ILLINOIS

UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN

Artificial Intelligence Qualifying Exam

Alli Nilles

October 5, 2017

University of Illinois at Urbana-Champaign

Outline

- Brief overview of my research projects

Outline

- Brief overview of my research projects
- *Understanding Black Box Predictions via Influence Functions*

Outline

- Brief overview of my research projects
- *Understanding Black Box Predictions via Influence Functions*
    - what is “influence”?

Outline

- Brief overview of my research projects
- *Understanding Black Box Predictions via Influence Functions*
    - what is “influence”?
    - validation

Outline

- Brief overview of my research projects
- *Understanding Black Box Predictions via Influence Functions*
    - what is “influence”?
    - validation
    - applications

Outline

- Brief overview of my research projects
- *Understanding Black Box Predictions via Influence Functions*
  - what is “influence”?
  - validation
  - applications
- *Generating Plans that Predict Themselves*

Outline

- Brief overview of my research projects
- *Understanding Black Box Predictions via Influence Functions*
    - what is “influence”?
    - validation
    - applications
- *Generating Plans that Predict Themselves*
    - defining what makes a plan t-predictable

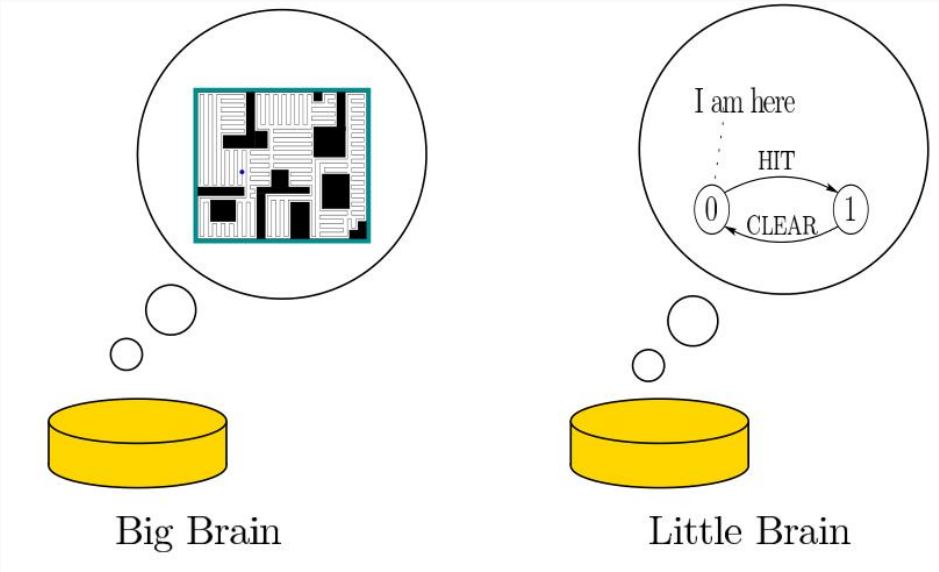
Outline

- Brief overview of my research projects
- *Understanding Black Box Predictions via Influence Functions*
    - what is “influence”?
    - validation
    - applications
- *Generating Plans that Predict Themselves*
    - defining what makes a plan *t*-predictable
    - instantiation and experiments

My Research

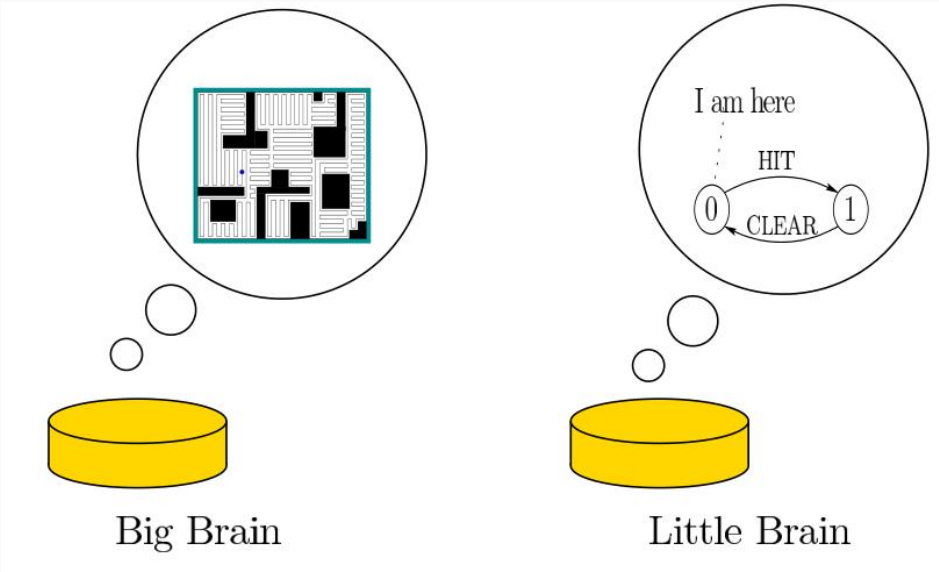
Simple Mobile Robots

- Mobile robots can vacuum floors, transport goods in warehouses, act as security robots (patrol), etc



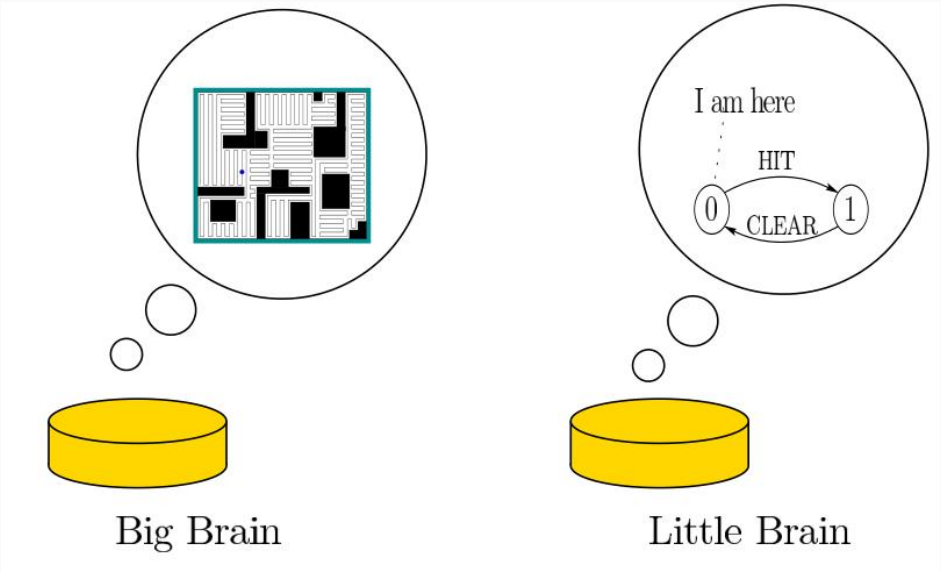
Simple Mobile Robots

- Mobile robots can vacuum floors, transport goods in warehouses, act as security robots (patrol), etc
- We want to **minimize** sensing, computation, actuation



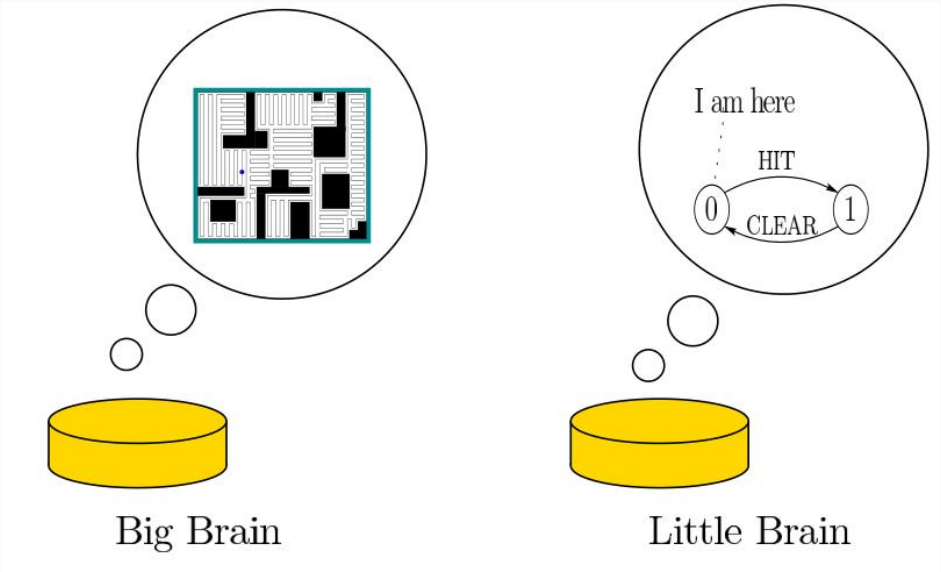
Simple Mobile Robots

- Mobile robots can vacuum floors, transport goods in warehouses, act as security robots (patrol), etc
- We want to **minimize** sensing, computation, actuation
    - make robots less expensive, more energy efficient



