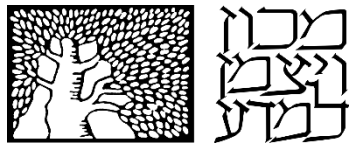
Controlling Neural Level Sets

Matan Atzmon

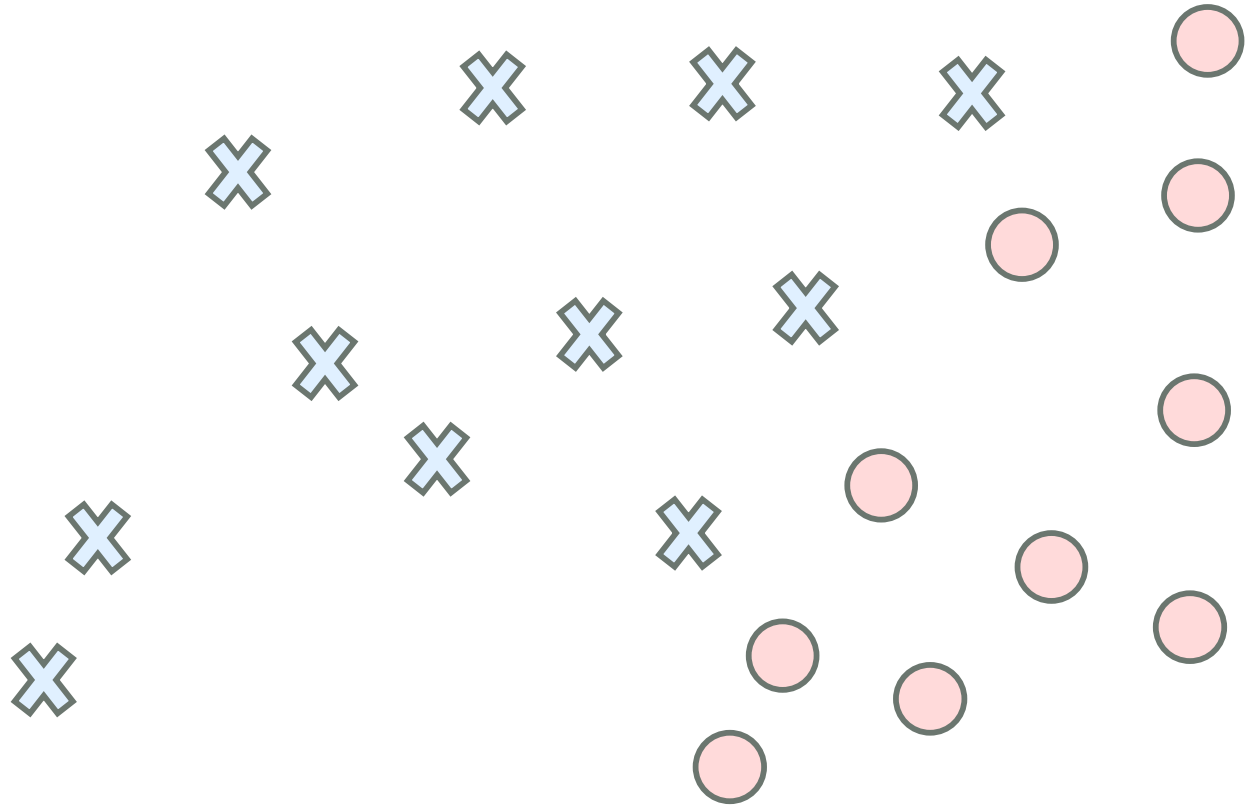
Joint work with Niv Haim, Lior Yariv, Ofer Israelov, Haggai Maron and Yaron Lipman

