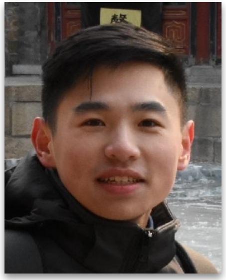
CRADLE: Cross-Backend Validation to Detect and Localize Bugs in Deep Learning Libraries

Hung Viet Pham[1]

Thibaud Lutellier[1]



Weizhen Qi[2]

Lin Tan[3]

[1]University of Waterloo, Canada
[2]University of Science and Technology of China, China
[3]Purdue University, USA

Deep learning (DL) is pervasive

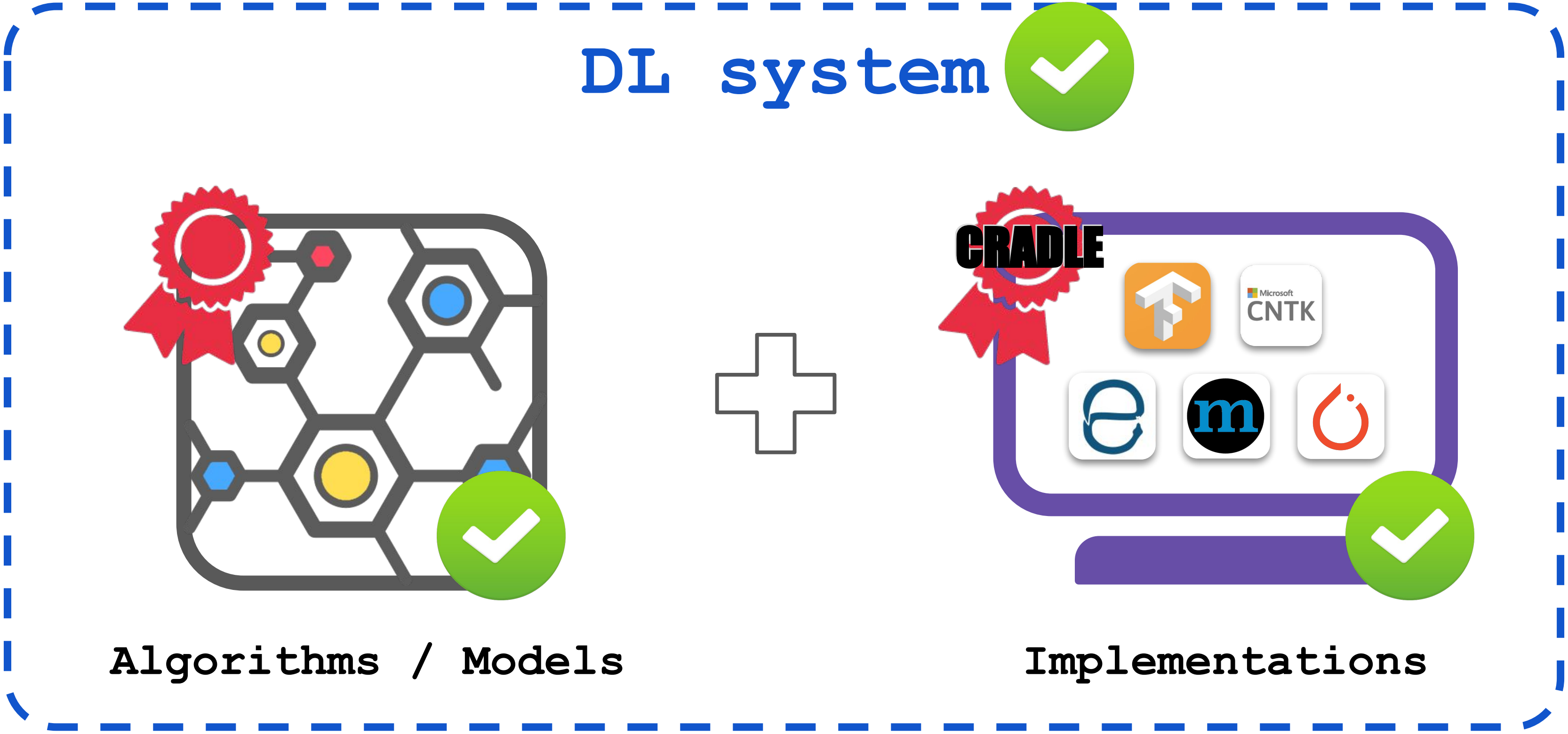
Machine translation

Alzheimer’s disease diagnosis

Autonomous driving cars

Virtual assistance

Correct DL systems require correct implementations



