Using Flip Points to Understand and Debug Deep Learning Models

March 2020

CCS

Dianne P. O'Leary

Using Flip Points to Understand and Debug Deep Learning Models

Dianne P. O'Leary

Computer Science Department and
Institute for Advanced Computer Studies
University of Maryland

Joint work with
Roozbeh Yousefzadeh
Postdoctoral Fellow, Yale University

Introduction: The Problem

Given: A mathematical model $\mathcal{F}(\boldsymbol{x}) = \boldsymbol{z}$, where

- $\mathcal{F}$ is a continuous function,
- $\sum_i z_i = 1$,
- $\max_i z_i = z_k$ means that $\boldsymbol{x}$ is said to be in class k.

Examples: This deep learning model might make classifications such as:

- Tumor is **Cancerous** or **Benign**.
- Person is **Low Risk** or **High Risk** if given probation.
- Input is an image of the numeral $0, 1, ...,$ or 9.
- Plasma density at a particular location and time is **Larger** or **Smaller** than some given value.

Introduction: The Problem

Given: A continuous model $\mathcal{F}(\boldsymbol{x}) = \boldsymbol{z}$, $\sum_i z_i = 1$, and $\max_i z_i = z_k$ means that $\boldsymbol{x}$ is said to be in class k.

We want to answer typical numerical analysis questions like:

- What is the least change in the input $\boldsymbol{x}$ that alters the classification?
- How trustworthy is the classification?
- How vulnerable is the model to adversarial inputs?

And, if the model is based on a set of training data, we want to:

- Gain insight about the dataset and classification boundaries.
- Improve the training of the model by identifying most/least influential training points and by generating synthetic data.
- Explain the model's decision.

The tool we use: **flip points.**

Our plan:

- Defining and computing flip points.
- Using flip points to:
  - assess uncertainty,
  - identify most/least influential training points,
  - identify most/least influential features,
  - improve the training of the model,
  - explain the model's decisions.
- The homotopy method for computing flip points
- Conclusions

Defining and Computing Flip Points

What is a flip point?

A flip point is any point on the decision boundary, the boundary between two classification regions.

For 2 classes, this is any point $\hat{\boldsymbol{x}}$ for which $z_1(\hat{\boldsymbol{x}}) = z_2(\hat{\boldsymbol{x}}) = 1/2$.

Of particular interest is the closest flip point $\hat{\boldsymbol{x}}^c$ to a particular input $\boldsymbol{x}$.

This point is the solution to the problem

$$\min_{\hat{\boldsymbol{x}}} \|\hat{\boldsymbol{x}} - \boldsymbol{x}\|,$$

where $\|.\|$ is a norm appropriate to the data.

Our only constraint is

$$z_1(\hat{\boldsymbol{x}}) = 1/2.$$

Notes on flip points

- Specific problems might require additional constraints; for example,
  - If $\boldsymbol{x}$ is an image, upper and lower bounds should be imposed on $\hat{\boldsymbol{x}}$.
  - Discrete inputs will require binary or integer constraints.
- It is possible that the solution $\hat{\boldsymbol{x}}$ is not unique, but the minimal distance is always unique.
- The optimization problem

$$\min_{\hat{\boldsymbol{x}}} \|\hat{\boldsymbol{x}} - \boldsymbol{x}\|,$$
$$z_1(\hat{\boldsymbol{x}}) = 1/2.$$

  is nonconvex.

What if there are more than 2 classes?

Then for a point $\boldsymbol{x}$ in class k, we define flip points $\hat{\boldsymbol{x}}_i^c$, for $i \neq k$ by solving

$$\min_{\hat{\boldsymbol{x}}} \|\hat{\boldsymbol{x}} - \boldsymbol{x}\|,$$

subject to the constraints

$$z_i(\hat{\boldsymbol{x}}) = z_k(\hat{\boldsymbol{x}}),$$

and, for $j \neq i, k$,

$$z_i(\hat{\boldsymbol{x}}) \geq z_j(\hat{\boldsymbol{x}}).$$

Constrained flip points can also be useful

Examples:

- What is the closest flip point with the same size tumor?
- What is the closest flip point for probation risk, given that the person is female?

Constraints reduce the number of variables and make the optimization easier.

