Opening the black box of deep learning
(+ take-aways for AI)

Sanjeev Arora

PRINCETON UNIVERSITY

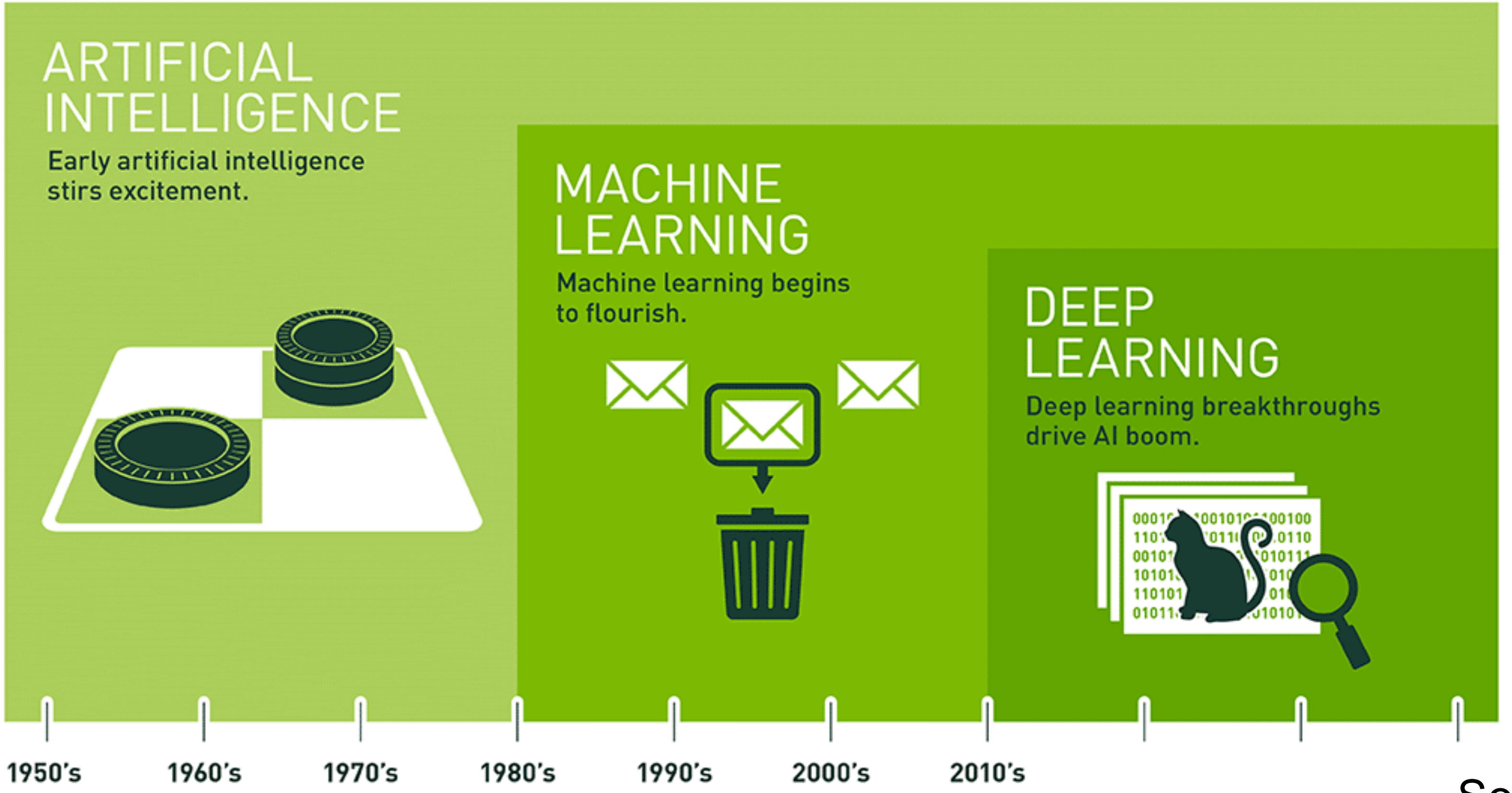
http://www.cs.princeton.edu/~arora/

Group website: unsupervised.princeton.edu

Blog: www.offconvex.org

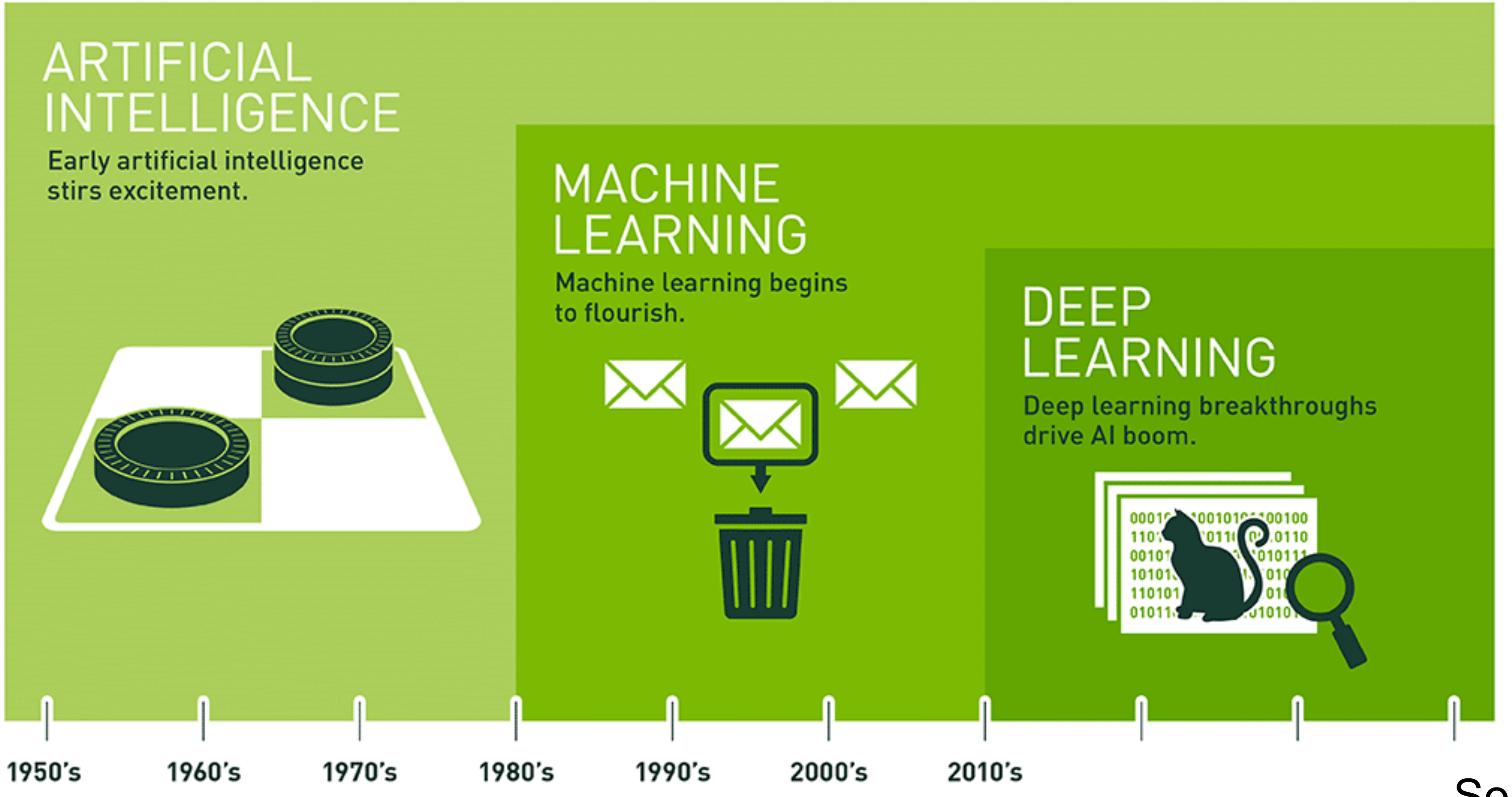
Twitter: @prfsanjeevarora

Grant support: NSF, Simons Foundation, ONR, DARPA/SRC, AWS



Source: Nvidia

Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

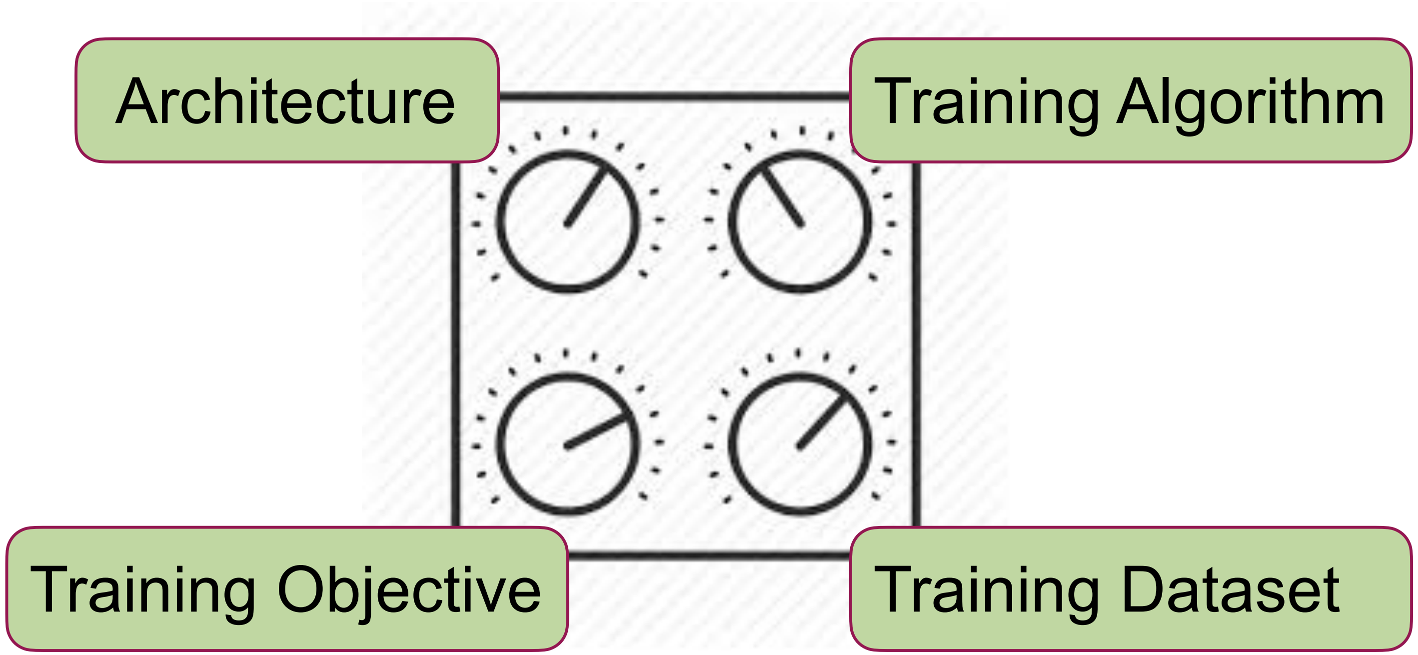


Since an early flush of optimism in the 1950s, smaller subsets of artificial intelligence – first machine learning, then deep learning, a subset of machine learning – have created ever larger disruptions.

Source: Nvidia

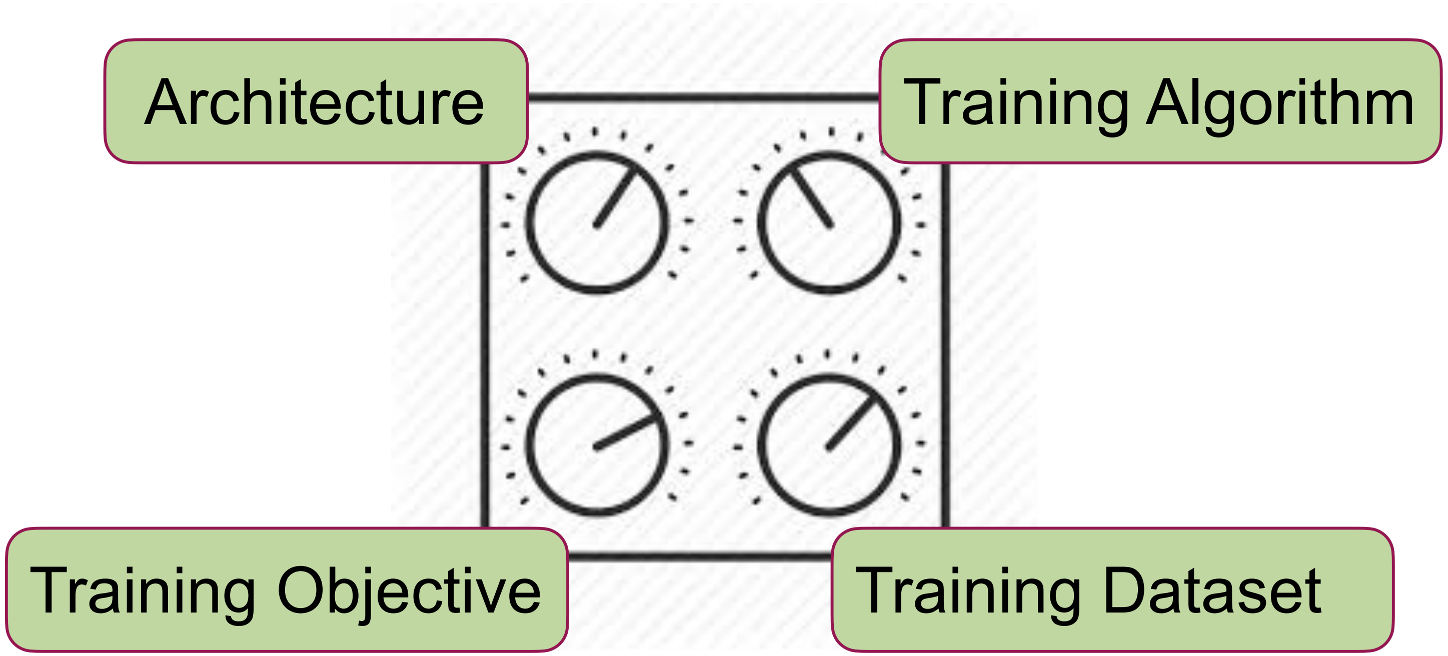
Programming based/ Human-Designed → Fitting model to data

Four control knobs of deep learning



