Machine Learning
Machine Learning in the Real World

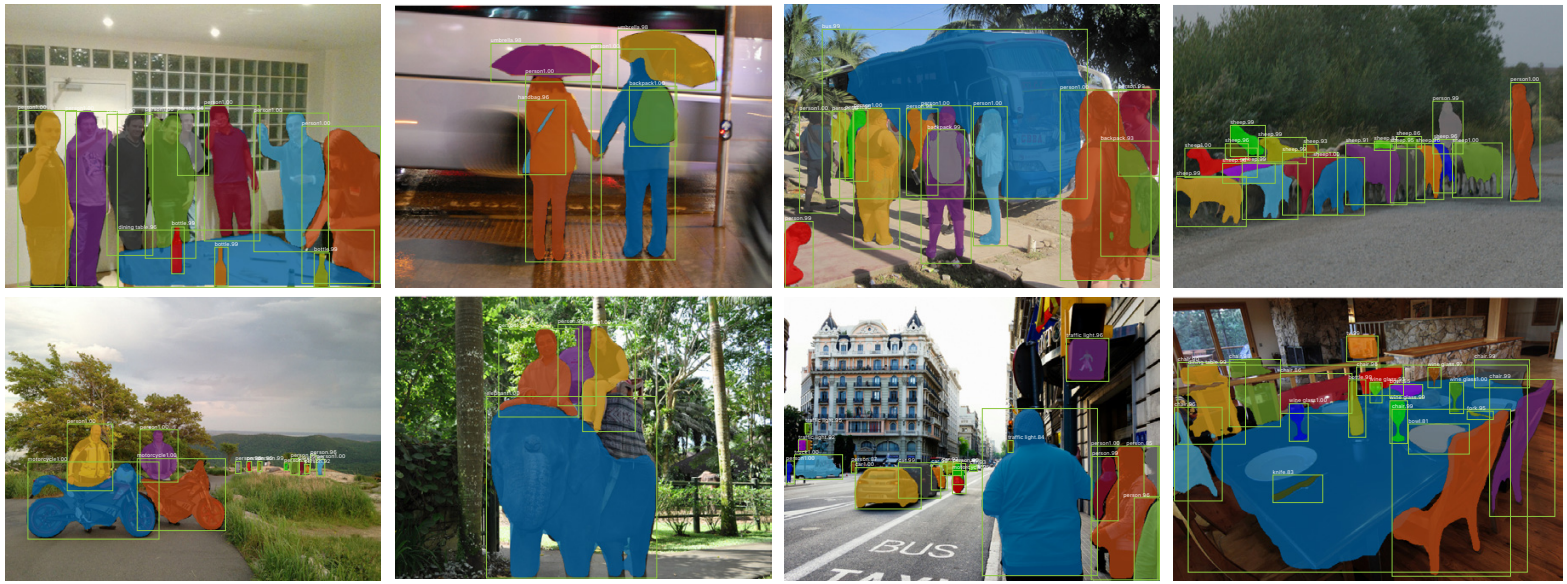
Rajesh Ranganath

Supervised Learning

Take some input **x** and predict **y**

Supervised Learning

Take some input **x** and predict **y**



[He+ 2015]

Supervised Learning

Take some input **x** and predict **y**

Input
Chest X-Ray Image

CheXNet
121-layer CNN

Output
Pneumonia Positive (85%)

[Rajpurkar+ 2017]

Bayesian Methods

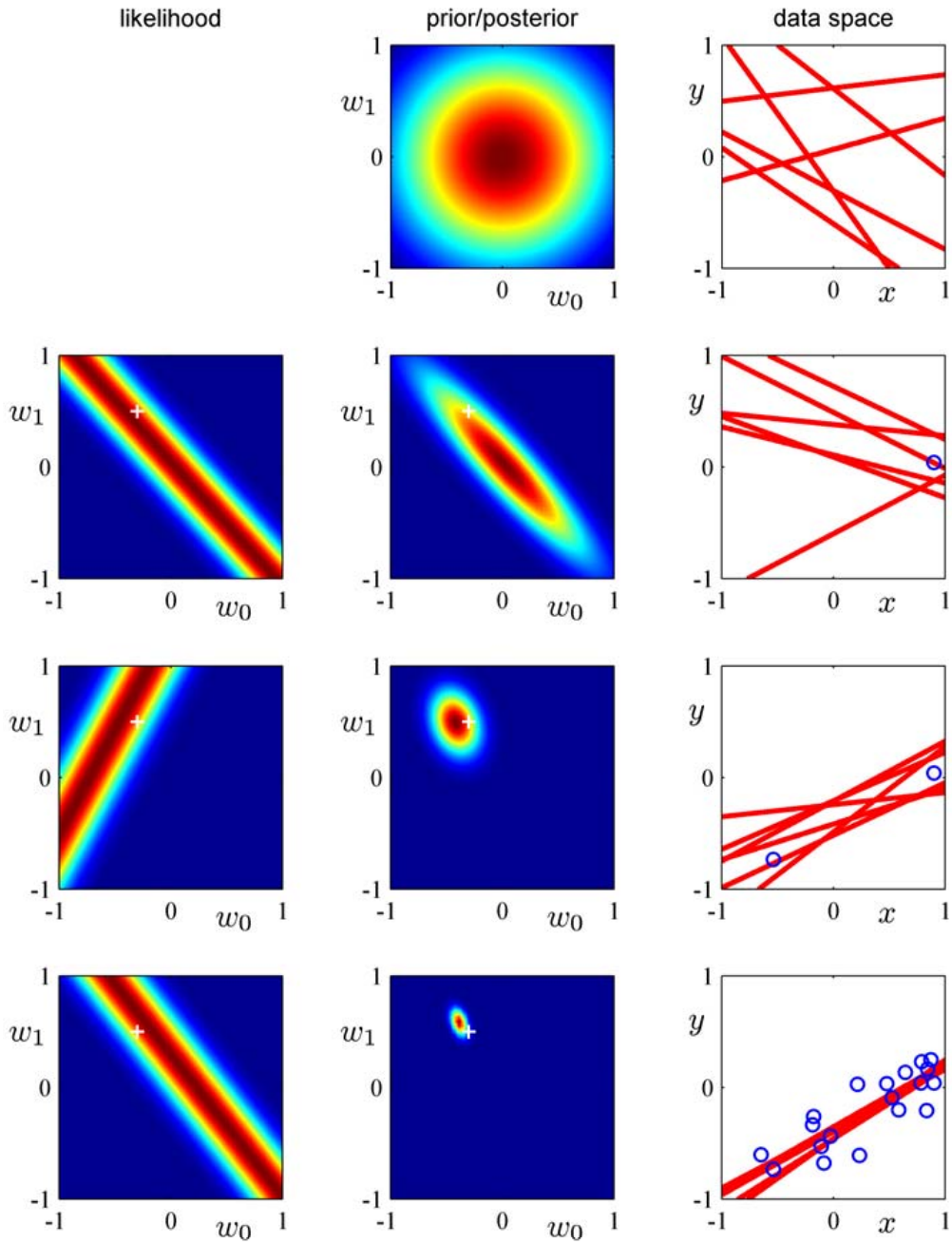


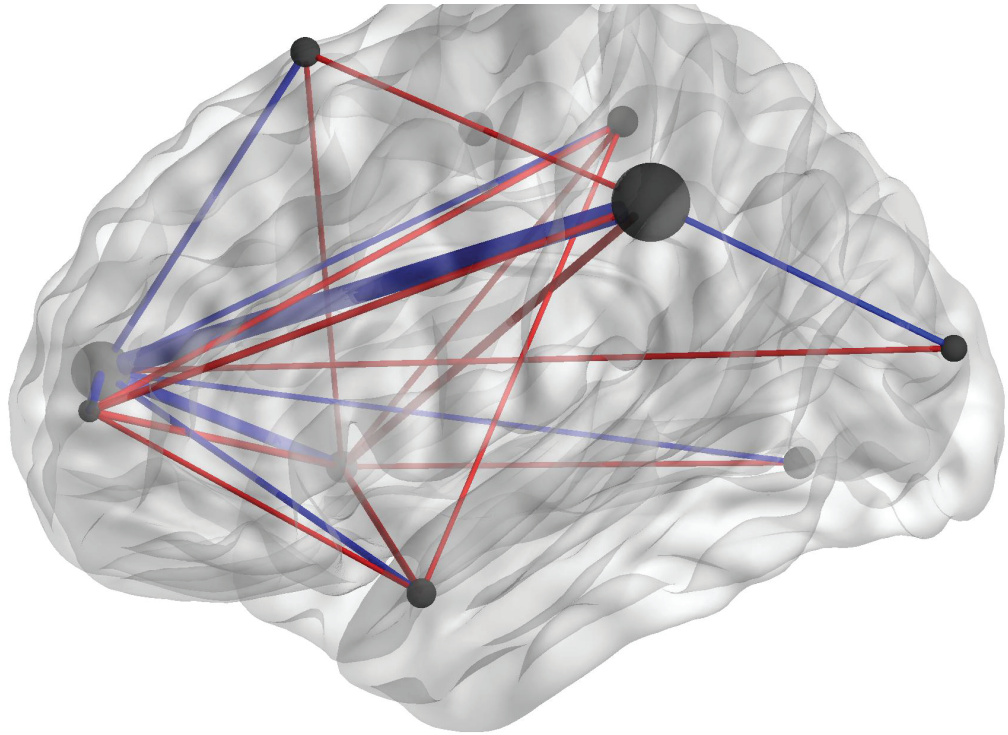
Figure 3.7 Illustration of sequential Bayesian learning for a simple linear model of the form $y(x, \mathbf{w}) = w_0 + w_1 x$. A detailed description of this figure is given in the text.

Graphical Models/Latent Variable Models

Take some input $\mathbf{x}$ understand relationships

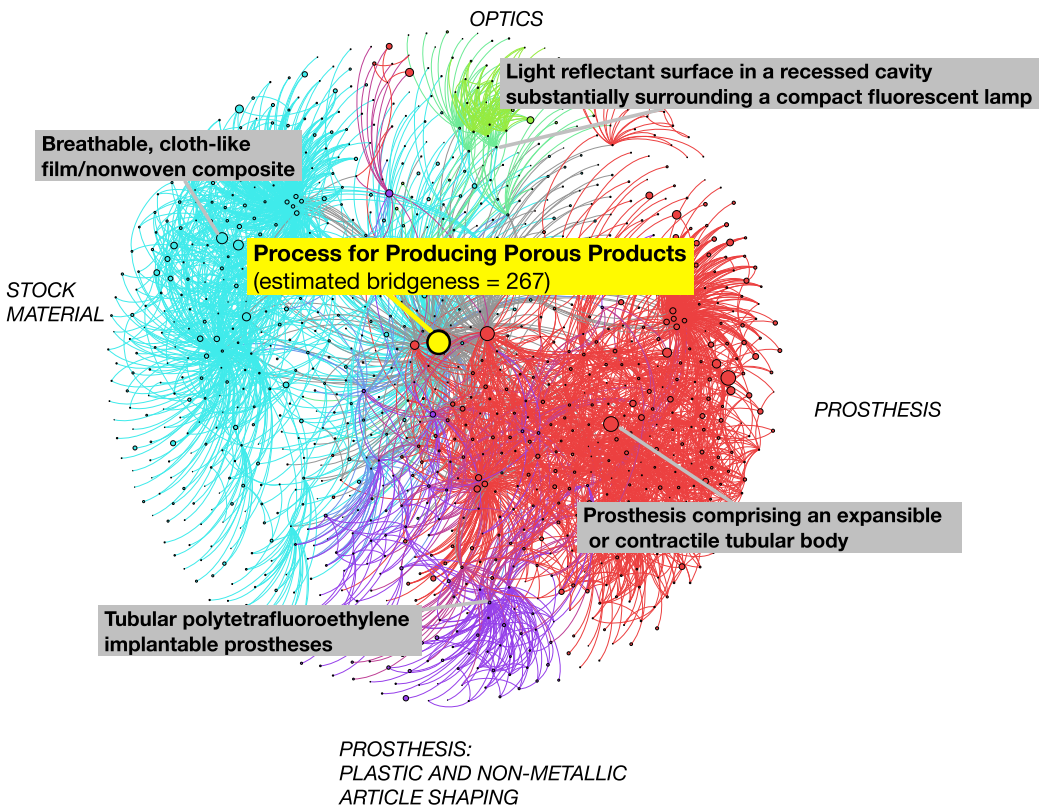
Graphical Models/Latent Variable Models

Take some input **x** understand relationships



Graphical Models/Latent Variable Models

Take some input **x** understand relationships



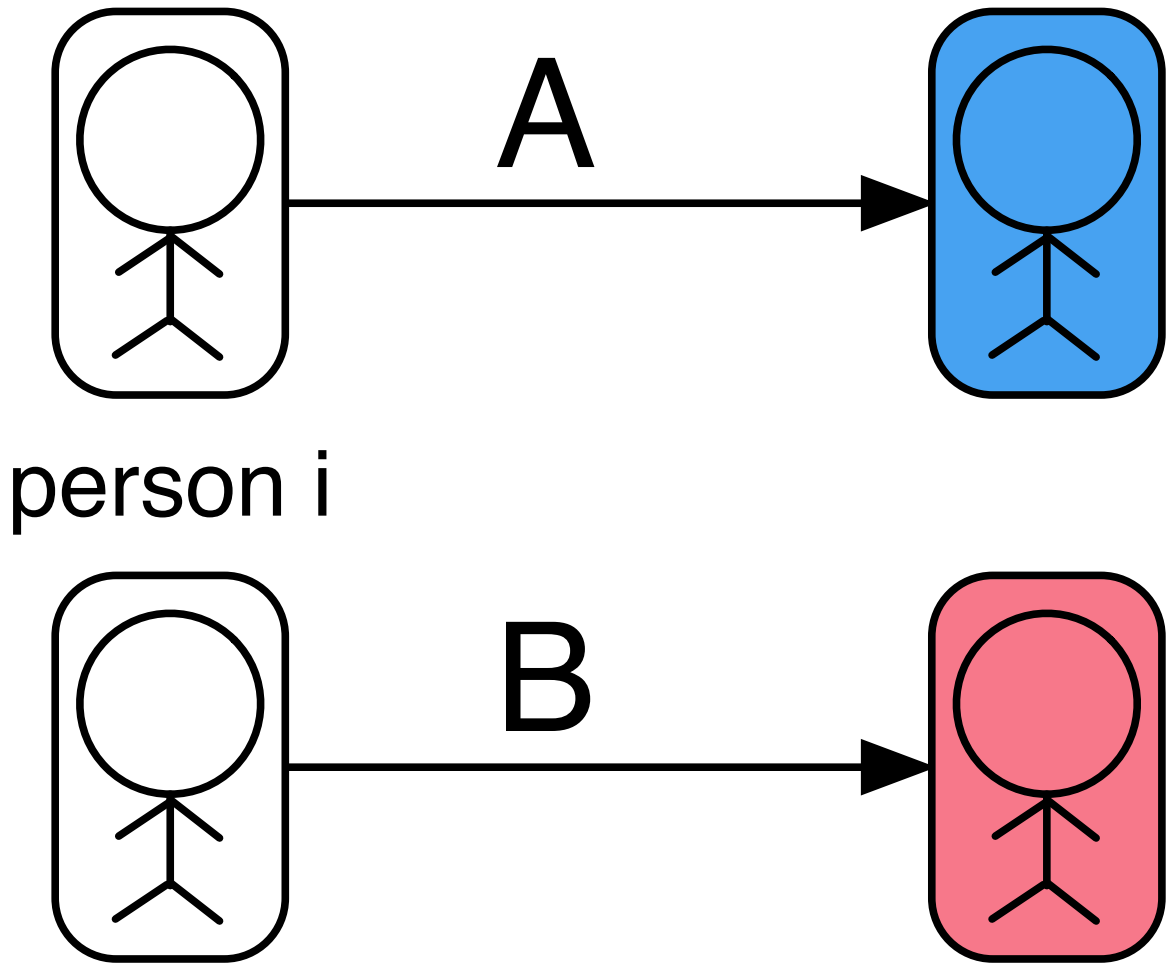
[Gopalan+ 2014]

