPAWN LOTS OF MODELS W DIFF HP

GOOD IF YOU HAVE LOTS OF SPARE COMP POWER

MY PANDA IS ACTUALLY A MIS-CLASSIFIED CAT BECAUSE I CAN'T DRAW PANDAS

MISC. EXTRAS

BATCH NORMALIZATION

NORMALIZE LAYER OUTPUT

- SPEEDS UP TRAINING
- MAKES WEIGHTS DEEPER IN NW MORE ROBUST (COVARIATE SHIFT)
- SLIGHT REGULARIZING EFFECT

MULTICLASS CLASSIFIC.

CAT FISH BABY CHICK OTHER

C = # CLASSES = 4

SOFTMAX ACTIVATION

$t = e^{(z^{[L]})}$

$a^{[L]} = \frac{t}{\sum t_i}$

p(OTHER | X)
p(CAT | X)
p(FISH | X)
p(BC | X)

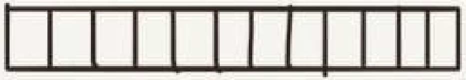
SUM: 1

EX: $z^{[L]} = \begin{bmatrix} 5 \\ 2 \\ -1 \\ 3 \end{bmatrix}$ $t = \begin{bmatrix} e^5 \\ e^2 \\ e^{-1} \\ e^3 \end{bmatrix} = \begin{bmatrix} 148.4 \\ 7.4 \\ 0.4 \\ 20.1 \end{bmatrix}$ 176.3

$\Rightarrow a^{[L]} = \frac{t}{176.3} = \begin{bmatrix} 0.842 \\ 0.042 \\ 0.02 \\ 0.114 \end{bmatrix}$ 11.4% PROB IT'S A BABY CHICK

OPTIMIZATION ALGORITHMS

MINI-BATCH GRAD. DESCENT



SPLIT YOUR DATA INTO MINI-BATCHES & DO GRAD DESCENT AFTER EACH BATCH
THIS WAY YOU CAN PROGRESS AFTER JUST A SHORT WHILE

CHOOSING THE MINIBATCH SIZE

SIZE = m → BATCH GRAD DESC.
SIZE = 1 → STOCHASTIC GRAD DESC

BATCH | MIDDLE | STOCHASTIC
TOO LONG PER ITERATION | LOOSE ALMOST ALL SPEED FROM VECTORIZATION

- SHORT ITER.
- USES VECT.

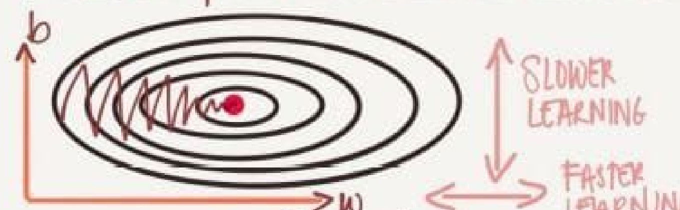
TIP
IF YOU HAVE < 2000 SAMPLES USE SIZE = 2000
OTHERWISE, USE 64, 128, 256...
SO X+Y FITS IN CPU/GPU CACHE

GRADIENT DESCENT W. MOMENTUM

WE WANT TO REDUCE OSCILLATION ↕ SO WE GET TO THE GOAL FASTER

SOLUTION: SMOOTH OUT THE CURVE BY TAKING AN EXPONENTIALLY WEIGHTED AVERAGE OF THE DERIVATIVES (i.e. LAST ONE HAS MORE IMPORTANCE)

RMSProp - ROOT MEAN SQUARED



NORMALIZE GRADIENT USING A MOVING AVG.

$S_{dw} = \beta S_{dw} + (1-\beta)\, dw^2$

$S_{db} = \beta S_{db} + (1-\beta)\, db^2$

$w = w - \alpha \frac{dw}{\sqrt{S_{dw}}} \quad b = b - \alpha \frac{db}{\sqrt{S_{db}}}$