Weizmann Institute of Science



NeurIPSI, November 2019

Classification with Neural Networks



Neural Network

Parameters

$$F(x;\theta): \mathbb{R}^d \times \mathbb{R}^m \rightarrow \mathbb{R}$$

Input

Classification with Neural Networks

$$\mathcal{S}(\theta) = \{x \in \mathbb{R}^d | F(x) = 0\}$$

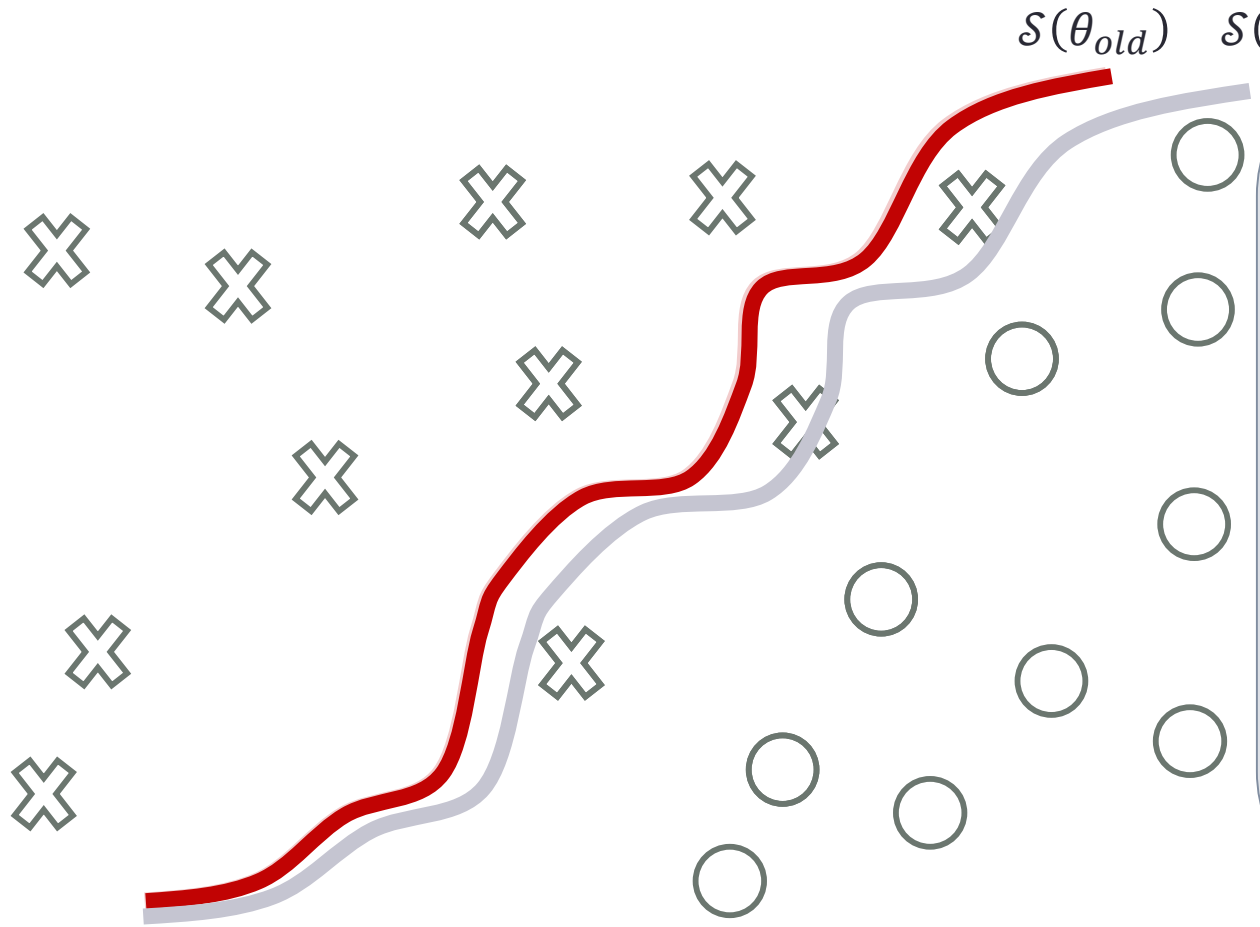
Neural Network

Parameters

$$F(x; \theta): \mathbb{R}^d \times \mathbb{R}^m \rightarrow \mathbb{R}$$

Input

Training Neural Networks



- Draw a batch of points $\{x_i\}_{i=1}^{N}$
- Evaluate the network $\{F(x_i))\}_{i=1}^{N}$
- Plug it in a loss function $\sum_i L(F(x_i), y_i)$
- Update θ to decrease the Loss

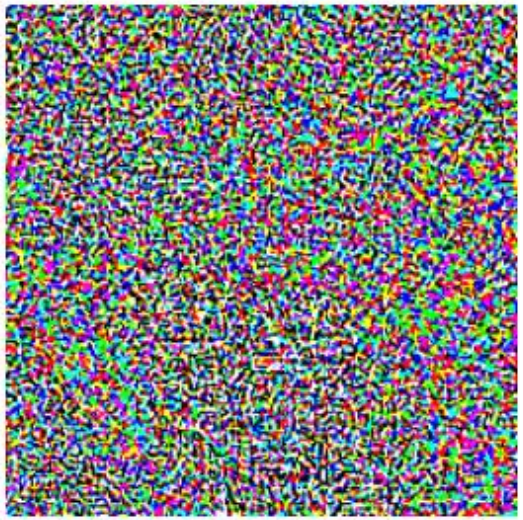
Motivation

- Loss functions(e.g., cross entropy loss) **only** measure network output values of training examples
- Therefore, decision boundary is only controlled **indirectly**

Motivation

$F(x)$ =“panda”