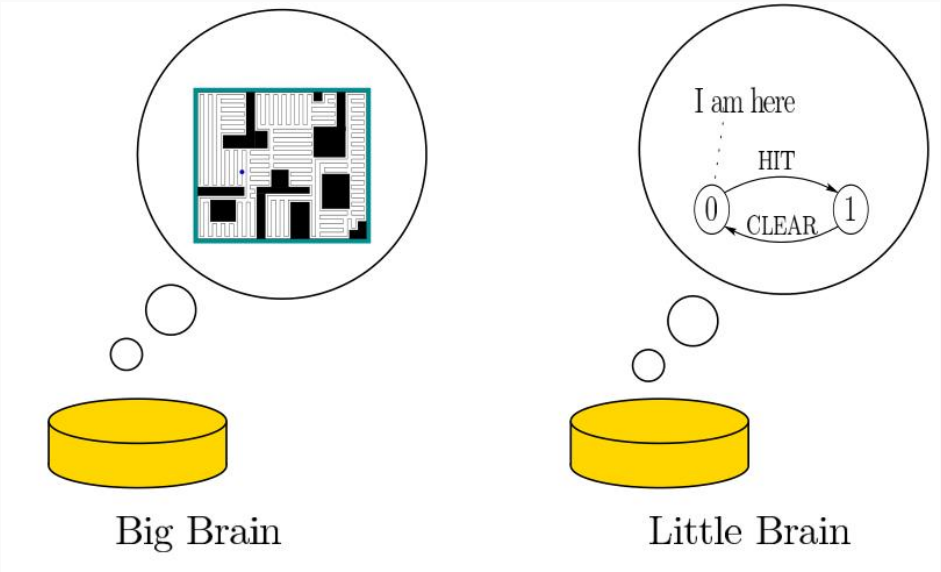
Simple Mobile Robots

- Mobile robots can vacuum floors, transport goods in warehouses, act as security robots (patrol), etc
- We want to **minimize** sensing, computation, actuation
    - make robots less expensive, more energy efficient
- Often, robots can bump into things and be ok!



Simple Mobile Robots

- Mobile robots can vacuum floors, transport goods in warehouses, act as security robots (patrol), etc
- We want to **minimize** sensing, computation, actuation
  - make robots less expensive, more energy efficient
- Often, robots can bump into things and be ok!
- How can we use **contact with the environment** as a strategy or source of information?



Blind, Bouncing Robots

Restrict the robot motion to:

- moving forward in straight lines until collision

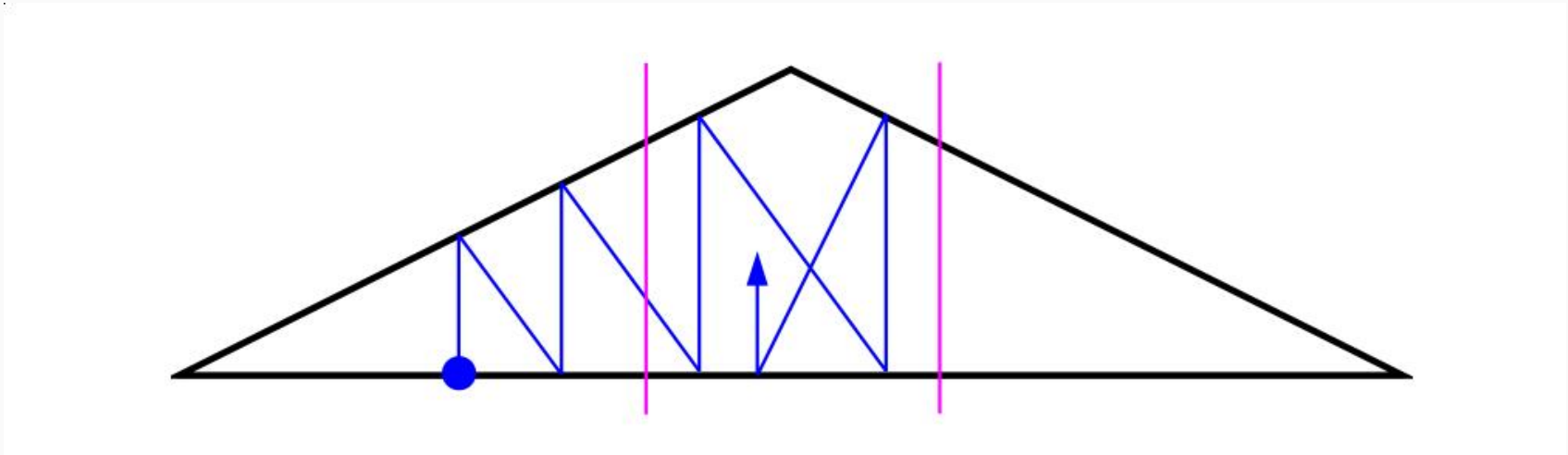


Figure 1. In this environment, bouncing at the normal, the robot will become trapped in the area between the purple lines.[2]

[2] [1], ICRA 13

Blind, Bouncing Robots

Restrict the robot motion to:

- moving forward in straight lines until collision
- when in contact with boundary, rotate in place to some angle θ, then move forward again

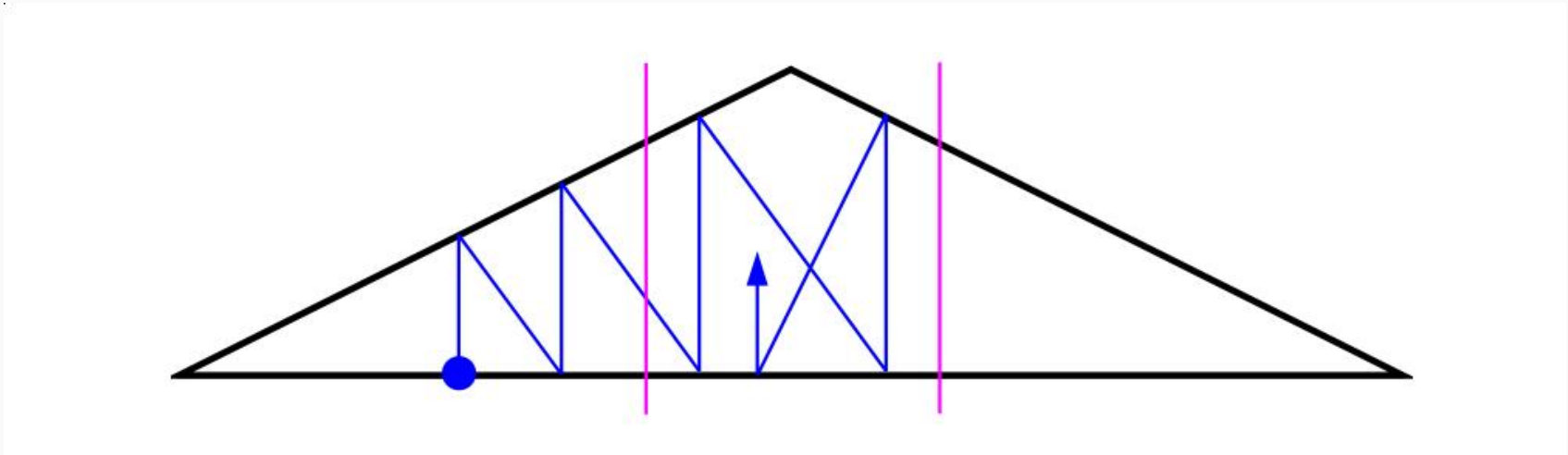


Figure 1. In this environment, bouncing at the normal, the robot will become trapped in the area between the purple lines.[2]

[2] [1], ICRA 13

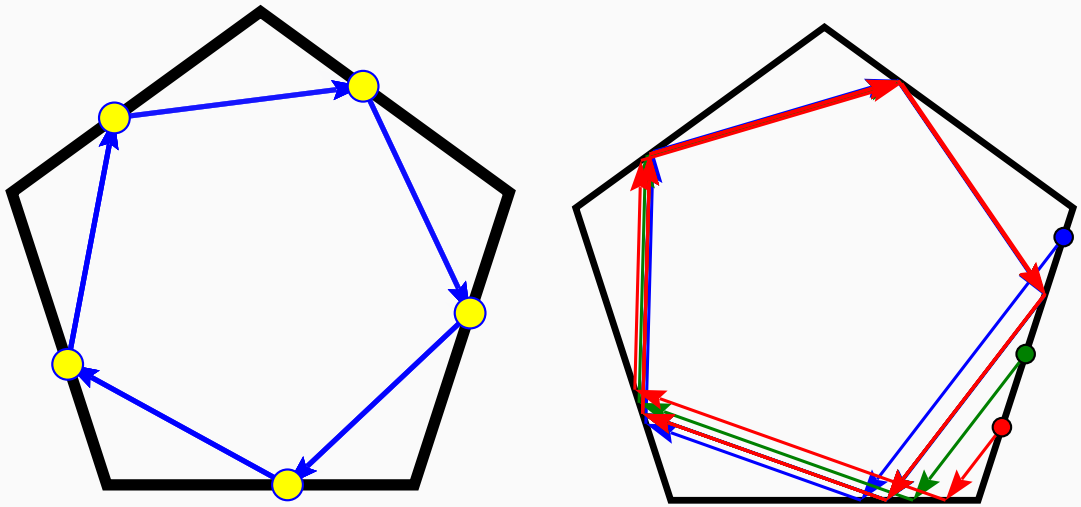
Research Questions

Given a constant control strategy, will the robot become “trapped” in a certain motion pattern (attractor)?