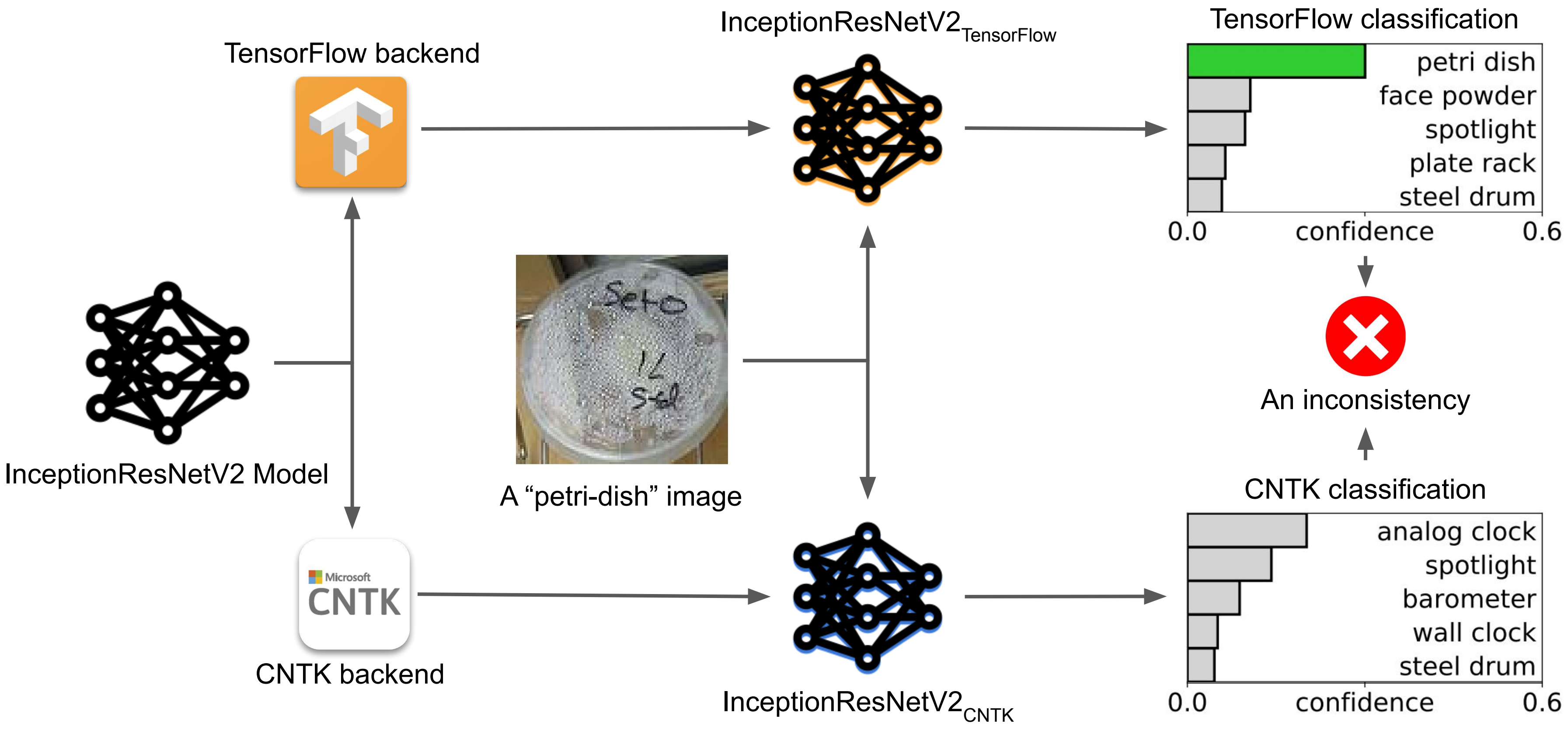
DL libraries are hard to test and debug

- 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
Expected output: tennis ball

***MobileNetV2* - TensorFlow:** banana
Ground-truth: banana

Idea: Differential testing



`Batch_normalization` bug

- The CNTK batch normalization formula was implemented incorrectly.
- The developers fixed the bug after we reported it.

```diff
-    return(x-mean)/(C.sqrt(var)+epsilon)*gamma+beta
+    return(x-mean)/ C.sqrt(var +epsilon)*gamma+beta
```

