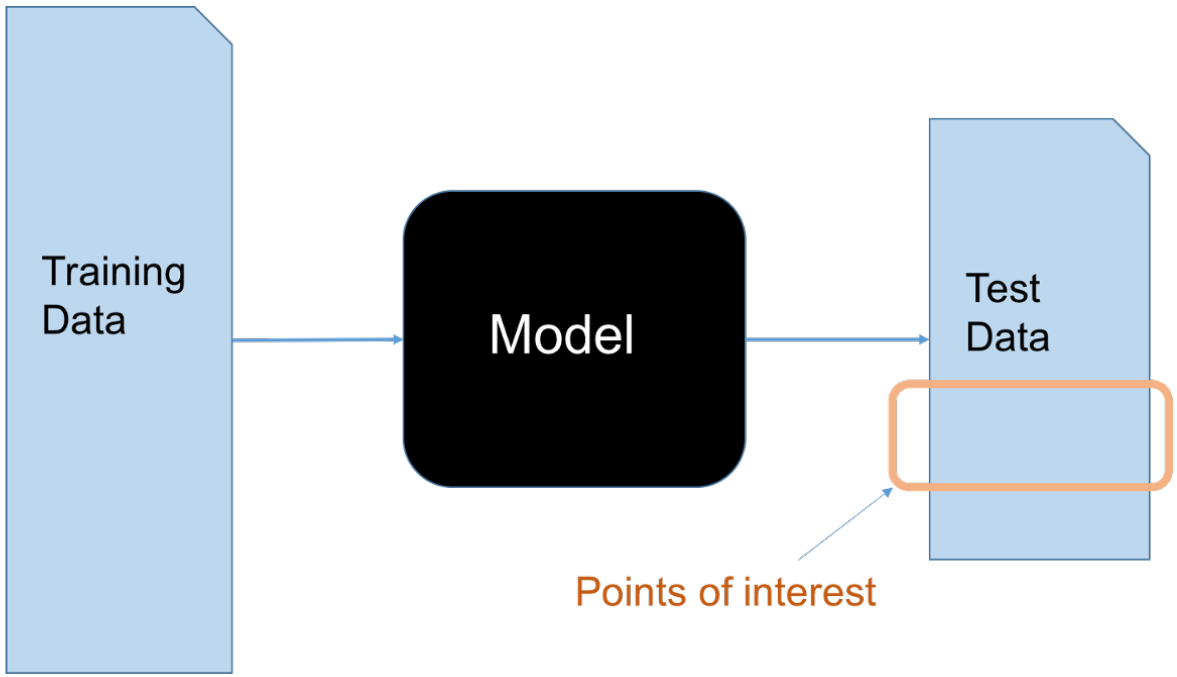
CS37300
PURDUE UNIVERSITY

NOV 29 2023

DATA MINING



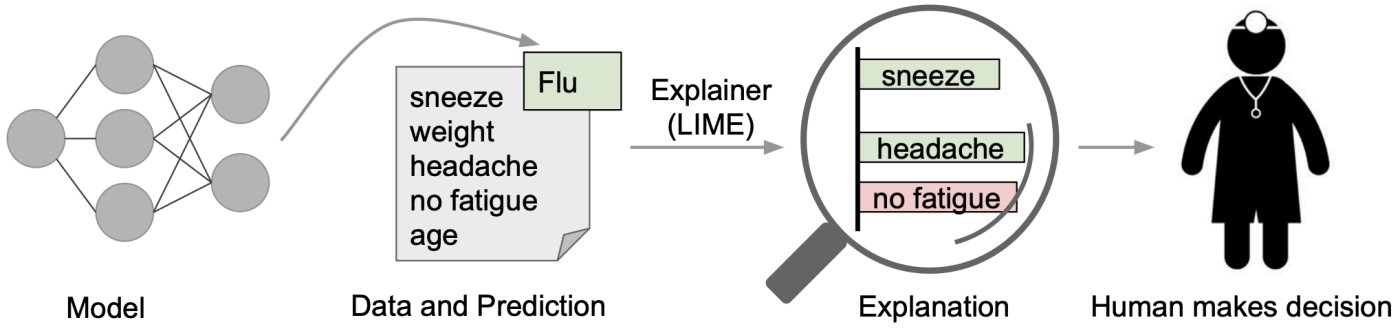
Which training data points are “most” responsible for predicting on points of interest ?