Computing flip points

- Suppose $z_1(\boldsymbol{x}) < 1/2$. Then the constraint $z_1(\hat{\boldsymbol{x}}) = 1/2$ is equivalent to $z_1(\hat{\boldsymbol{x}}) \geq 1/2$, and this can be useful.
- There are many applicable algorithms: NLOPT, IPOPT, Matlab's Optimization Toolbox, ...
- For models that are neural networks, we have had great success with a homotopy method:
  - somewhat more reliable,
  - somewhat less expensive.
- Computing a closest flip point is quite fast, under 1 second for the MNIST, CIFAR- 10, and Wisconsin Breast Cancer datasets using a 2017 MacBook. Calculating the closest flip point for the Adult Income dataset takes about 5 seconds, because it has both discrete and continuous variables.

Some previous work:

- Spangher et al. (2018) (independently) defined a flip set as the set of changes in the input that can flip the prediction of a *linear* classifier. They use flip sets to explain the least changes in individual inputs but not to debug the model.
- Wachter et al. (2018) defined counterfactuals as the possible changes in the input that can produce a different output label and used them to explain the decision of a model. For a continuous model, the closest counterfactual is ill-defined, since there are points arbitrarily close to the decision boundaries. They use enumeration for discrete features, applicable only to a small number of features.
- Russell (2019) suggested formulation as an integer programming problem. His examples are linear with small dimensionality.
- Many authors believed that computing flip points was impractical and instead relied on line searches or "small" perturbations to find relevant boundary points.

Neural networks

For the remainder of the talk, I'll use neural networks as an example family of models.

- Normalization of the output is done using softmax:

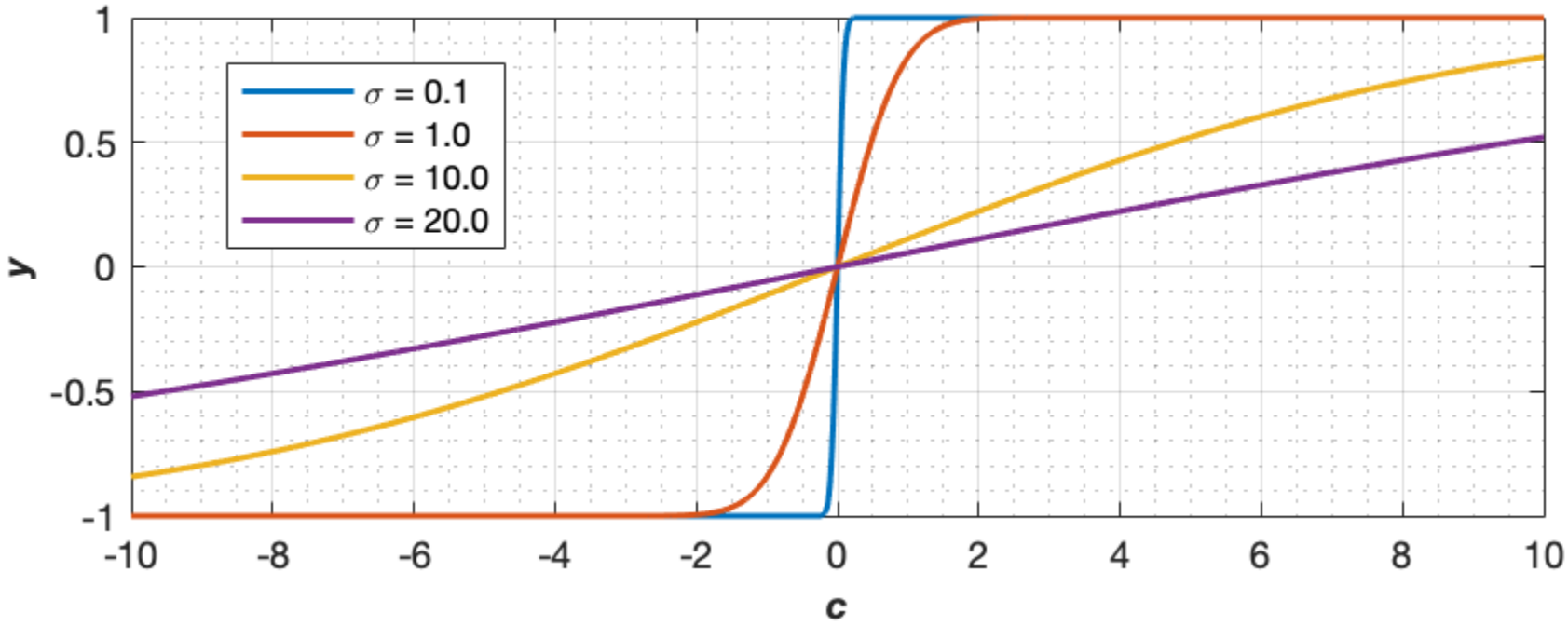
$$\mathsf{softmax}(\boldsymbol{y}) = \frac{e^{\boldsymbol{y}}}{\mathsf{sum}(e^{\boldsymbol{y}})}$$