+



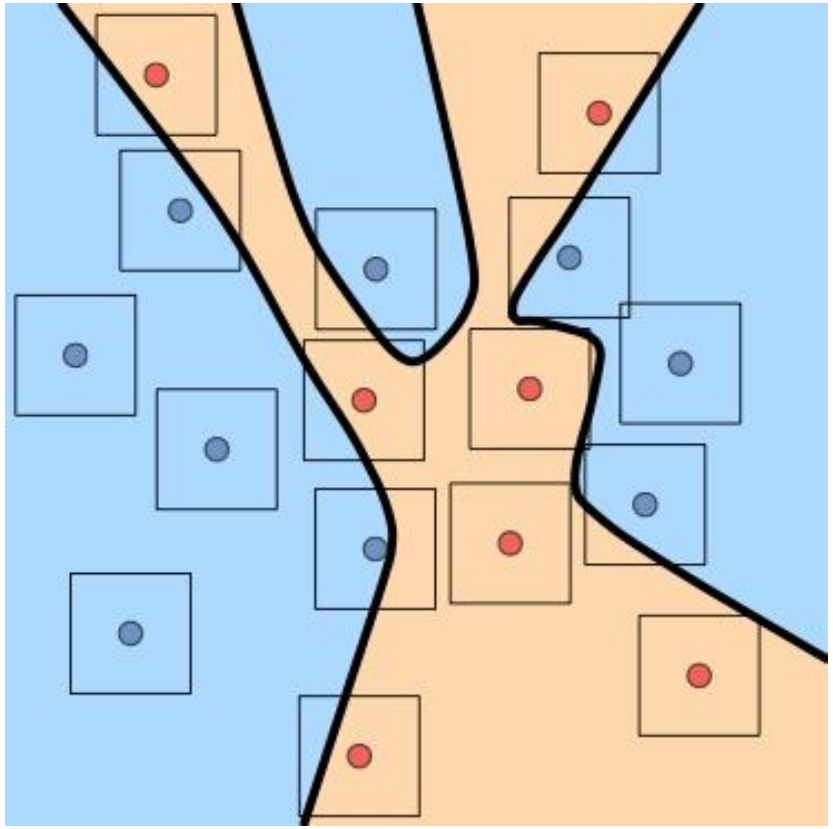
adversarial perturbation

=

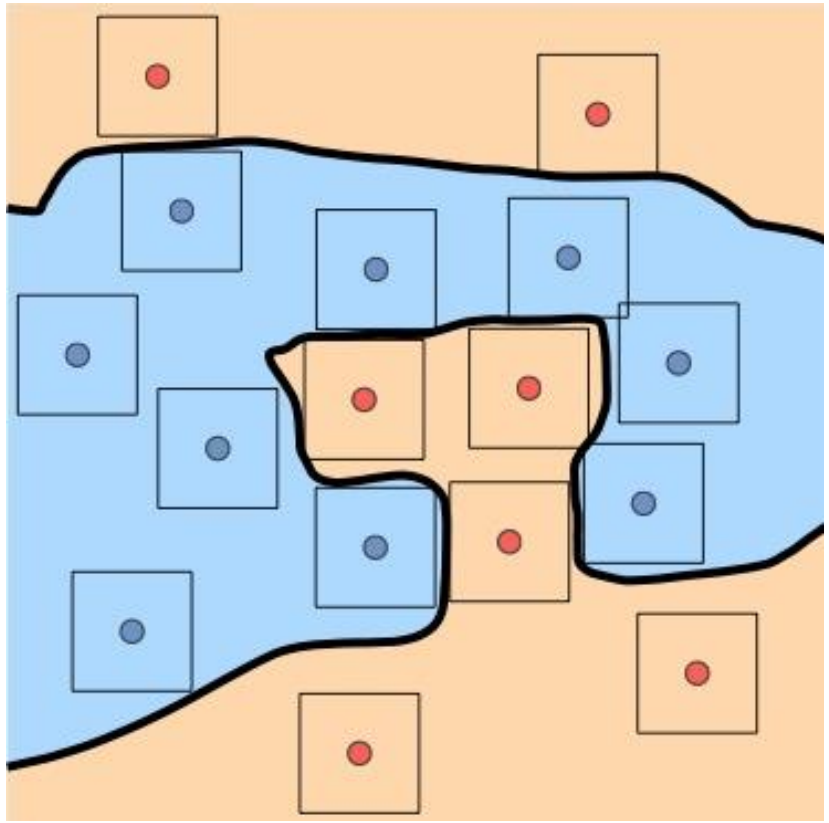
$F(x)$ =“gibbon”

Image taken from [Goodfellow et al., 2014]