Very different from
old AI (e.g.,Lisp code)!

Today's main point



This simplistic, "black-box" view of deep learning may not suffice for getting us to where we want to go in AI (i.e., for designing very flexible learners)

Basic deep learning paradigm

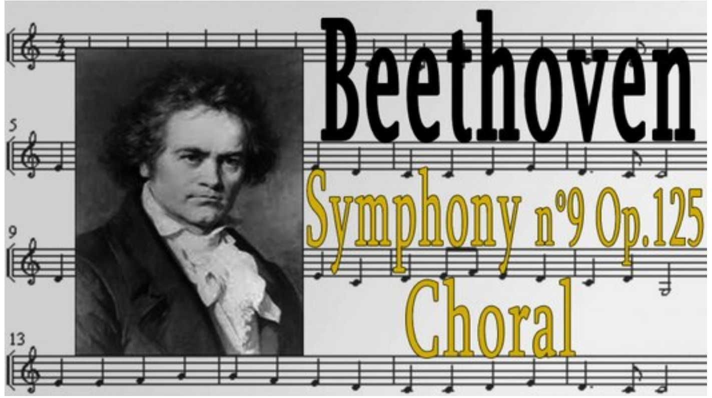
$\ell(w)$: training objective/loss

(w = parameter vector)

Gradient Descent (GD):

$$w^{(t+1)} \leftarrow w^{(t)} - \eta \nabla \ell(w^{(t)})$$

Usual view: Objective ≃ score



e.g.,

Stuart Russell: Will we choose the right objective for AI before it destroys us all?

Stuart Russell, author of a textbook on AI, and the popular volum Human Compatible, says humanity needs to get its act together and think about what the right objectives are to make sure machines more intelligent than ourselves don't annihilate the human race.