Causal Inference

Understand the effect of altering **x** on **y**

Causal Inference

Understand the effect of altering **x** on **y**

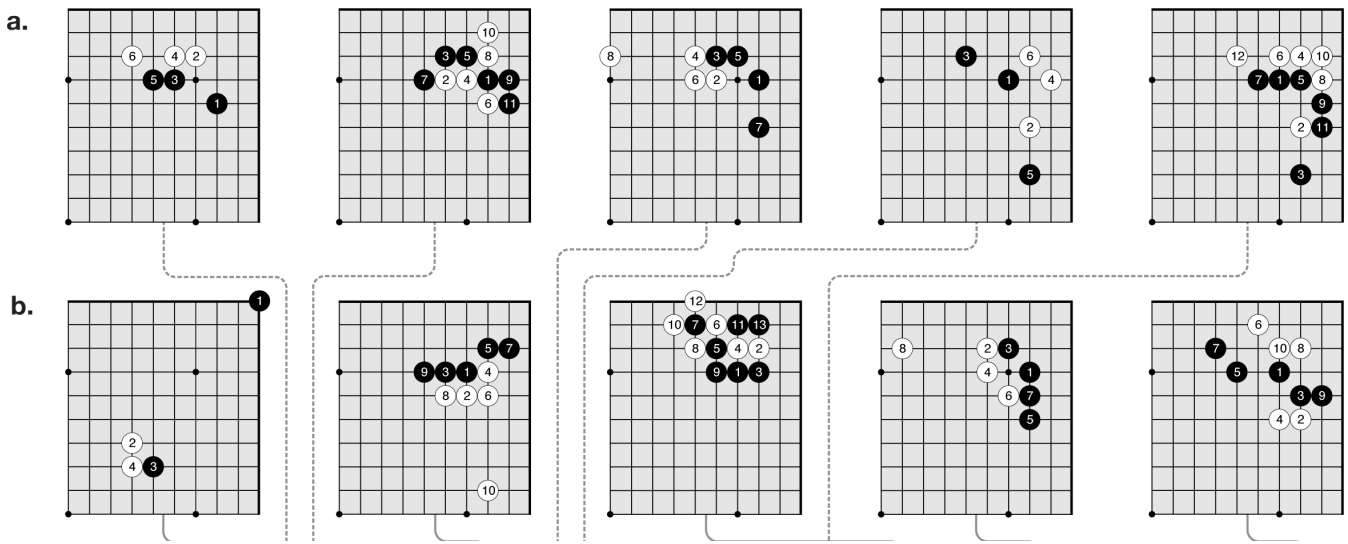


Reinforcement Learning

Understand how to take a sequence of actions to meet a goal

Reinforcement Learning

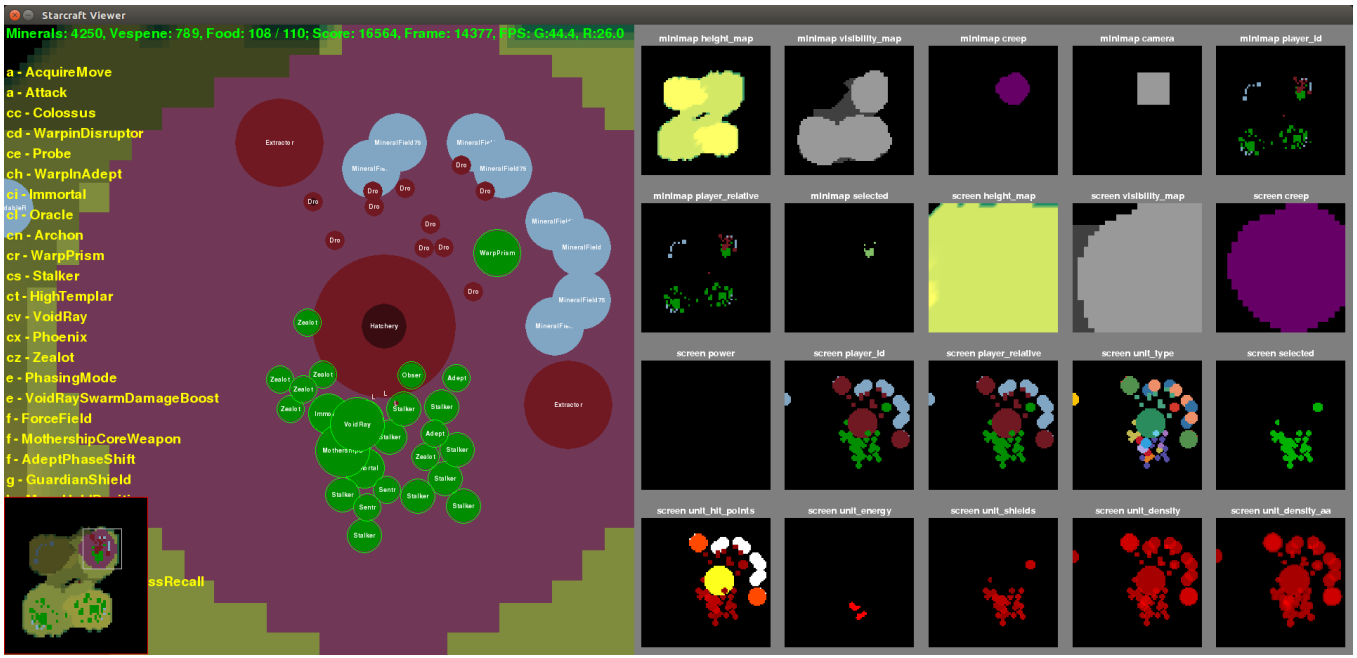
Understand how to take a sequence of actions to meet a goal



[Silver+ 2016]

Reinforcement Learning

Understand how to take a sequence of actions to meet a goal

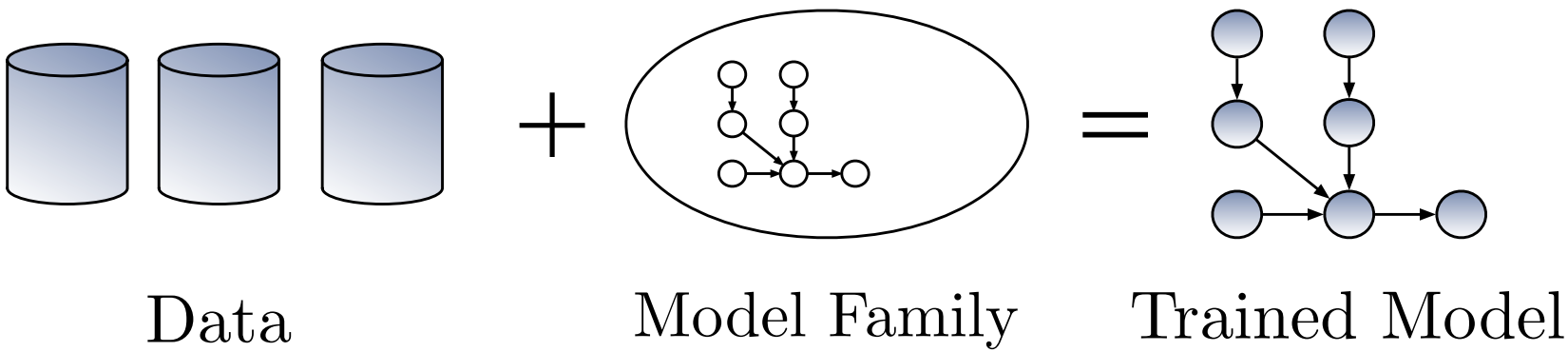


[Vinayals+]

Do They Work?

From twitter:

How Does Machine Learning Learn?



Fundamentally assumption dependent to say something about an input that was never seen

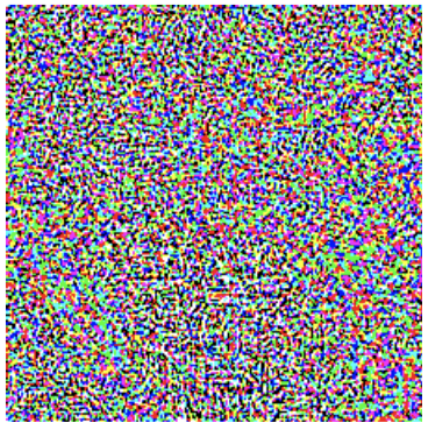
Real World Worries

Worry 1

$\boldsymbol{x}$

"panda"
57.7% confidence

$+ .007 \times$



$\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

"nematode"
8.2% confidence

$=$

$\boldsymbol{x} + \epsilon\text{sign}(\nabla_{\boldsymbol{x}} J(\boldsymbol{\theta}, \boldsymbol{x}, y))$

"gibbon"
99.3 % confidence

[Goodfellow+ 2017]

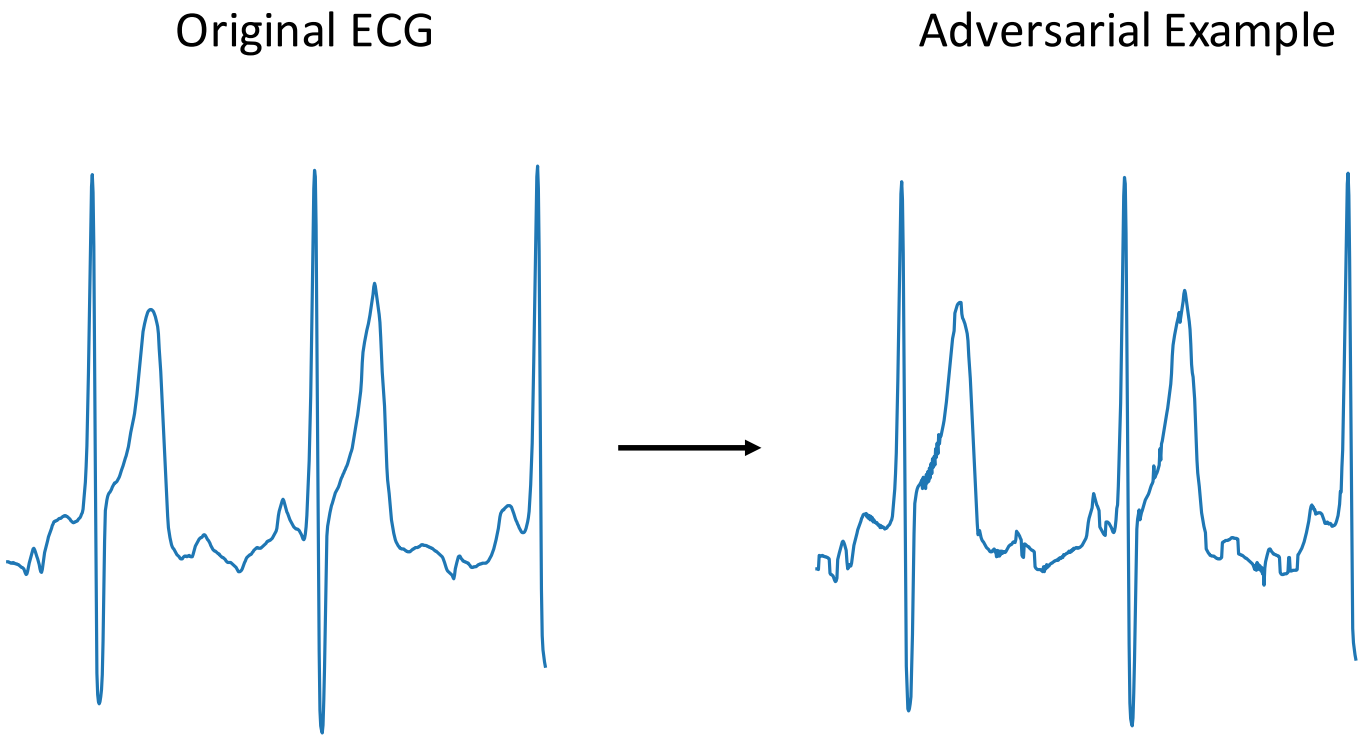
Recipe for Making Adversarial Examples

Bad loss means misclassification

1. Start with trained model
2. Compute gradient with respect to loss function with respect to input
3. Follow gradient to increase the loss
4. Limit the movement to a norm

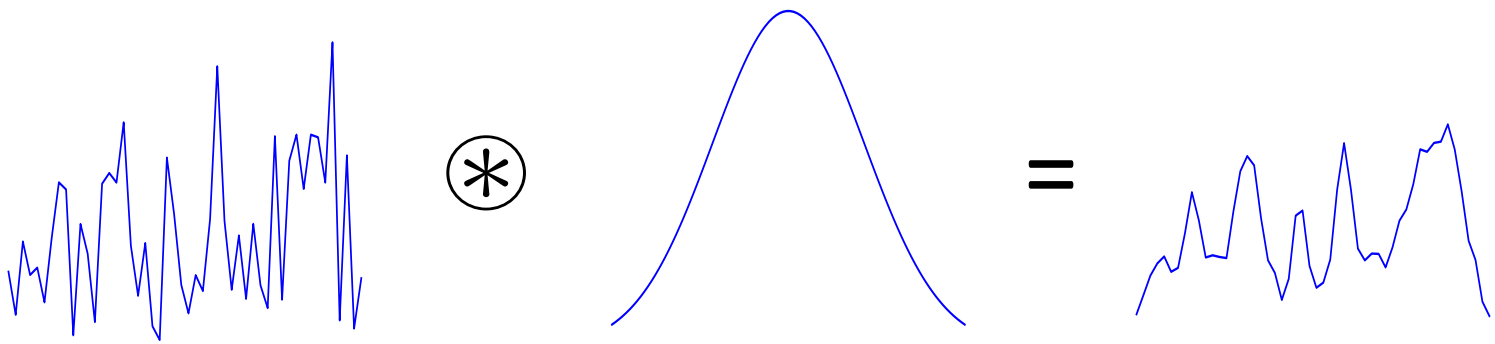
Popular technique: Projected Gradient Descent [Madry+ 2017]