- We use erf as the activation function:

$$\mathsf{erf}_\sigma(c) = \frac{1}{\sqrt{\pi}} \int_{-\frac{c}{\sigma}}^{+\frac{c}{\sigma}} e^{-t^2} dt,$$

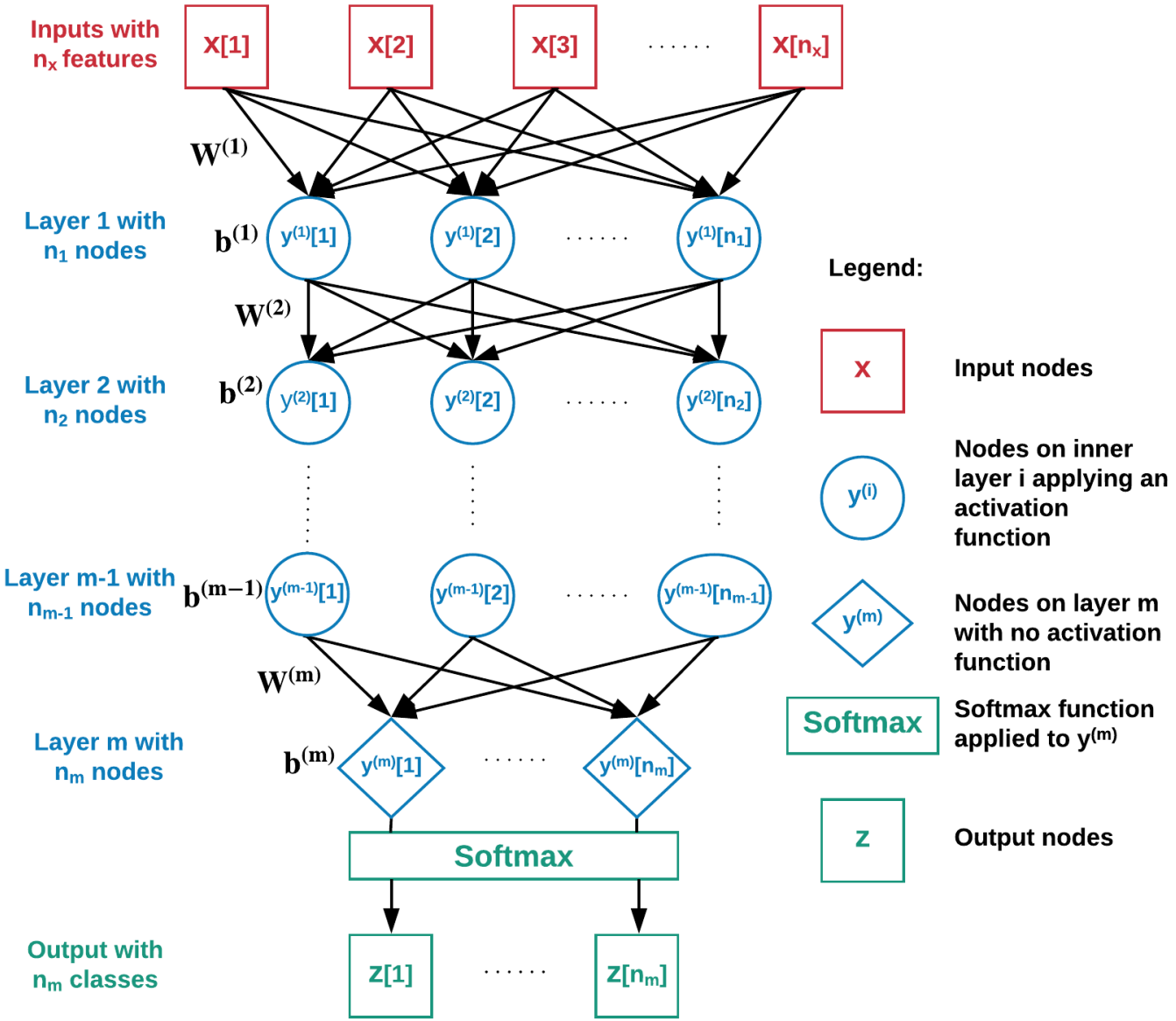
  where σ is a parameter in the neural network.