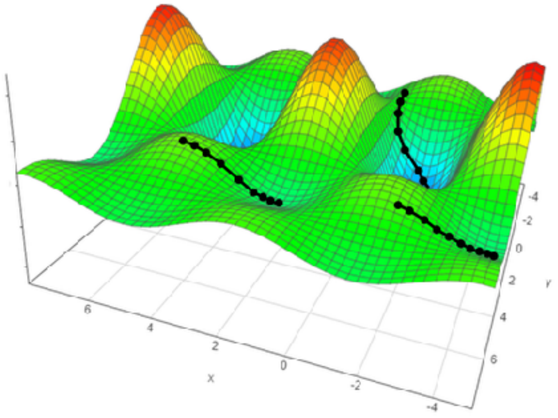
ZDNet Feb 13 2020

Objective **doesn't** fix functionality

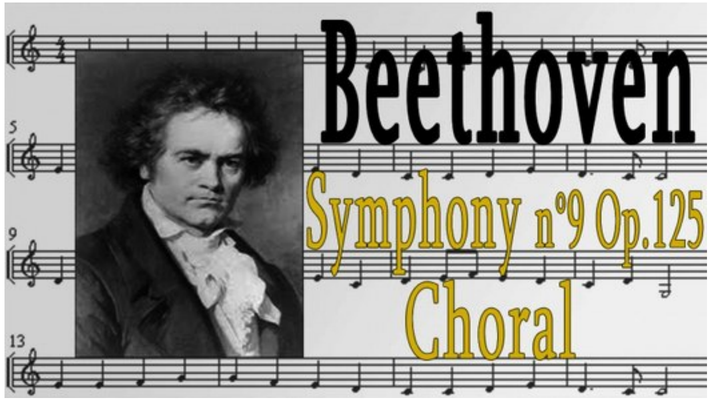
$\ell(w)$: training objective/loss
(w = parameter vector)
Gradient Descent (GD):

$$w^{(t+1)} \leftarrow w^{(t)} - \eta \nabla \ell(w^{(t)})$$

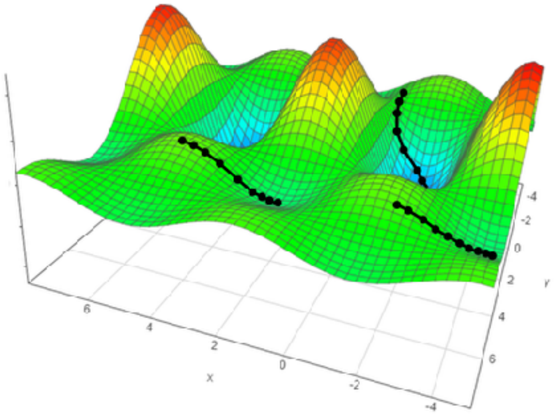
Reality:
$\ell()$ nonconvex; has multiple optima (w/ different properties)



Usual view: Objective ≃ score



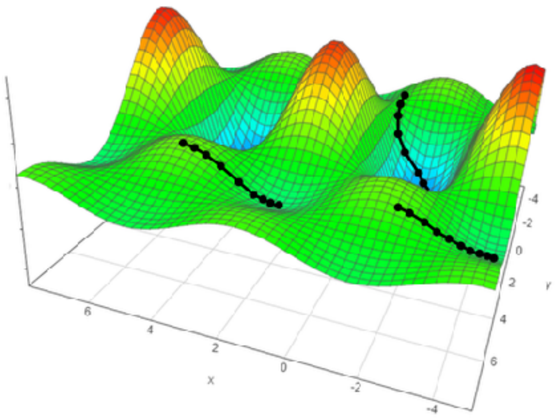
Training algorithm selects a solution..
(we have little idea how)



Why worry about multiple optima? Hasn't stopped progress so far.

Answer: For classification tasks (e.g., ImageNet Challenge) trained net can be tested on held-out data to check model's goodness.

But AI seeks flexible learners that can handle new situations…
(We'll return to this point a few times…)



Part 1: Architecture + dataset + objective don't suffice to determine behavior of trained deep model.

(Need to look at **dynamics** of **training algorithm**)

(Coming up: Vignettes from theory + Takeaways)

Vignette: Mathematical understanding of GD on **Linear** Nets

(“Even though objective looks nonsensical, GD picks meaningful solution;
Better than classical hand-designed algorithm.)

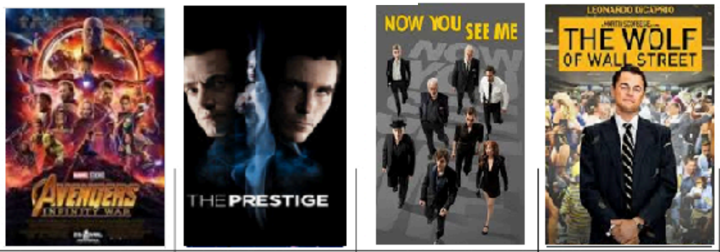
Matrix Completion Problem

Unknown *low rank* *$n \times n$ matrix M.*

Entries revealed in a random subset Ω of locations

Goal: Recover M.

MOVIES



USERS

Bob	4	?	?	4
Alice	?	5	4	?
Joe	?	5	?	?
Sam	5	?	?	?

[Srebro et al'05] Find matrix with best least-squares fit and smallest nuclear norm (convex!)

$$\sum_{ij\in\Omega}(M_{ij}-b_{ij})^2+\underbrace{\lambda|M|_*}_{\text{regularizer}}$$

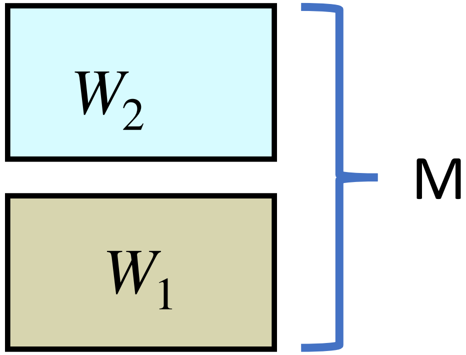
$|M|_*$ = sum of singular values of M
(Convex surrogate for low rank)

[Candes, Recht'10]: This is statistically "optimal" !

Linear nets for matrix completion

[Gunasekar et al'17] Find M as product of 2 matrices (depth 2 linear net); no regularization or rank constraint!

$$\sum_{i,j\in\Omega} ((W_2 W_1)_{ij} - b_{ij}))^2$$

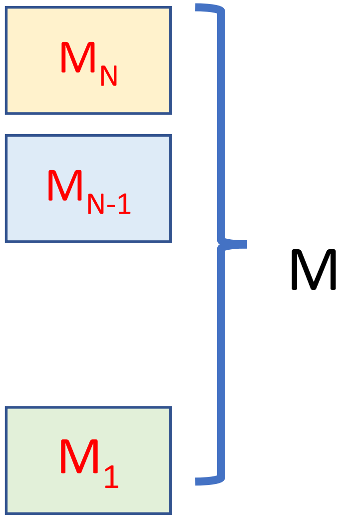


Infinitely many solns exist; most nonsensical.
Empirical finding: GD finds soln as good as nuclear norm minimization!

Linear nets for matrix completion (contd)

[A., Cohen, Luo, Wu'19] Find M as product of N matrices (depth-N linear net); no regularization!

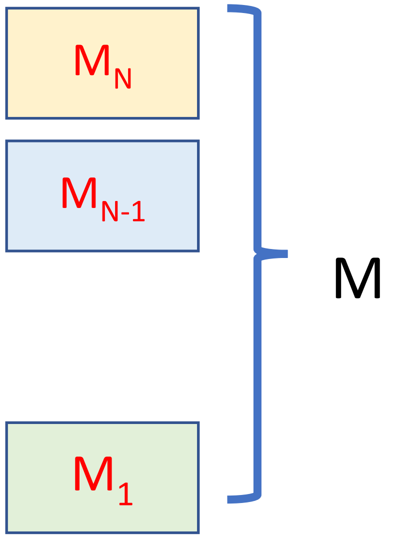
$$\sum_{i,j\in\Omega} ((W_N \cdots W_2 W_1)_{ij} - b_{ij})^2$$



Now GD finds soln better than nuclear norm minimization!

Mathematical analysis of GD trajectory in [A., Cohen, Luo, Wu ICML'19].
Complete characterization (≈"Greedy low rank learning") in [Li, Luo, Lyu ICLR'21]

Mathematical analysis of gradient flow (nontrivial!)