Use PGD to build adversarial examples on 2017 CinC winner



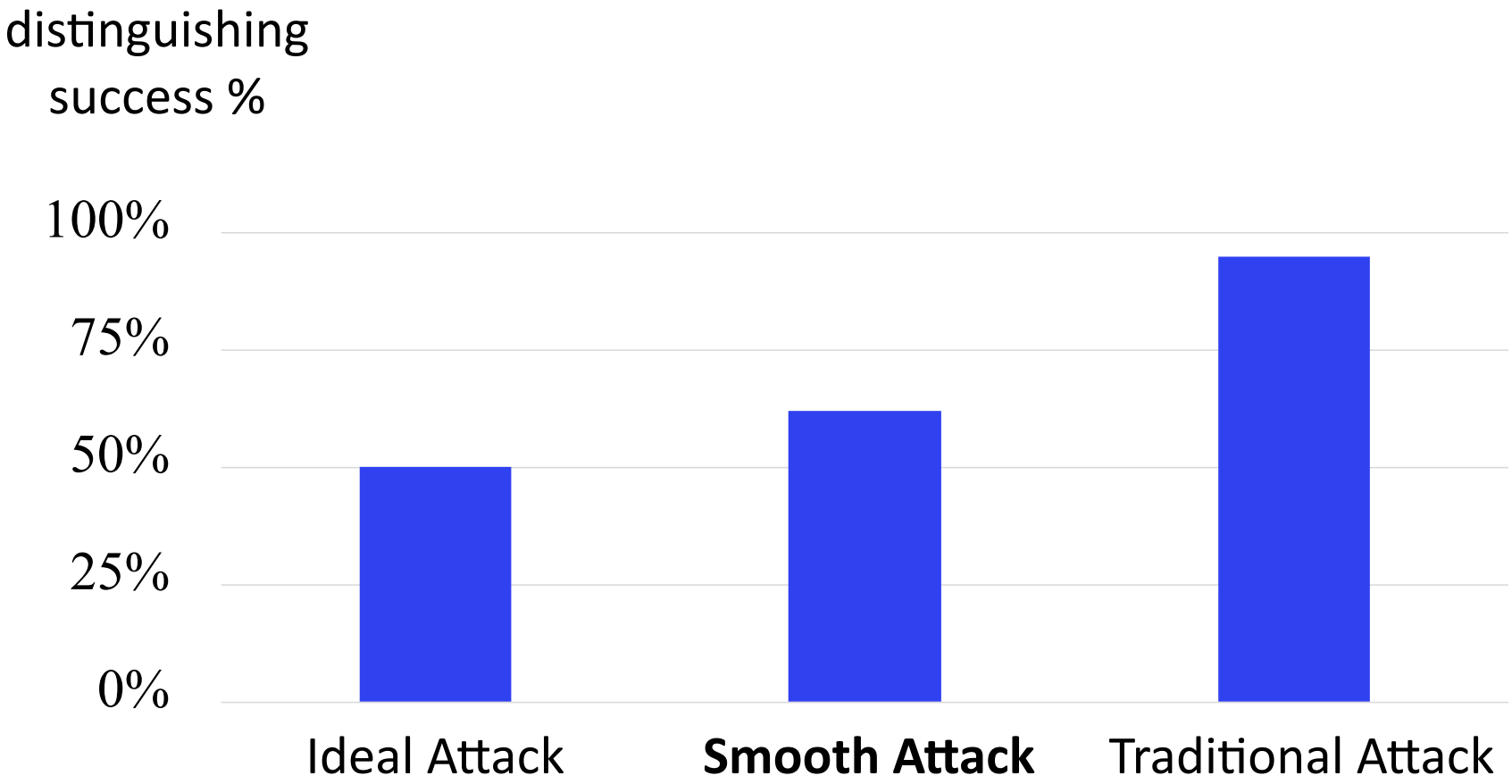
Aren't real adversarial examples!

Construct Smooth Adversarial Examples



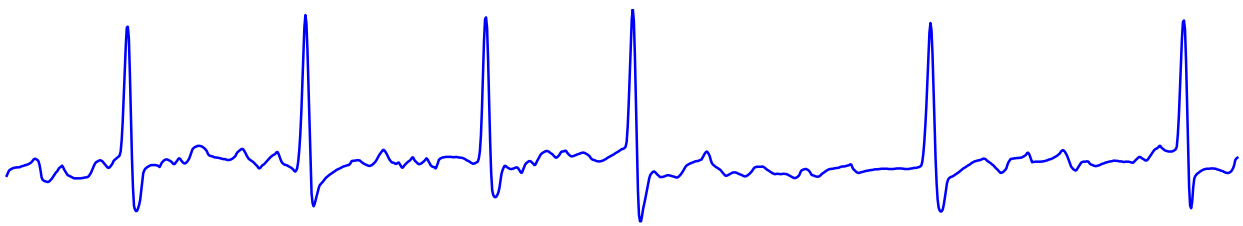
Smoothed Adversarial Perturbations (SAP)

Clinician's Ability to Distinguish



Adversarial Examples: AF to Normal

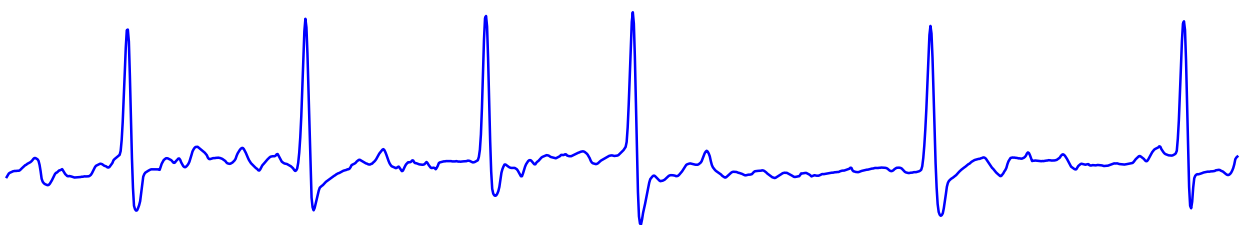
Original tracing
Prediction: AF
100% confidence



+

Smooth Perturbation

Combined tracing
Prediction: Normal
100% confidence



Adversarial Examples: Normal to AF

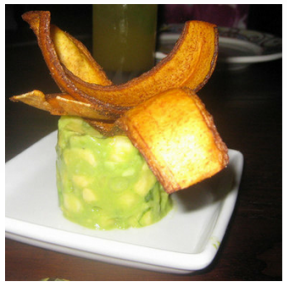
Worry 2

Maybe the model isn’t good enough

CLIP

- Language/Image model released by Open AI
- CLIP model trains on 256 GPUs for 2 weeks
- Over million 400M images processed
- Trained by predicting captions associated with images

guacamole (90.1%) Ranked 1 out of 101 labels



✓ a photo of **guacamole**, a type of food.

× a photo of **ceviche**, a type of food.

× a photo of **edamame**, a type of food.

× a photo of **tuna tartare**, a type of food.

× a photo of **hummus**, a type of food.

YOUTUBE-BB

airplane, person (89.0%) Ranked 1 out of 23

✓ a photo of a **airplane**.

× a photo of a **bird**.

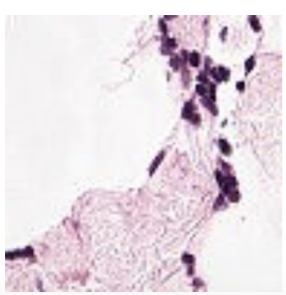
× a photo of a **bear**.

× a photo of a **giraffe**.

× a photo of a **car**.

PATCHCAMELYON (PCAM)

healthy lymph node tissue (22.8%) Ranked 2 out of 2



× this is a photo of **lymph node tumor tissue**

✓ this is a photo of **healthy lymph node tissue**

CLIP Failures

Granny Smith	85.6%
iPod	0.4%
library	0.0%
pizza	0.0%
toaster	0.0%
dough	0.1%

Granny Smith	0.1%
iPod	99.7%
library	0.0%
pizza	0.0%
toaster	0.0%
dough	0.0%

Any Idea Why?

Let's think about training a classifier

$$p(y \mid \mathbf{x})$$

It exists only when $p(\mathbf{x}) > 0$. Since

$$p(y \mid \mathbf{x}) = \frac{p(y, \mathbf{x})}{p(\mathbf{x})}$$

Is $p(\mathbf{x}) > 0$

Granny Smith	85.6%
iPod	0.4%
library	0.0%
pizza	0.0%
toaster	0.0%
dough	0.1%

Granny Smith	0.1%
iPod	99.7%
library	0.0%
pizza	0.0%
toaster	0.0%
dough	0.0%

Is $p(\mathbf{x}) > 0$

Granny Smith	85.6%
iPod	0.4%
library	0.0%
pizza	0.0%
toaster	0.0%
dough	0.1%

Granny Smith	0.1%
iPod	99.7%
library	0.0%
pizza	0.0%
toaster	0.0%
dough	0.0%

- In one sense yes, this image could exist on the internet
- In another sense no, this image probably did not exist on internet

Is $p(\mathbf{x}) > 0$

Granny Smith	85.6%
iPod	0.4%
library	0.0%
pizza	0.0%
toaster	0.0%
dough	0.1%

Granny Smith	0.1%
iPod	99.7%
library	0.0%
pizza	0.0%
toaster	0.0%
dough	0.0%

Again needs some kind of extrapolation

Is $p(\mathbf{x}) > 0$

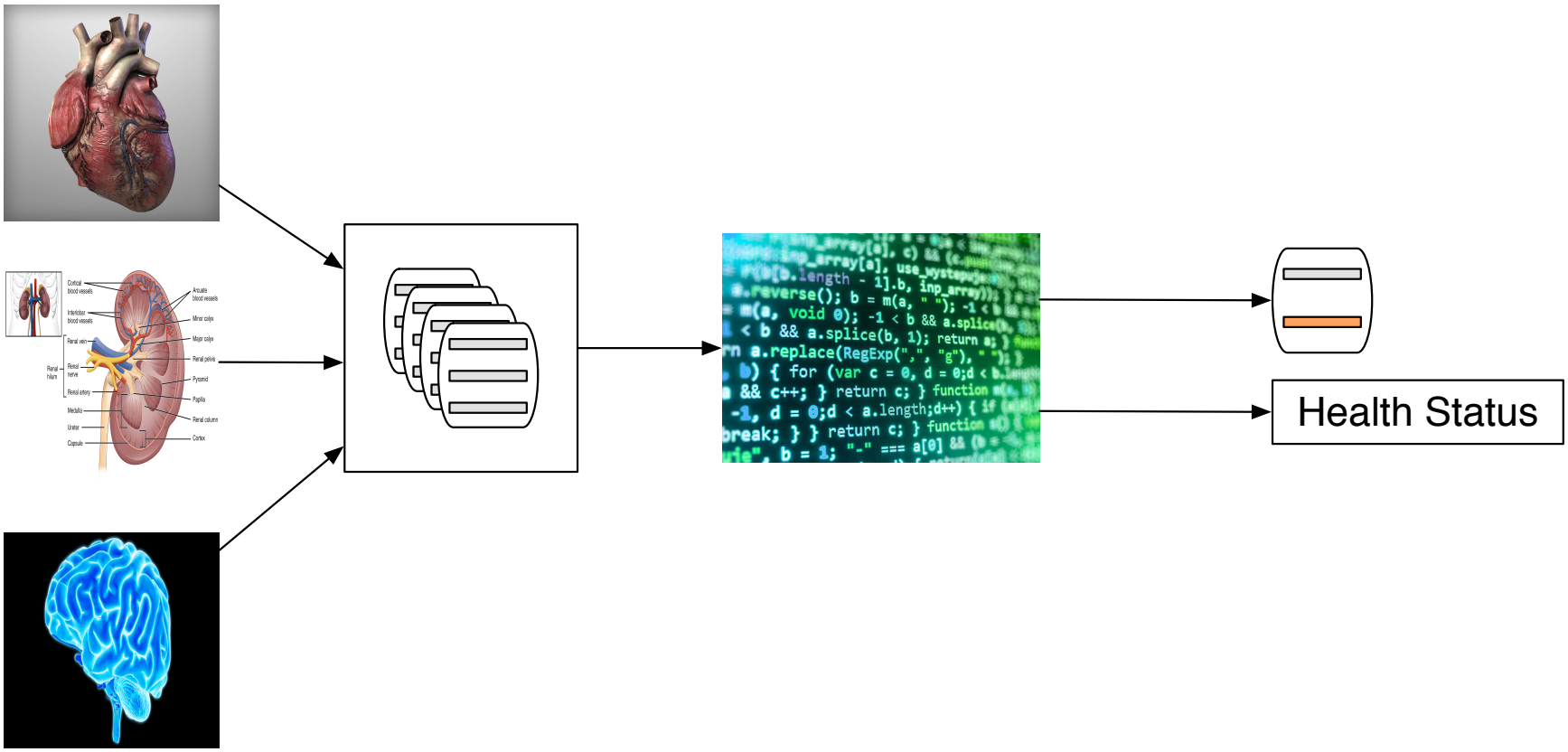
Granny Smith	85.6%
iPod	0.4%
library	0.0%
pizza	0.0%
toaster	0.0%
dough	0.1%

Granny Smith	0.1%
iPod	99.7%
library	0.0%
pizza	0.0%
toaster	0.0%
dough	0.0%

- If almost all things with the text iPod are iPods, is it a bad extrapolation?
- If text in images generally reveals the answer, is it bad to use?

Worry 3

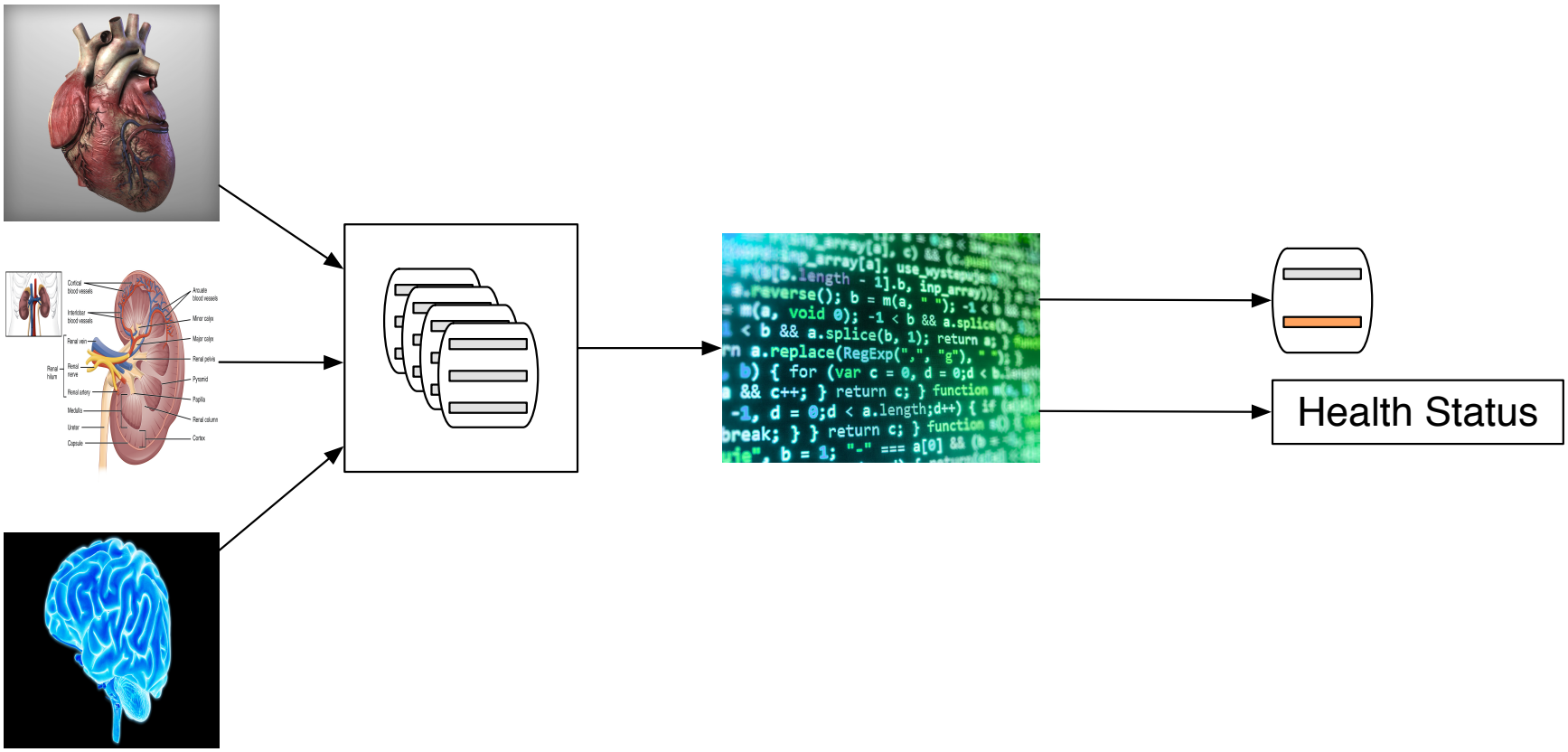
What is Model Explanation?