What does erf look like?



Note that erf can mimic the commonly used activation functions (e.g., ReLU), by adjusting σ, but also allows nonlinear behavior and avoids zero/infinite derivatives.

Our neural network: Pictorially



Our neural network: Mathematically

Defining $\boldsymbol{y}^{(0)} = \boldsymbol{x}$, the output for layers 1 through $m-1$ is

$$\boldsymbol{y}^{(i)} = \mathsf{erf}_{\sigma_i}(\boldsymbol{y}^{(i-1)}\ \boldsymbol{W}^{(i)} + \boldsymbol{b}^{(i)}).$$

Then

$$\boldsymbol{y}^{(m)} = \boldsymbol{y}^{(m-1)}\, \boldsymbol{W}^{(m)} + \boldsymbol{b}^{(m)},$$

and the output of the network is

$$\boldsymbol{z}(\boldsymbol{x}) = \mathsf{softmax}(\boldsymbol{y}^{(m)}).$$

The parameters $\boldsymbol{W}^{(i)}$, $\boldsymbol{b}^{(i)}$, and σ_i are determined by the training process.

Using Flip Points to Assess Uncertainty

Some previous approaches to assessing uncertainty

Previous approaches have some drawbacks.

- Nguyen et al. 2015, Guo et al 2017 found Softmax to be unreliable.
- Some methods are more expensive and don't answer quite the right question.
  - Gal and Ghahramani 2016: use dropout to assess uncertainty.
  - Guo et al. 2017 use a separate model to calibrate.
  - Lakshmiarayanan et al. 2017 use an ensemble of neural networks.

Fundamental principle behind our approach:

Small distance to the closest flip point means that small perturbations in the input can change the prediction of the model, while large distance to the flip point means that a larger change is necessary.

Therefore, the flip point defines the sensitivity.

When the difference between $\boldsymbol{x}$ and its closest flip point is less than the uncertainty in the measurements, then the prediction made by the model is quite possibly incorrect, and the user should be told!

Example of uncertainty reporting: MNIST

Consider the MNIST dataset, 60,000 28×28 images of digits 0 through 9.

First we reduce the 784 "features" to the 100 most important using wavelets and a pivoted QR decomposition.

Why wavelets/shearlets are a good representation of image features: see Schug, Easley, O'Leary (2015, 2017)