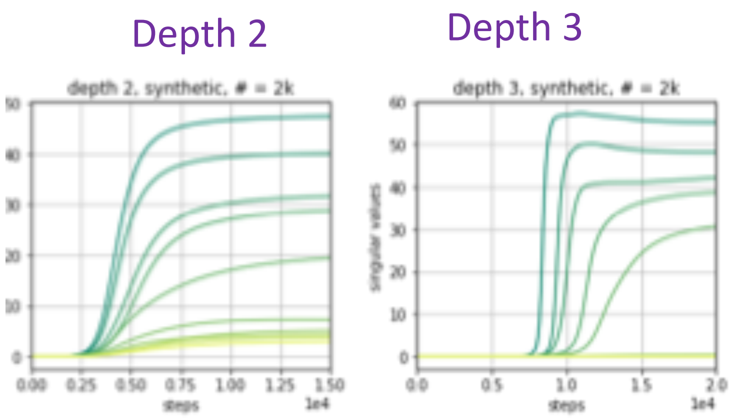
1. Show that singular values and sing. vectors of end-to-end matrix M are analytic functions of time t.

2. **Theorem 3.** *The signed singular values of the product matrix W evolve by:*

$$\dot{\sigma}_r(t) = -N \cdot \left(\sigma_r^2(t)\right)^{1-1/N} \cdot \left\langle \nabla \ell(W(t)), \mathbf{u}_r(t)\mathbf{v}_r^\top(t) \right\rangle \quad ,$$

"Rich get richer"; promotes low rank

(Interpretation: "Goal" of GD = Make singular vectors of M align with those of $\nabla(\ell(W(t)))$. Sing. directions grow one by one, not all at once.)



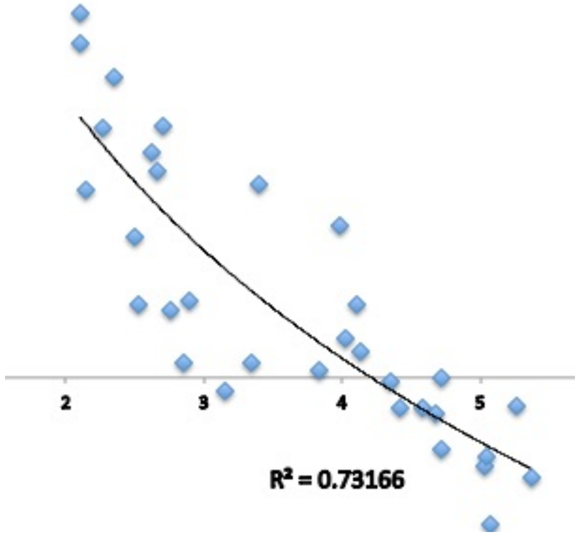
Evolution of sing. values w/ time

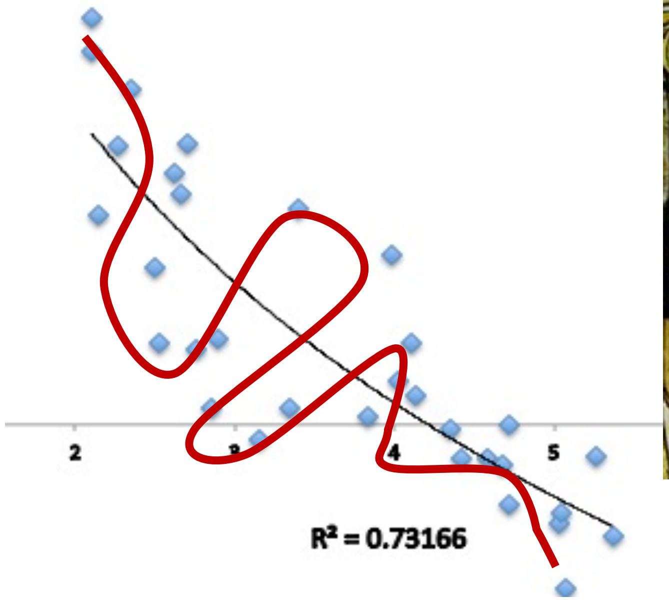
Vignette: Training of **infinitely** wide* Deep Nets

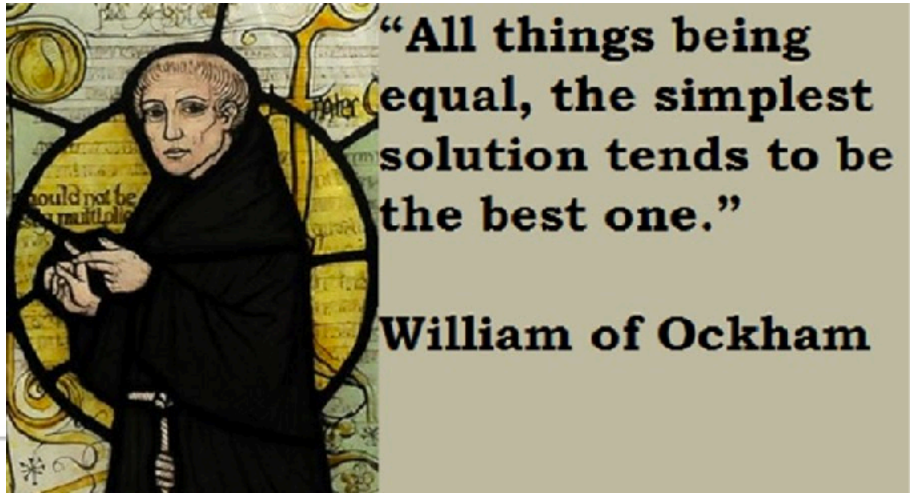
(“Architecture looks vacuous, but GD picks a meaningful solution out of infinitely many possibilities”)

(* Motivations: “Thermodynamic limit” + “Gaussian Process View of DL”)

Motivation: NO Overfitting mystery of deep learning

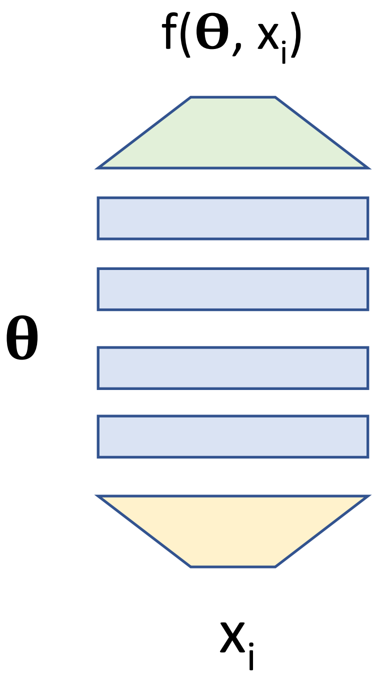






Rule of thumb: Overcomplicated models (e.g. when # parameters >> # datapoints) overfit and do not "generalize" to explaining new data.

Overparametrized nets (capable of fitting random data [Zhang et al.17]) outperform smaller nets.



Dataset: UCI Primary Tumor
(multiclass; 17-dimensional input, # training samples= 339)

Want to train fully connected 5-layer net on it. Infinitely wide!

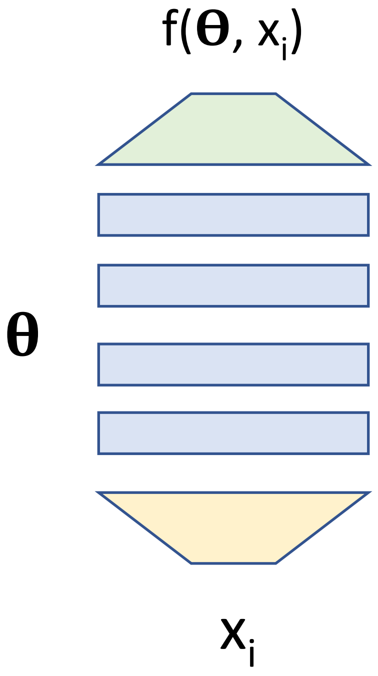
Means: Keep input and output layer fixed, but allow width of inner layers $\rightarrow \infty$
(initialize with suitably-scaled Gaussians so expected node value is equal at all layers)

Too expressive! Will overfit to training data.
(Arbitrarily wide 2-layer nets can represent every finite function, so # of zero-loss solutions $\rightarrow \infty$*)*
Plus, infeasible to train!