Explanations are domain specific

- Could be sparsity of inputs or features
- In health, could be organ or system based

What is Model Explanation?



Roughly understanding what inputs relate to the output

Is this important to do?

Is this important to do?

- We do AI
- We fit good models
- We check if models are good on test data

What makes a model?

A probabilistic model $\mathscr{M}$ trained with data $(\mathbf{x}, y) \sim q$

$$\mathscr{M}_{trained} = q + \text{Generalization-Error}$$

What makes a model?

A probabilistic model $\mathscr{M}$ trained with data $(\mathbf{x}, y) \sim q$

$$\mathscr{M}_{trained} = q + \text{Generalization-Error}$$

A little more formally:

$$\mathscr{M}_{trained} - q = \text{Generalization-Error}$$

What makes a model?

A probabilistic model $\mathscr{M}$ trained with data $(\mathbf{x}, y) \sim q$

$$\mathscr{M}_{trained} = q + \text{Generalization-Error}$$

A little more formally:

$$\mathbb{E}_{\mathbf{x}} KL(q(y \mid \mathbf{x}) || \mathscr{M}_{trained}(y \mid \mathbf{x})) = \text{Generalization-Error}$$

What makes a model?

A probabilistic model $\mathscr{M}$ trained with data $(\mathbf{x}, y) \sim q$

$$\mathscr{M}_{trained} = q + \text{Generalization-Error}$$

A little more formally:

$$\mathbb{E}_{\mathbf{x}} KL(q(y \mid \mathbf{x}) || \mathscr{M}_{trained}(y \mid \mathbf{x})) = \text{Generalization-Error}$$

For a well-trained model

$$\mathscr{M}_{trained} \approx q$$

Note that well-trained does not imply good predictions

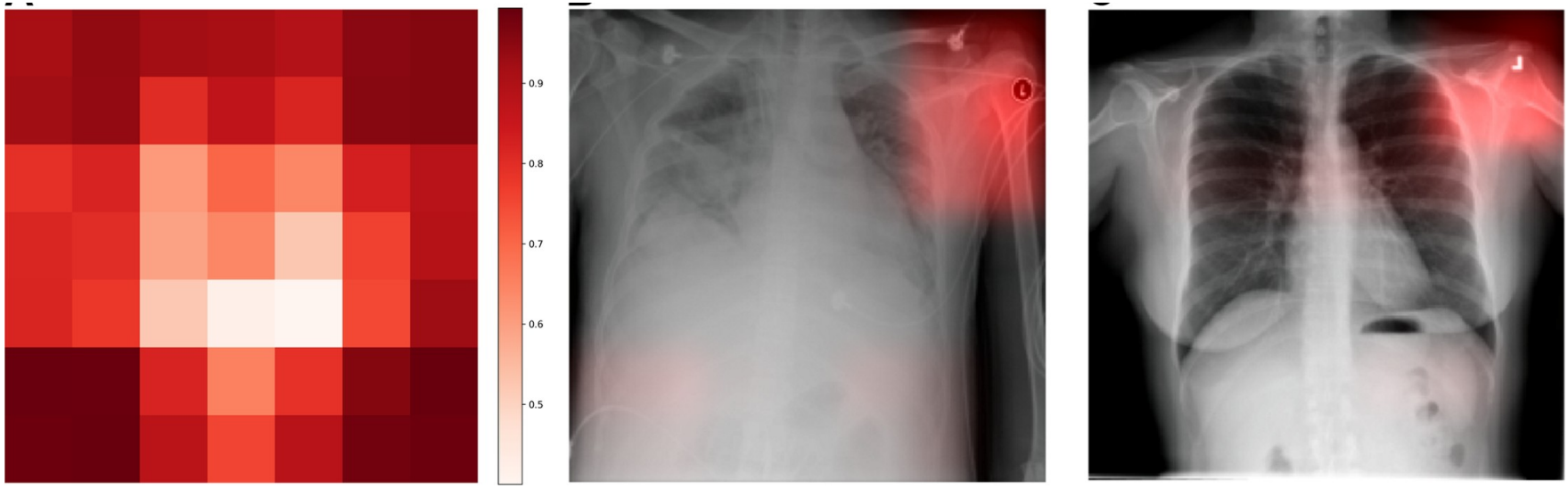
For a well trained model

$$\mathcal{M}_{trained} \approx q$$

Explaining this model means explaining the data distribution

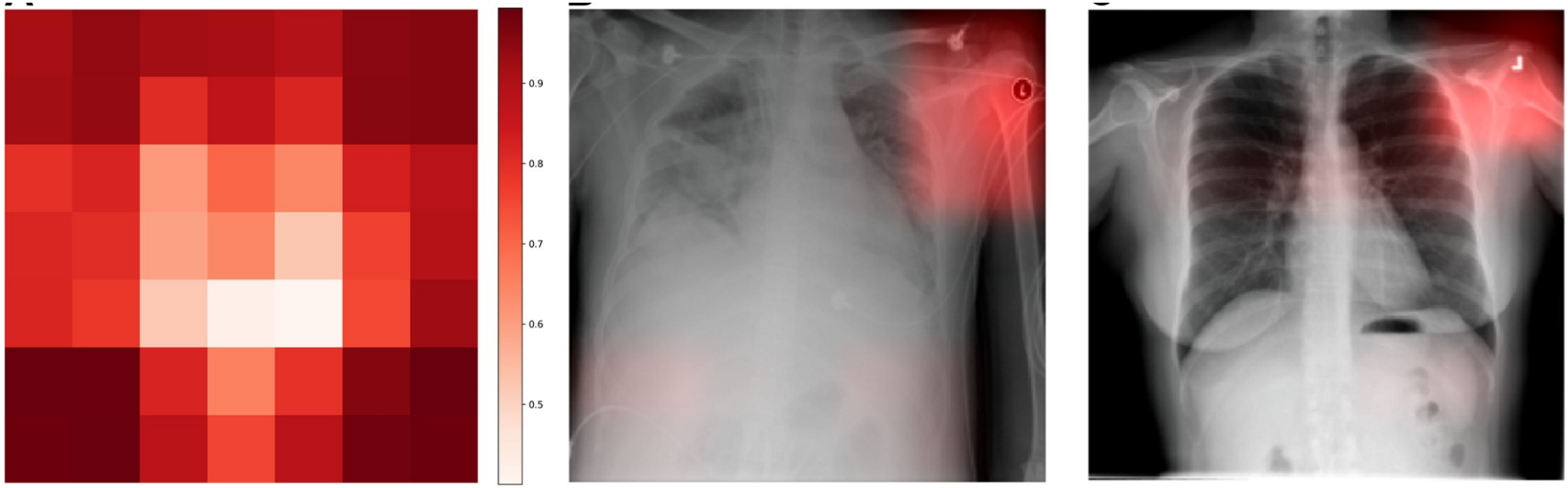
Why would we do this?

Predicting Hospitals from X-rays



[Zech+ 2018]

Predicting Hospitals from X-rays



[Zech+ 2018]

If

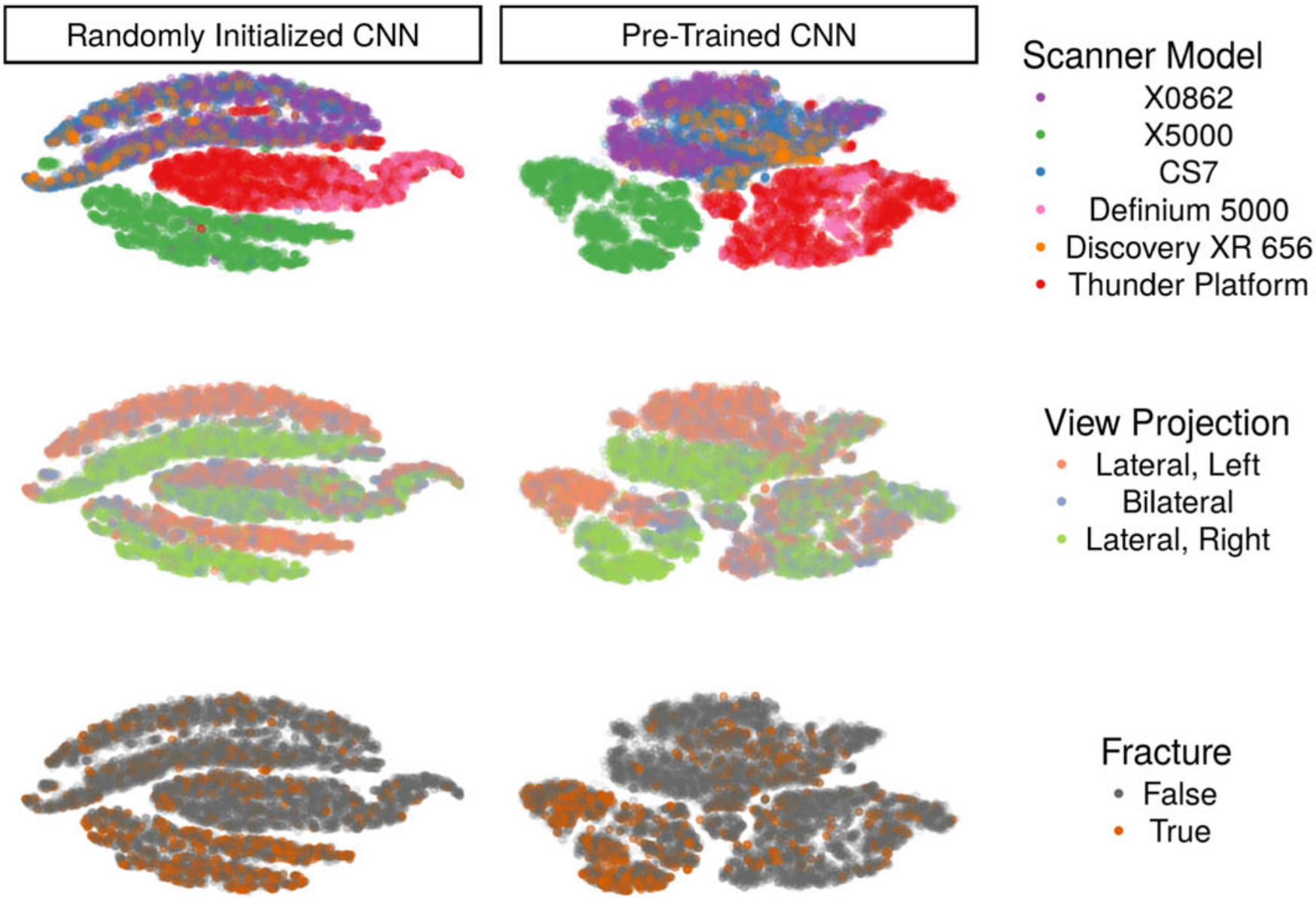
$$y \not\perp \text{metal-token}$$

and

$$\text{metal-token} \not\perp \text{x-ray}$$

Potentially won't generalize. *An issue with data*

Predicting from Hip X-rays



[Badgeley+ 2019]

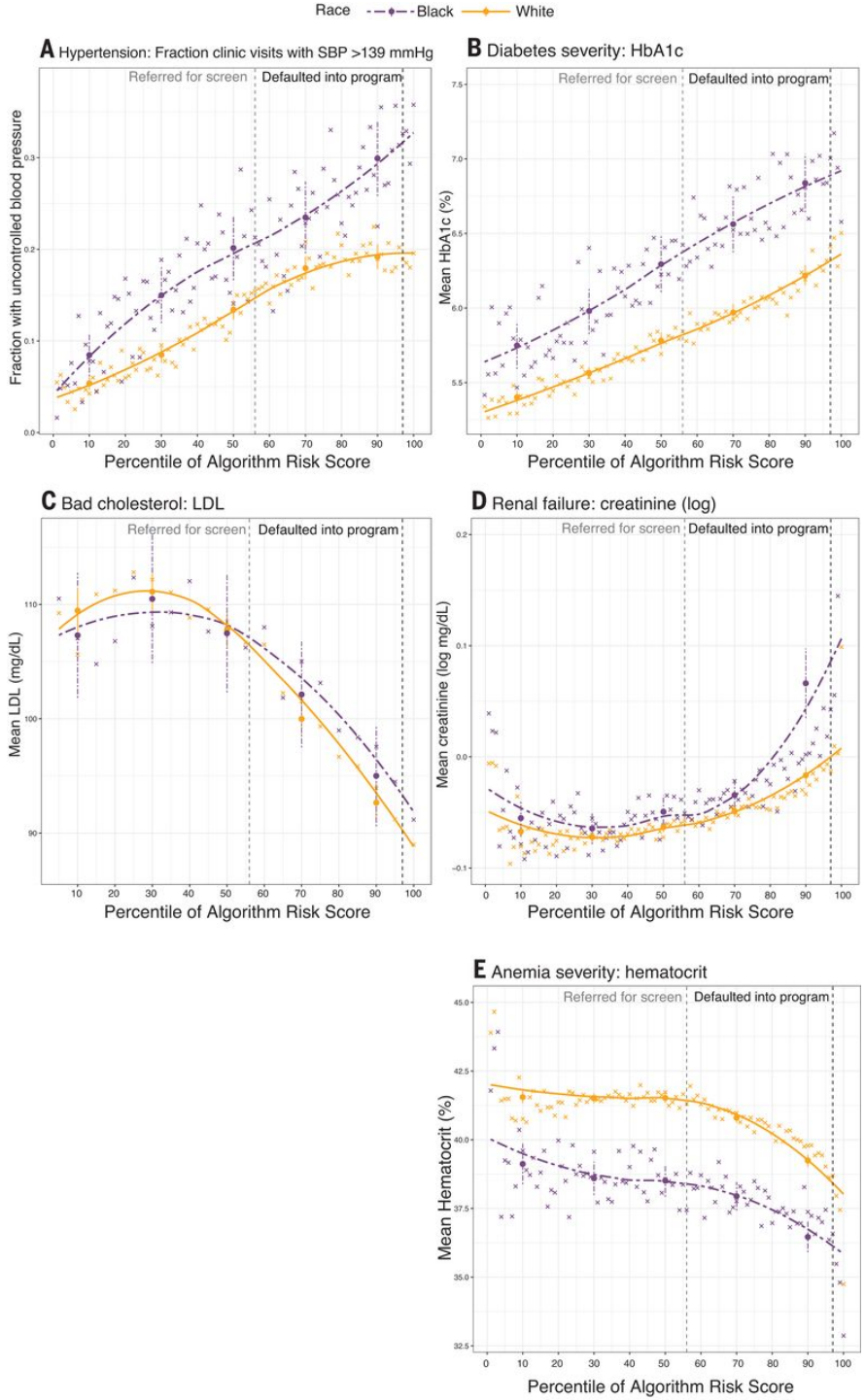
- Bias and causal associations in observational research [Grimes+ 2002]
- Biases in electronic health record data due to processes within the healthcare system: retrospective observational study [Agniel+ 2018]
- Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: A cross-sectional study [Zech+ 2018]
- Deep learning predicts hip fracture using confounding patient and healthcare variables [Badgeley+ 2019]
- Hidden Stratification Causes Clinically Meaningful Failures in Machine Learning for Medical Imaging [Oakden-Rayner+ 2019]