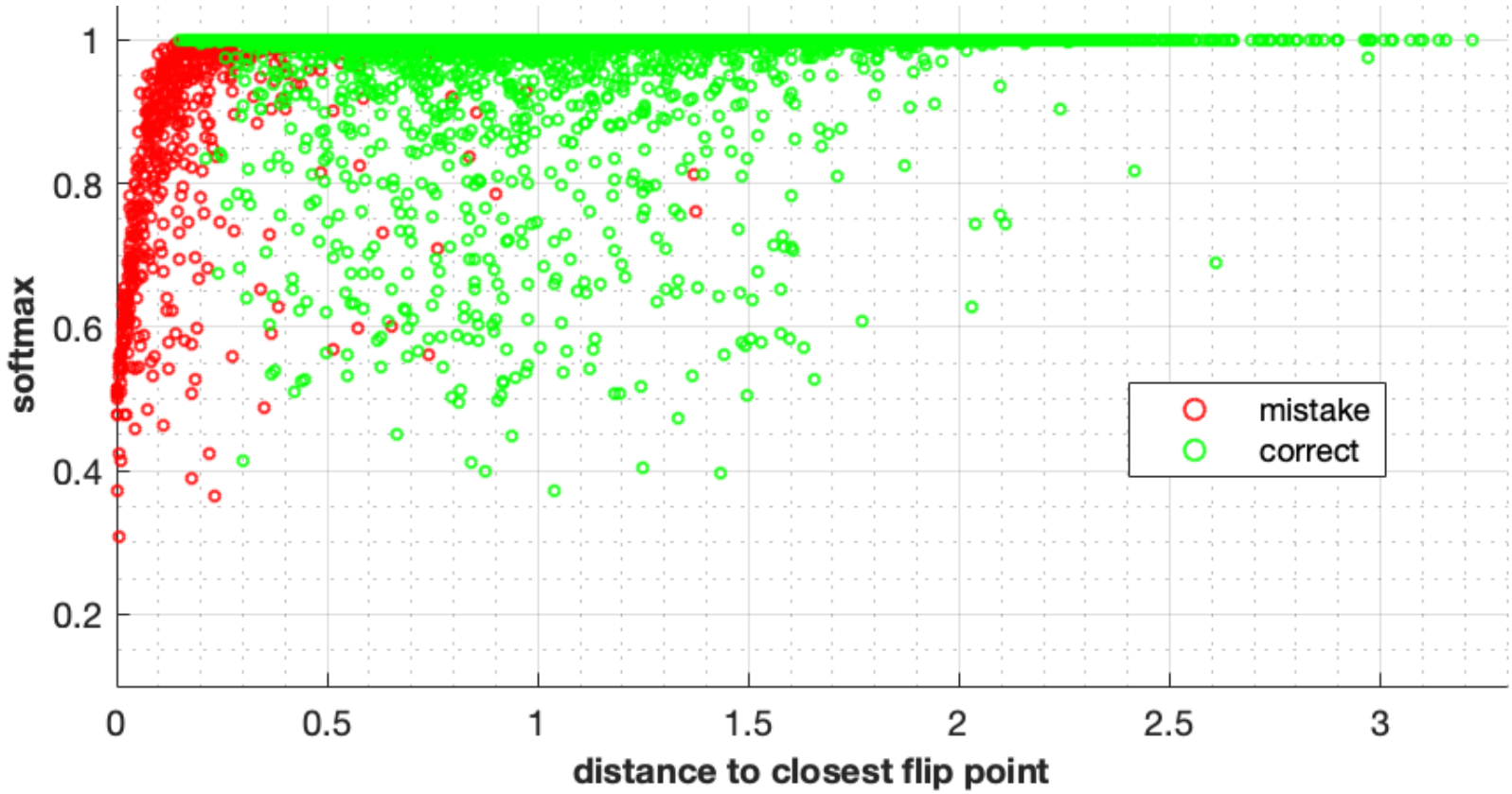
Train a feed-forward neural network with 12 layers using Tensorflow, with Adam optimizer and learning rate of 0.001.

Flip points identify uncertainties: MNIST

This 8 is mistakenly classified as a 3 by our neural network, but distances to the nearest flip points tells us that we might be wrong.