PERFORMANCE ATTRIBUTION

- Feature attribution
  - Linear regression – sparsity
  - Decision trees
- Instance attribution

FEATURE ATTRIBUTION

- From “Why should I trust you?”



DESIGN CRITERIA FOR FEATURE ATTRIBUTION

- Complicated models are hard to understand for laymen
- Choose few features (sparse)
- How do those features impact the prediction?
  - Simpler models are easy to interpret
  - Use simple approximations of the model
- Model agnostic
- High “fidelity”

LIME

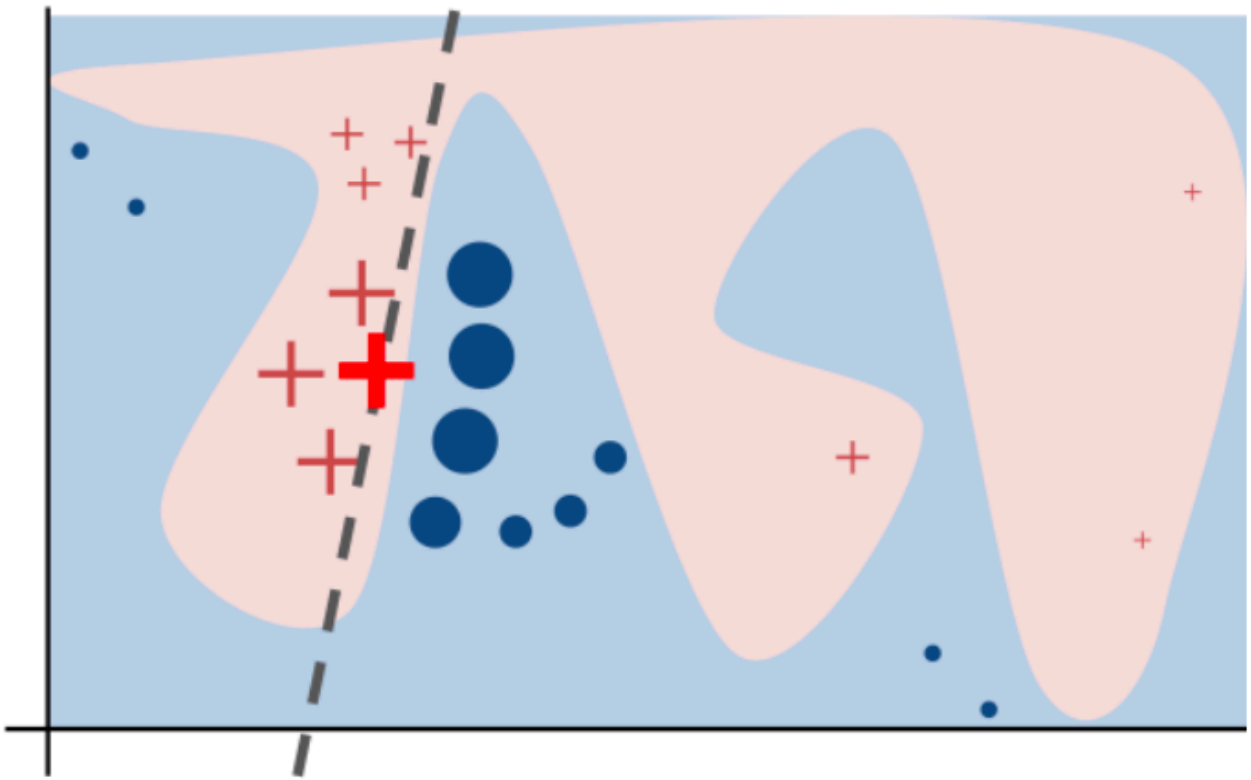
- LIME = Locally Interpretable Model-agnostic Explanations
- Let g() be an easily interpretable simple model
- Let f() be the actual model to interpret
- The idea is to build a g() that closely approximates f() “locally”
-

