Vehicle Tutorial Chapter 4: Property-Driven Training

Today's presentors: Ekaterina Komendantskaya and Luca Arnaboldi (live),
Matthew Daggitt (online), on behalf of the Vehicle team

In this chapter...

We will discuss:

- ... why and how training is a part of verification of neural networks

In this chapter...

We will discuss:

- ... why and how training is a part of verification of neural networks
- ... what choices exist for implementing this, generally

In this chapter...

We will discuss:

- ... why and how training is a part of verification of neural networks
- ... what choices exist for implementing this, generally
- ... what choices **Vehicle** makes in this respect

In this chapter...

We will discuss:

- ... why and how training is a part of verification of neural networks
- ... what choices exist for implementing this, generally
- ... what choices **Vehicle** makes in this respect
- ... theoretical and practical issues with the chosen methods, and **Vehicle**'s take on them

Recap: four PL problems

Recap: four PL problems

- I^O Interoperability – properties are not portable between training/counter-example search/ verification.
- I^P Interpretability – code is not easy to understand.
- $I^\int$ Integration – properties of networks cannot be linked to larger control system properties.
- E^G Embedding gap – little support for translation between problem space and input space.

Why Training is a part of Verification?

```
vehicle verify \
  --specification acasXu.vcl \
  --verifier Marabou \
  --network acasXu:acasXu_1_7.onnx \
  --property property3

Verifying properties:
property3 [==========================================] 1/1 queries
  result: counterexample found
  x: [1799.9886669999978, 1.9509286320000003e-2,
                        3.09999732192, 980.0, 1017.6036]
```

Why Training is a part of Verification?

For Chapter 3 exercise on verifying a small Fashion MNIST network, the answer would be:

	$\epsilon = 0.01$	$\epsilon = 0.05$	$\epsilon = 0.1$	$\epsilon = 0.5$
Success rate:	82.6 % (413/500)	29.8 % (149/500)	3.8 % (19/500)	0 % (0/500)

A few words on the context

1943 Perceptron by McCullogh and Pitts

90-2000 Rise of machine learning applications

2013 C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. 2013. (10000+ citations on GS)

2013-.. Tens of thousands of papers on adversarial training (in the attack-defence style)

A. C. Serban, E. Poll, J. Visser. Adversarial Examples - A Complete Characterisation of the Phenomenon. 2019.

A few words on the context

1943 Perceptron by McCullogh and Pitts

90-2000 Rise of machine learning applications

2013 C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks. 2013. (10000+ citations on GS)

2013-.. Tens of thousands of papers on adversarial training (in the attack-defence style)

A. C. Serban, E. Poll, J. Visser. Adversarial Examples - A Complete Characterisation of the Phenomenon. 2019.

2017 First Neural network verification attempts

G. Katz, C.W. Barrett, D.L. Dill, K. Julian, M.J. Kochenderfer: Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks. CAV (1) 2017: 97-117.

X. Huang, M. Kwiatkowska, S. Wang, M. Wu. Safety Verification of Deep Neural Networks. CAV (1) 2017: 3-29.

2017-.. Hundreds of papers on neural network verification

Training for Robustness

Training for Robustness

Training generally:

1. depends on data
2. depends on loss functions
3. some other parameters like shape of the functions

1. Data Augmentation

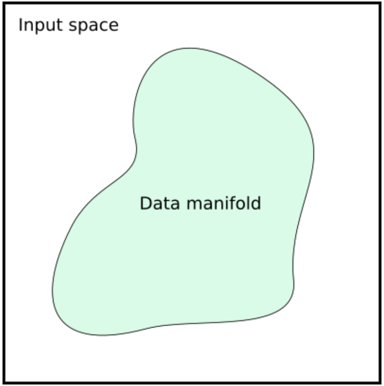
Suppose we are given a data set $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), \dots, (\mathbf{x}_n, \mathbf{y}_n)\}$. Prior to training, generate new training data samples close to existing data and label them with the same output as the original data.

C. Shorten, T.M. Khoshgoftaar: A survey on image data augmentation for deep learning. J. Big Data 6, 60 (2019)

1. Data Augmentation

Suppose we are given a data set $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\}$. Prior to training, generate new training data samples close to existing data and label them with the same output as the original data.