Worry 4

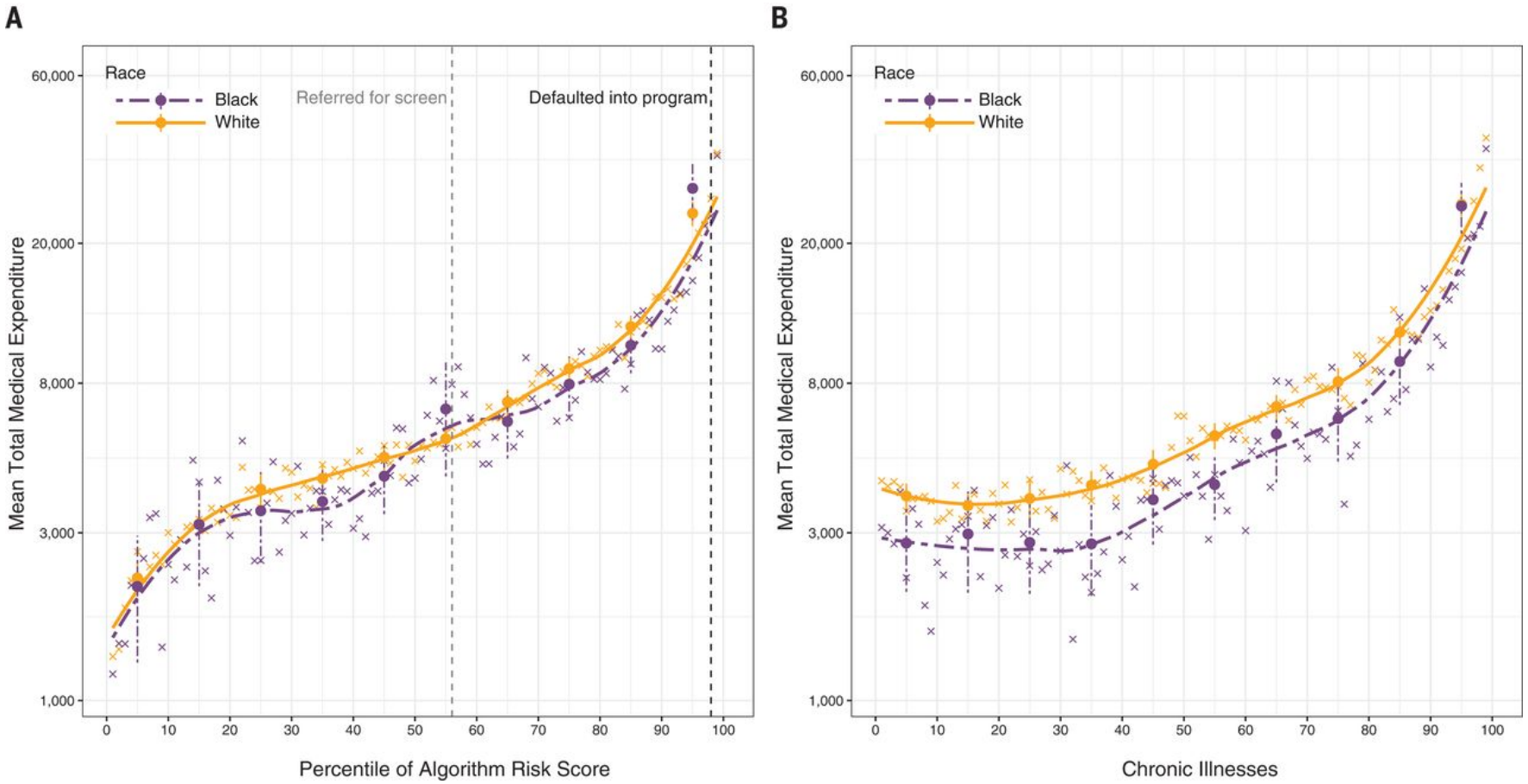
Insurance asks us to build a model to reduce costs

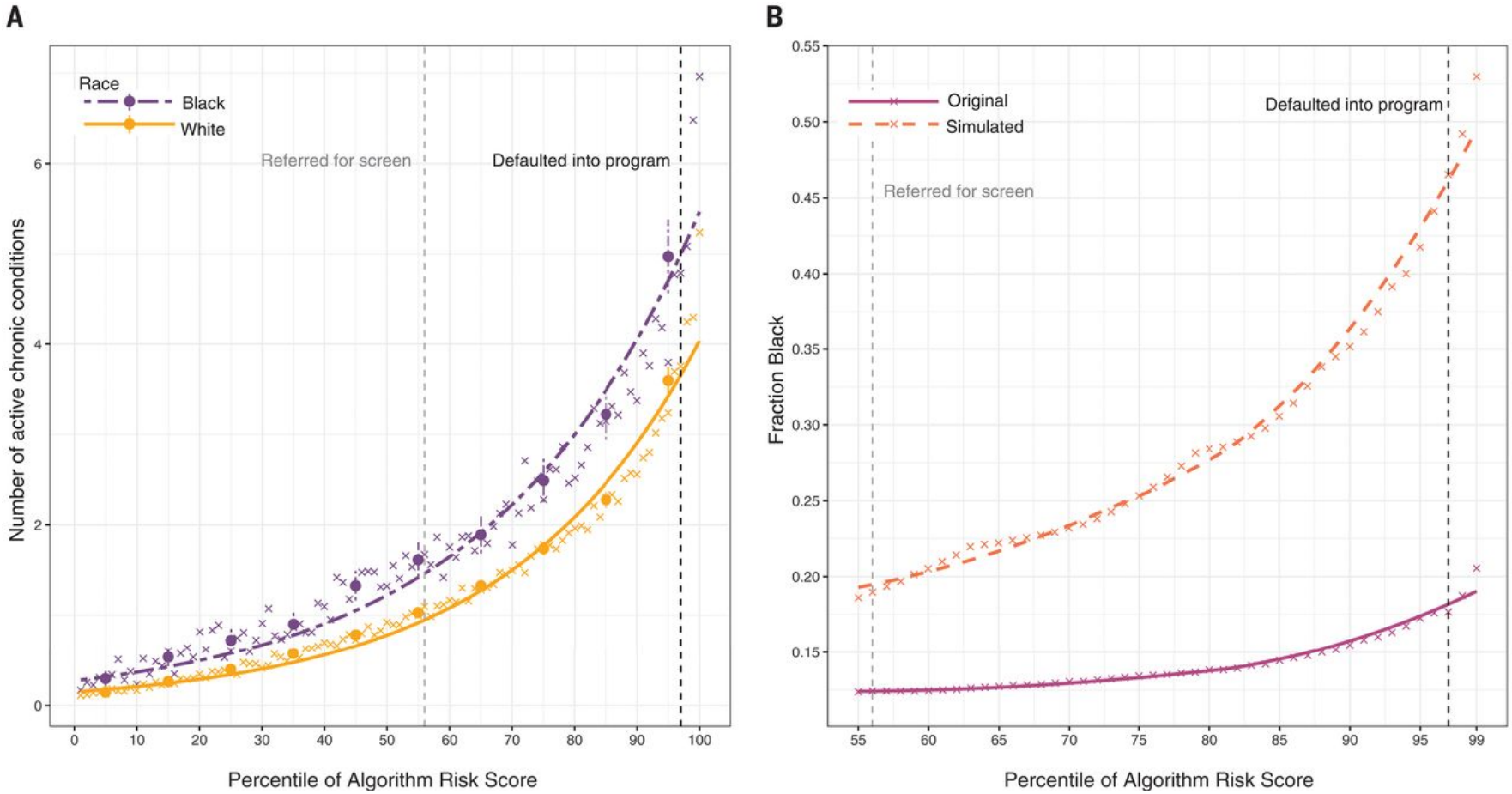
- *Idea take data and predict who will have more cost*
- Then give the predicted high-cost people extra attention early to lower cost

Building such a model works well [Obermeyer+ 2019]



Something wrong?


A
Race
Black
White
Referred for screen
Defaulted into program
Mean Total Medical Expenditure
60,000
20,000
8,000
3,000
1,000
0
10
20
30
40
50
60
70
80
90
100
Percentile of Algorithm Risk Score
B
Race
Black
White
Mean Total Medical Expenditure
60,000
20,000
8,000
3,000
1,000
0
10
20
30
40
50
60
70
80
90
100
Chronic Illnesses


A
Race
Black
White
Referred for screen
Defaulted into program
Number of active chronic conditions
0
2
4
6
0
10
20
30
40
50
60
70
80
90
100
Percentile of Algorithm Risk Score
B
Original
Simulated
Defaulted into program
Referred for screen
Fraction Black
0.15
0.20
0.25
0.30
0.35
0.40
0.45
0.50
0.55
55
60
65
70
75
80
85
90
95
99
Percentile of Algorithm Risk Score

What went wrong?

- Predicting costs is not inherently wrong
- Costs are a function of utilization, which differs across groups
- Why does it differ?

Model amplifies existing inequity

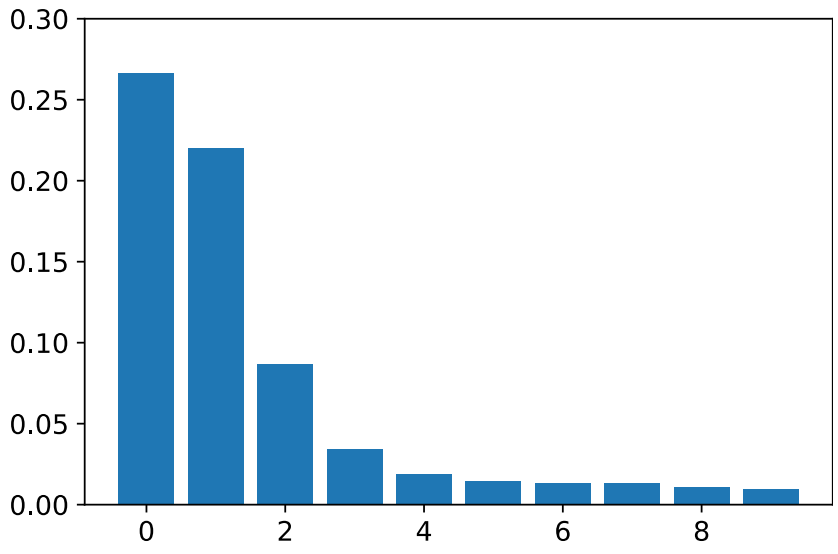
- What about credit scoring?
- What about criminal recidivism?

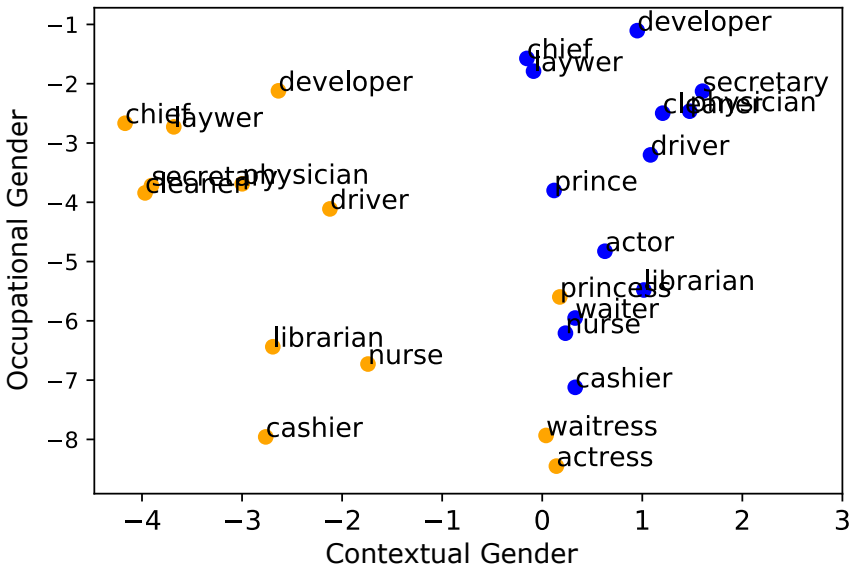
Worry 5

Modern Language Models are Quite Powerful

System	Dev		Test	
	EM	F1	EM	F1
Top Leaderboard Systems (Dec 10th, 2018)				
Human	-	-	82.3	91.2
#1 Ensemble - nlnet	-	-	86.0	91.7
#2 Ensemble - QANet	-	-	84.5	90.5
Published				
BiDAF+ELMo (Single)	-	85.6	-	85.8
R.M. Reader (Ensemble)	81.2	87.9	82.3	88.5
Ours				
$\text{BERT}_{\text{BASE}}$ (Single)	80.8	88.5	-	-
$\text{BERT}_{\text{LARGE}}$ (Single)	84.1	90.9	-	-
$\text{BERT}_{\text{LARGE}}$ (Ensemble)	85.8	91.8	-	-
$\text{BERT}_{\text{LARGE}}$ (Sgl.+TriviaQA)	**84.2**	**91.1**	**85.1**	**91.8**
$\text{BERT}_{\text{LARGE}}$ (Ens.+TriviaQA)	**86.2**	**92.2**	**87.4**	**93.2**

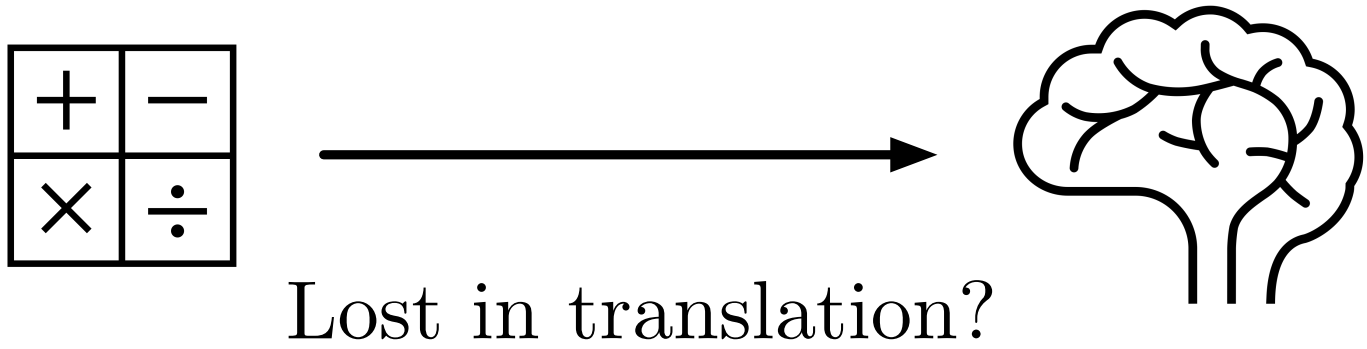
Modern Language Models Encode Biases





Worry 6: The methodology

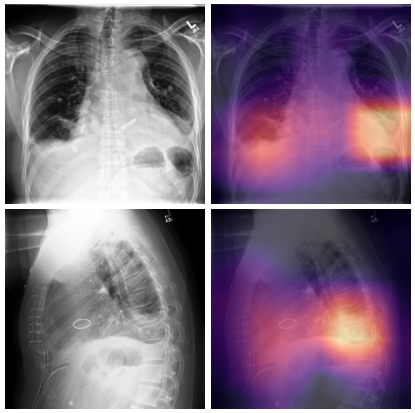
What is an Interpretation?