Motivation

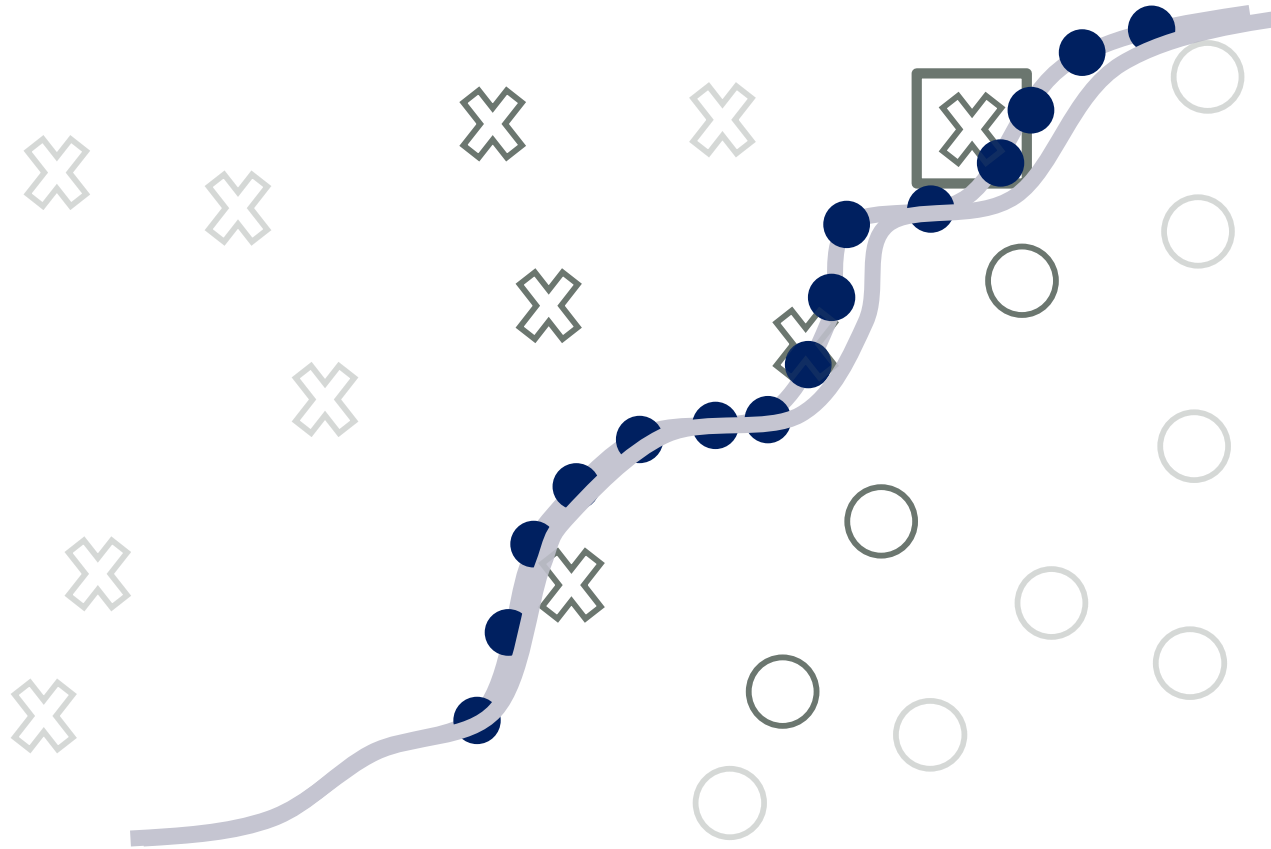


Training with Cross Entropy Loss



Training with our method

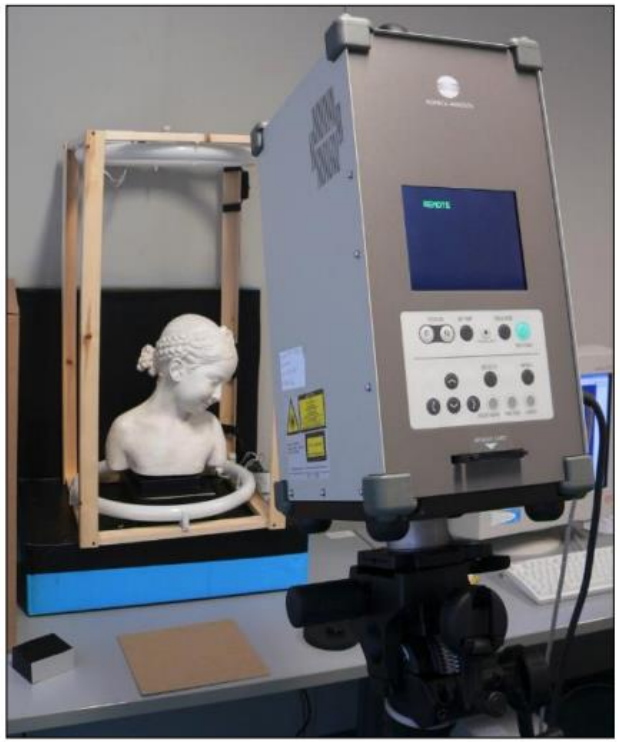
Idea – Direct Control of Level Sets



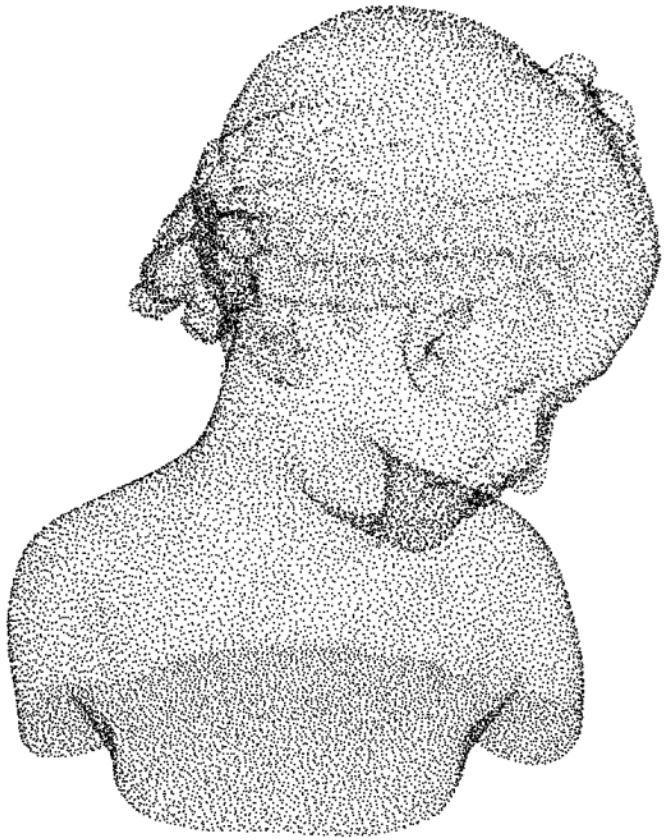
- Draw a sample of points $\{p_i\}$ from the decision boundary
- Draw a batch of points $\{x_i\}_{i=1}^N$
- Relate the samples to network parameters $p_i \mapsto p_i(\theta)$
- Evaluate the network $\{F(x_i)\}_{i=1}^N$
- Incorporate samples into a loss function $\max(0, \varepsilon - \text{sign}(F(x_i)) \|x - p_i(\theta)\|_\infty)$
- Plug it in a loss function $\sum_i L(F(x_i), y_i)$
- Updating θ moves the decision boundary
- Update θ to decrease the Loss in a controlled fashion