Test accuracy: 51.5

(Random Forest: 48.5, Gaussian Kernel: 48.4)

Details (fully connected nets)



$$f(\boldsymbol{\theta},\boldsymbol{x}) = \boldsymbol{W}^{(L+1)} \cdot \sqrt{\frac{c_\sigma}{d_L}}\sigma\left(\boldsymbol{W}^{(L)} \cdot \sqrt{\frac{c_\sigma}{d_{L-1}}}\sigma\left(\boldsymbol{W}^{(L-1)} \cdots \sqrt{\frac{c_\sigma}{d_1}}\sigma\left(\boldsymbol{W}^{(1)}\boldsymbol{x}\right)\right)\right)$$

W: Gaussian Initialization

- Square Loss: $L(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^{n}(f(\boldsymbol{\theta},\boldsymbol{x}_i) - y_i)^2$
- Dynamics of Gradient Descent on $L(\cdot)$, (shorthand: $u_i(t) = \mathrm{f}\big(\Theta(t)\ x_i\big)$
  $\dot{u}(t) = H(t)(u(t) - y)$

$$H_{ij}(t) = \left\langle \frac{\partial u_i(t)}{\partial W(t)}, \frac{\partial u_j(t)}{\partial W(t)} \right\rangle$$

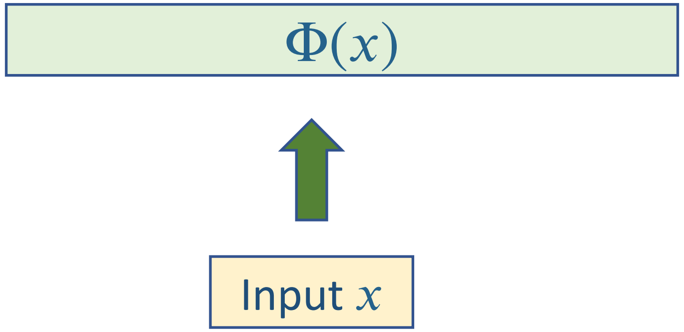
Thm ([Jacot et al.,'18] + followup papers): As $width \to \infty$, $\forall t\ \ H(t) \to H^*$
Implication: GD trajectory $\to \ell_2$ regression w.r.t. kernel H^*
(classic algorithm, but a new kernel: Neural Tangent Kernel)

(NB: "Infinitely wide nets" definable in multiple ways; not all reduce to NTK regression.)

Previous slide unpacked : Kernel regression/SVM reminder

Kernel trick: l_2 regression possible if can compute $< \Phi(x_1),\ \Phi(x_2) >$ for any given input pair $x_1,\ x_2$

"Reproducing Kernel Hilbert Space"



(e.g., polynomial kernel, Gaussian kernel,..)

Neural Tangent Kernel H^*:

Each coordinate of $\Phi(x)$ corresponds to parameter w in the net.

Corresponding entry is $\partial(output)/\partial w$ at $t = 0$

To do regression wrt H^* only need algorithm to compute $< \Phi(x_1),\ \Phi(x_2) >$ for any given input pair $x_1,\ x_2$

- Dynamic programming algorithm to compute H^* (also convnets) and thus l_2 regression.
  GPU friendly!
  Main idea: Treat infinitely wide layers as representing a continuous distribution of values.

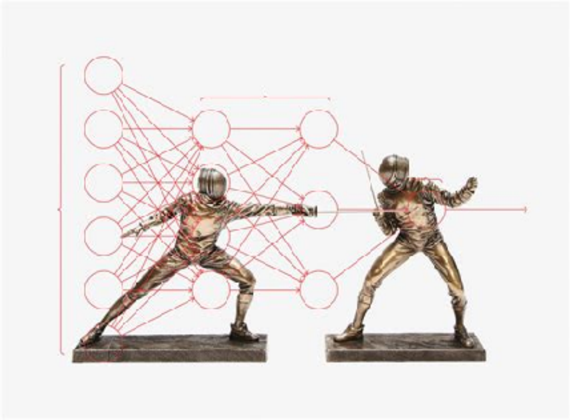
- Empirical finding: Pretty good performance in small-data setting. Competitive with old champions like Random Forests… [“Harnessing the power of infinitely wide nets for small-data tasks”..]

- Open: Tight analysis of generalization of kernel regression.

  (“To understand deep learning we need to understanding kernel learning”, [Belkin et al’18])

- Open: analysis of what other algorithms (SGD, Adam, BN etc) do with infinitely wide nets

Exact computation via infinitely wide nets (with convolution + global avg. pooling)
Applied to CIFAR10. [A., Du, Hu, Li, Salakhutdinov, Wang, NeurIPS 2019]

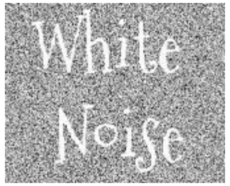


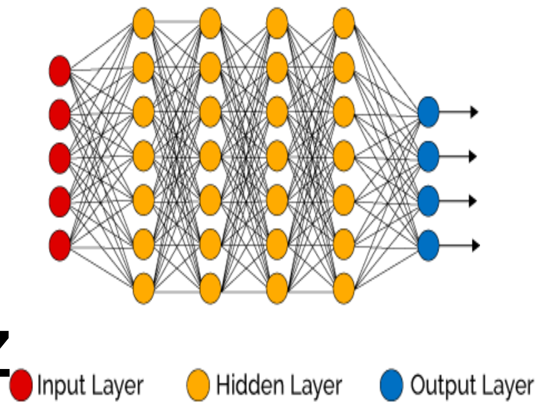
("Duelling AIs" MIT Tech Review)

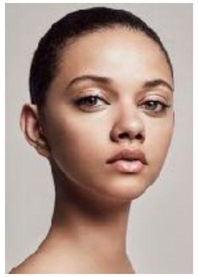
Part 2: Mode Collapse in Generative Adversarial Nets (GANs)

(i)Training Objective may deliver less than you expect
(ii) Caution warranted in multi-objective/multiplayer settings..

Deep generative models
(e.g., Variational AutoEncoders)



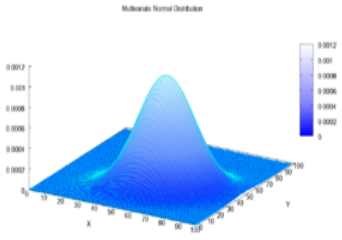




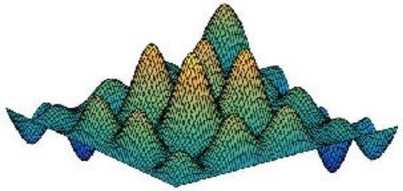
Code Z **Image X**

Usual training:
Max E_x[log p(x)]

(“log likelihood”)



N(O, I)



D_{real}

Implicit assumption: D_{real} *generatable by deep net of reasonable size.*

Generative Adversarial Nets (GANs)
[Goodfellow et al. 2014]

Motivations :

(1) Avoid loglikelihood objective;
it favors outputting fuzzy images.

(2) Instead of loglikelihood, use power of discriminative deep learning (i.e., classification) to improve the generative model.