COST FUNCTION

- Here $\mathscr{L}()$ is the loss that approximates f() with g() in the local region π_x around the instance in question x
- Even though f() may be very complex *globally,* it may still be possible to approximate it *locally.*
- Ω is the complexity measure

$$\xi(x) = \underset{g \in G}{\operatorname{argmin}} \ \mathcal{L}(f, g, \pi_x) + \Omega(g)$$

EXAMPLE



EXAMPLE

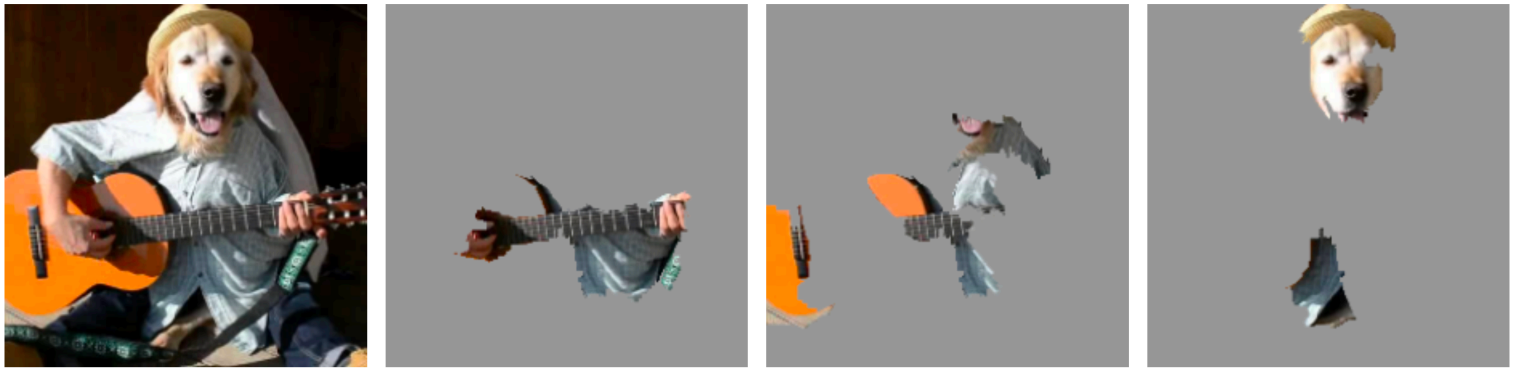
- Let g() be linear functions
- Let $\pi_x(z) := \exp(-D(x,z)/\sigma^2)$ be an exponential kernel for some distance function D
- Loss: $\mathscr{L} = \sum_{z,x \in Z} \pi_x(z)(f(x) - g(x))^2$

EXAMPLE (CONTD)

- Ω is measure of complexity *acceptable to the end user*
  - For text: this is could be “use up to k words”
  - For images it could be “use up to k super pixels”
  -

IMAGE EXPLANATION

- Segment the image into super pixels which are x
- Build simple linear model around vicinity of each super pixel to obtain high local fidelity



(a) Original Image (b) Explaining *Electric guitar* (c) Explaining *Acoustic guitar* (d) Explaining *Labrador*

Figure 4: Explaining an image classification prediction made by Google's Inception neural network. The top 3 classes predicted are "Electric Guitar" ($p = 0.32$), "Acoustic guitar" ($p = 0.24$) and "Labrador" ($p = 0.21$)

EXAMPLE: EXPLAINING A BAD CLASSIFICATION

(a) Husky classified as wolf (b) Explanation

Figure 11: Raw data and explanation of a bad model's prediction in the "Husky vs Wolf" task.