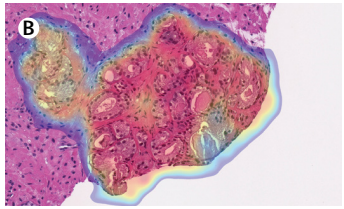


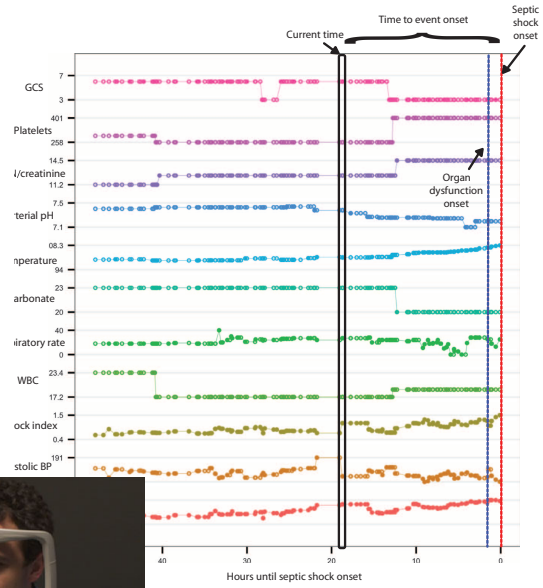
- Math defines a quantity, quantity needs to be translated
- Issues faced no different than in linear regression

$$f(\mathbf{x}) = \boldsymbol{\beta}^\top \mathbf{x}$$

Individual features less well-defined







Need something more tailored to an instance

Gradient of data generating distribution

$$\nabla_{\mathbf{x}} F(y \,|\, \mathbf{x})$$

is well defined math

- But it's a local quantity. Does any gradient stay on data manifold?
- Simplicity and speed make it popular
- Gradient methods fail evaluations like predictability
  [Hooker+ 2019]

Fidelity to data

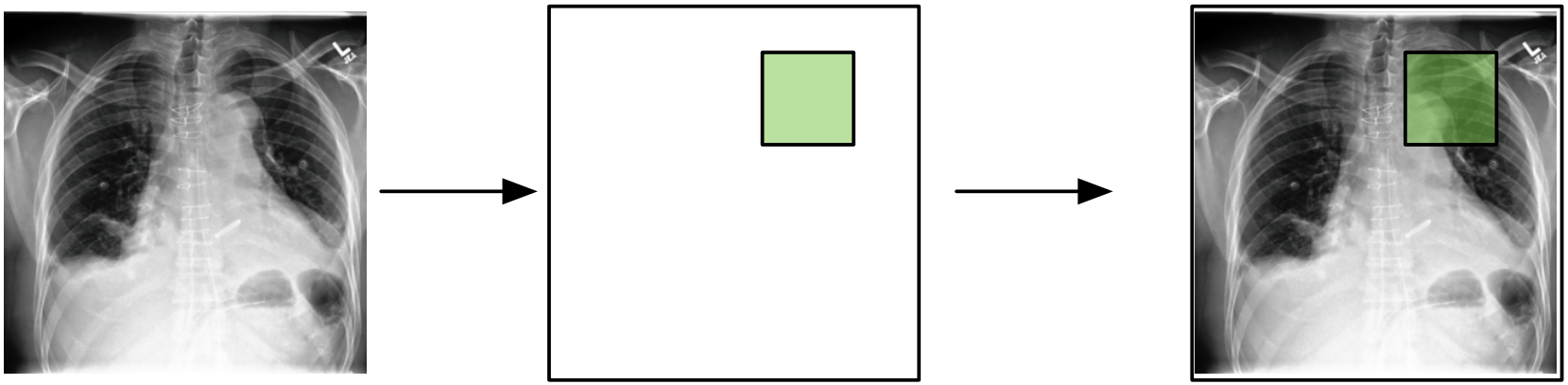
Simple to follow

Fast to compute

Fidelity to data | Simple to follow | Fast to compute

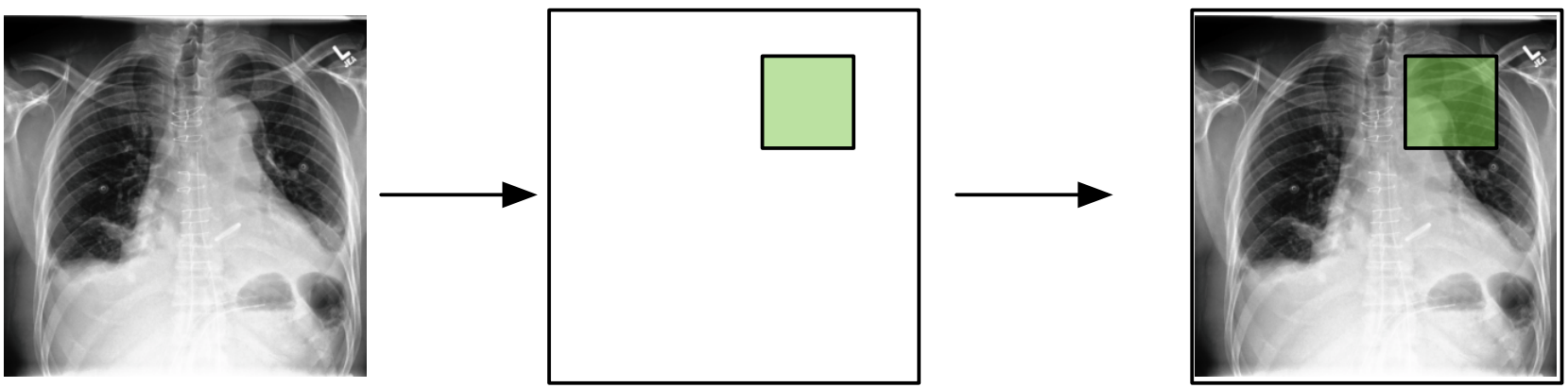
One thing: Subset of **x** that retains a lot of predictability

Learning to Explain



$\boldsymbol{x}$ $\boldsymbol{b} \sim S_{\boldsymbol{\beta}}(\boldsymbol{x})$ $m(\boldsymbol{b}, \boldsymbol{x})$

Learning to Explain

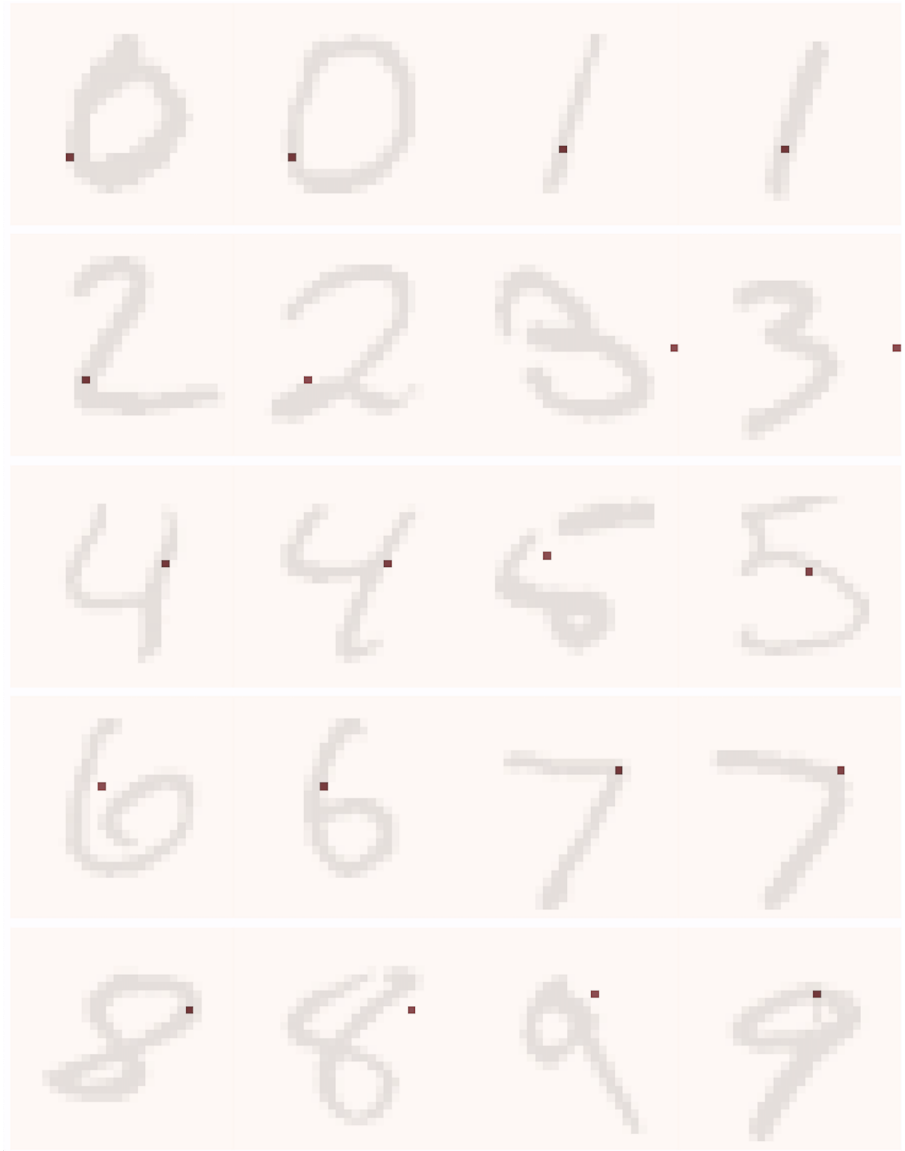


$\boldsymbol{x}$ $\quad$ $\boldsymbol{b} \sim S_{\boldsymbol{\beta}}(\boldsymbol{x})$ $\quad$ $m(\boldsymbol{b}, \boldsymbol{x})$

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\beta}) = E_{F(\mathbf{x},y)} E_{\mathbf{b}\sim S_{\boldsymbol{\beta}}(\mathbf{x})} [\underbrace{\log p(y \mid m(\mathbf{b}, \mathbf{x}); \boldsymbol{\theta})}_{\text{predict from selection}} - \underbrace{\lambda R(\mathbf{b})}_{\text{to encourage simplicity}}]$$

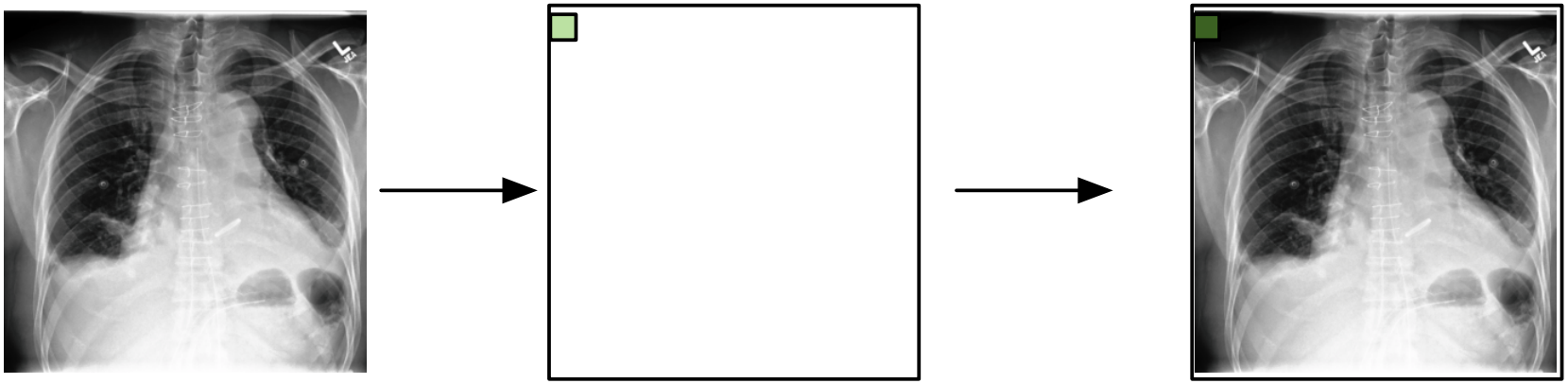
[L2X: Chen+ 2018, INVASE: Yoon+ 2019]

Learning to Explain on MNIST



Gets 96% accuracy. What's going on?

What's going on?

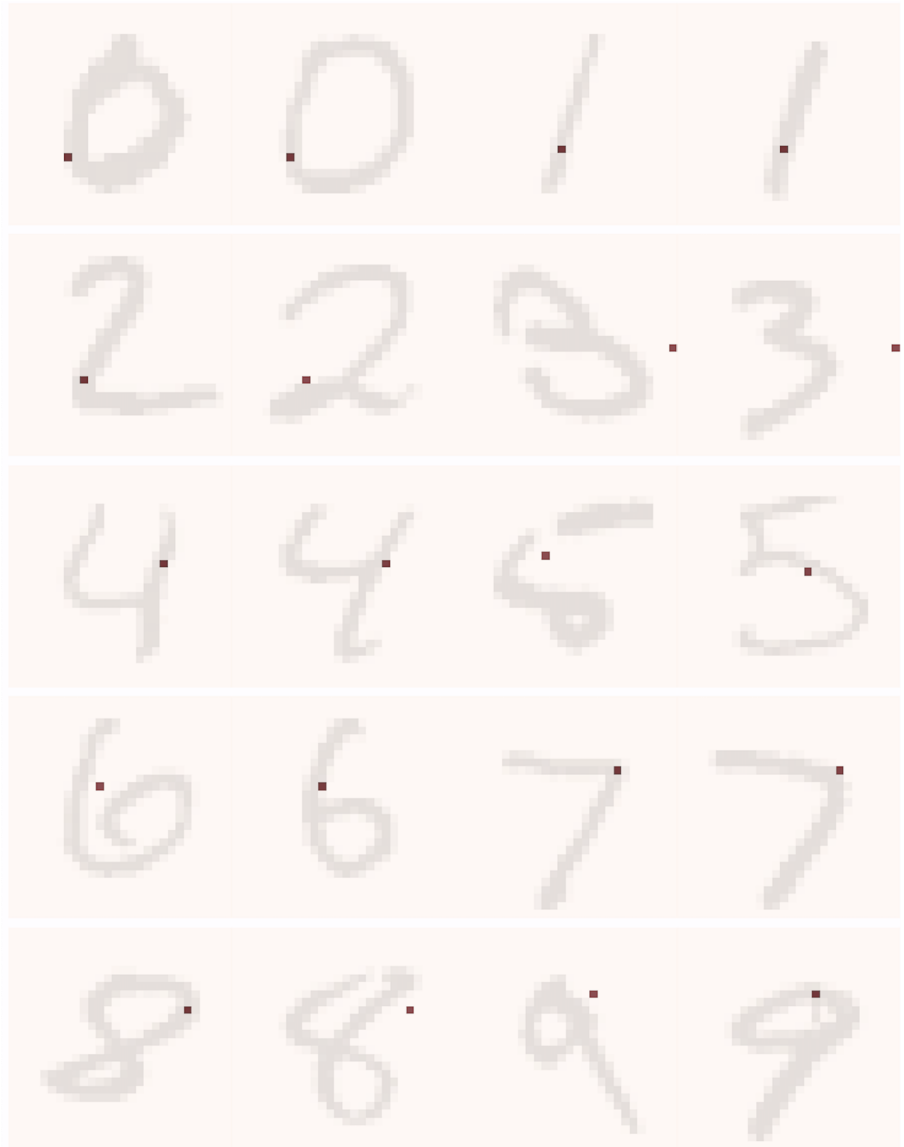


$\boldsymbol{x}$ $\quad$ $\boldsymbol{b} \sim S_{\boldsymbol{\beta}}(\boldsymbol{x})$ $\quad$ $m(\boldsymbol{b}, \boldsymbol{x})$