Research Questions

Given a constant control strategy, will the robot become “trapped” in a certain motion pattern (attractor)?

We show that such a robot can perform the task of **patrolling**: periodically following the same path.



Results

- limit cycles in regular polygons [2]

Results

- limit cycles in regular polygons [2]
- limit cycles in convex polygons (upcoming, with Israel Becerra, postdoc)

Results

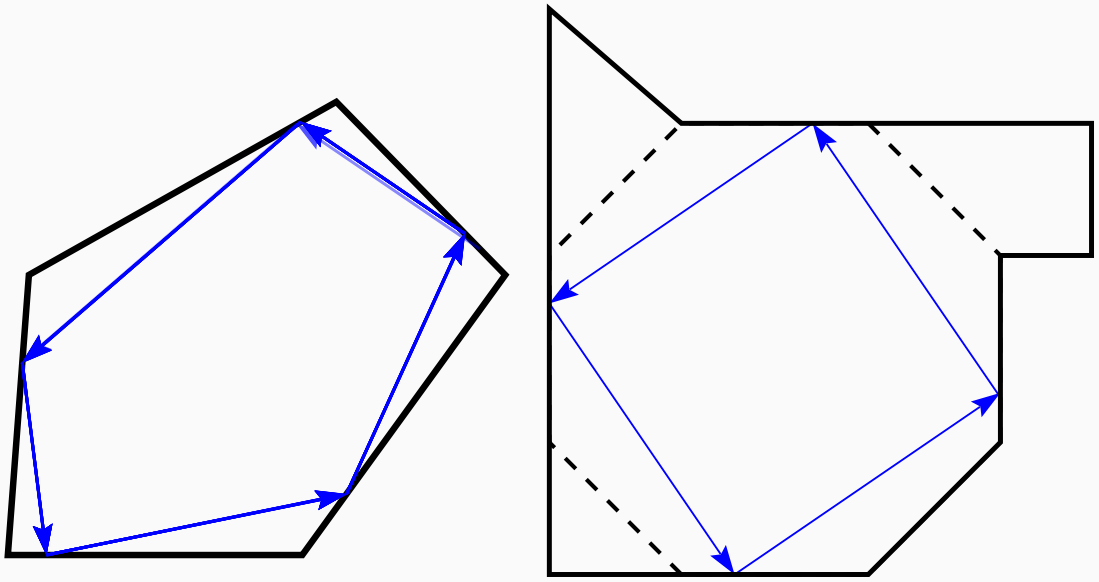
- limit cycles in regular polygons [2]
- limit cycles in convex polygons (upcoming, with Israel Becerra, postdoc)
- next steps: incorporate feedback control, and explore design space (other sensors, actuation strategies, etc), multiple robots, etc

Results

- limit cycles in regular polygons [2]
- limit cycles in convex polygons (upcoming, with Israel Becerra, postdoc)
- next steps: incorporate feedback control, and explore design space (other sensors, actuation strategies, etc), multiple robots, etc

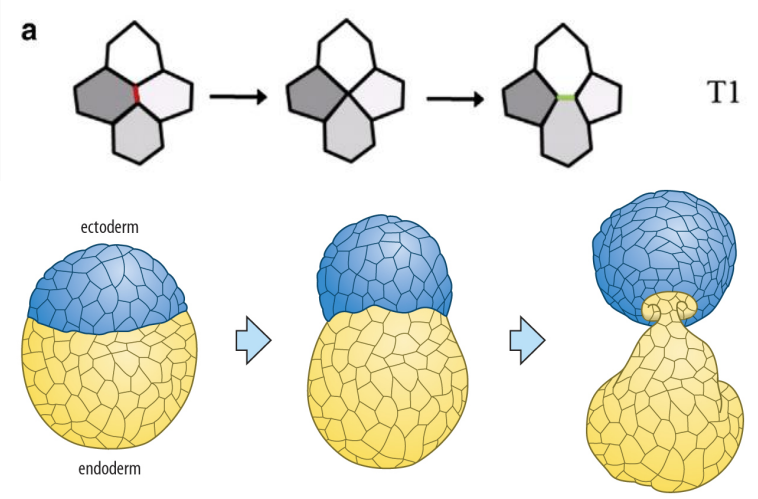
Results

- limit cycles in regular polygons [2]
- limit cycles in convex polygons (upcoming, with Israel Becerra, postdoc)
- next steps: incorporate feedback control, and explore design space (other sensors, actuation strategies, etc), multiple robots, etc



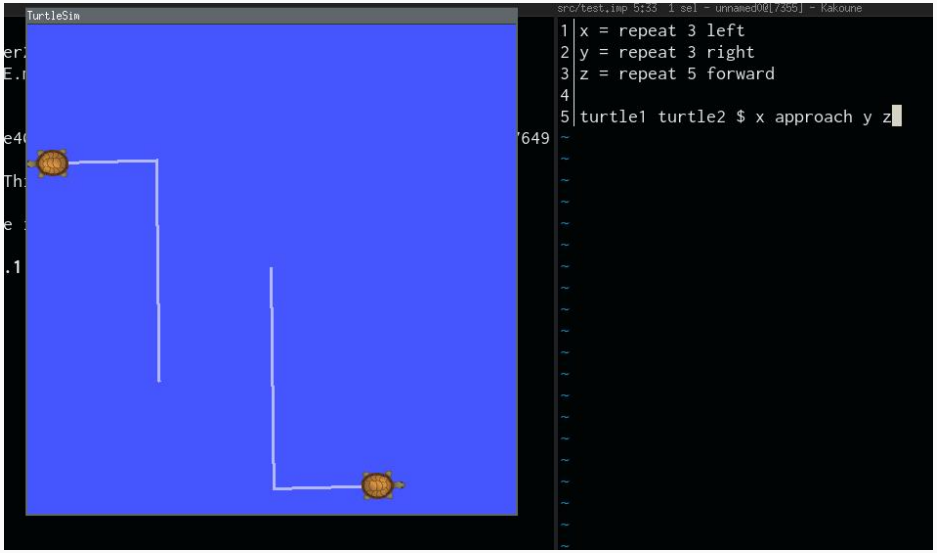
Other Projects

With Dr. Yuliy Baryshnikov: morphogenesis from local cell reconfigurations. Figures from [3] [4]

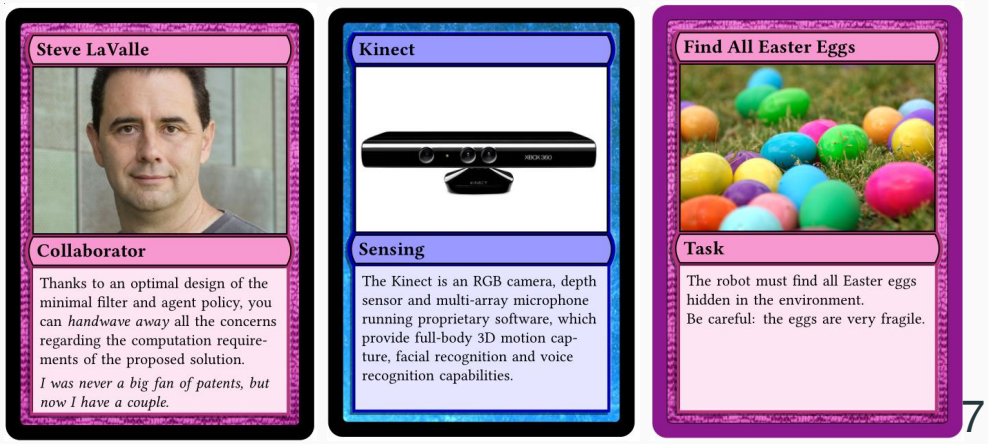


Weaselball assemblies (undergradates run this project!)