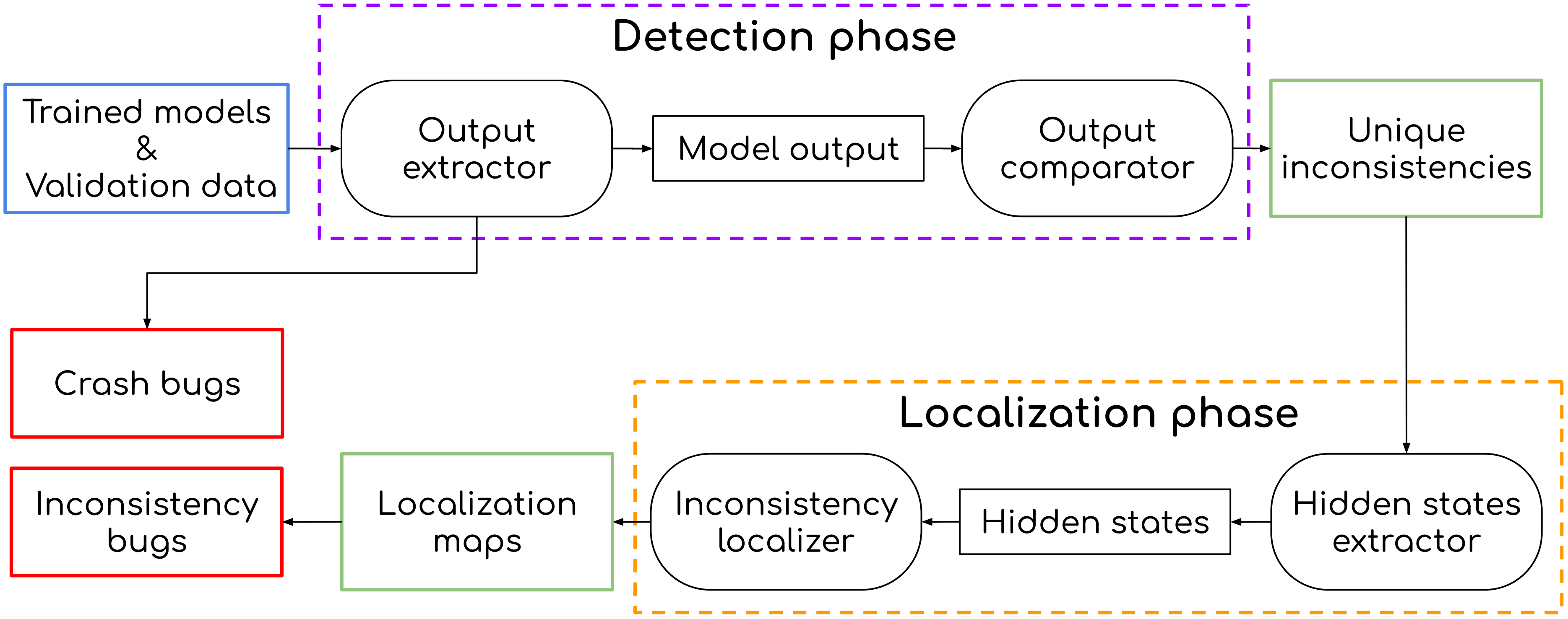
Differential testing: Challenges

- 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?

