

Analysis and Testing for AI software

Wei Le

April 29, 2020

Neural Networks

- ▶ training the model using massive amount of data and then predict the labels given new inputs
- ▶ autonomous vehicles
- ▶ aircraft collision avoidance detection

Testing for Autonomous Vehicles at Uber

- ▶ collect real-world data
- ▶ generate synthetic data based on real-world data, e.g., lack of collision scenarios
- ▶ constantly running testing for the Autonomous Vehicles system
 - ▶ can you recognize the background data
 - ▶ can you recognize moving object
 - ▶ can you predict the next movement of the moving object

Neural Networks Input and Output

- ▶ Input: 3 Dimensional inputs (e.g., colored images)
- ▶ Reshape the input to vectors
- ▶ Output: labels (e.g., is it 1, 2, ...9)
- ▶ Using massive amount of labeled data to "parameterize" neural network for a specific problem, which we call *models*, models then used like a classifier

Neural Networks Internal

- ▶ consists of sequences of layers, e.g., input/output layer, and hidden layers
- ▶ each layer consists of *neurons* (also called *perceptron*)
- ▶ *feed-forward*: the output of a neuron is not feedback to the previous layer
- ▶ *convolutional neural networks*: *fully connected layers*, *pooling layers* and *convolutional layers*

Challenges





- ▶ Robustness: are neural network vulnerable to *adversarial examples* – slightly perturbing an input classified correctly leads to mis-classification?
- ▶ Testing: How to test models so we know it is a good model and ready to go for application?
- ▶ Debugging: if the prediction is wrong, is it a problem of insufficient training data, implementation errors in networks, or it is an error expected by the algorithm?

Can program analysis help?

- ▶ differential analysis: compare the output of neurons for correct and incorrect predictions [4]
- ▶ compare with other versions of software [5]
- ▶ abstract interpretation [1]

AI² Safety and Robustness Certification of Neural Networks with Abstract Interpretation

- ▶ AI²: *abstract interpretation* for artificial intelligence
- ▶ Goal: prove safety properties (e.g, robustness)
- ▶ Example:
 - ▶ FGSM attack: Adding a particular noise vector multiplied by a small number ϵ , can neural net still correctly classify the digit
 - ▶ Brighten attack: change all pixels above the threshold $1-\delta$ to the brightest possible value

Attack	Original	Perturbed
FGSM [12], $\epsilon = 0.3$		
Brightening, $\delta = 0.085$		

- ▶ *abstract interpretation*: Sound, computable and precise finite approximation of potentially infinite sets of behaviors
- ▶ Construct an *abstract element* that captures all perturbed images (more than 10^{1154} images)
- ▶ Construct *abstract transformer* that compute the effect of neural net on the abstract element
- ▶ if the abstract output satisfies the property, all concrete inputs satisfy the property (soundness of abstract interpretation)

Abstract Interpretation

See Alex Aiken's slides for an example of abstract interpretation

Mapping different layers to CAT functions

Done manually based on algorithms

- ▶ activation function (ReLU) to CAT
- ▶ Pooling layer (extract features), convolution layer (extract features) and fully connected layer (classification) to CAT

Defining CAT functions

represent neural network as *conditional affine transformations (CAT)*

$$\begin{aligned} f(\bar{x}) &::= W \cdot \bar{x} + \bar{b} \\ &| \text{ case } E_1: f_1(\bar{x}), \dots, \text{ case } E_k: f_k(\bar{x}) \\ &| f(f'(\bar{x})) \\ E &::= E \wedge E \mid x_i \geq x_j \mid x_i \geq 0 \mid x_i < 0 \end{aligned}$$

Abstract Interpretation

Fig. 6 shows a CAT function $f: \mathbb{R}^2 \rightarrow \mathbb{R}^2$ that is defined as $f(\bar{x}) = \begin{pmatrix} 2 & -1 \\ 0 & 1 \end{pmatrix} \cdot \bar{x}$ and four input points for the function f , given as $X = \{(0, 1), (1, 1), (1, 3), (2, 2)\}$. Let the property be $C = \{(y_1, y_2) \in \mathbb{R}^2 \mid y_1 \geq -2\}$, which holds in this example. To reason about all inputs simultaneously, we lift the definition of f to be over a set of inputs X rather than a single input:

$$T_f: \mathcal{P}(\mathbb{R}^m) \rightarrow \mathcal{P}(\mathbb{R}^n), \quad T_f(X) = \{f(\bar{x}) \mid \bar{x} \in X\}.$$

The function T_f is called the *concrete transformer* of f . With T_f , our goal is to determine whether $T_f(X) \subseteq C$ for a given input set X . Because the set X can be very large (or infinite), we cannot enumerate all points in X to compute $T_f(X)$. Instead, AI overapproximates sets with abstract elements (drawn from some abstract domain \mathcal{A}) and then defines a function, called an *abstract transformer* of f , which works with these abstract elements and overapproximates the effect of T_f . Then, the property C can be checked on the resulting abstract element returned by the abstract transformer.

Deep Gauge: Multi-Granularity Testing Criteria for Deep Learning Systems

- ▶ each neuron computes an output based on an input
- ▶ each layer computes an output based on an input
- ▶ cover all possible output values
- ▶ design a family of test criteria for neuron level and layer level

Neuron Level Criteria

- ▶ k-multisection neuron coverage: given a neuron n , the criterion measures how thoroughly the given set of test inputs T covers the range $[low_n, high_n]$
- ▶ neuron boundary coverage: how many tests cover the corner cases of $(high_n, \infty)$, and $(-\infty, low_n)$
- ▶ strong neuron activation coverage: how many corner cases w.r.t. the upper boundary value $high_n$ has been covered

Layer Level Criteria

- ▶ define "active neurons": for a given test input x and neuron n_1 and n_2 , we say n_1 is more active than n_2 given x if the output of n_1 regarding x is larger than the output of n_2
- ▶ test data should uncover more active neurons
- ▶ top k neuron coverage: how many neurons of a layer has been the most active k neurons
- ▶ top k neuron patterns: how many top k neuron patterns are covered

Experimental Setup

- ▶ test dataset: MNIST and ImageNet
- ▶ include also adversarial test dataset

Findings

- ▶ the data set cover both main function region and corner cases, but cover the main function region more than corner cases
- ▶ the adversarial test dataset boost the coverage criteria
- ▶ lower region is more difficult to cover than the higher region

Further Reading

1. AI²: Safety and Robustness Certification of Neural Networks with Abstract Interpretation
2. Deep Gauge: Multi-Granularity Testing Criteria for Deep Learning Systems
3. Brian McClendon's talk that covers testing for Autonomous Vehicles at Uber
4. MODE: Automated Neural Network Model Debugging via State Differential Analysis and Input Selection
5. CRADLE: Cross-Backend Validation to Detect and Localize Bugs in Deep Learning Libraries