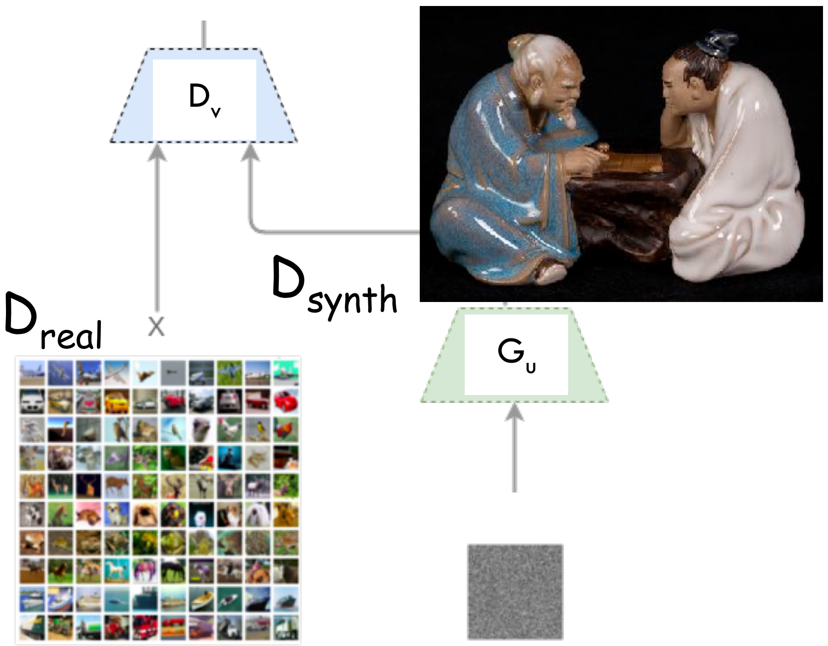
"Difference in expected output on real vs synthetic images"
Wasserstein GAN [Arjovsky et al'17]
**

Generative Adversarial Net

[Goodfellow et al. 2014]

Real (1) or
Fake (0)

$$\min_{u \in \mathcal{U}} \max_{v \in \mathcal{V}} \mathbf{E}_{x \sim \mathcal{D}_{real}}[D_v(x)] - \mathbf{E}_h[D_v(G_u(h))].$$



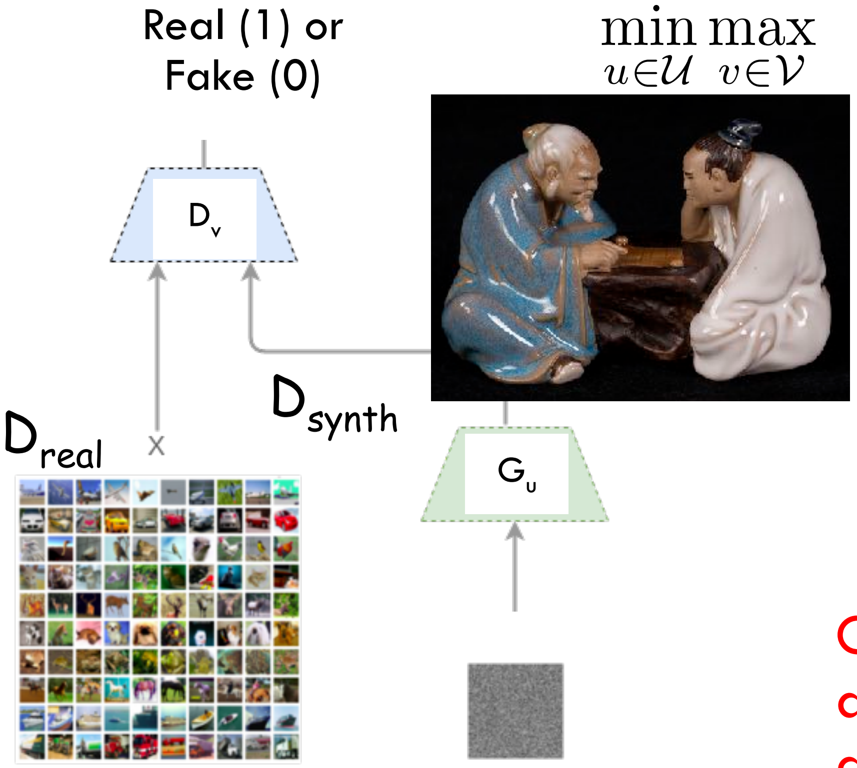
- Discriminator trained to output 1 on real inputs, and 0 on synthetic inputs.
- Generator trained to produce synthetic outputs that make discriminator output high values.

[Excellent resource: [Goodfellow's survey]

u= trainable parameters of Generator net
v = trainable parameters of Discriminator net

Generative Adversarial Nets (GANs) [Goodfellow et al. 2014]

$$\min_{u \in \mathcal{U}} \max_{v \in \mathcal{V}} \; \mathbf{E}_{x \sim \mathcal{D}_{real}}[D_v(x)] - \mathbf{E}_h[D_v(G_u(h))].$$



- Discriminator trained to output 1 on real inputs, and 0 on synthetic inputs.
- Generator trained to produce synthetic outputs that make discriminator output high values.

u= trainable parameters of Generator
v = trainable parameters of Discriminator

Generator "wins" if objective ≈ 0 and further training of discriminator doesn't help. ("Equilibrium.")

What spoils a GANs trainer's day: Mode Collapse

- Since discriminator only learns from a few samples, it may be unable to teach generator to produce distribution D_{synth} with sufficiently large diversity

- (many ad hoc qualitative checks for mode collapse..)

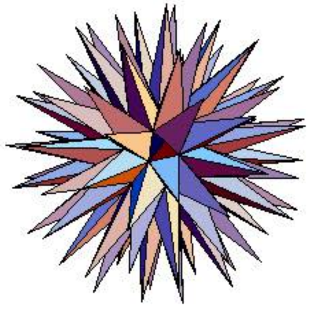
New Insight from theory: problem not with # of training samples, but size ("capacity") of the discriminator!

Thm [A., Ge, Liang, Ma, Zhang ICML'17] : If discriminator size = N, then ∃ generator that generates a distribution supported on O(Nlog N) images, and still wins against all possible discriminators.
(tweaking objectives or increasing training set doesn't help..)

(NB: D_{real} presumably has infinite support..)

→ Small discriminators inherently incapable of detecting "mode collapse."

Pf sketch: Consider generator that learns to produce O(N logN) random real images. Consider "all possible discriminators of size N" (suffices to consider "ε-net"). Use concentration bounds to argue that none of them can distinguish D_{real} from this low-support distribution.



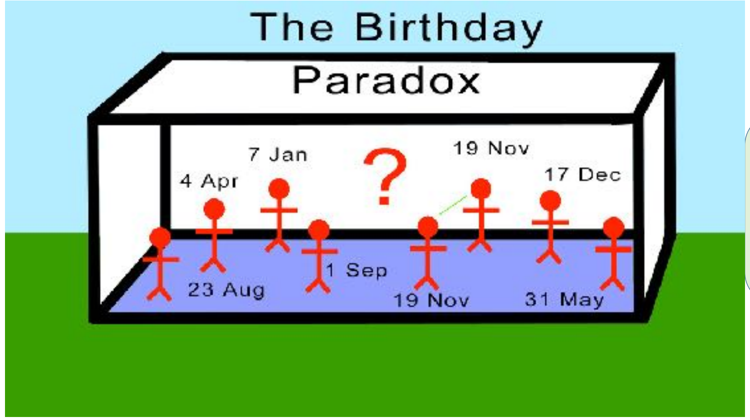
How to check support size of generator's distribution??

Theory suggests GANs training objective not guaranteed to avoid mode-collapse.

Does this happen during real life training???

Empirically detecting mode collapse (Birthday Paradox Test)

(A, Risteski, Zhang ICLR'18)



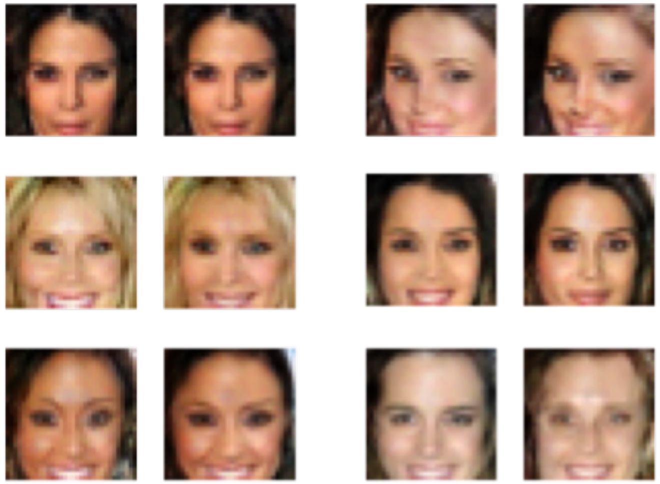
If you put 23 random people in a room, chance is > 1/2 that two of them share a birthday.

Suppose a distribution is supported on N images. Then Pr[sample of size √N has a duplicate image] > ½.