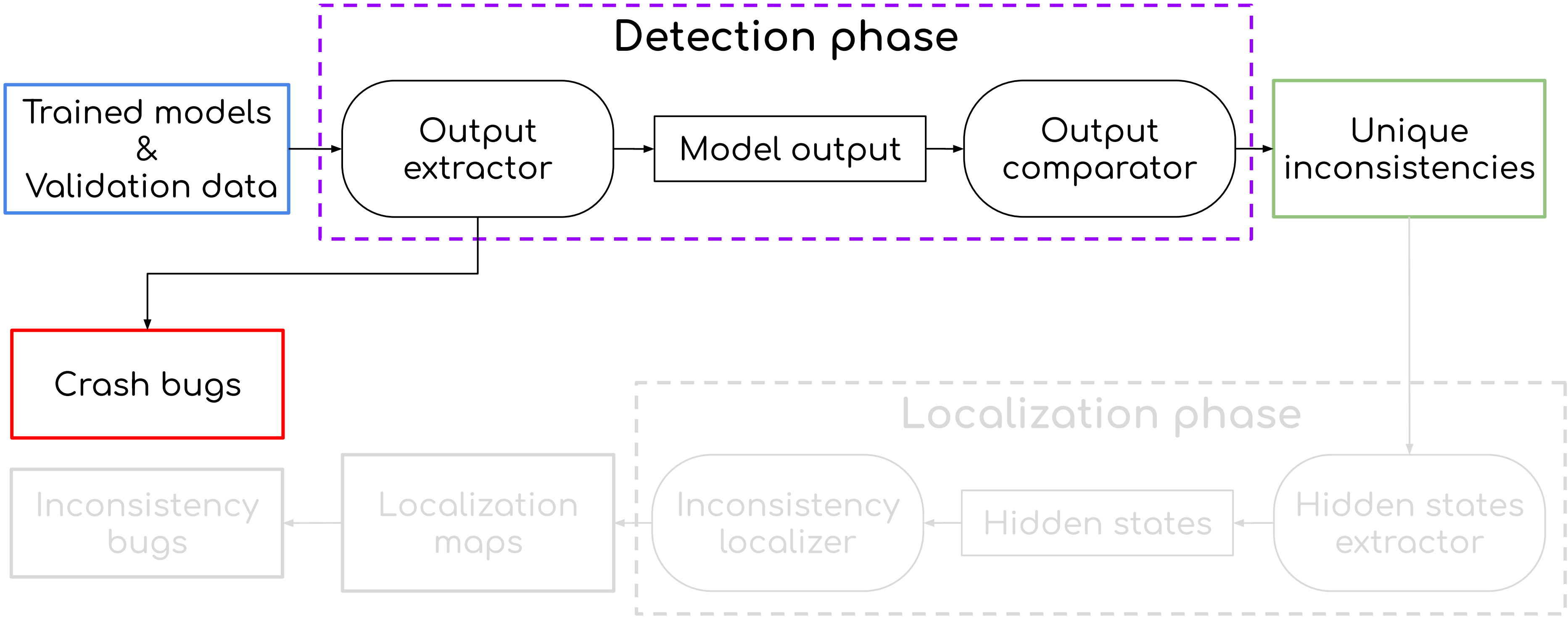
Differential testing: Ideas

- 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

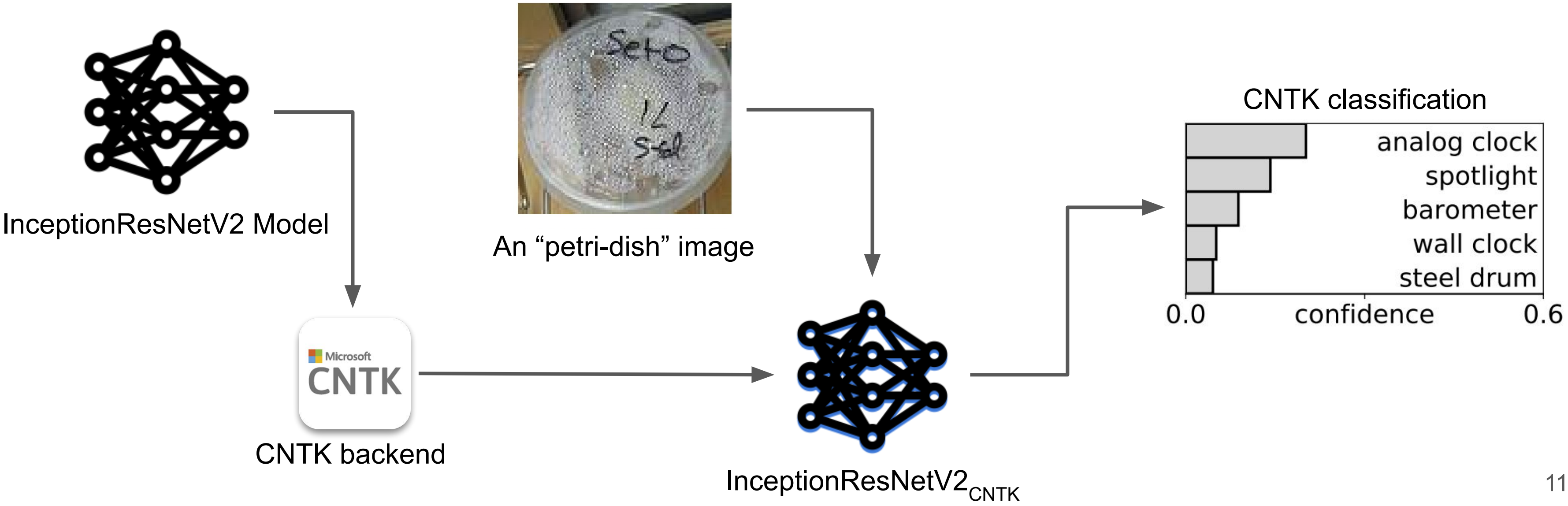


CRADLE: Detection phase



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)