Robot “live coding” language for ROS - with Chase Gladish, Drs. Amy LaViers, Mattox Beckman

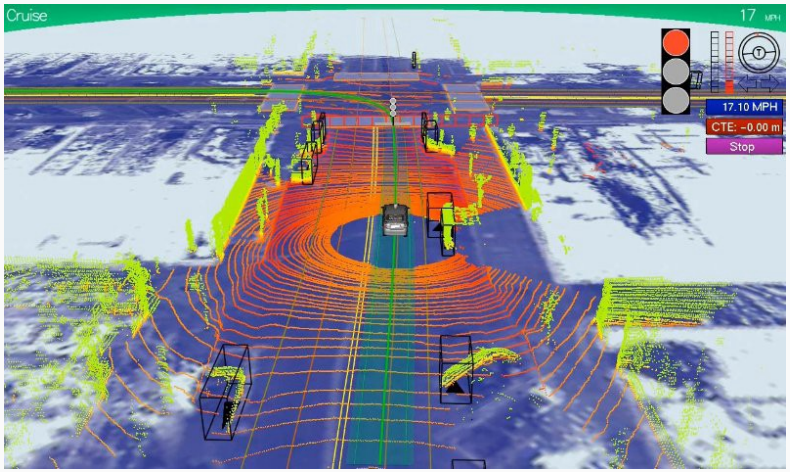


Robot Design Game (RSS 17 Workshop)



Understanding Black Box Predictions via Influence Functions

Motivation

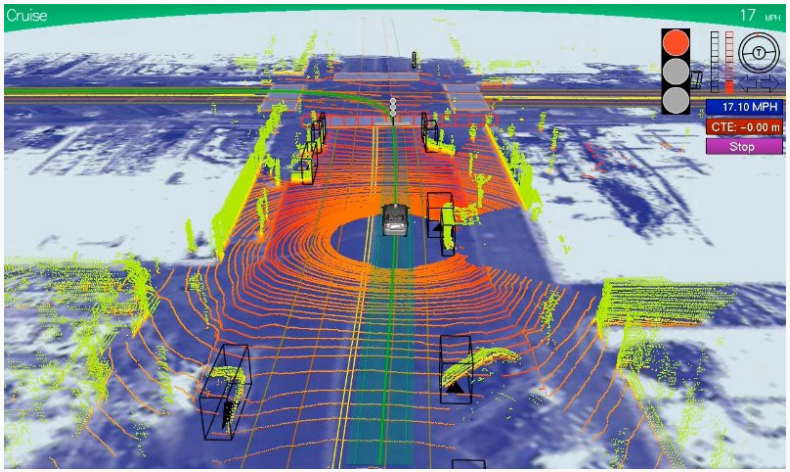


3,4

- How can we interpret trained models, and perform sanity checks?

[3] From Voyage Auto, “An Introduction to Lidar”
[4] From Google Research

Motivation



3,4

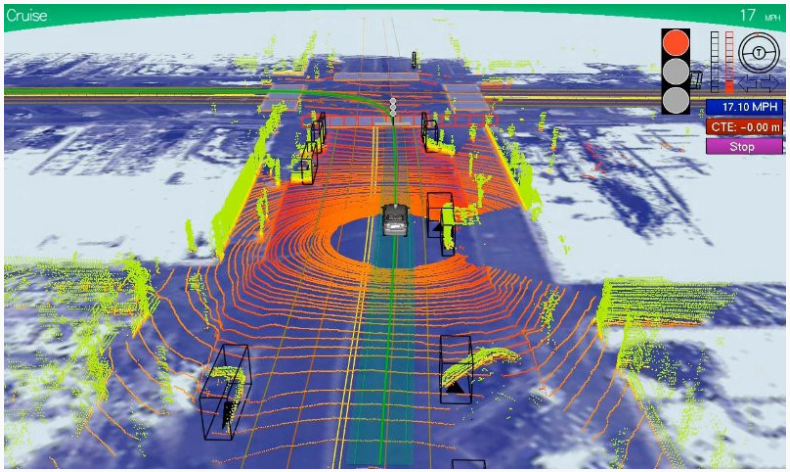
- How can we interpret trained models, and perform sanity checks?
- How can we avoid possible training-set and test-set attacks?

---

[3] From Voyage Auto, “An Introduction to Lidar”

[4] From Google Research

Motivation



3,4

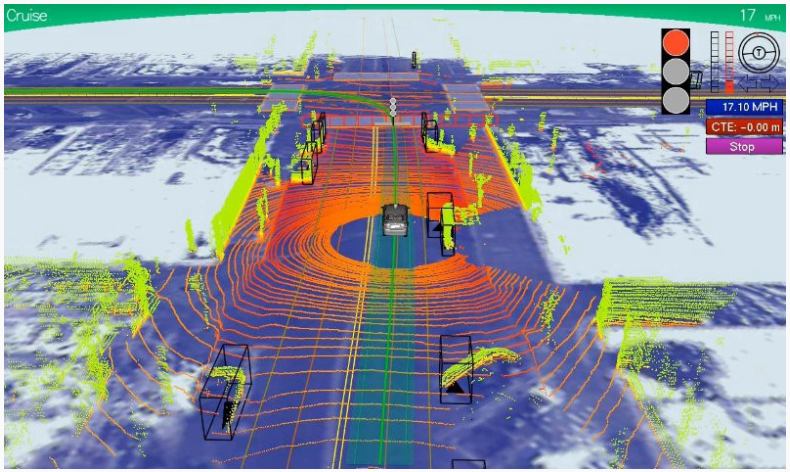
- How can we interpret trained models, and perform sanity checks?
- How can we avoid possible training-set and test-set attacks?
- How robust are our models to noise?

---

[3] From Voyage Auto, “An Introduction to Lidar”

[4] From Google Research

Motivation



3,4

- How can we interpret trained models, and perform sanity checks?
- How can we avoid possible training-set and test-set attacks?
- How robust are our models to noise?
- Strong need for quantitative analysis tools

[3] From Voyage Auto, “An Introduction to Lidar”
[4] From Google Research

Paper Contributions

- A scalable implementation of influence functions, parameterized over loss function

Paper Contributions

- A scalable implementation of influence functions, parameterized over loss function
- Evidence of usefulness for model understanding, generating adversarial training examples, debugging domain mismatch, and fixing mislabeled examples

Paper Contributions

- A scalable implementation of influence functions, parameterized over loss function
- Evidence of usefulness for model understanding, generating adversarial training examples, debugging domain mismatch, and fixing mislabeled examples

Paper Contributions

- A scalable implementation of influence functions, parameterized over loss function
- Evidence of usefulness for model understanding, generating adversarial training examples, debugging domain mismatch, and fixing mislabeled examples

Much more work remains to be done! This is an analysis tool - what to do with results of analysis?

Context and Related Work

- statistics: Cook, Weisberg 1980: *Residuals and influence in regression*

Context and Related Work

- statistics: Cook, Weisberg 1980: *Residuals and influence in regression*
    - focused on linear models, exact solutions

Context and Related Work

- statistics: Cook, Weisberg 1980: *Residuals and influence in regression*
    - focused on linear models, exact solutions
- *Robustness of Convex Risk Minimization Models* [5]

Context and Related Work

- statistics: Cook, Weisberg 1980: *Residuals and influence in regression*
  - focused on linear models, exact solutions
- *Robustness of Convex Risk Minimization Models* [5]
  - $n = 500$, SVM with different kernels, focus on effect of adding a data point

Context and Related Work

- statistics: Cook, Weisberg 1980: *Residuals and influence in regression*
  - focused on linear models, exact solutions
- *Robustness of Convex Risk Minimization Models* [5]
  - $n = 500$, SVM with different kernels, focus on effect of adding a data point
- *Model selection in kernel based regression using the influence function* [6]

Context and Related Work

- statistics: Cook, Weisberg 1980: *Residuals and influence in regression*
  - focused on linear models, exact solutions
- *Robustness of Convex Risk Minimization Models* [5]
  - $n = 500$, SVM with different kernels, focus on effect of adding a data point
- *Model selection in kernel based regression using the influence function* [6]
- *“Influence Sketching”: Finding Influential Samples In Large-Scale Regressions* [7]

Context and Related Work

- statistics: Cook, Weisberg 1980: *Residuals and influence in regression*
  - focused on linear models, exact solutions
- *Robustness of Convex Risk Minimization Models* [5]
  - $n = 500$, SVM with different kernels, focus on effect of adding a data point
- *Model selection in kernel based regression using the influence function* [6]
- *"Influence Sketching": Finding Influential Samples In Large-Scale Regressions* [7]
  - randomized algorithm for approximating influence, specific to generalized linear models. $n = 2$ million

Context and Related Work

- statistics: Cook, Weisberg 1980: *Residuals and influence in regression*
  - focused on linear models, exact solutions
- *Robustness of Convex Risk Minimization Models* [5]
  - $n = 500$, SVM with different kernels, focus on effect of adding a data point
- *Model selection in kernel based regression using the influence function* [6]
- *"Influence Sketching": Finding Influential Samples In Large-Scale Regressions* [7]
  - randomized algorithm for approximating influence, specific to generalized linear models. $n = 2$ million
- adversarial examples and training-set attacks

Background

- Pang Wei Koh, and his advisor Percy Liang
- Stanford and Microsoft Research
- ICML 2017 Best Paper Award

“otherwise high-performing models are still difficult to debug and fail catastrophically in the presence of changing data distributions and adversaries. . . it is critical to build tools to help us make machine learning more reliable ‘in the wild.’ ” – Percy Liang

Problem Formulation

What does it mean for a training point to be *influential*?

Problem Formulation

What does it mean for a training point to be *influential*?