Birthday paradox test* [A, Risteski, Zhang] : If a sample of size s has near-duplicate images with prob. > 1/2, then distribution has only s^2 distinct images.

Implementation: Draw sample of size s; use heuristic method to flag possible near-duplicates. Rely on human in the loop to verify duplicates.

Estimated support size from well-known GANs



CelebA (faces):
200k training images

DC-GAN [Radford et al'15]: Duplicates in 500 samples. Support size $(500)^2 = 250K$

BiGAN [Donohue et al'17]
and ALI (Dumoulin et al'17]:
Support size = $(1000)^2 = 1M$

(Similar results on CIFAR10)

Followup: [Santurkar et al'17] Different test; confirms lack of diversity.

[Bauetal'19] Confirms continued lack of diversity in more recent models.

formalization of "learnt skills"?

Part 3: Why does training on Task A help later with solving Task B?

(e.g., major in math, later do well in law school)

Recall: Canonical ML framework

Datapoints come from a distribution $\mathscr{D}$
S = Training Datapoints

Train model parameters w by minimizing $E_{x \in S}[\ell_x(w)]$

Learning works if $E_{x \in \mathscr{D}}[\ell_x(w)] \approx E_{x \in S}[\ell_x(w]$
(i.e. test loss is similar to training loss)

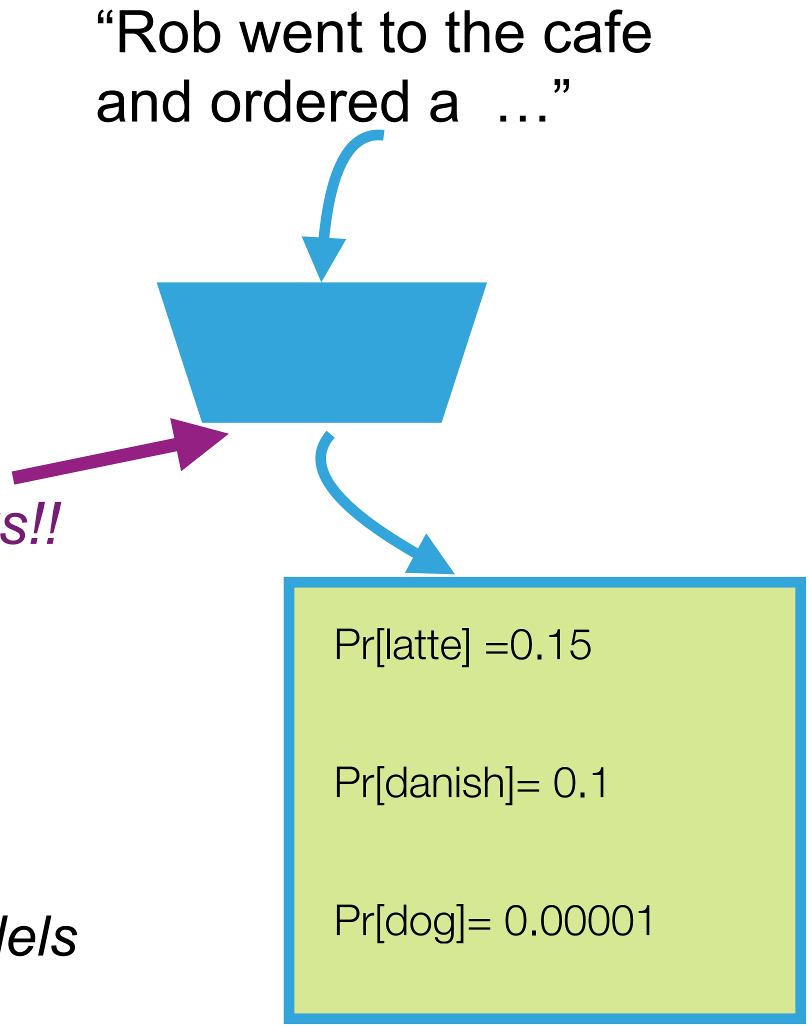
If test task $\neq$ training task, must look for "learned skills" inside trained model w^*.

Ex1: Language models

Training of language models like GPT-3 involves "fill in the blank" tasks (uses "log likelihood")

$$\boxed{\ell_{xent}\left(\left\{p_{\cdot|s}\right\}\right) = \mathbb{E}_{s,w}\left[-\log\left(p_{\cdot|s}(w)\right)\right]}$$

GPT-3's internal representation of text $f(\,.\,)$ turn out to be useful (with no further fine-tuning) for other language tasks!!



Is this something special about
* language modeling itself,
* current deep architectures, or
* training algorithms?

Next few slides: [*Mathematical exploration of why Language Models help solve Downstream Tasks:* Saunshi, Malladi, A. ICLR'21**]**

Classification tasks often can be cast as next-word prediction

Ex: (Sentiment Classification) Given movie review, classify as +ve or -ve

Review + " I think the movie was…"

s

Sentiment ought to be apparent from $p_{\cdot|s}(w)$ for w= "good," "terrible," "amazing", "meh", etc.!

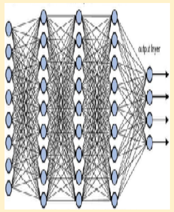
Natural task: Solvable via linear classifier on the vector of probabilities $p^*_{\cdot|s}(w)$

(Empirical finding: Suffices for classifier to look at $p_{\cdot|s}(w)$ for 10-20 words. Part of "natural" defn!)

Relating text embedding to probabilities

Context: s

Language Model



$\boldsymbol{f}(\boldsymbol{s}) \in \mathbb{R}^{\boldsymbol{d}}$

$\boldsymbol{\Phi} \in \mathbb{R}^{\boldsymbol{d \times V}}$

softmax on $\boldsymbol{\Phi}^\top \boldsymbol{f}(\mathrm{s})$

$$\begin{pmatrix} 0.3 \\ \cdots \\ 0.2 \\ 0.05 \\ 0.0001 \end{pmatrix}$$

Φ = (Fixed) Matrix of word embeddings

1st order optimality condition: For fixed Φ, f^* that minimizes ℓ_{xent} satisfies $\Phi p_{f^*(s)} = \Phi p^*_{\cdot|s}$

(i.e., Only guaranteed to learn $p^*_{\cdot|s}$ up to "projection" in d-dimensional **subspace spanned by $\boldsymbol{\Phi}$**

Claim: If Φ respects synonym structure (i.e. synonyms have similar embeddings) then natural task continue to be solvable via $p^*_{\cdot|s}$ that satisfy subspace constraint

Better language model $\Longrightarrow$ Better classification

$$\ell_{\mathcal{T}}\left(\left\{\Phi p_{f(s)}\right\}\right) \leq \tau + \mathcal{O}\left(\sqrt{\epsilon}\right)$$

Logistic regression loss for task using d-dimensional $\Phi p_{f(s)}$

Measure of naturalness of the task

Suboptimality of LM wrt likelihood
$\epsilon = \ell_{xent}\left(\left\{p_{f(s)}\right\}\right) - \ell_{xent}\left(\left\{p_{f^*(s)}\right\}\right)$

NB: Assumes (i) natural task (ii) Φ respects synonym structure (iii) 1st order optimality condition of language model objective

Ex 2: Self-supervised learning: QuickThought

[Logeswaran & Lee, ICLR'18] "like word2vec.."

[For image tasks, x, x^+ are frames from same video [Wang-Gupta'15]..

