CRADLE: Cross-Backend Validation to Detect and Localize Bugs in Deep Learning Libraries

Hung Viet Pham¹
Thibaud Lutellier¹
Weizhen Qi²
Lin Tan³

¹University of Waterloo, Canada
²University of Science and Technology of China, China
³Purdue University, USA
Deep learning (DL) is pervasive

- Machine translation
- Alzheimer’s disease diagnosis
- Autonomous driving cars
- Virtual assistance
Correct DL systems require correct implementations

*DL system*

Algorithms / Models + Implementations
DL libraries are hard to test and debug

- Intrinsic complexity
- DL system expected output is unknown
  - Correct programs should output expected output.
  - The ground truth is not the expected output because models are not perfect.

*MobileNetV2 - TensorFlow: banana
Ground-truth: banana*
Idea: Differential testing

TensorFlow backend → InceptionResNetV2\textsubscript{TensorFlow} → TensorFlow classification

InceptionResNetV2 Model → A “petri-dish” image → InceptionResNetV2\textsubscript{CNTK} → CNTK classification

An inconsistency

TensorFlow classification:
- petri dish: 0.6
- face powder
- spotlight
- plate rack
- steel drum

CNTK classification:
- analog clock: 0.3
- spotlight
- barometer
- wall clock
- steel drum
The CNTK batch normalization formula was implemented incorrectly.

The developers fixed the bug after we reported it.

\[
\text{return}\left(\frac{x-\text{mean}}{(C\sqrt{\text{var}}+\epsilon)\times\gamma+\beta}\right)
\]

\[
\text{return}\left(\frac{x-\text{mean}}{C\sqrt{\text{var}+\epsilon}}\times\gamma+\beta\right)
\]
Differential testing: Challenges

- How to compare two implementations?
  - What metric to use?
  - What should be considered bugs?
- How to localize the faults?
  - How to find faults in the complex model executions?
Differential testing: Ideas

- Two metrics measure the severity of the inconsistency for a set of input instances.
- Localization map compares intermediate states of DL models for fault localization.
CRADLE: Overview

**Detection phase**

- Trained models & Validation data
- Output extractor
- Model output
- Output comparator
- Unique inconsistencies

**Localization phase**

- Crash bugs
- Inconsistency bugs
- Localization maps
- Inconsistency localizer
- Hidden states
- Hidden states extractor
CRADLE: Detection phase

Detection phase:
- Trained models & Validation data
  - Output extractor
    - Model output
      - Output comparator
        - Unique inconsistencies

Localization phase:
- Crash bugs
  - Inconsistency bugs
  - Localization maps
    - Inconsistency localizer
      - Hidden states
        - Hidden states extractor
Output extractor

- Executes the models on different backends to obtain output
- Detects crashes
Output comparator: Distance metrics

Metrics calculate difference relatively to the ground-truth.

CLASS-based (Classification)

\[
\sigma_{C,Y} = \begin{cases} 
2^{k-\text{rank}_{C,Y}} & \text{if } \text{rank}_{C,Y} \leq k \\
0 & \text{otherwise}
\end{cases}
\]

\[
D_{\text{CLASS}}_{C,Y,Y'} = |\sigma_{C,Y} - \sigma_{C,Y'}|
\]

MAD-based (Regression)

\[
\delta_{O,Y} = \frac{1}{N} \sum_{i=1}^{N} |Y_i - O_i|
\]

\[
D_{\text{MAD}}_{O,Y,Y'} = \frac{|\delta_{O,Y} - \delta_{O,Y'}|}{\delta_{O,Y} + \delta_{O,Y'}}
\]
CLASS-based distance example