Surface Reconstruction

Motivation – Surface Reconstruction



Laser scanning



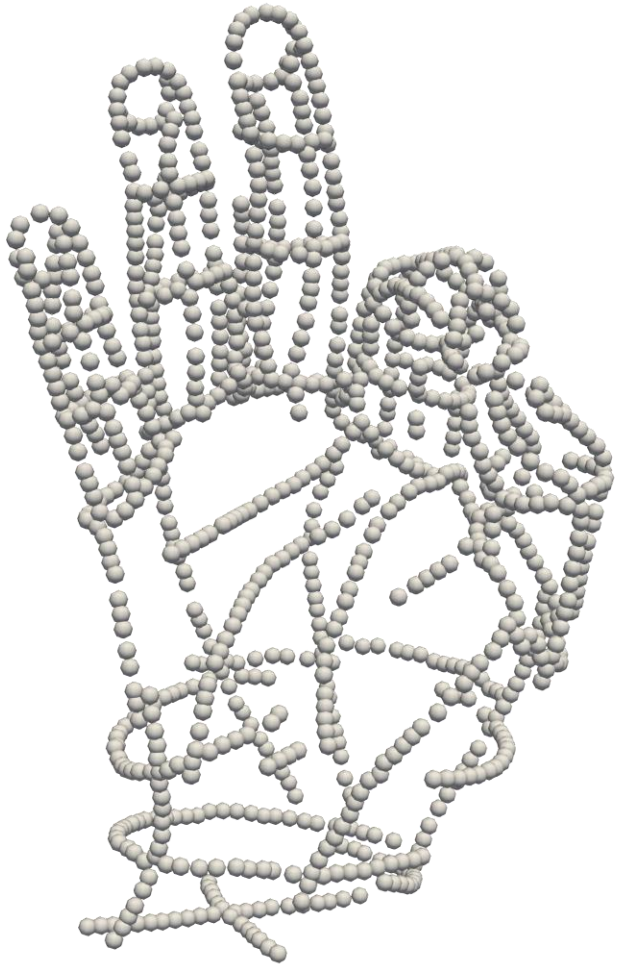
Point Cloud

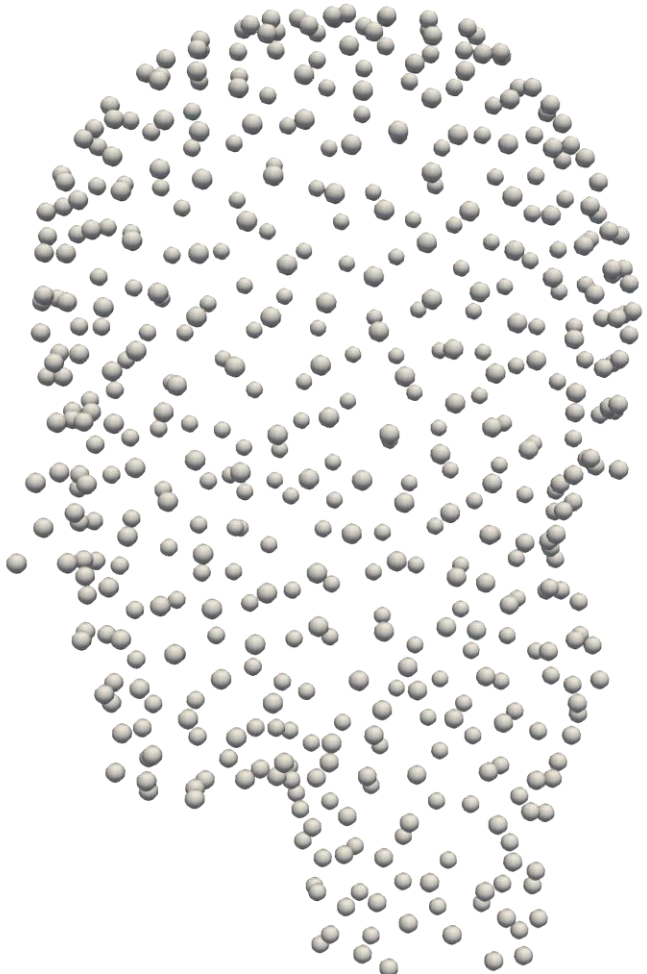
Reconstructed Surface

Slide taken from Pierre Alliez

Surface Reconstruction Challenges

- Varying Sampling Density
- Noise
- Outliers
- Missing data





Surfaces with Neural Networks

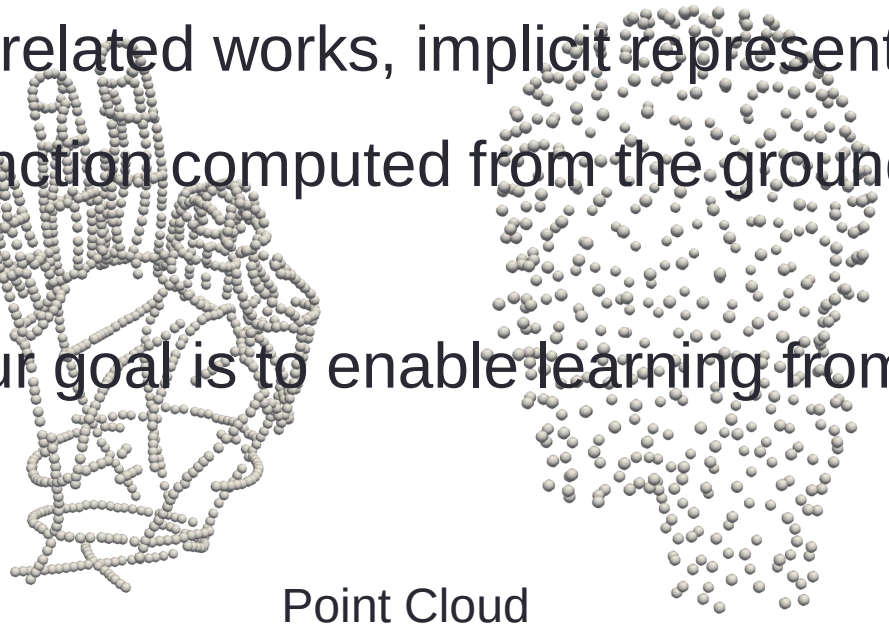
- Model a surface implicitly using a level set of a neural network

$$\mathcal{S}(\theta) = \{x \in \mathbb{R}^3 | F(x) = 0\}$$

- In related works, implicit representation were learned using regression, to a function computed from the ground truth surface

[Park et al. 2019, Chen et al. 2019, Mescheder etl a. 2019]