ADAM OPTIMIZATION

COMBO OF GD W MOMENTUM & RMSProp

LEARNING RATE DECAY

IDEA: USE A LARGE α IN THE BEGINNING THEN DECREASE AS WE GET CLOSER TO GOAL

OPTION 1: $\alpha = \frac{1}{1 + \text{DECAYRATE} \cdot \text{EPOCH}} \alpha_0$

EXPONENTIAL: $\alpha = 0.95^{\text{EPOCH}} \alpha_0$

OPTION 3: $\alpha = \frac{k}{\sqrt{\text{EPOCH}}} \alpha_0$

OPTION 4: $\alpha = \frac{k}{\sqrt{t}} \alpha_0$

OPTION 5: DISCRETE STAIRCASE

OPTION 6: MANUAL

EPOCH = 1 PASS THROUGH THE DATA

IMPROVING DEEP NEURAL NETS · COURSERA

REGULARIZATION

PREVENTING OVERFITTING

L2 REGULARIZATION

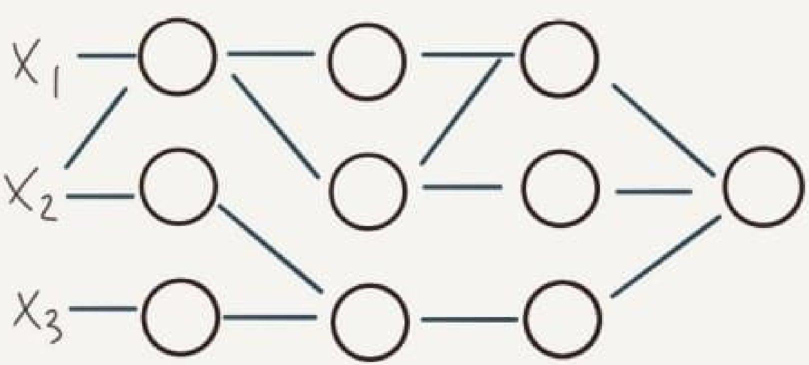
$$\text{COST}: J(w,b) = \frac{1}{m}\sum_{i=1}^{m} \mathcal{L}(\hat{y},y) + \frac{\lambda}{2m}\|w\|_2^2$$

EUCLIDEAN NORM

L1 REGULARIZATION

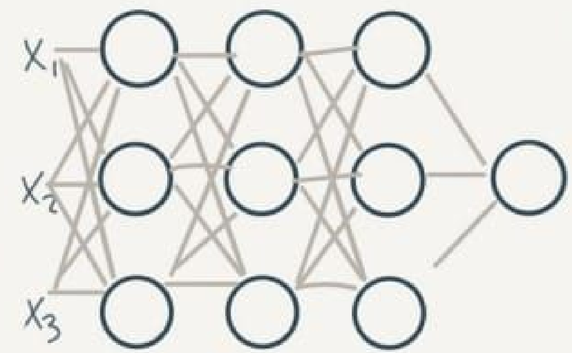
$$\text{COST}: J(w,b) = \frac{1}{m}\sum_{i=1}^{m} \mathcal{L}(\hat{y},y) + \frac{\lambda}{m}\|w\|_1$$

BOTH PENALIZE LARGE WEIGHTS ⇒
SOME WILL BE CLOSE TO Ø ⇒
SIMPLER NETWORKS

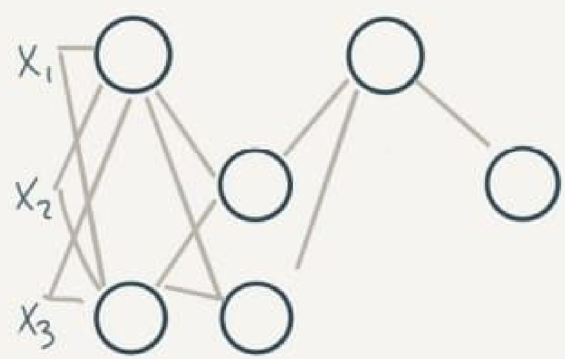


IMPROVING DEEP NEURAL NETS • COURSERA

DROPOUT



FOR EACH ITERATION & SAMPLE
SOME NODES ARE RANDOMLY
DROPPED (BASED ON KEEP_PROB)