SHAPLEY VALUES

- From “A Unified Approach to Interpreting Model Predictions”
- Builds from LIME
- Additive feature attribution for the simple model class g() with $z' \in \{0,1\}, \phi_i \in R$

$$g(z') = \phi_0 + \sum_{i=1}^{M} \phi_i z'_i$$

- Proposes mathematical properties that a LIME-like system should satisfy

INSTANCE ATTRIBUTION

- Similar idea: Remove a point, re-train model to calculate effect
- "Counter-factual" to calculate influence of a point – how would the model prediction change if the given point did not exist?
- Leave-one-out (LOO) error
  - n-fold Cross-validation for n data points
  - Stability of the model
  - Expensive

GOAL OF LOO

- To calculate the influence of training data point, upweigh its weight by a small amount (recall boosting)
- How does the parameter θ change ?
- $\hat{\theta} \overset{\text{def}}{=} \arg\min_{\theta \in \Theta} \frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta)$
- $\hat{\theta}_{-z} \overset{\text{def}}{=} \arg\min_{\theta \in \Theta} \sum_{z_i \neq z} L(z_i, \theta)$
- Can we evaluate $\hat{\theta}_{-z} - \hat{\theta}$?

ALTERNATIVE TO LOO

- From “Understanding Black-box Predictions via Influence Functions”
- Upweigh the point to be removed by an ϵ
- $\hat{\theta}_{\epsilon,z} \overset{\text{def}}{=} \arg\min_{\theta\in\Theta} \frac{1}{n}\sum_{i=1}^{n} L(z_i,\theta) + \epsilon L(z,\theta)$
- Notice $\epsilon = -\frac{1}{n}$ recovers LOO
- Strategy : Assume n is large so for small ϵ we can use derivatives to approximate LOO
- Find $\frac{\partial\hat{\theta}_{\epsilon,z}}{\partial\epsilon}$ at $\epsilon = 0$

NOTATION

- Say:

$$R(\theta) \stackrel{\text{def}}{=} \frac{1}{n} \sum_{i=1}^{n} L(z_i, \theta)$$

$$H_{\hat{\theta}} \stackrel{\text{def}}{=} \nabla^2 R(\hat{\theta}) = \frac{1}{n} \sum_{i=1}^{n} \nabla^2_{\theta} L(z_i, \hat{\theta})$$

- Want:

$$\hat{\theta}_{\epsilon,z} = \arg\min_{\theta \in \Theta} \{R(\theta) + \epsilon L(z, \theta)\}$$

$$\Delta_\epsilon = \hat{\theta}_{\epsilon,z} - \hat{\theta},$$

$$\frac{d\hat{\theta}_{\epsilon,z}}{d\epsilon} = \frac{d\Delta_\epsilon}{d\epsilon}$$