For a given learned model (with known loss function):

- How would the model's predictions change if we **omit** a specific training point?

Problem Formulation

What does it mean for a training point to be *influential*?

For a given learned model (with known loss function):

- How would the model's predictions change if we **omit** a specific training point?
- How would the model's predictions change if we **perturb** a specific training point?

Problem Formulation

What does it mean for a training point to be *influential*?

For a given learned model (with known loss function):

- How would the model's predictions change if we **omit** a specific training point?
- How would the model's predictions change if we **perturb** a specific training point?

Problem Formulation

What does it mean for a training point to be *influential*?

For a given learned model (with known loss function):

- How would the model's predictions change if we **omit** a specific training point?
- How would the model's predictions change if we **perturb** a specific training point?

To approach these questions, study the *derivative* of the *optimal parameters*, or of the *loss*, with respect to different perturbations of a single training point.

Problem Formulation

What does it mean for a training point to be *influential*?

For a given learned model (with known loss function):

- How would the model's predictions change if we **omit** a specific training point?
- How would the model's predictions change if we **perturb** a specific training point?

To approach these questions, study the *derivative* of the *optimal parameters*, or of the *loss*, with respect to different perturbations of a single training point.

When this value is larger, that training point is more *influential*.

Definitions

predictor: $\mathcal{X} \to \mathcal{Y}$

Definitions

predictor: $\mathcal{X} \to \mathcal{Y}$

given training points $z_1, \ldots, z_n$, where $z_i \in \mathcal{X} \times \mathcal{Y}$

Definitions

predictor: $\mathcal{X} \to \mathcal{Y}$

given training points $z_1, \ldots, z_n$, where $z_i \in \mathcal{X} \times \mathcal{Y}$

trained parameters $\theta \in \Theta$

Definitions

predictor: $\mathcal{X} \to \mathcal{Y}$

given training points $z_1, \ldots, z_n$, where $z_i \in \mathcal{X} \times \mathcal{Y}$

trained parameters $\theta \in \Theta$

loss $L(z, \theta)$ and empirical risk $R(\theta) = \frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta)$

- approach is agnostic to loss (but assumes convex, twice-differentiable wrt θ)
- we will often use $H_{\hat{\theta}} \stackrel{\text{def}}{=} \frac{1}{n} \sum_{i=1}^{n} \nabla_\theta^2 L(z_i, \hat{\theta})$

Definitions

predictor: $\mathcal{X} \to \mathcal{Y}$

given training points $z_1, \ldots, z_n$, where $z_i \in \mathcal{X} \times \mathcal{Y}$

trained parameters $\theta \in \Theta$

loss $L(z, \theta)$ and empirical risk $R(\theta) = \frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta)$

- approach is agnostic to loss (but assumes convex, twice-differentiable wrt θ)
- we will often use $H_{\hat{\theta}} \stackrel{\text{def}}{=} \frac{1}{n} \sum_{i=1}^{n} \nabla_\theta^2 L(z_i, \hat{\theta})$

empirical risk minimizer $\hat{\theta} = \arg\min_{\theta \in \Theta} R(\theta)$

Removing and Perturbing Training Points

Similar methods can derive the following:

$$\begin{aligned}\mathcal{I}_L(z, z_{test}) &\overset{\text{def}}{=} \left.\frac{dL(z_{test}, \hat{\theta}_{\epsilon,z})}{d\epsilon}\right|_{\epsilon=0} \\ &= -\nabla_\theta L(z_{\text{test}}, \hat{\theta})^\top H_{\hat{\theta}}^{-1} \nabla_\theta L(z, \hat{\theta})\end{aligned}$$

which measures influence on the loss, not just the parameters.

Removing and Perturbing Training Points

Similar methods can derive the following:

$$\mathcal{I}_L(z, z_{test}) \overset{\text{def}}{=} \left. \frac{dL(z_{test}, \hat{\theta}_{\epsilon,z})}{d\epsilon} \right|_{\epsilon=0}$$
$$= -\nabla_\theta L(z_{\text{test}}, \hat{\theta})^\top H_{\hat{\theta}}^{-1} \nabla_\theta L(z, \hat{\theta})$$

which measures influence on the loss, not just the parameters.

We can also measure the influence of perturbing the **value** of a training input, $z_\delta = (x + \delta, y)$:

Removing and Perturbing Training Points

Similar methods can derive the following:

$$\begin{aligned}\mathcal{I}_L(z, z_{test}) &\overset{\text{def}}{=} \left.\frac{dL(z_{test}, \hat{\theta}_{\epsilon,z})}{d\epsilon}\right|_{\epsilon=0} \\ &= -\nabla_\theta L(z_{\text{test}}, \hat{\theta})^\top H_{\hat{\theta}}^{-1} \nabla_\theta L(z, \hat{\theta})\end{aligned}$$

which measures influence on the loss, not just the parameters.

We can also measure the influence of perturbing the **value** of a training input, $z_\delta = (x + \delta, y)$:

$$\begin{aligned}\left.\frac{d\hat{\theta}_{\epsilon,z_\delta,-z}}{d\epsilon}\right|_{\epsilon=0} &= \mathcal{I}_{\hat{\theta}}(z_\delta) - \mathcal{I}_{\hat{\theta}}(z) \\ &= -H_{\hat{\theta}}^{-1}\left(\nabla_\theta L(z_\delta, \hat{\theta}) - \nabla_\theta L(z, \hat{\theta})\right). \qquad (1)\end{aligned}$$

Analysis - Remove Terms from Influence

Let $p(y \mid x) = \sigma(y\theta^\top x)$, with $y \in \{-1, 1\}$ and $\sigma(t) = \frac{1}{1+\exp(-t)}$.

Analysis - Remove Terms from Influence

Let $p(y \mid x) = \sigma(y\theta^\top x)$, with $y \in \{-1, 1\}$ and $\sigma(t) = \frac{1}{1+\exp(-t)}$.

For a training point $z = (x, y)$,

$$L(z, \theta) = \log(1 + \exp(-y\theta^\top x))$$

$$\nabla_\theta L(z, \theta) = -\sigma(-y\theta^\top x)yx$$

$$H_\theta = \frac{1}{n}\sum_{i=1}^{n} \sigma(\theta^\top x_i)\sigma(-\theta^\top x_i)x_i x_i^\top$$

Analysis - Remove Terms from Influence

Let $p(y \mid x) = \sigma(y\theta^\top x)$, with $y \in \{-1, 1\}$ and $\sigma(t) = \frac{1}{1+\exp(-t)}$.

For a training point $z = (x, y)$,

$$
\begin{aligned}
L(z, \theta) &= \log(1 + \exp(-y\theta^\top x)) \\
\nabla_\theta L(z, \theta) &= -\sigma(-y\theta^\top x) yx \\
H_\theta &= \frac{1}{n} \sum_{i=1}^{n} \sigma(\theta^\top x_i)\sigma(-\theta^\top x_i) x_i x_i^\top
\end{aligned}
$$

and $\mathcal{I}_L(z, z_{\text{test}})$ is

$$
-y_{\text{test}} y \cdot \sigma(-y_{\text{test}}\theta^\top x_{\text{test}}) \cdot \sigma(-y\theta^\top x) \cdot x_{\text{test}}^\top H_{\hat{\theta}}^{-1} x.
$$

Analysis - Remove Terms from Influence

Let $p(y \mid x) = \sigma(y\theta^\top x)$, with $y \in \{-1, 1\}$ and $\sigma(t) = \frac{1}{1+\exp(-t)}$.