$$\sigma_{C,Y} = \begin{cases} 2^{k-rank_{C,Y}} & \text{if } rank_{C,Y} \leq k \\ 0 & \text{otherwise} \end{cases}$$

$$\text{D\_CLASS}_{C,Y,Y'} = |\sigma_{C,Y} - \sigma_{C,Y'}|$$

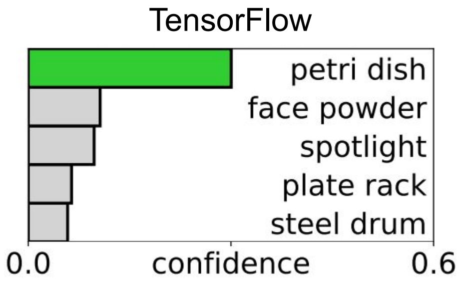
MAD-based (Regression)

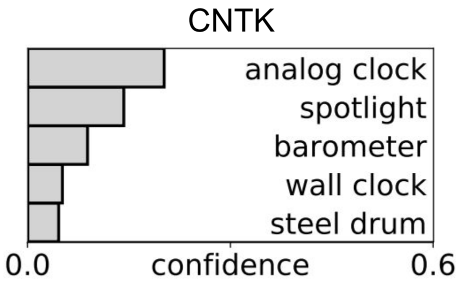
$$\delta_{O,Y} = \frac{1}{N} \sum_{i=1}^{N} |Y_i - O_i|$$

$$\text{D\_MAD}_{O,Y,Y'} = \frac{|\delta_{O,Y} - \delta_{O,Y'}|}{\delta_{O,Y} + \delta_{O,Y'}}$$

CLASS-based distance example

Top-5 classification





$$\sigma_{C,Y} = \begin{cases} 2^{k - rank_{C,Y}} & \text{if } rank_{C,Y} \leq k \\ 0 & \text{otherwise} \end{cases}$$

$Rank_{\text{petri-dish,TF}} = 1$

$Rank_{\text{petri-dish,CN}} > 5$

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

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

$$\text{D\_CLASS}_{C,Y,Y'} = |\sigma_{C,Y} - \sigma_{C,Y'}|$$

$|\sigma_{\text{petri-dish,CN}} - \sigma_{\text{petri-dish,CN}}| = 16$

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.

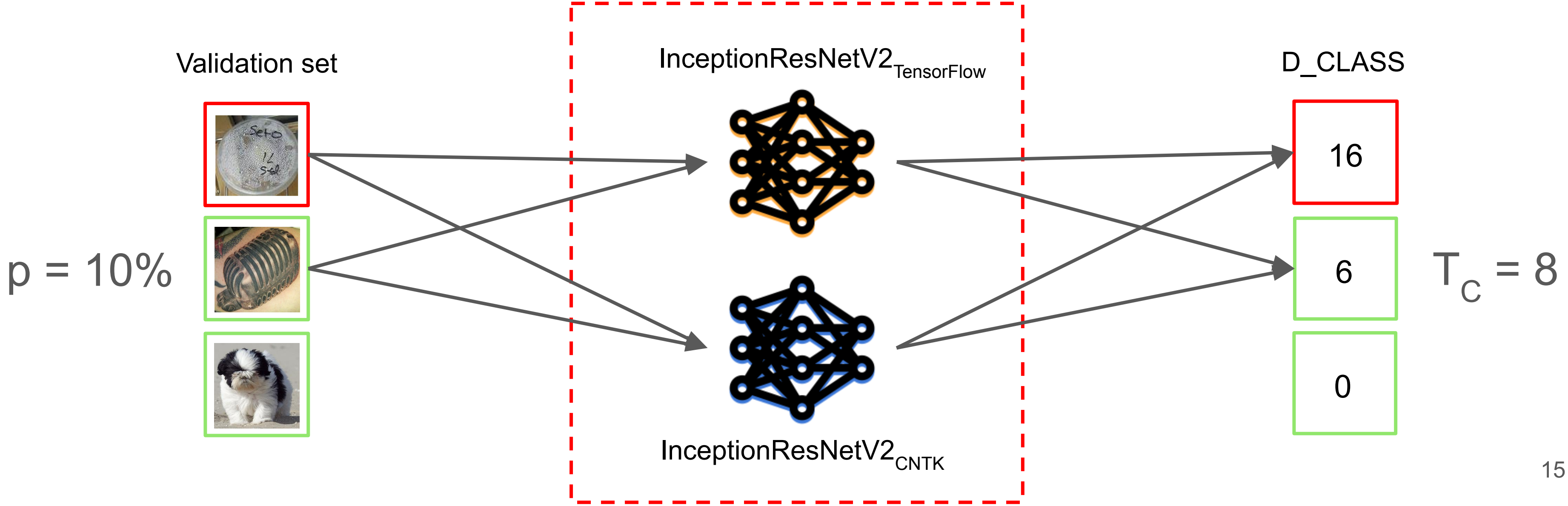
Theano: Indian elephant
TensorFlow: groom
CNTK: groom