Outputs $\boldsymbol{b}[i] = 1$

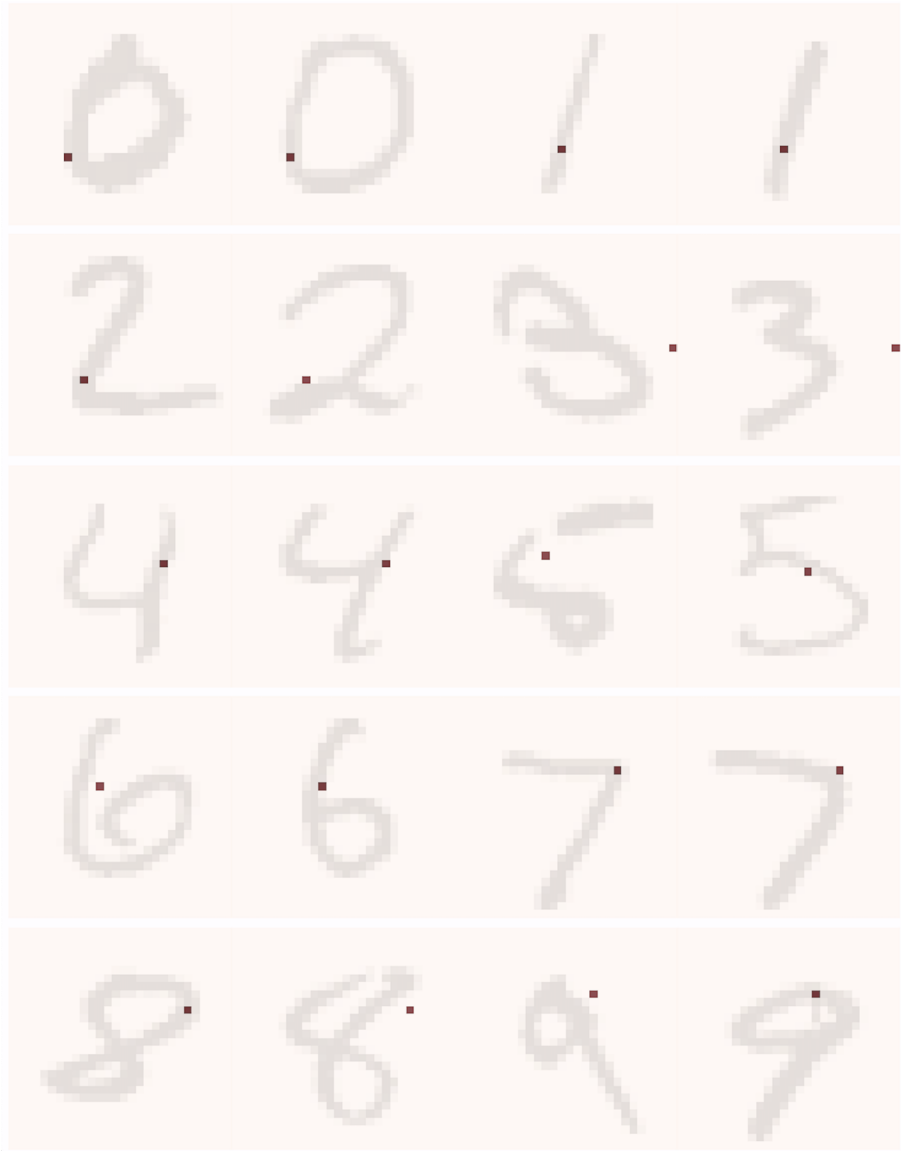
$\iff$ Prediction of $\boldsymbol{x} = i$

$$\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\beta}) = E_{F(\mathbf{x},y)} E_{\mathbf{b}\sim S_{\boldsymbol{\beta}}(\mathbf{x})} [\underbrace{\log p(y \mid m(\mathbf{b}, \mathbf{x}); \boldsymbol{\theta})}_{\text{predict from selection}} - \underbrace{\lambda R(\mathbf{b})}_{\text{to encourage simplicity}}]$$



- The selector makes the prediction
- Selection locations can transmit lots of information
- Need something that explains the selector...

Why do evaluations fail?



- Retraining evaluations like ROAR [Hooker+ 2019] would fail
- Retrained models can coadapt to selections because selections encode the prediction
- Retraining is a good idea to avoid evaluating on OOD inputs

Worry 7: The methodology

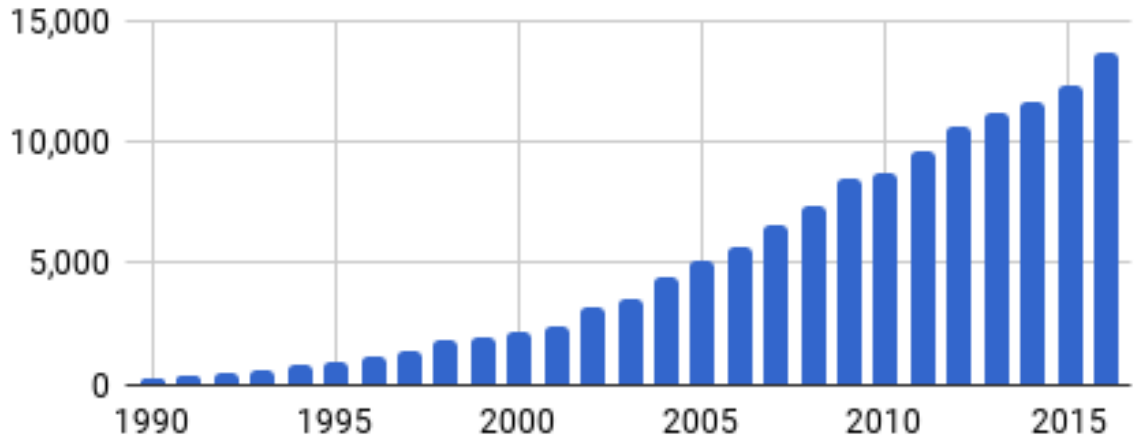


Figure 1: Growth of published reinforcement learning papers. Shown are the number of RL-related publications (y-axis) per year (x-axis) scraped from Google Scholar searches.

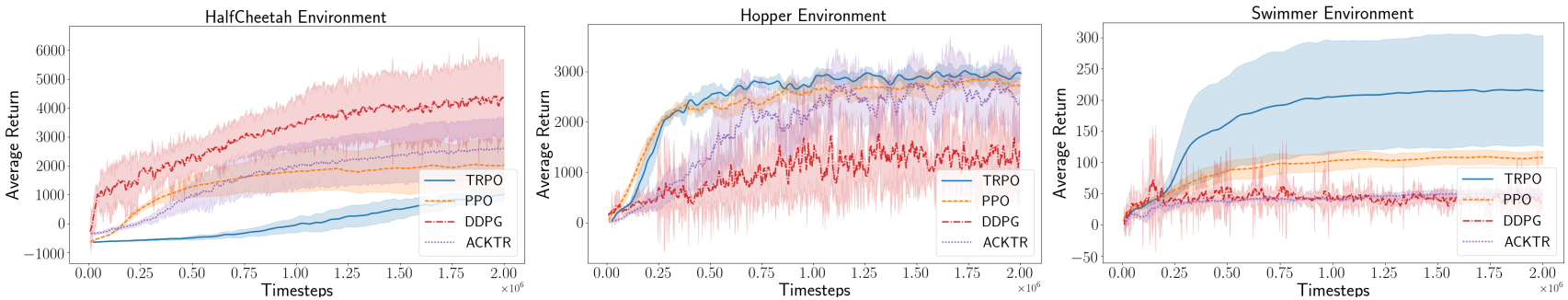
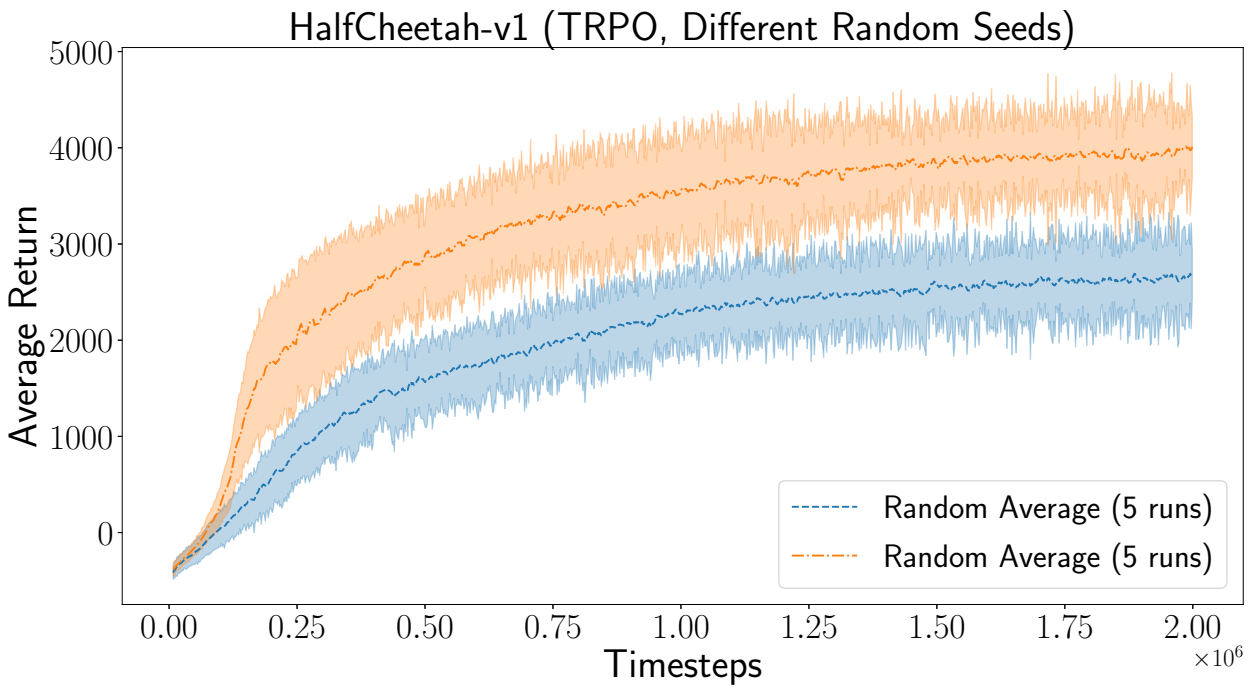


Figure 4: Performance of several policy gradient algorithms across benchmark MuJoCo environment suites


HalfCheetah-v1 (TRPO, Different Random Seeds)
Average Return
5000
4000
3000
2000
1000
0
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Timesteps
×10^6
Random Average (5 runs)
Random Average (5 runs)

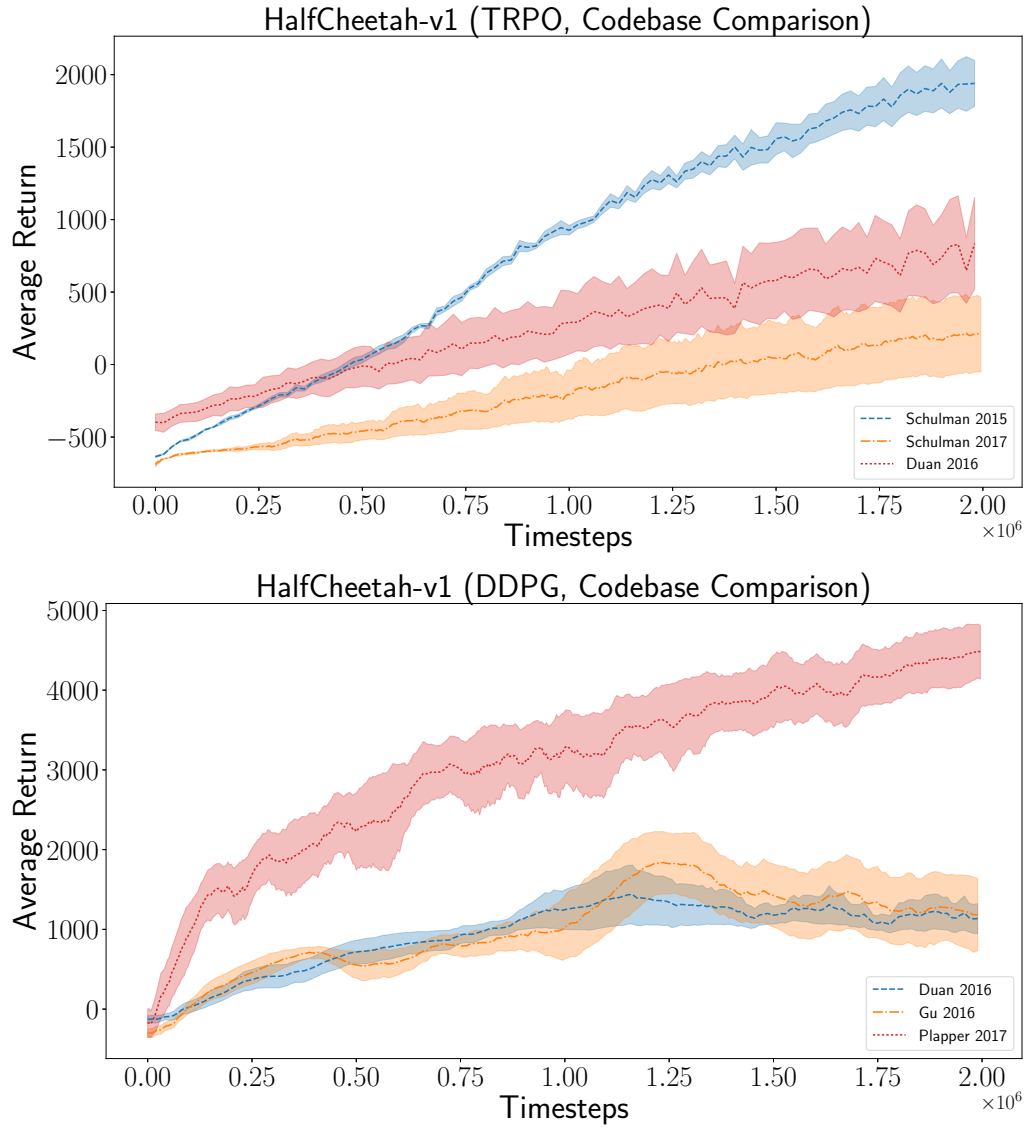
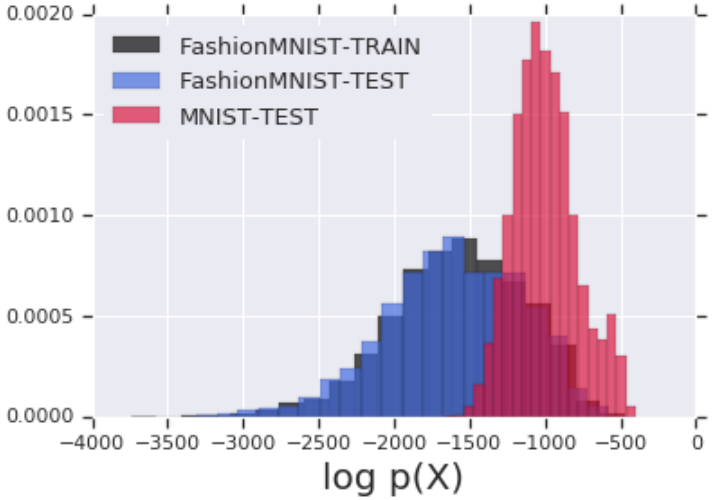
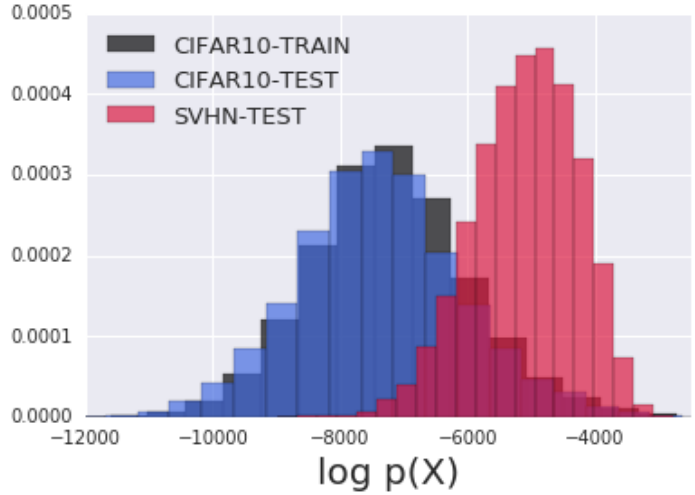


Figure 6: TRPO codebase comparison using our default set of hyperparameters (as used in other experiments).