Top-5 classification

\[ \sigma_{C,Y} = \begin{cases} 2^{k - \text{rank}_{C,Y}} & \text{if } \text{rank}_{C,Y} \leq k \\ 0 & \text{otherwise} \end{cases} \]

\[ D_{\text{CLASS}}_{C,Y,Y'} = |\sigma_{C,Y} - \sigma_{C,Y'}| \]

\[ \sigma_{\text{petri-dish},\text{TF}} = 2^{5-1} = 16 \]

\[ \text{Rank}_{\text{petri-dish},\text{TF}} = 1 \]

\[ \sigma_{\text{petri-dish},\text{CN}} = 0 \]

\[ \text{Rank}_{\text{petri-dish},\text{CN}} > 5 \]

\[ |\sigma_{\text{petri-dish},\text{CN}} - \sigma_{\text{petri-dish},\text{CN}}| = 16 \]
Inconsistency triggering input (ITI)

- An input instance triggers a distance larger than a threshold ($T_C$ and $T_M$)
  - E.g.,: “petri-dish” image is an ITI given $T_C = 8$.

<table>
<thead>
<tr>
<th>Theano</th>
<th>TensorFlow</th>
<th>CNTK</th>
</tr>
</thead>
<tbody>
<tr>
<td>Indian elephant</td>
<td>groom</td>
<td>Arabian camel</td>
</tr>
<tr>
<td>CNTK: groom</td>
<td>TensorFlow: banana</td>
<td>Theano: hen</td>
</tr>
</tbody>
</table>
Detect inconsistency

- An inconsistency is a pair of implementations that triggers more than $\rho\%$ of ITIs over the validation set.
CRADLE: Localization phase

Detection phase

- Trained models & Validation data
- Output extractor
- Model output
- Output comparator
- Unique inconsistencies

Localization phase

- Crash bugs
- Inconsistency bugs
- Localization maps
- Inconsistency localizer
- Hidden states
- Hidden states extractor
Hidden state extractor

- The “most inconsistent” input per inconsistency is used.
- The network structure + hidden states are considered as the network execution graph.
- Hidden states are output of hidden layers.
MAD differences

```
TensorFlow: jean

Conv2D
BatchNorm
 Activation
GloAvgPool

δ = 0.0
δ = 0.0002
δ = 0.1480
δ = 0.0860
δ = 0.0004

Input: jean

CNTK: mail bag

Conv2D
BatchNorm
Activation
GloAvgPool

δ = 0.0
δ = 0.0
δ = 0.0860
δ = 0.0004
```
Inconsistency introduction rate

- Calculate the rate of change
  - $\in$ prevent division by zero
- Highlight executions with $R$ above the third quantile

\[
R_L = \frac{\delta S_L, S'_L - \delta_{pre}}{\delta_{pre} + \epsilon}
\]

\[
\delta_{pre} = \max_{l \in \text{pre}(L)} (\delta S_l, S'_l)
\]
Result

104 unique inconsistencies
3 backends
28 models 11 datasets
7 inconsistency bugs 5 crash bugs
7 inconsistency bugs

- Batch normalization
- Padding scheme
- Pooling scheme
- Parameter organization
- BatchNormalization
- Conv2D variant
- AveragePooling2D
- Trainable Conv
Localization is helpful

Relevant to the causes of all 104 unique inconsistencies
Conclusion

● CRADLE applies differential testing on DL implementations and localize faulty functions by tracking error propagation.
  ○ Detects 7 confirmed inconsistency bugs and 5 crash bugs
  ○ Helps find root causes of all 104 unique inconsistencies using localization maps

● Inconsistencies are common and widespread.