Using text corpus (eg Wikipedia) train deep representation function f to minimize

$$\mathbb{E}\left[\log\left(1 + e^{f(x)^T f(x^-) - f(x)^T f(x^+)}\right)\right]$$

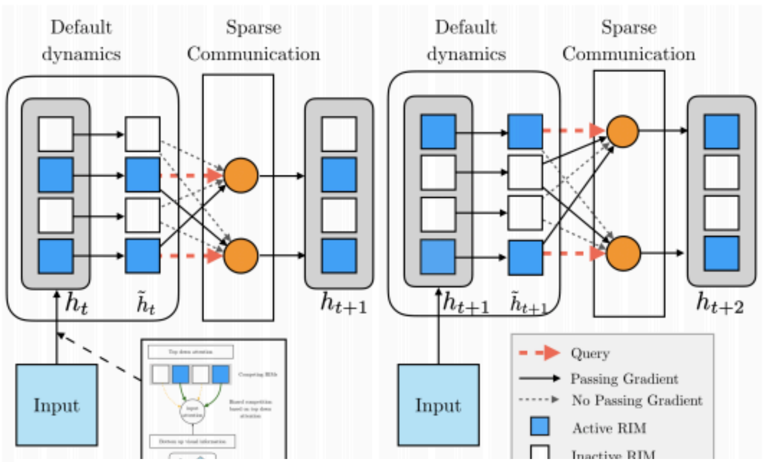
x, x^+ are adjacent sentences, x^- is random sentence from corpus

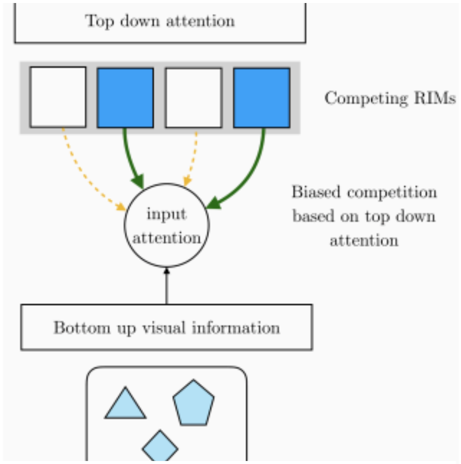
("Make adjacent sentences have high inner product, while random pairs of sentences have low inner product.")

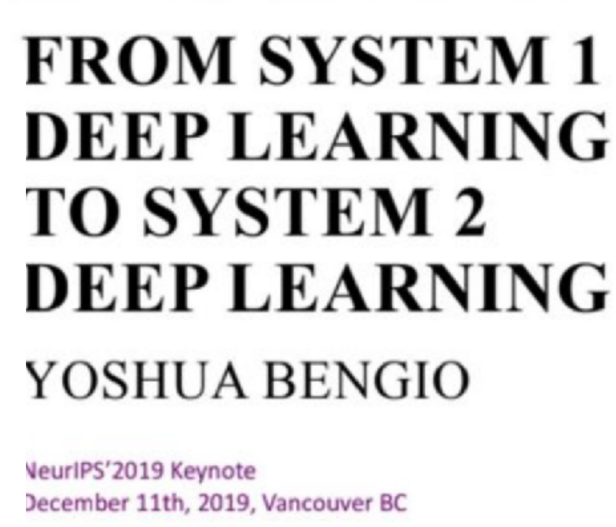
Many classification tasks solvable via **linear** classifers when sentence s is represented as $f(s)$

[A theoretical analysis of contrastive unsupervised representation learning, A., Khandeparker, Khodak, Plevrakis, Saunshi ICML'19]

Part 4 (speculative): Flexible & Reliable AI agents may need new design principles



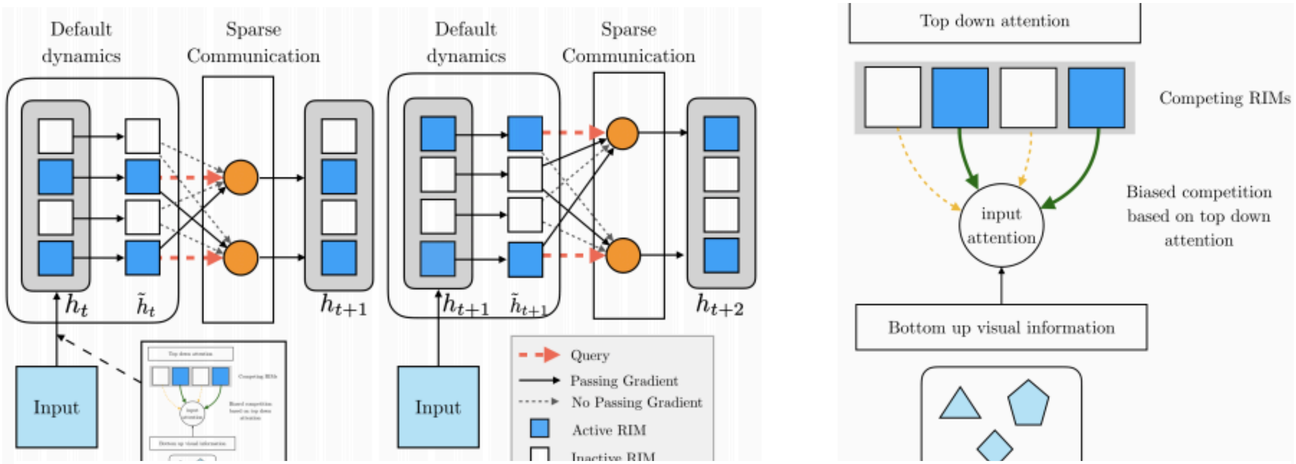




RECURRENT INDEPENDENT MECHANISMS

Anirudh Goyal[1], Alex Lamb[1], Jordan Hoffmann[1,2,*], Shagun Sodhani[1,*], Sergey Levine[4]
Yoshua Bengio[1,], Bernhard Schölkopf[3,**]**

(Rough summary): Society of agents who choose (using attention mechanism) to compete or collaborate. Credit assignment via suitable gradients



Recurrent Independent Mechanisms

Anirudh Goyal[1], Alex Lamb[1], Jordan Hoffmann[1,2,*], Shagun Sodhani[1,*], Sergey Levine[4]
Yoshua Bengio[1,**], Bernhard Schölkopf[3,**]

FROM SYSTEM 1 DEEP LEARNING TO SYSTEM 2 DEEP LEARNING

YOSHUA BENGIO

NeurIPS'2019 Keynote
December 11th, 2019, Vancouver BC

Operating way beyond traditional ML framework (multiobjective, no fixed training distribution,..)

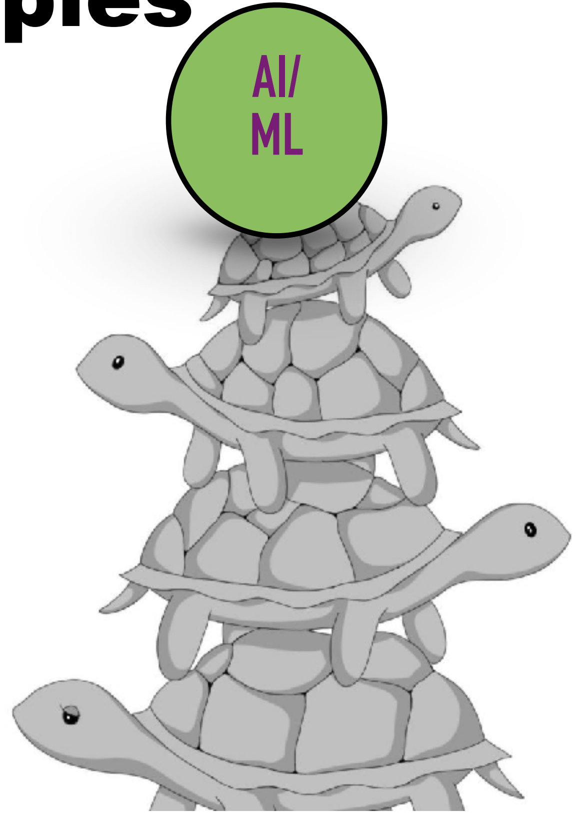
(Rough summary): Society of agents who choose (using attention mechanism) to compete or collaborate. Credit assignment via suitable gradients

Need for new theory and principles

Recall: ML relies on avg. performance on data (training / test) drawn from a **fixed** distribution.

Many Proposed Alternatives! (Variants on "distributions on distributions", e.g. Meta-Learning, Bayesian frameworks)

Possibly insufficient to capture richness of everyday interactions? (Low-probability events an important test of flexible agents?)



Independent samples from fixed distribution

(NB: Conceivable that huge datasets can allow a way around this…)

In conclusion

- We're starting to open black box of deep learning
- In some (admittedly stylized) settings properties of trained models proven to arise from complicated interaction of architecture, objective, training algorithm and dataset.
- Formal understanding could be crucial for design of flexible, reliable AI agents.

THANK YOU!

http://www.cs.princeton.edu/~arora/

Group website: unsupervised.princeton.edu

Blog: www.offconvex.org