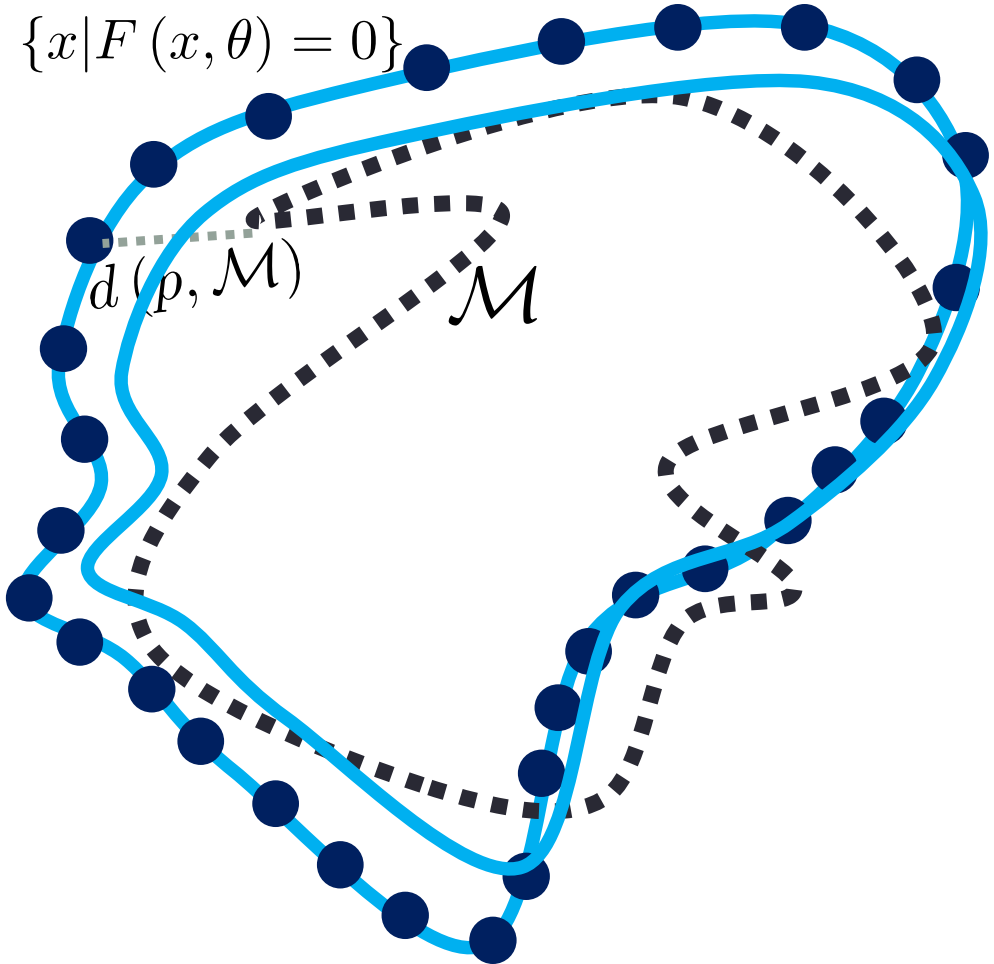
- Our goal is to enable learning from raw data



Point Cloud

0 Level set of a neural network

Idea – Direct Control of Level Sets



- Draw a sample $\{p_i\}$ from the network zero level set
- Relate the samples to network parameters

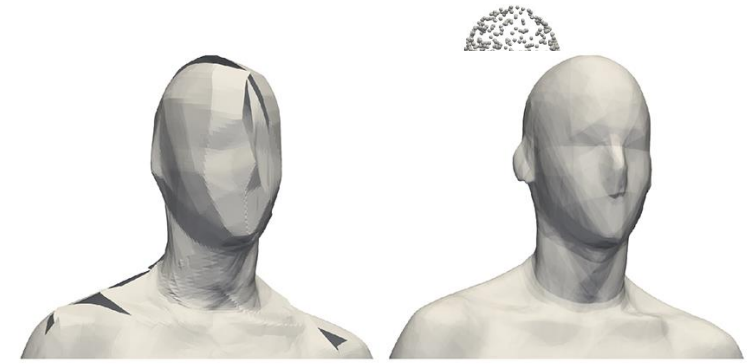
$$p_i \mapsto p_i(\theta)$$

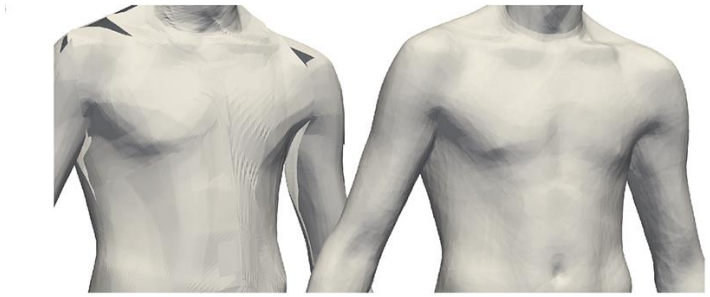
- Incorporate samples into a loss function

$$\int_{\mathcal{S}(\theta)} \min_{x\in\mathcal{M}} \|x - p(\theta)\|_2 \, dv(p) + \lambda \sum_{x\in\mathcal{M}} |F(x)|$$