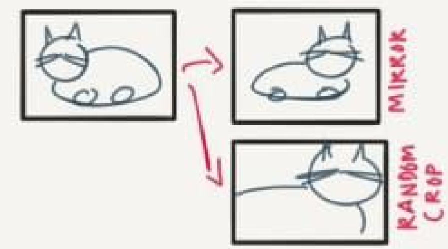


WE GET SIMPLER NWs
& LESS CHANCE TO RELY ON
SINGLE FEATURES

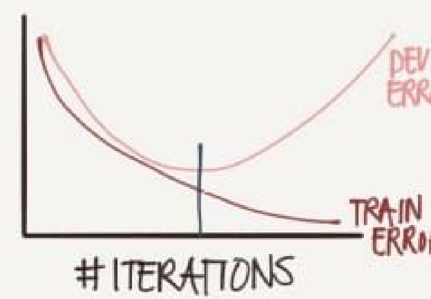
OTHER REGULARIZATION TECHNIQUES

DATA AUGMENTATION

GENERATE NEW PICS FROM EXISTING



EARLY STOPPING



PROBLEM: AFFECTS BOTH
BIAS & VARIANCE

SETTING UP
YOUR ML APP

CLASSIC ML

100 – 10000 SAMPLES

TRAIN	DEV	TEST
60%	20%	20%

DEEP LEARNING

1M SAMPLES

TRAIN	D	T
98%	1%	1%

ALL FROM SAME PLACE

DISTRIBUTION

EX: TRAIN  PRO CAT PICS FROM INTERNET

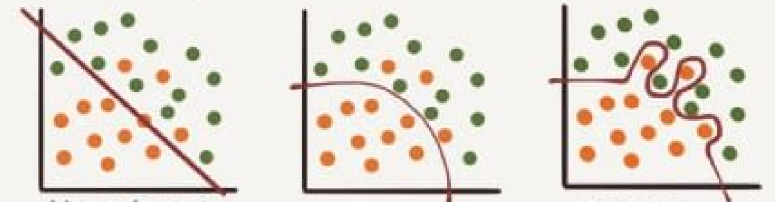
DEV/TEST  BLURRY CAT PICS FROM APP

TIP

DEV & TEST SHOULD COME FROM SAME DISTRIBUTION

IDEA → CODE → EXPERIMENT → IDEA

BIAS/VARIANCE



ERROR				
TRAIN	1%	15%	15%	0.5%
TEST	11%	16%	30%	1%
	HIGH VARIANCE	HIGH BIAS	HIGH BIAS & VARIANCE	LOW BIAS & VARIANCE

ASSUMING HUMANS GET 0% ERROR

THE ML RECIPE

HIGH BIAS —y→ BIGGER NETWORK, TRAIN LONGER, (DIFF NN ARCHITECTURE)

↓ no

HIGH VARIANCE —y→ MORE DATA (TRAIN), REGULARIZATION, (DIFF NN ARCHITECTURE)

↓ no

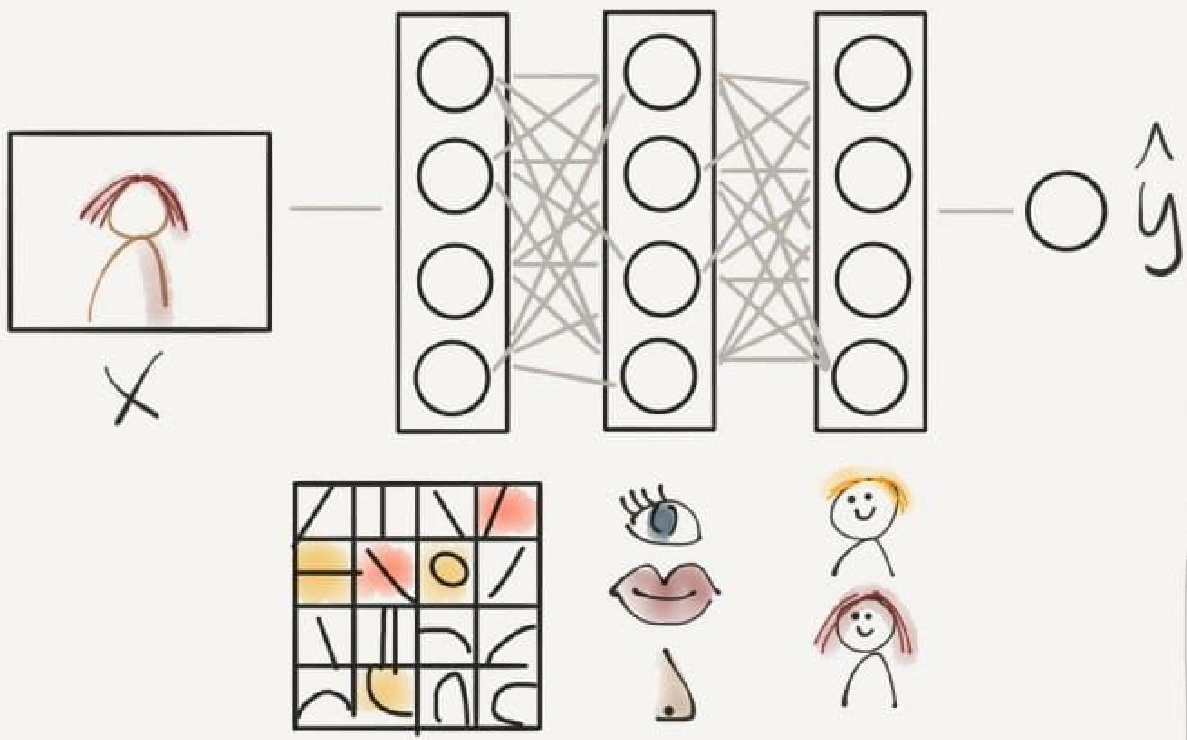
DONE

IMPROVING DEEP NEURAL NETS · COURSERA

DEEP
NEURAL NETS

THERE ARE FUNCTIONS A SMALL DEEP NET CAN COMPUTE THAT SHALLOW NETS NEED EXP. MORE UNITS TO COMP.

WHY DEEP NEURAL NETS?



LOW LEVEL AUDIO WAVE FEATURES — PHONEMES — WORDS — SENTENCES

↗ ↘ PITCH

C A T

VERY DATA HUNGRY

NEED LOTS OF COMPUTER POWER

ALWAYS VECTORIZE

VECTOR MULT · CHEAPER THAN FOR LOOPS

COMPUTE ON GPUs

LOTS OF HYPERPARAMS

- LEARNING RATE α
- # ITERATIONS
- # HIDDEN LAYERS
- # HIDDEN UNITS
- CHOICE OF ACTIVATION
- MOMENTUM
- MINI-BATCH SIZE
- REGULARIZATION

NEURAL NETWORKS & DEEP LEARNING • COURSERA • ANDREW NG • WK 4