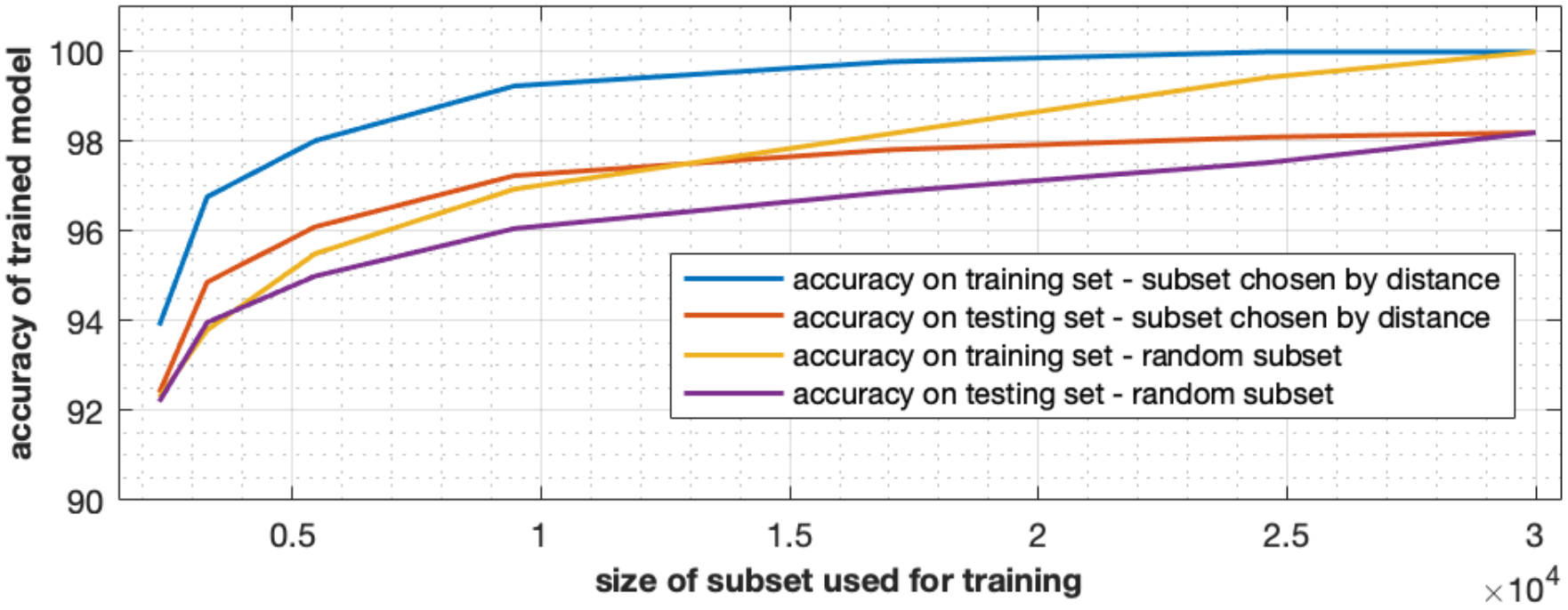
Class	0	1	2	3	4	5	6	7	8	9
Distance	1.27	1.32	0.58	**0.00**	2.16	0.56	1.45	1.51	**0.16**	0.90

Distance is a more reliable indication of uncertainty than softmax: MNIST



Using Flip Points to Identify Most/least Influential Training Points

Distance also identifies points important in the training: MNIST



Using Flip Points to Identify Most/Least Influential Features

Which features are most/least influential?

- We form a matrix with one row $(\hat{\boldsymbol{x}}^{\mathcal{C}} - \boldsymbol{x})$ for each data point.
- PCA analysis of this matrix identifies the most influential directions for flipping the outputs in the dataset and thus the most influential features.

Aside: A pivoted QR decomposition of the matrix of training data (one row per sample) should be used to eliminate redundant features before training.

Influential features: Adult Income Dataset

Adult Income dataset

- 32,561 points (people) in the training set and 16,281 in the testing set.
- Two classes: annual income $>$ 50K or not.
- Normalize continuous variables to the range (0, 100), and also use these ranges to constrain the search for flip points.
- Each categorical type (work-class, education level, marital status, occupation, relation- ship, race, sex, native country) is represented by one binary feature. When searching for a flip point, we require exactly one binary element to be equal to 1 for each of the categorical variables.

Consider the subset of directions that flip a $\leq$ 50K income to $>$50K.

Influential features identified for Adult Income Dataset

Results: The first principal component reveals that the features with most positive impact on the decision of our neural network are:

- *having a master's degree*,
- *having capital gains*,
- and *working in the private sector*.

Features with most negative impact are:

- *having highest education of Preschool*,
- *working without pay*,
- and *having capital loss*.

Pivoted QR decomposition of the matrix of directions reveals that some features, such as *having a Prof-school degree*, have no impact.

Influential features identified for Adult Income Dataset, cont.

PCA on the directions between the mistakes in the training set and their closest flip points shows that *native country United States* has the largest coefficient in the first principal component, followed by *being a wife* and *having capital gain*.

The most significant features with negative coefficient are *being a husband* and *native country Cambodia* and *Ireland*.