For a training point $z = (x, y)$,

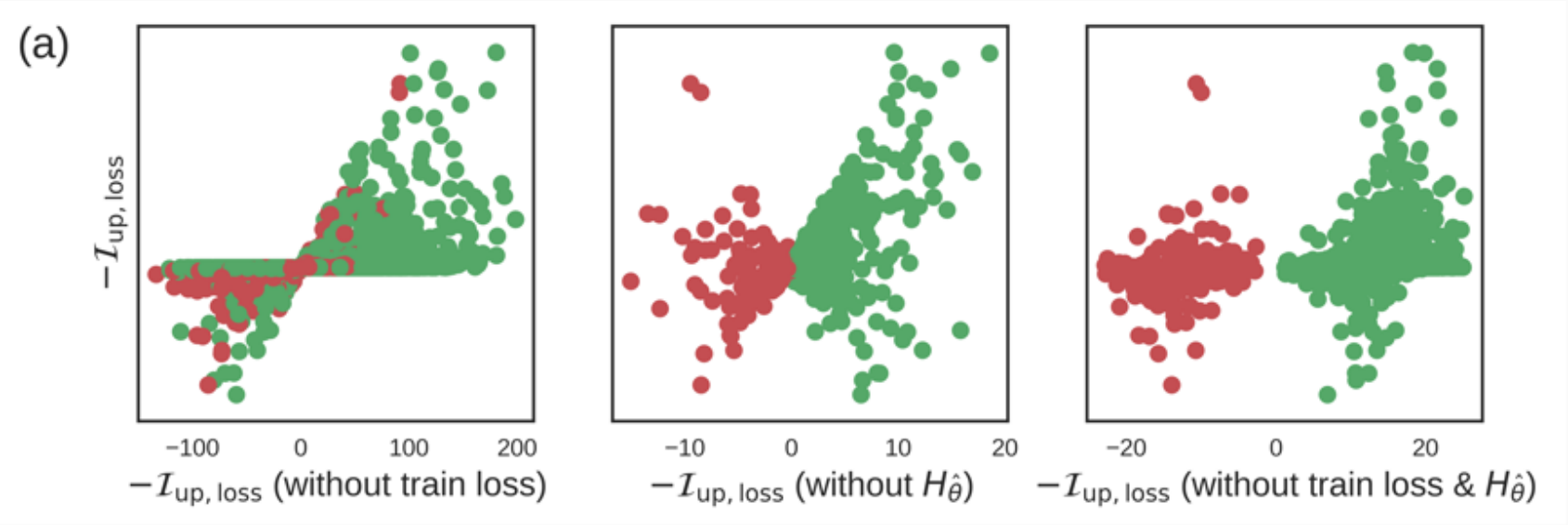
$$L(z, \theta) = \log(1 + \exp(-y\theta^\top x))$$
$$\nabla_\theta L(z, \theta) = -\sigma(-y\theta^\top x)yx$$
$$H_\theta = \frac{1}{n}\sum_{i=1}^{n} \sigma(\theta^\top x_i)\sigma(-\theta^\top x_i)x_i x_i^\top$$

and $\mathcal{I}_L(z, z_{\text{test}})$ is

$$-y_{\text{test}} y \cdot \sigma(-y_{\text{test}}\theta^\top x_{\text{test}}) \cdot \sigma(-y\theta^\top x) \cdot \mathbf{x}_{\mathbf{test}}^\top H_{\hat{\theta}}^{-1}\mathbf{x}.$$

Analysis - Remove Terms from Influence

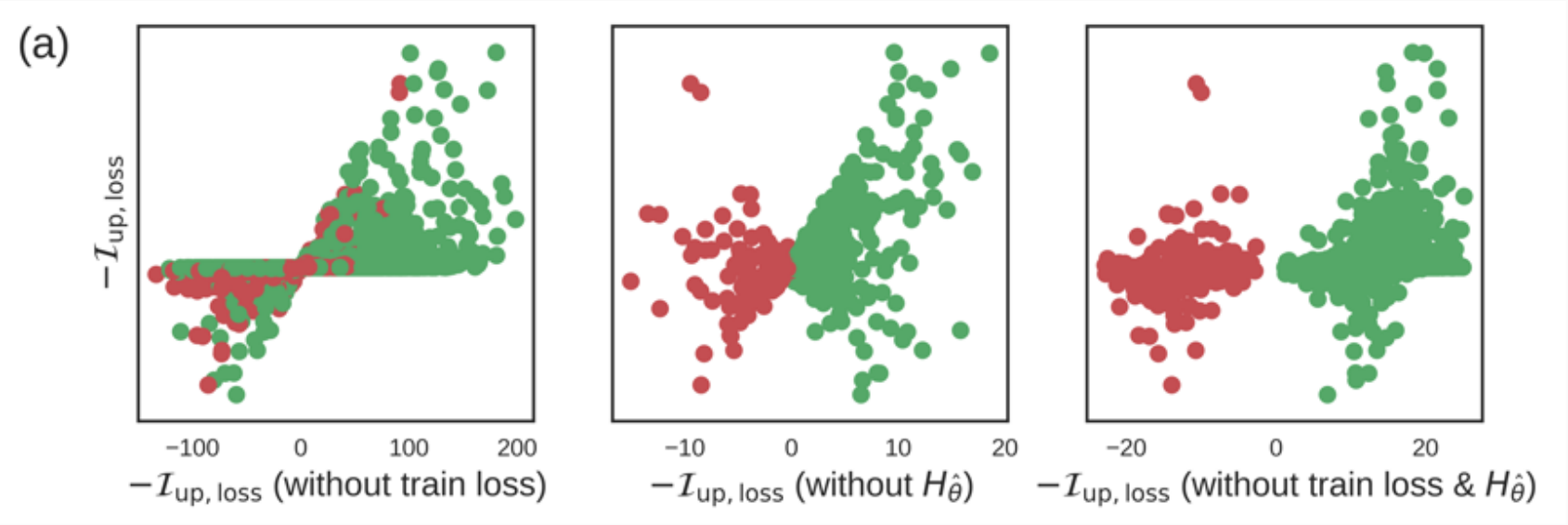
$$-y_{\text{test}} y \cdot \sigma(-y_{\text{test}} \theta^\top x_{\text{test}}) \cdot \sigma(-y\theta^\top x) \cdot x_{\text{test}}^\top H_{\hat{\theta}}^{-1} x.$$



left: $\sigma(-y\theta^\top x)$ gives points with low training loss less influence: without it, we overestimate the influence of training points

Analysis - Remove Terms from Influence

$$-y_{\text{test}} y \cdot \sigma(-y_{\text{test}} \theta^\top x_{\text{test}}) \cdot \sigma(-y \theta^\top x) \cdot x_{\text{test}}^\top H_{\hat{\theta}}^{-1} x.$$



middle/right: the weighted covariance matrix $H_{\hat{\theta}}^{-1}$ measures the "resistance" of the other training points to the removal of z. Without it, all same-label points are helpful, all opposite-label points are harmful.

Efficiency

Two challenges:

1. Forming and inverting $H_{\hat{\theta}} = \frac{1}{n} \sum_{i=1}^{n} \nabla_{\theta}^2 L(z_i, \hat{\theta})$
   - n training points, $\theta \in \mathbb{R}^p$ requires $\mathcal{O}(np^2 + p^3)$ ops
2. Often want to calculate influence across all training points for a specific test point

How to Make Faster?

Overall approach:

- Efficiently approximate $s_{test} \overset{\text{def}}{=} H_{\hat{\theta}}^{-1} \nabla_{\theta} L(z_{test}, \hat{\theta})$

How to Make Faster?

Overall approach:

- Efficiently approximate $s_{test} \overset{\text{def}}{=} H_{\hat{\theta}}^{-1} \nabla_\theta L(z_{test}, \hat{\theta})$
- Use this to efficiently compute $\mathcal{I}_L(z, z_{test})$ by just multiplying s_{test} by $\nabla_\theta L(z, \theta)$ as needed!

How to Make Faster?

Overall approach:

- Efficiently approximate $s_{test} \overset{\text{def}}{=} H_{\hat{\theta}}^{-1} \nabla_\theta L(z_{test}, \hat{\theta})$
- Use this to efficiently compute $\mathcal{I}_L(z, z_{test})$ by just multiplying s_{test} by $\nabla_\theta L(z, \theta)$ as needed!

How to Make Faster?

Overall approach:

- Efficiently approximate $s_{test} \stackrel{\text{def}}{=} H_{\hat{\theta}}^{-1} \nabla_\theta L(z_{test}, \hat{\theta})$
- Use this to efficiently compute $\mathcal{I}_L(z, z_{test})$ by just multiplying s_{test} by $\nabla_\theta L(z, \theta)$ as needed!

Conjugate Gradients

Stochastic Estimation

Both automatically handled in systems like TensorFlow, Theano - users just specify L.

Speeds up calculating influence for all training points on a given test point to $\mathcal{O}(np)$.

How to Know if it Works?