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

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

RESULT

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

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

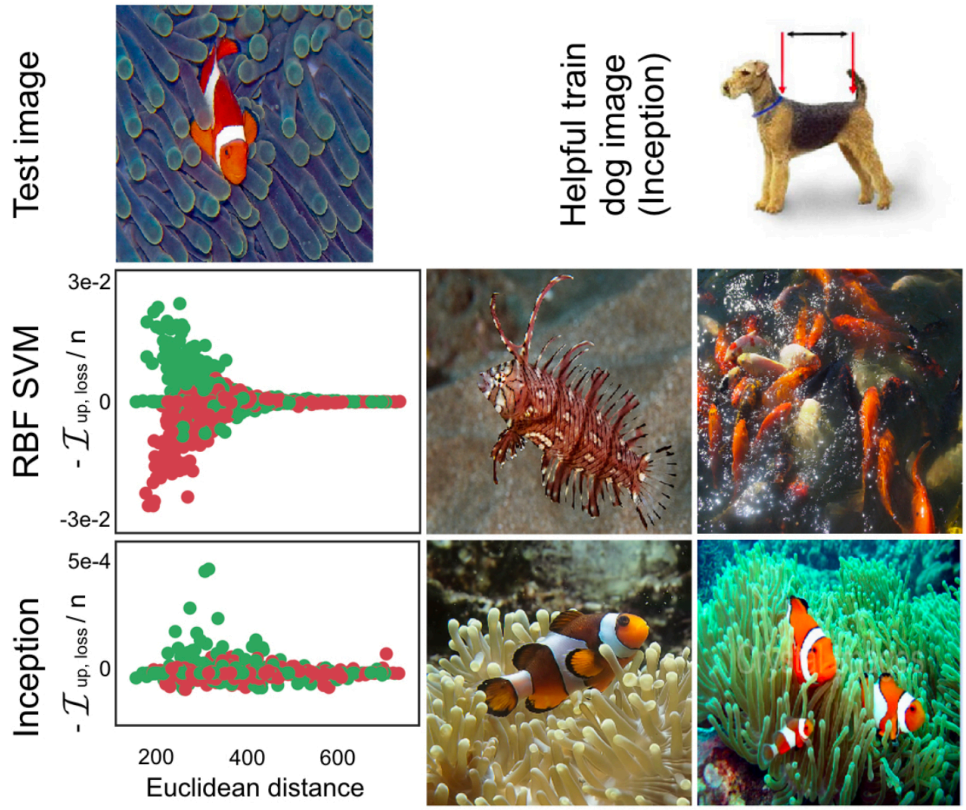
INFLUENCE FUNCTIONS

- $$\mathcal{I}_{\text{up,params}}(z) \stackrel{\text{def}}{=} \left.\frac{d\hat{\theta}_{\epsilon,z}}{d\epsilon}\right|_{\epsilon=0} = -H_{\hat{\theta}}^{-1} \nabla_\theta L(z, \hat{\theta})$$
- Can approximate change in the parameter $$\hat{\theta}_{-z} - \hat{\theta} \approx -\frac{1}{n}\mathcal{I}_{\text{up,params}}(z)$$
- Chain rule:
- $$\begin{aligned}\mathcal{I}_{\text{up,loss}}(z, z_{\text{test}}) &\stackrel{\text{def}}{=} \left.\frac{dL(z_{\text{test}}, \hat{\theta}_{\epsilon,z})}{d\epsilon}\right|_{\epsilon=0} \\ &= \nabla_\theta L(z_{\text{test}}, \hat{\theta})^\top \left.\frac{d\hat{\theta}_{\epsilon,z}}{d\epsilon}\right|_{\epsilon=0} \\ &= -\nabla_\theta L(z_{\text{test}}, \hat{\theta})^\top H_{\hat{\theta}}^{-1} \nabla_\theta L(z, \hat{\theta})\end{aligned}$$