Influential features: Wisconsin Breast Cancer Dataset

Wisconsin breast cancer database: 30 features extracted from digitized images of fine needle aspirate of 569 breast masses.

Normalization: divide features by the mean over the training data.

We perform PCA on the matrix of directions between each *Benign* input and its closest flip point, and look at the first principal component.

Influential features identified for Wisconsin Breast Cancer Dataset

For our neural network

- From PCA on the matrix of directions for **Benign** inputs: The most prominent features that flip the decision to **Malignant** are changes in *standard error of radius* and *standard error of texture*.
- From PCA on the matrix of directions for **Malignant** inputs: The most prominent features that flip the decision to **Benign** are changes in *standard error of texture* and *worst area*.

A clinician can use this information

- for scientific discovery,
- to validate the trained neural network as a computational tool,
- or to develop alternate normalizations to improve the neural network.

Using Flip Points to Improve the Training of the Model

Using Flip Points to Improve the Training of the Model

- We can use flip points as synthetic data, adding them to the training set to move the output boundaries of a neural network insightfully and effectively.

- If a model is biased for or against certain features of the inputs, we could alter that bias using synthetic data.

Using Flip Points to Explain the Model's decisions

Using Flip Points to Explain the Model's decisions

Closest flip points and closest constrained flip points enable us to generate explanations of the model's recommendations.

Example: FICO dataset, report on Person 1:

The model predicts that this person has **Bad** risk.

The prediction would remain **Bad** if:

- any individual feature is changed,
- or only features about *Delinquencies* are changed,
- or *Months Since Most Recent Trade Open* is changed in combination with any other individual feature.

The prediction would change to **Good** if:

- *Percentage of Trades Never Delinquent* changes to 100% and *Months Since Most Recent Inquiry Excluding Last 7 Days* is greater than 6,
- or *Number of Satisfactory Trades* is greater than 20 and *Average Months in File* is greater than 96,
- or $<$ changes in the other 29 feature pairs $>$.

Example: FICO dataset, report on Person 1, cont.

The smallest change in features that changes
the model's prediction to **Good** is:

- *Average Months in File* increases from 84 to 111.2, and
- *Number of Satisfactory Trades* changes from 20 to 24, and
- *Months Since Most Recent Delinquency* changes from 2 to 0, and
- *Number of Trades Open in Last 12 Months* changes from 1 to 2, and
- *Net Fraction Revolving Burden* changes from 33 to 8.5.

The Homotopy Method

Our homotopy method for computing flip points

To find the closest flip point for $\boldsymbol{x}$ for the model defined by $\boldsymbol{\sigma}$:

1. Find a $\hat{\boldsymbol{\sigma}}$ and a bias vector $\hat{\boldsymbol{b}}^m$ (for the last layer) so that $\boldsymbol{x}$ itself is the flip point. (The details are ugly but easily executed.)
2. Follow the path of flip points from this starting neural network to the original one by considering networks parameterized by

$$\alpha\boldsymbol{\sigma} + (1-\alpha)\hat{\boldsymbol{\sigma}},$$
$$\alpha\boldsymbol{b}^m + (1-\alpha)\hat{\boldsymbol{b}}^m,$$