- Updating θ moves the zero level set towards the input

Results on Faust scans dataset

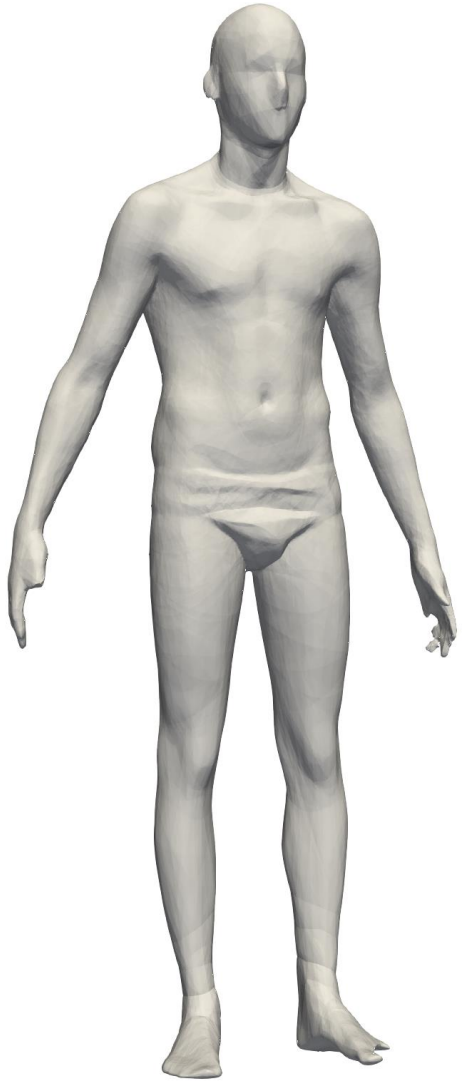




AtlasNet

Ours

	Chamfer L1	Chamfer L2
AtlasNet-1 sphere	23.56 ± 2.91	17.69 ± 2.45
AtlasNet-1 patch	18.67 ± 3.45	13.38 ± 2.66
AtlasNet-25 patches	11.54 ± 0.53	7.89 ± 0.42
Ours	$\mathbf{10.71} \pm 0.63$	$\mathbf{7.32} \pm 0.46$



Ours

Results on Adversarial Robustness

Method	Dataset	Attack	Test Acc.	Rob. Acc. Xent	Rob. Acc. Margin
Standard	MNIST	$\text{PGD}^{40}(\varepsilon = 0.3)$	99.34%	13.59%	0.00%
Madry et al. [2]	MNIST	$\text{PGD}^{40}(\varepsilon = 0.3)$	99.35%	96.04%	96.11%
TRADES [4]	MNIST	$\text{PGD}^{40}(\varepsilon = 0.3)$	98.97%	96.75%	96.74%
Ours	MNIST	$\text{PGD}^{40}(\varepsilon = 0.3)$	99.35%	99.23%	97.35%
Standard	CIFAR10	$\text{PGD}^{20}\ (\varepsilon = 0.031)$	83.67%	0.00%	0.00%
Madry et al. [2]	CIFAR10	$\text{PGD}^{20}\ (\varepsilon = 0.031)$	71.86%	39.84%	38.18%
TRADES [4]	CIFAR10	$\text{PGD}^{20}\ (\varepsilon = 0.031)$	71.24%	41.89%	38.4%
Ours	CIFAR10	$\text{PGD}^{20}\ (\varepsilon = 0.031)$	71.96%	38.45%	38.54%

Results of L_∞-bounded attacks compared to other methods.

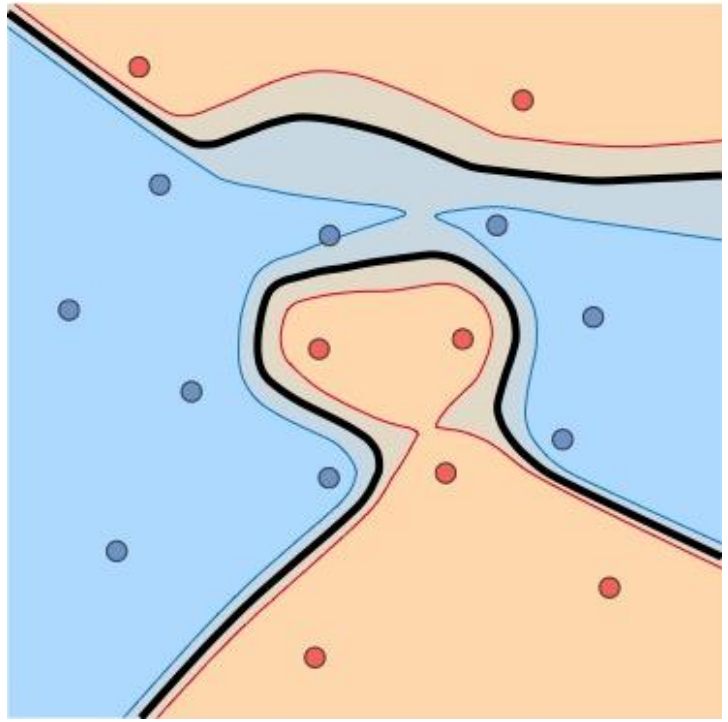
Our Approach

- Sampling of neural level sets
- Relating the samples' positions to the network parameters
- Achieved by adding a fixed linear layer to the original network

$$p(\theta) = p - D_x F(p;\theta_0)^\dagger F(p;\theta)$$

Summary

- Incorporate level sets of neural networks into new loss functions
- Robustness to adversarial examples
- Surface reconstruction from raw data
- Generalize SVM to Neural Networks (Not covered)



Future Directions

- Investigating control of intermediate layers' level sets
- Developing sampling conditions to ensure coverage of neural level sets
- Employing additional geometrical regularization to the neural level sets

The End

- Code is online: https://github.com/matanatz/ControllingNeuralLevelsets
- Support
  - ERC Consolidator Grant (LiftMatch)
  - Israel Science Foundation
- Thanks for listening