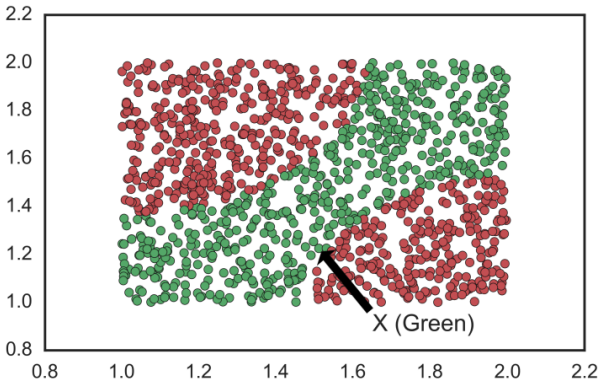
COMPARISON WITH EUCLIDEAN SIMILARITY

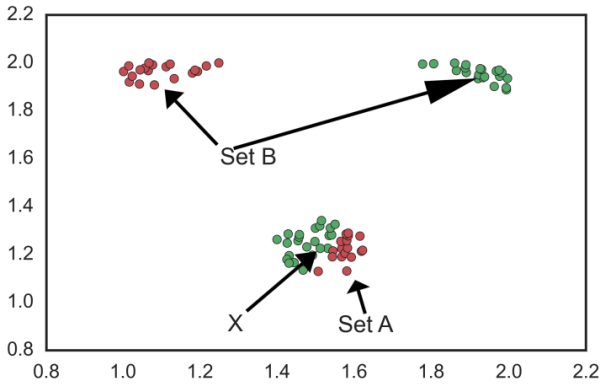
- For logistic =

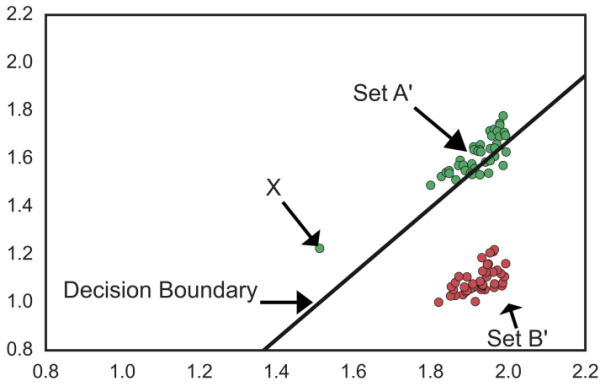
$$-y_{\text{test}} y \cdot \sigma(-y_{\text{test}} \theta^\top x_{\text{test}}) \cdot \sigma(-y\theta^\top x) \cdot x_{\text{test}}^\top H_{\hat{\theta}}^{-1} x$$



- Set A : closest 50 points, Set B: farthest





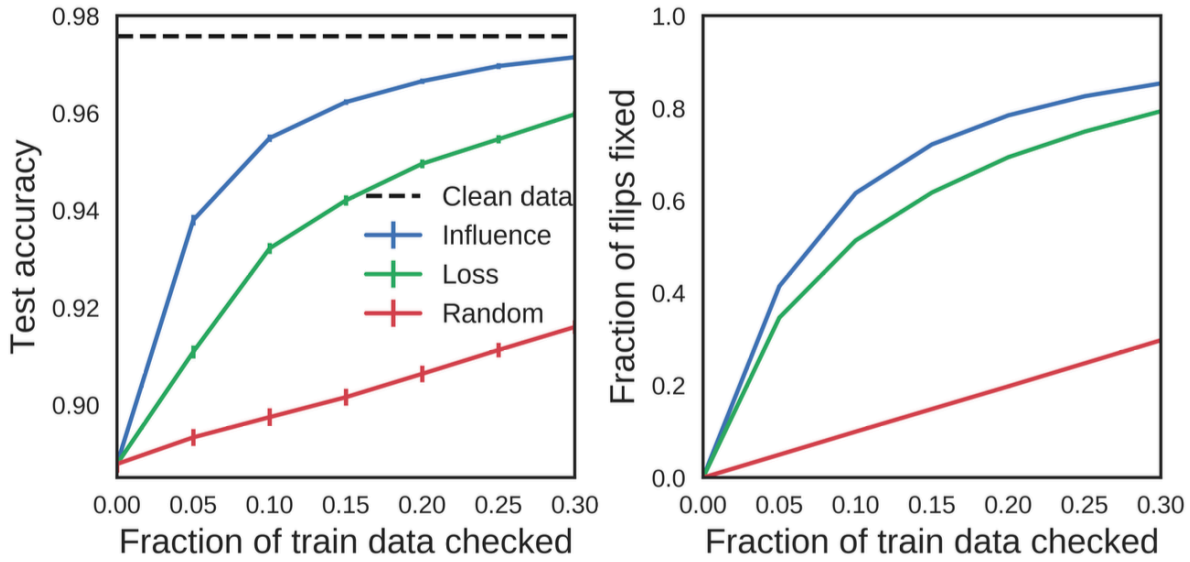


PRACTICAL CONSIDERATIONS

- Need to evaluate influence for all training points for a given test point
  - Expensive!
- Exact Inverse computation is generally not a good idea
- Need $H^{-1}v$
- There are iterative and sampling methods to do this efficiently
- Non-convexity/Non-invertible H, non-differentiable losses