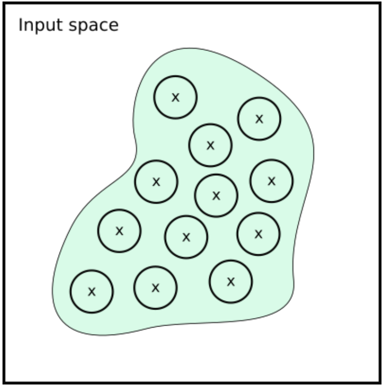
C. Shorten, T.M. Khoshgoftaar: A survey on image data augmentation for deep learning. J. Big Data 6, 60 (2019)



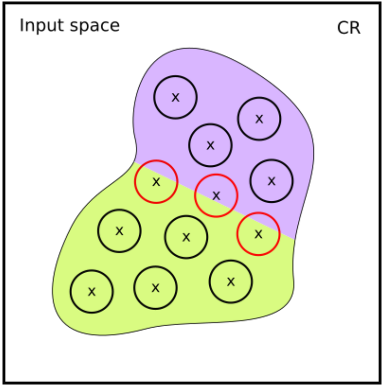
1. Data Augmentation

Suppose we are given a data set $\mathcal{D} = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\}$. Prior to training, generate new training data samples close to existing data and label them with the same output as the original data.

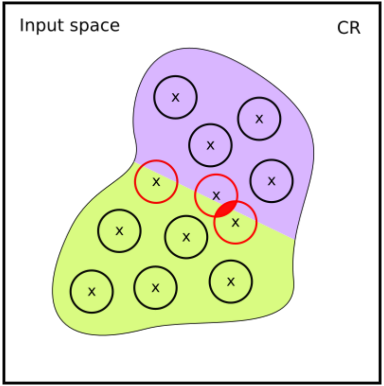
C. Shorten, T.M. Khoshgoftaar: A survey on image data augmentation for deep learning. J. Big Data 6, 60 (2019)



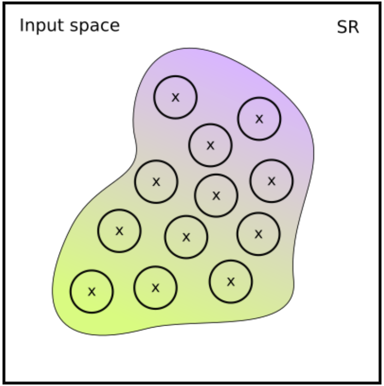
However,



However,



Adversarial Training



2. Solutions Involving Loss Functions

Given a data set $\mathcal{D}$, a function $f : \mathbb{R}^n \to \mathbb{R}^m$, a loss function $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ computes a penalty proportional to the difference between the output of f on a training input x and a desired output y.

2. Solutions Involving Loss Functions

Given a data set $\mathcal{D}$, a function $f : \mathbb{R}^n \to \mathbb{R}^m$, a loss function $\mathcal{L} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ computes a penalty proportional to the difference between the output of f on a training input x and a desired output y.

Example (Cross Entropy Loss Function)

Given a function $f : \mathbb{R}^n \to [0,1]^m$, the cross-entropy loss is defined as

$$\mathcal{L}_{ce}(\mathbf{x}, \mathbf{y}) = -\sum_{i=1}^{m} \mathbf{y}_i \, \log(f(\mathbf{x})_i) \tag{1}$$

where $\mathbf{y}_i$ is the true probability for class i and $f(\mathbf{x})_i$ the probability for class i as predicted by f when applied to $\mathbf{x}$.

2. Adversarial Training for Robustness

- gradient descent minimises loss $\mathcal{L}(\hat{\mathbf{x}}, \mathbf{y})$ between the predicted value $f_\theta(\hat{\mathbf{x}})$ and the true value $\mathbf{y}$, for each entry $(\hat{\mathbf{x}}, \mathbf{y})$ in $\mathcal{D}$:

$$\min_\theta \mathcal{L}(\hat{\mathbf{x}}, \mathbf{y})$$

2. Adversarial Training for Robustness

- gradient descent minimises loss $\mathcal{L}(\hat{\mathbf{x}}, \mathbf{y})$ between the predicted value $f_\theta(\hat{\mathbf{x}})$ and the true value $\mathbf{y}$, for each entry $(\hat{\mathbf{x}}, \mathbf{y})$ in $\mathcal{D}$:

$$\min_\theta \mathcal{L}(\hat{\mathbf{x}}, \mathbf{y})$$

- instead minimise the loss with respect to the worst-case perturbation of each sample in $\mathcal{D}$.
  - Replace the standard training objective with:

$$\min_\theta \max_{\forall \mathbf{x}: |\mathbf{x} - \hat{\mathbf{x}}| \leq \epsilon} \mathcal{L}(\mathbf{x}, \mathbf{y})$$