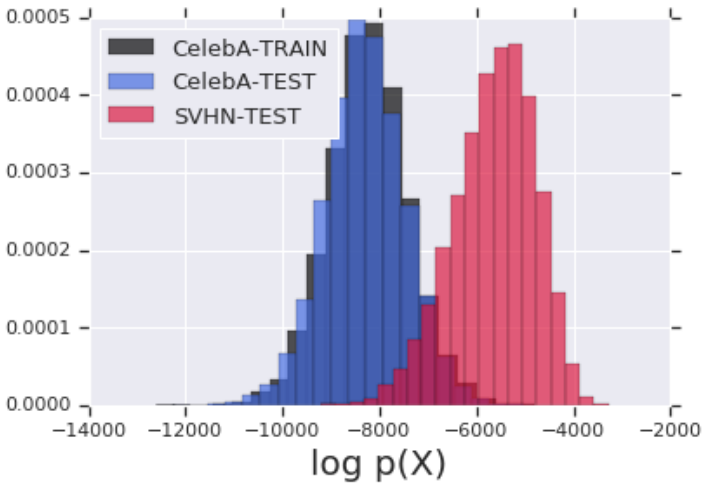
Worry 8: Generative Models



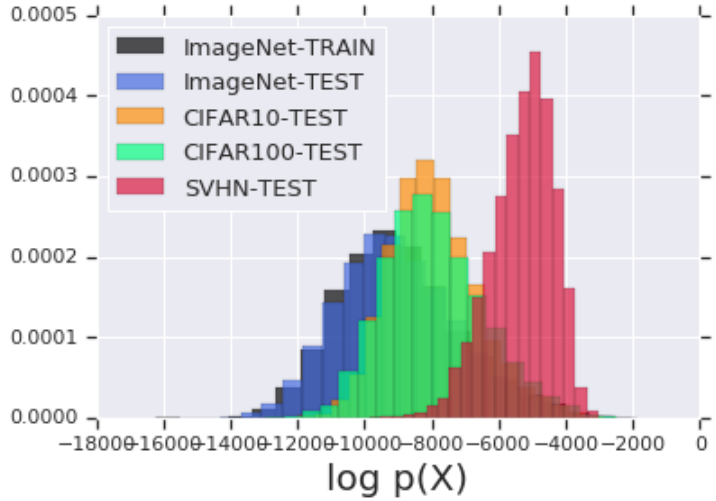
(a) Train on FashionMNIST, Test on MNIST



(b) Train on CIFAR-10, Test on SVHN



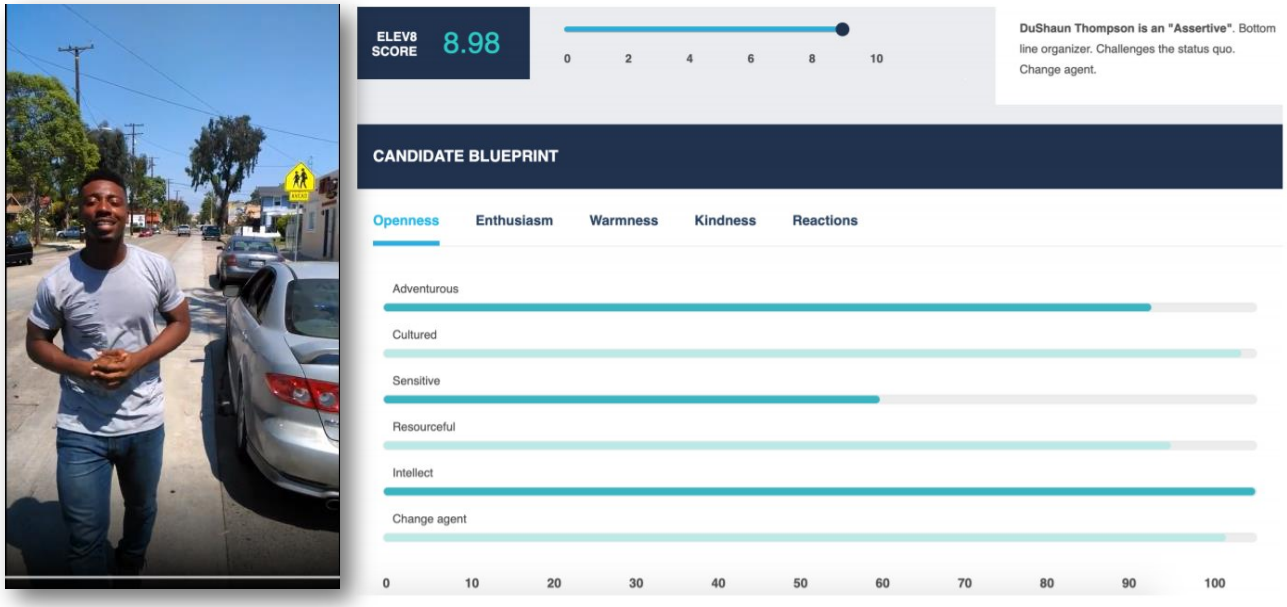
(c) Train on CelebA, Test on SVHN



(d) Train on ImageNet,
Test on CIFAR-10 / CIFAR-100 / SVHN

Worry 9: Why was this class so focused on principles?

Assessing personality & job suitability from 30-second video



[Arvind Narayanan, Raghavan+ 2019]

Vendor name	Funding	# of employees	Location
8 and Above	–	1-10	WA, USA
ActiView	$6.5M	11-50	Israel
Applied	£2M	11-50	UK
Assessment Innovation	$1.3M	1-10	NY, USA
Good&Co	$10.3M	51-100	CA, USA
Harver	$14M	51-100	NY, USA
HireVue	$93M	251-500	UT, USA
impress.ai	$1.4M	11-50	Singapore
Knockri	–	11-50	Canada
Koru	$15.6M	11-50	WA, USA
LaunchPad Recruits	£2M	11-50	UK
myInterview	$1.4M	1-10	Australia
Plum.io	$1.9M	11-50	Canada
PredictiveHire	A$4.3M	11-50	Australia
pymetrics	$56.6M	51-100	NY, USA
Scoutible	$6.5M	1-10	CA, USA
Teamscope	€800K	1-10	Estonia
ThriveMap	£781K	1-10	UK
Yobs	$1M	11-50	CA, USA

Not all these companies offer AI assessments of job candidates, but most do.

[Arvind Narayanan, Raghavan+ 2019]

Be wary of AI snake oil

["Snake Oil": From Arvind Narayanan]

Bonus


From survival prediction
to treatment decision
in lung cancer
Wouter A.C. van Amsterdam

Bonus


From survival prediction
to treatment decision
in lung cancer
Wouter A.C. van Amsterdam

What we've learned already

- Variational autoencoders - have to do inference with latent variables
- Normalizing flows - need functions that integrate to 1
- GANs - Minimax problems and problems with overlap

What we've learned already

- Variational autoencoders - have to do inference with latent variables
- Normalizing flows - need functions that integrate to 1
- GANs - Minimax problems and problems with overlap

A new/old entrant: energy-based models

$$p(\mathbf{x}) \propto \exp(-E_{\boldsymbol{\theta}}(\mathbf{x})) = \exp(-\nabla_{\mathbf{x}} \log p(\mathbf{x}))$$

where the normalization constant is

$$C(\boldsymbol{\theta}) = \log \int_{\text{support}} \exp(-E_{\boldsymbol{\theta}}(\mathbf{x}))d\mathbf{x}$$

- Unlike normalizing flows no need for $E_{\boldsymbol{\theta}}$ to integrate to 1
- Unlike GANs not obviously minimax

For maximum likelihood

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{F(\mathbf{x})}[\log \exp(-E_{\boldsymbol{\theta}}(\mathbf{x}) - C(\boldsymbol{\theta}))] = -\mathbb{E}_{F(\mathbf{x})}[\nabla_{\boldsymbol{\theta}} E_{\boldsymbol{\theta}}(\mathbf{x})] - \nabla_{\boldsymbol{\theta}} C(\boldsymbol{\theta})$$

$$= -\mathbb{E}_{F(\mathbf{x})}[\nabla_{\boldsymbol{\theta}} E_{\boldsymbol{\theta}}(\mathbf{x})] - \nabla_{\boldsymbol{\theta}} \log \int_{\text{support}} \exp(-E_{\boldsymbol{\theta}}(\mathbf{x}))d\mathbf{x}$$

$$= -\mathbb{E}_{F(\mathbf{x})}[\nabla_{\boldsymbol{\theta}} E_{\boldsymbol{\theta}}(\mathbf{x})] - \frac{\nabla_{\boldsymbol{\theta}} \int_{\text{support}} \exp(-E_{\boldsymbol{\theta}}(\mathbf{x}))d\mathbf{x}}{\int_{\text{support}} \exp(-E_{\boldsymbol{\theta}}(\mathbf{x}))d\mathbf{x}}$$

$$= -\mathbb{E}_{F(\mathbf{x})}[\nabla_{\boldsymbol{\theta}} E_{\boldsymbol{\theta}}(\mathbf{x})] - \frac{\int_{\text{support}} -\nabla_{\boldsymbol{\theta}} E_{\boldsymbol{\theta}}(\mathbf{x}) \exp(-E_{\boldsymbol{\theta}}(\mathbf{x}))d\mathbf{x}}{\int_{\text{support}} \exp(-E_{\boldsymbol{\theta}}(\mathbf{x}))d\mathbf{x}}$$

$$= -\mathbb{E}_{F(\mathbf{x})}[\nabla_{\boldsymbol{\theta}} E_{\boldsymbol{\theta}}(\mathbf{x})] - \frac{\int_{\text{support}} -\nabla_{\boldsymbol{\theta}} E_{\boldsymbol{\theta}}(\mathbf{x}) \exp(-E_{\boldsymbol{\theta}}(\mathbf{x}))d\mathbf{x}}{\exp(C(\boldsymbol{\theta}))}$$

$$= -\mathbb{E}_{F(\mathbf{x})}[\nabla_{\boldsymbol{\theta}} E_{\boldsymbol{\theta}}(\mathbf{x})] - \int_{\text{support}} -\nabla_{\boldsymbol{\theta}} E_{\boldsymbol{\theta}}(\mathbf{x}) \exp(-E_{\boldsymbol{\theta}}(\mathbf{x}) - C(\boldsymbol{\theta}))d\mathbf{x}$$

$$= -\mathbb{E}_{F(\mathbf{x})}[\nabla_{\boldsymbol{\theta}} E_{\boldsymbol{\theta}}(\mathbf{x})] + \mathbb{E}_{p_{\boldsymbol{\theta}}}[\nabla_{\boldsymbol{\theta}} E_{\boldsymbol{\theta}}(\mathbf{x})]$$

Needs samples from the model, which is hard

Desiderata

Would like a model

- Doesn't require working hard for inference
- Can easily sample from it
- Can use generic functions
- Not a minimax problem

An idea

Make a latent variable, so inference is easy

Fact: Take an $\mathbf{x} \sim F$, then run

$$d\mathbf{x}_t = -\frac{1}{2}\mathbf{x}_t dt + dW_t,$$

then $\mathbf{x}_t$ converges in distribution to a sample from Normal$(0, 1)$.

- Regardless of the data distribution, this process converts data to a sequence of latent variables with known distribution.
- Inference is fixed to something easy

How do you pick a model?

- Can easily sample from it
- Can use generic functions

Idea make the model, the inference process in reverse

$$d\mathbf{x}_t = h_{\boldsymbol{\theta}}(\mathbf{x}_t, t)dt + dW_t,$$

How do you pick a model?

- Can easily sample from it
- Can use generic functions

Idea make the model, the inference process in reverse

$$d\mathbf{x}_t = h_{\boldsymbol{\theta}}(\mathbf{x}_t, t)dt + dW_t,$$

- Sampling is easy just run the process
- No restrictions on $h_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$

Training yields and ELBO like other latent variable models

Results

[Kingma+ 2022]

Conditional Generation

Really the goal is to generate some $\mathbf{x}$ given some $\mathbf{y}$

Setting

$$h_{\boldsymbol{\theta}}(\mathbf{x}_t, t) = s_{\boldsymbol{\theta}}(\mathbf{x}_t, T - t) + \frac{1}{2}\mathbf{x}_t$$

means the optimal $s_{\boldsymbol{\theta}}$ is the score $\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t)$.

For the conditional, the needed score is

$$\nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t \,|\, \mathbf{y}) = \nabla_{\mathbf{x}_t} \log q(\mathbf{x}_t, \mathbf{y}) = \nabla_{\mathbf{x}_t} \log q(\mathbf{y} \,|\, \mathbf{x}) + \nabla_{\mathbf{x}_t} \log q(\mathbf{x})$$

Convert the marginal trained model into conditional model by using the derivative of a classifier

Cooler results

Text-to-Image Synthesis on LAION. 1.45B Model.

'A street sign that reads "Latent Diffusion" '	*'A zombie in the style of Picasso'*	*'An image of an animal half mouse half octopus'*	*'An illustration of a slightly conscious neural network'*	*'A painting of a squirrel eating a burger'*	*'A watercolor painting of a chair that looks like an octopus'*	*'A shirt with the inscription: "I love generative models!" '*

[Rombach+ 2022]

I am excited to see all the projects next week