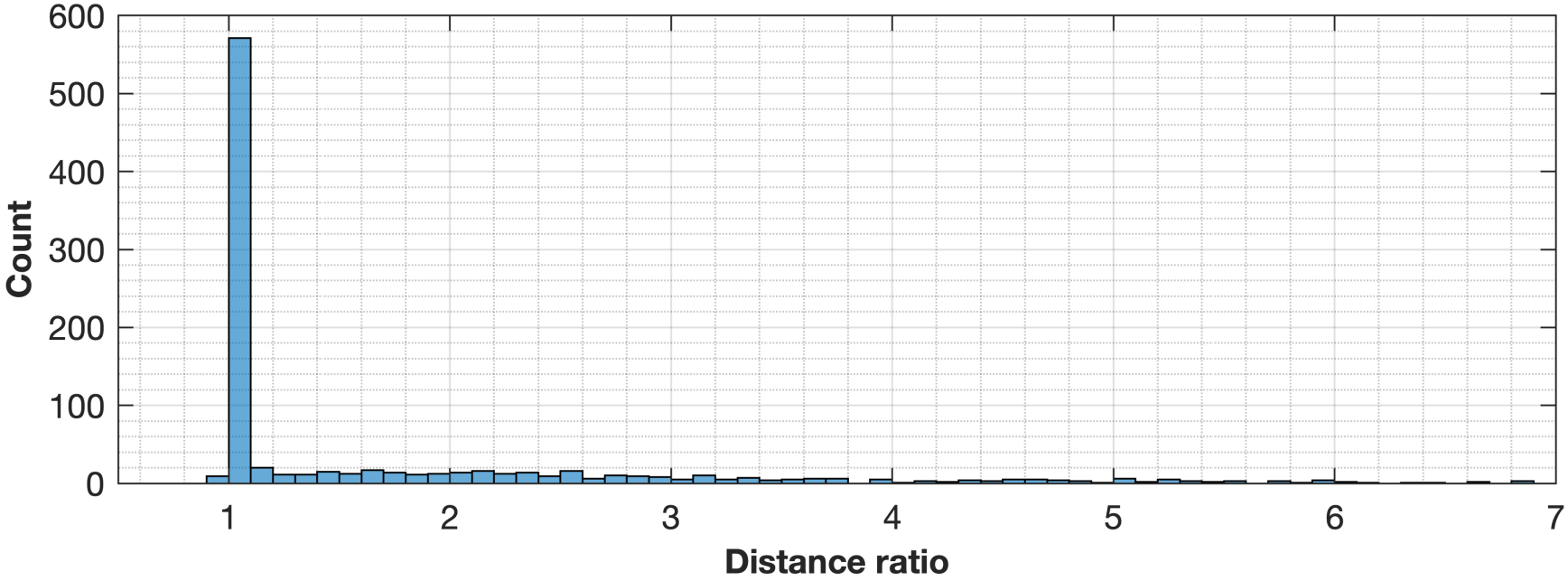
for $0 \leq \alpha \leq 1$.

We choose to do this by finding closest flip points for $\alpha = \delta, 2\delta, \ldots 1$ using an off-the shelf optimizer, given the previous closest flip point as a starting point.

How effective is the homotopy algorithm?

Results on Adult Income dataset, with a combination of continuous and discrete features: 100 data points randomly chosen from testing set, measuring

$$\frac{\text{distance for best off-the-shelf solution}}{\text{distance to flip point reported by homotopy solution}}$$



- For nearly 60% of the data points, the distance ratio is very close to 1.
- The range of the ratio is $[0.95, 7])$.
- For 1.4% of the data points, other algorithms do not yield a feasible flip point, while the homotopy algorithm always finds a feasible flip point.

How much overhead are we adding?

We can use an off-the-shelf algorithm that relies on derivatives for the continuous variables.

- Derivatives of $\boldsymbol{z}$ w.r.t. the parameters $\boldsymbol{\sigma}$ and $\boldsymbol{b}^m$ are easy.
- The cost for a given $\boldsymbol{x}$ is less than computing the derivative of the loss function of the network for a single input $\boldsymbol{x}$ and one choice of parameters and a single layer of the network (done during training).
- These training derivatives are computed for every training input and every choice of parameter and every layer of the network.
- Assuming a modest number of iterations for our algorithm, our cost (for a number of points comparable to the training set) is small relative to the training cost.

As noted above, on a laptop, we compute a flip point in 1-5 sec.

Conclusions: What's new? We ...

- Defined flip points and constrained flip points for deep learning models.
- Demonstrated how they can be computed using off-the-shelf algorithms or a new homotopy method.
- Advocate using erf as a tunable activation function for neural networks.
- Advocate using wavelet/shearlet encoding for image input.
- Quantified sensitivity/uncertainty in the model output using flip points.
- Identified influential training points.
- Used PCA and flip points to identify most/least influential features.
- Added flip points to the training set to improve the model.
- Provide clear explanations of a model's decisions, even for black box models. The approach is model agnostic.

References:

- These slides: www.cs.umd.edu/users/oleary/talkview.pdf
- See papers on arXiv.org by Yousefzadeh and O'Leary.

Roozbeh Yousefzadeh

Thank you!