● We call for more attention to testing of DL libraries.
DL system overview

High-level Libraries
- TensorFlow
- Theano
- CNTK

Low-level Libraries
- User code
  - Keras

Hardware
- CPU
- GPU
Group unique inconsistency

- A group of inconsistencies with the same inconsistency pattern between the same pair of implementations
  - Inconsistency pattern is the distribution of metric distance

<table>
<thead>
<tr>
<th>Id</th>
<th>Keras</th>
<th>Backends</th>
<th>Model</th>
<th>16</th>
<th>15-8</th>
<th>7-4</th>
<th>3-2</th>
<th>1</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2.2.2</td>
<td>TF-CN</td>
<td>Xception</td>
<td>10</td>
<td>202</td>
<td>147</td>
<td>100</td>
<td>85</td>
<td>4456</td>
</tr>
<tr>
<td>2</td>
<td>2.2.2</td>
<td>TF-CN</td>
<td>NASNetLarge</td>
<td>5</td>
<td>132</td>
<td>86</td>
<td>77</td>
<td>65</td>
<td>4635</td>
</tr>
<tr>
<td>3</td>
<td>2.2.1</td>
<td>TF-CN</td>
<td>Xception</td>
<td>10</td>
<td>202</td>
<td>147</td>
<td>100</td>
<td>85</td>
<td>4456</td>
</tr>
<tr>
<td>4</td>
<td>2.2.1</td>
<td>TF-CN</td>
<td>NASNetLarge</td>
<td>5</td>
<td>132</td>
<td>86</td>
<td>77</td>
<td>65</td>
<td>4635</td>
</tr>
</tbody>
</table>
Suggested settings

● Grid search on $T_C$, $T_M$, and $p$ values
● Optimal settings (most inconsistency without false negative and false positive) are:
  ○ CLASS-based: $T_C = 8$ and $p = 0\%$
  ○ MAD-based: $T_M = 0.2$ and $p = 0\%$
● Confirm using cross-validation
Dataset and hardware

- **Dataset:**
  - 11 datasets including ImageNet, MNIST, Udachi Driving Challenge 2, etc.
  - 30 pre-trained models

- **Hardware:**
  - Xeon E5-2695
  - NVIDIA Titan Xp
## Detected inconsistencies

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Instances</th>
<th># of Inconsistencies</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>TH-TF</td>
</tr>
<tr>
<td>ImageNet</td>
<td>5,000</td>
<td>10(34)</td>
</tr>
<tr>
<td>Driving</td>
<td>5,614</td>
<td>3(9)</td>
</tr>
<tr>
<td>MNIST</td>
<td>10,000</td>
<td>3(9)</td>
</tr>
<tr>
<td>Thai MNIST</td>
<td>1,665</td>
<td>1(3)</td>
</tr>
<tr>
<td>KGS Go game</td>
<td>12,288</td>
<td>2(14)</td>
</tr>
<tr>
<td>Anime Faces</td>
<td>14,490</td>
<td>1(5)</td>
</tr>
<tr>
<td>Dogs VS Cats</td>
<td>832</td>
<td></td>
</tr>
<tr>
<td>Dog species</td>
<td>835</td>
<td></td>
</tr>
<tr>
<td>Faces</td>
<td>466</td>
<td>2(14)</td>
</tr>
<tr>
<td>Pokedex</td>
<td>1,300</td>
<td>1(14)</td>
</tr>
<tr>
<td>GTSRB sign</td>
<td>12,630</td>
<td>2(14)</td>
</tr>
<tr>
<td><strong>Total</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>18(95)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The numbers outside and (inside) brackets are the unique and (total) number of inconsistencies respectively.
Comparison to accuracy

- Detect inconsistency if the top-k accuracy difference is above a threshold $T_{\text{AC}}$
- We pick $k$ between 1 to 5 and $T_{\text{AC}}$ between 0 and 50
- With $T_{\text{AC}} = 0$, top-1 accuracy detects the most inconsistencies (305) but still missed 35
  - E.g., for the *Dog species* model, the *Batch_normalization* bugs induce inconsistency between TensorFlow and CNTK
  - However, those backends got identical top-1 (29.9%) and top-5 (64.4%) accuracies
Future work

● Detect inconsistencies and bugs in training code
  ○ Harder since training is non-deterministic
● Generate mutated models using fuzzing to expand testing set
● Testing with only one backend with equivalent models