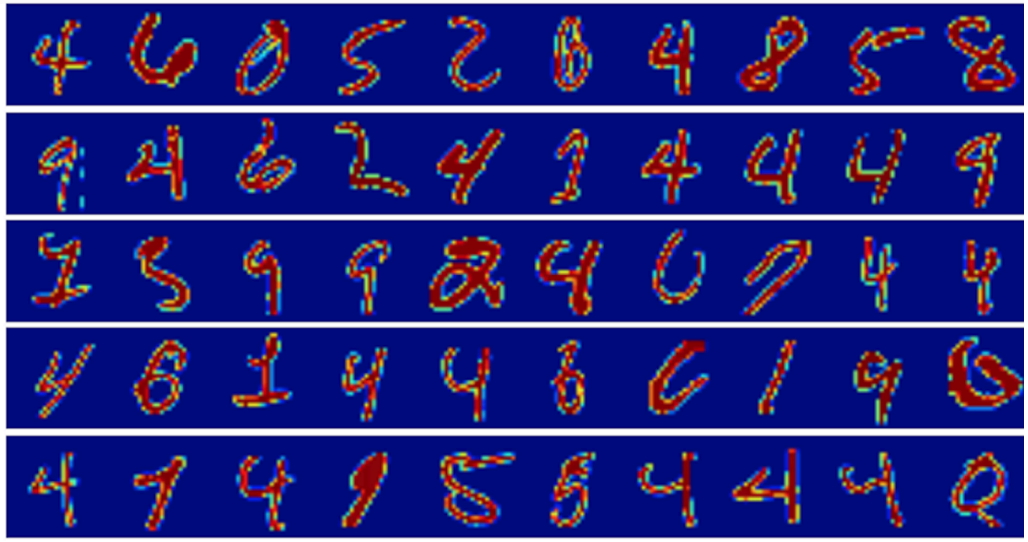
USE-CASE

- Fixing mislabelled examples

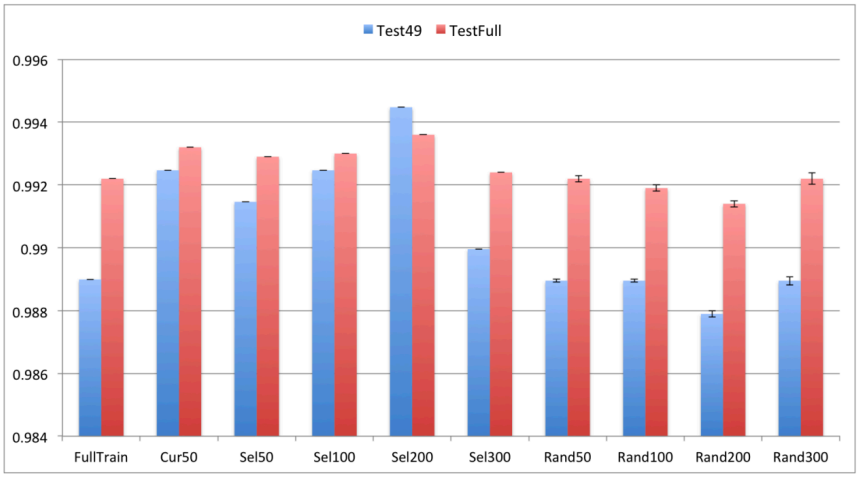


USE-CASE

- Data cleaning



(a) A subset of selected prototypes responsible for misclassifying 4s and 9s in the test set



(b) Accuracy fractions on test data 4s and 9s (Test49), and the full test set after removing random (Rand), algorithm selected (Sel), or Curated (Cur) prototypes.

SHORTCOMINGS

- Unstable – especially for over-parameterized NNs
- Limited to single-point test queries
  - Fixed later through Fisher kernels
- Selecting multiple `top' training points is a combinatorial problem
  - Greedy heuristic
- Several other approaches proposed later