  - often referred to as the method of "projected gradient descent" (PGD)

I.J. Goodfellow, J. Shlens, C. Szegedy: Explaining and harnessing adversarial examples. 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings (2015)

3. Lipshitz Continuity

Optimise for:

$$\forall \mathbf{x} : |\mathbf{x} - \hat{\mathbf{x}}| \leq \epsilon \Rightarrow |f(\mathbf{x}) - f(\hat{\mathbf{x}})| \leq L|\mathbf{x} - \hat{\mathbf{x}}|$$

P. Pauli, A. Koch, J. Berberich, P. Kohler, F. Allgower: Training robust neural networks using Lipschitz bounds. IEEE Control Systems Letters (2021)

H. Gouk, E. Frank, B. Pfahringer, M.J. Cree: Regularisation of neural networks by enforcing Lipschitz continuity. Machine Learning 110(2), 393–416 (2021)

and much more...

Ok, great!

Machine Learning Community knows how to make our networks robust, and maybe even verifiable!

Ok, great!

Machine Learning Community knows how to make our networks robust, and maybe even verifiable!

But remember:

- I^O Interoperability – properties are not portable between training/counter-example search/ verification.
- I^P Interpretability ...
- $I^\int$ Integration ...
- E^G Embedding gap ...

Interpretation of adversarial training in connection to verification properties

- Recall the epsilon ball robustness:
  $\forall \mathbf{x} \in \mathbb{B}(\hat{\mathbf{x}}, \epsilon).\ robust(f(\mathbf{x}))$

M. Casadio, E. Komendantskaya, M. L. Daggitt, W. Kokke, G. Katz, G. Amir, and I. Rafaeli. 2022. Neural Network Robustness as a Verification Property: A Principled Case Study. CAV'22.

Interpretation of adversarial training in connection to verification properties

- Recall the epsilon ball robustness:
  $\forall \mathbf{x} \in \mathbb{B}(\hat{\mathbf{x}}, \epsilon).\ robust(f(\mathbf{x}))$
- We can map different kinds of adversarial training to formal properties:

Training style	Definition of *robust*
Data Augmentation	$argmax\ f(\mathbf{x}) = c$
Adversarial Training	$\lvert f(\mathbf{x}) - f(\hat{\mathbf{x}})\rvert \leq \delta$
Lipschitz Continuity	$\lvert f(\mathbf{x}) - f(\hat{\mathbf{x}})\rvert \leq L\lvert\mathbf{x} - \hat{\mathbf{x}}\rvert$
...	...

M. Casadio, E. Komendantskaya, M. L. Daggitt, W. Kokke, G. Katz, G. Amir, and I. Rafaeli. 2022. Neural Network Robustness as a Verification Property: A Principled Case Study. CAV'22.

Problem Recap

- one kind of robustness does not necessarily imply another;
- It is easy to get it wrong, and, while optimising for a wrong kind of robustness, achieve little in verification success rates
- And what to do with properties that are not ϵ-ball robustness?

Problem Recap

- one kind of robustness does not necessarily imply another;
- It is easy to get it wrong, and, while optimising for a wrong kind of robustness, achieve little in verification success rates
- And what to do with properties that are not ϵ-ball robustness?

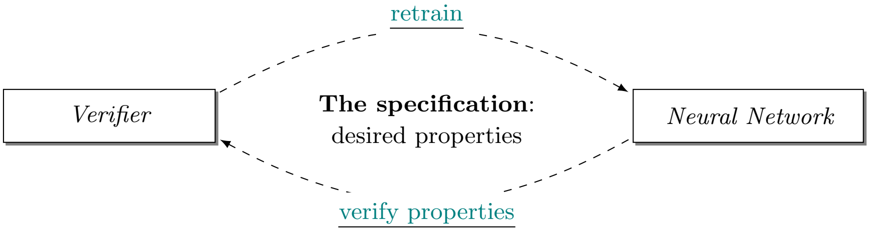
I^O Interoperability – properties are not portable between training/counter-example search/ verification.

I^P Interpretability …

$I^\int$ Integration …

E^G Embedding gap …

The solution we are looking for



In Vehicle terms,