Validation: Influence matches leave-one-out retraining

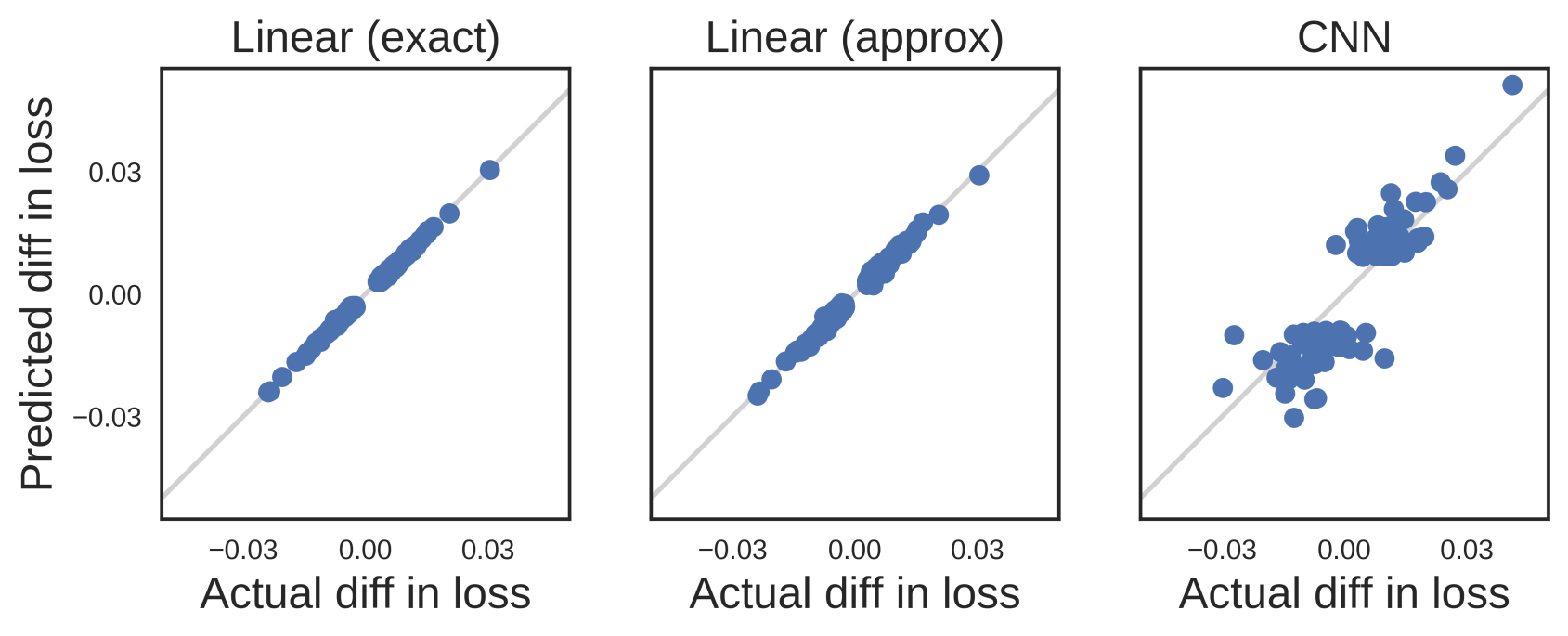
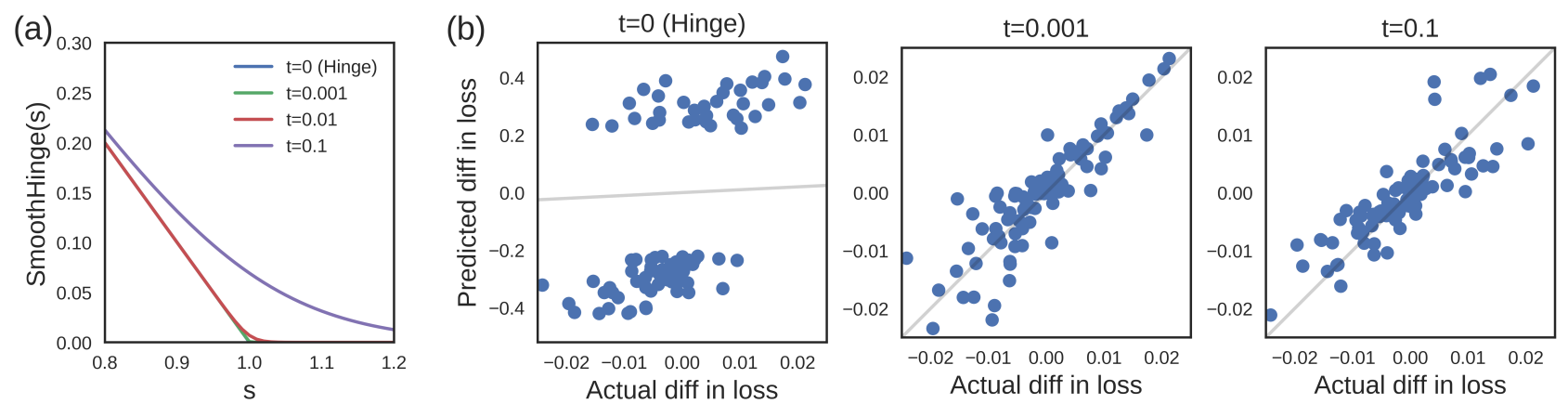


Figure 2. Left: For each of the 500 training points with largest influence, we plotted $-\frac{1}{n} \cdot \mathcal{I}_L(z, z_{\text{test}})$ against the actual change in test loss after removing that point and retraining. The inverse HVP was solved exactly with CG. **Mid:** Same, but with the stochastic approximation. **Right:** The same plot for a CNN, computed on the 100 most influential points with CG. For the actual difference in loss, we removed each point and retrained from $\tilde{\theta}$ for 30k steps

Non-differentiable losses



- SVM with hinge loss
  - approximate with $smoothHinge(s, t) = t \log(1 + \exp(\frac{1-s}{t}))$
- set derivative at hinge to 0, lose second derivative information
- t=0.001, Pearson's R=0.95
- t=0.1, Pearson's R=0.91

What to Use it For?

Understanding Model Behavior

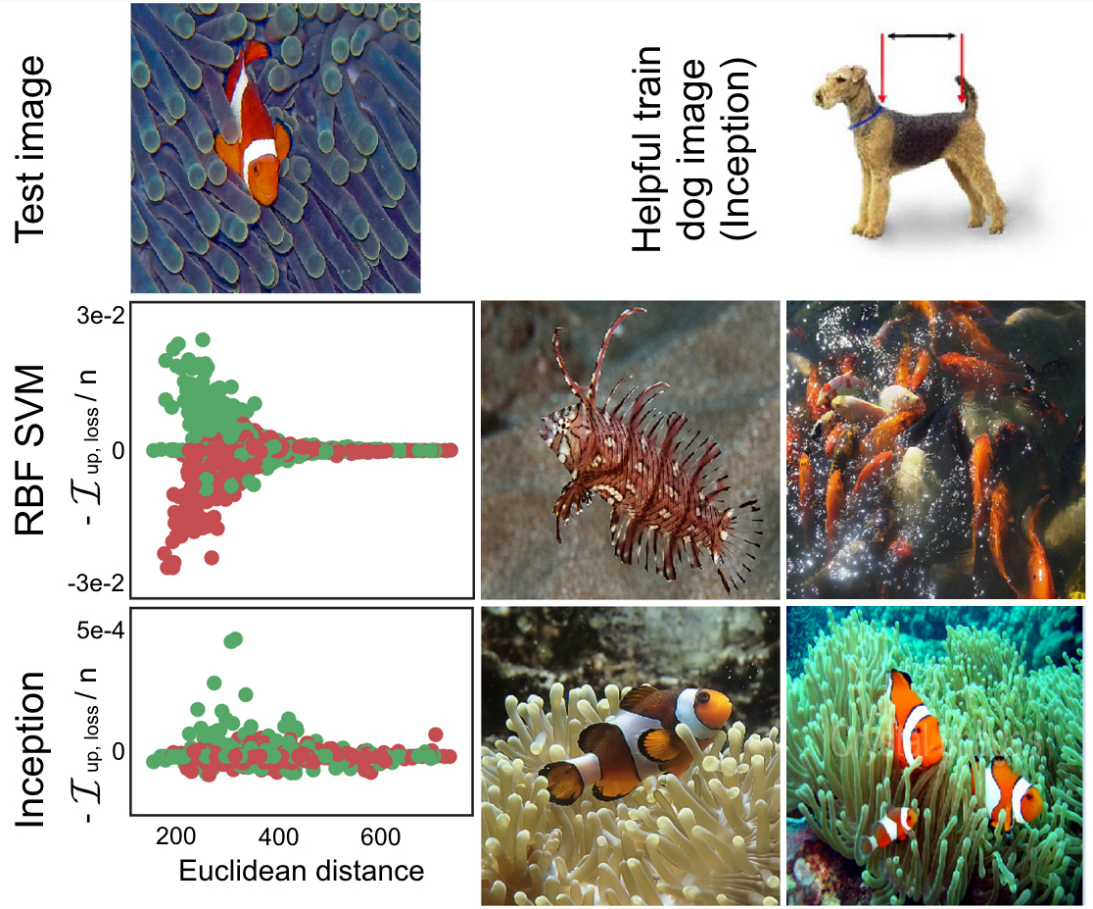


Figure 3. Bottom left: $-\mathcal{I}_L(z, z_{\text{test}})$ vs. $\|z - z_{\text{test}}\|_2^2$. Green dots are fish and red dots are dogs. **Bottom right:** The two most helpful training images, for each model, on the test. **Top right:** An image of a dog in the training set that helped the Inception model correctly classify the test image as a fish.

Adversarial Training Examples

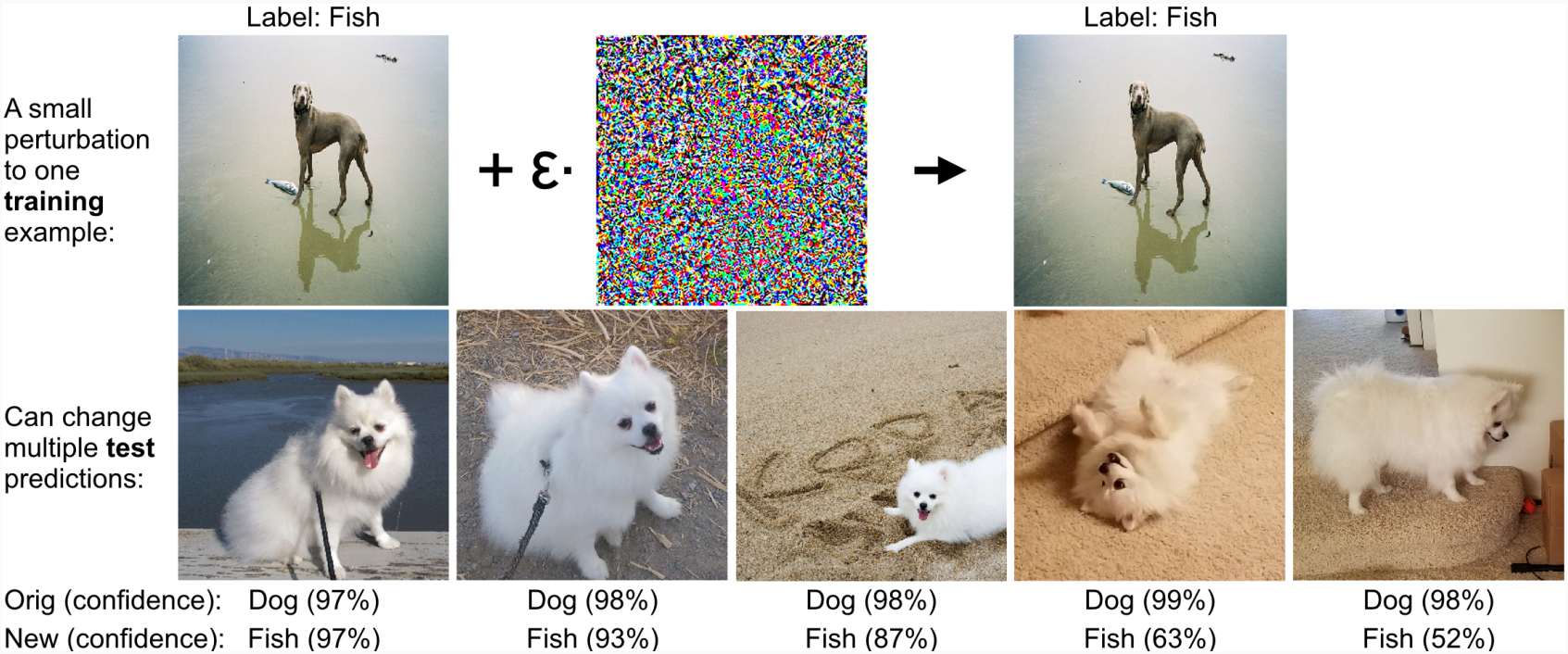


Figure 4. We targeted a set of 30 test images featuring the **first author's dog** in a variety of poses and backgrounds. By maximizing the average loss over these 30 images, we created a visually-imperceptible change to the particular training image (shown on top) that flipped predictions on 16 test images.

Conclusion

- Why Best Paper?
  - Connects statistical technique with large-scale applications
  - Relatively usable “out of the box”: code and datasets available, parameterized over loss
  - Addresses important question for safety-critical systems
- What could be better / remaining questions?
  - Analysis of CNN example still leaves questions: How to linearize loss? How to apply to other architectures?
  - Should analyze nonconvexity and nonconvergence separately, not together.
  - How to make datasets uniformly influential?

Now For Something Completely Different!

Robots!


x k c d
when we open the lab each morning, we tell the robot to kill
it's our little joke
but secretly
we're just afraid
to tell it to love
r munroe

Challenges in Robotics

- We have low-level planning mostly figured out (thanks Steve!)
    - Even in real time: see, dynamic replanning, *Robot Motion Planning on a Chip*
- More than ever, robots “just work” (but don’t ask me to demo)
- Now, we can focus on the hugely important problem of human-robot interaction

Robots and Humans, Working Together!

Thank you!

Appendix

Sketch of Derivation

We want to find change in model parameters if training point z is removed, but we don't want to retrain

Instead, weight z by ϵ:

$$\hat{\theta}_{\epsilon,z} = \arg\min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta) + \epsilon L(z, \theta)$$

Sketch of Derivation

We want to find change in model parameters if training point z is removed, but we don't want to retrain

Instead, weight z by ϵ:

$$\hat{\theta}_{\epsilon,z} = \arg\min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta) + \epsilon L(z, \theta)$$

With $\Delta_\epsilon = \hat{\theta}_{\epsilon,z} - \hat{\theta}$, we can calculate influence as:

$$\mathcal{I}_{\hat{\theta}}(z) \stackrel{\text{def}}{=} \frac{d\hat{\theta}_{\epsilon,z}}{d\epsilon} = \frac{d\Delta_{\epsilon,z}}{d\epsilon}$$

Sketch of Derivation

$\hat{\theta}_{\epsilon,z}$ minimizes $R(\theta) + \epsilon L(z, \theta)$:

$$0 = \nabla R(\hat{\theta}_{\epsilon,z}) + \epsilon \nabla L(z, \hat{\theta}_{\epsilon,z})$$

Sketch of Derivation

$\hat{\theta}_{\epsilon,z}$ minimizes $R(\theta) + \epsilon L(z, \theta)$:

$$0 = \nabla R(\hat{\theta}_{\epsilon,z}) + \epsilon \nabla L(z, \hat{\theta}_{\epsilon,z})$$

Taylor expand the right hand side around $\hat{\theta}$

$$0 \approx \left[\nabla R(\hat{\theta}) + \epsilon \nabla L(z, \hat{\theta})\right] + \left[\nabla^2 R(\hat{\theta}) + \epsilon \nabla^2 L(z, \hat{\theta})\right] \Delta_\epsilon$$

Sketch of Derivation

$\hat{\theta}_{\epsilon,z}$ minimizes $R(\theta) + \epsilon L(z, \theta)$:

$$0 = \nabla R(\hat{\theta}_{\epsilon,z}) + \epsilon \nabla L(z, \hat{\theta}_{\epsilon,z})$$

Taylor expand the right hand side around $\hat{\theta}$

$$0 \approx \left[\nabla R(\hat{\theta}) + \epsilon \nabla L(z, \hat{\theta})\right] + \left[\nabla^2 R(\hat{\theta}) + \epsilon \nabla^2 L(z, \hat{\theta})\right] \Delta_\epsilon$$

and solve for Δ_ϵ

$$\Delta_\epsilon \approx -\left[\nabla^2 R(\hat{\theta}) + \epsilon \nabla^2 L(z, \hat{\theta})\right]^{-1} \left[\nabla R(\hat{\theta}) + \epsilon \nabla L(z, \hat{\theta})\right].$$

Sketch of Derivation

But $\nabla R(\hat{\theta}) = 0$. Keeping only $O(\epsilon)$ terms, we have

$$\Delta_\epsilon \approx -\nabla^2 R(\hat{\theta})^{-1} \nabla L(z, \hat{\theta})\epsilon.$$

Sketch of Derivation

But $\nabla R(\hat{\theta}) = 0$. Keeping only $O(\epsilon)$ terms, we have

$$\Delta_\epsilon \approx -\nabla^2 R(\hat{\theta})^{-1} \nabla L(z, \hat{\theta})\epsilon.$$

We conclude that:

$$\left.\frac{d\hat{\theta}_{\epsilon,z}}{d\epsilon}\right|_{\epsilon=0} = -H_{\hat{\theta}}^{-1} \nabla L(z, \hat{\theta}) \stackrel{\text{def}}{=} \mathcal{I}_{\hat{\theta}}(z).$$

References i

[1] L. H. Erickson and S. M. LaValle, “Toward the design and analysis of blind, bouncing robots,” in *IEEE international conference on robotics and automation*, 2013.

[2] A. Nilles, I. Becerra, and S. M. LaValle, “Periodic trajectories of mobile robots,” *IROS*, 2017.

[3] A. G. Fletcher, M. Osterfield, R. E. Baker, and S. Y. Shvartsman, “Vertex models of epithelial morphogenesis,” *Biophysical journal*, vol. 106, no. 11, pp. 2291–2304, 2014.

[4] B. E. Staveley, “Molecular and developmental biology (biol3530),” *Department of Biology, Memorial University of Newfoundland*.

[5] A. Christmann and I. Steinwart, “On robustness properties of convex risk minimization methods for pattern recognition,” *Journal of Machine Learning Research*, vol. 5, no. Aug, pp. 1007–1034, 2004.

[6] M. Debruyne, M. Hubert, and J. A. Suykens, “Model selection in kernel based regression using the influence function,” *Journal of Machine Learning Research*, vol. 9, no. Oct, pp. 2377–2400, 2008.

[7] M. Wojnowicz, B. Cruz, X. Zhao, B. Wallace, M. Wolff, J. Luan, and C. Crable, “‘Influence sketching’: Finding influential samples in large-scale regressions,” in *Big data (big data), 2016 ieee international conference on*, 2016, pp. 3601–3612.