TensorFlow: banana
CNTK: tennis ball
Theano: tennis ball

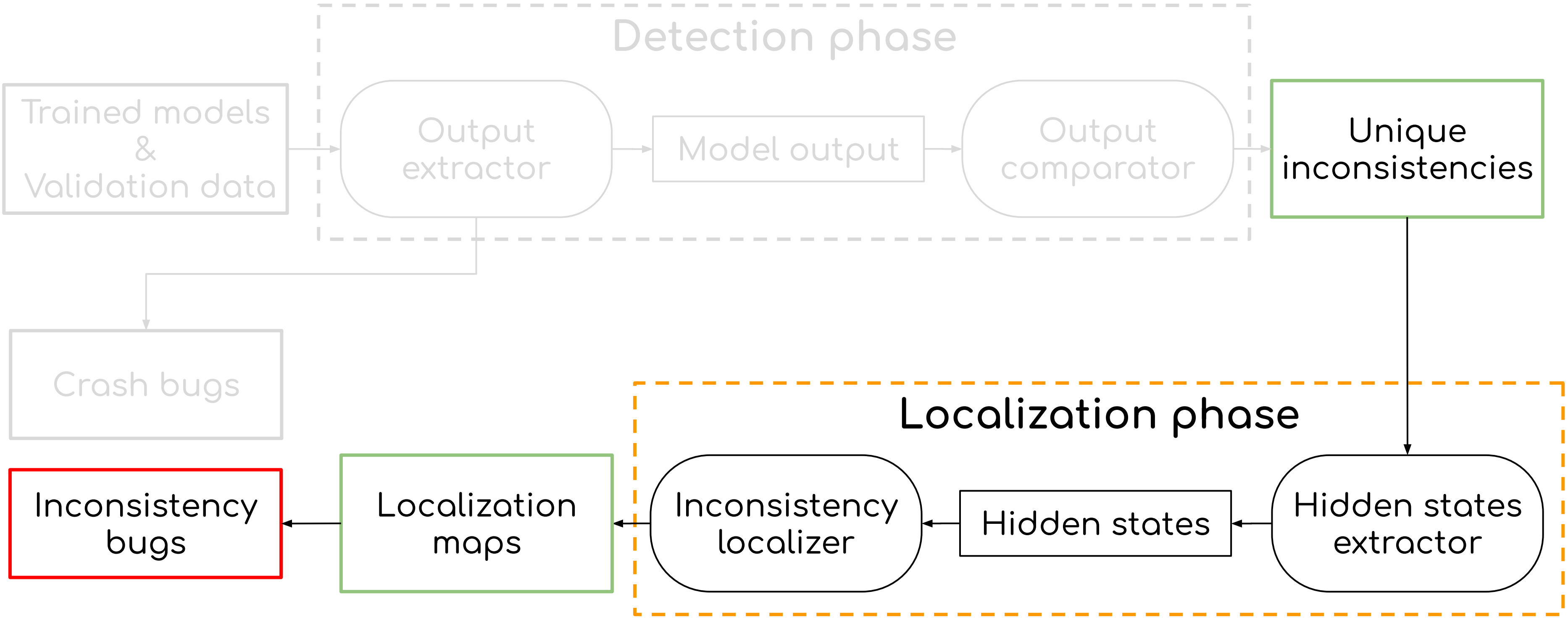
CNTK: Arabian camel
TensorFlow: hen
Theano: hen

Detect inconsistency

- An inconsistency is a pair of implementations that triggers more than *p*% of ITIs over the validation set

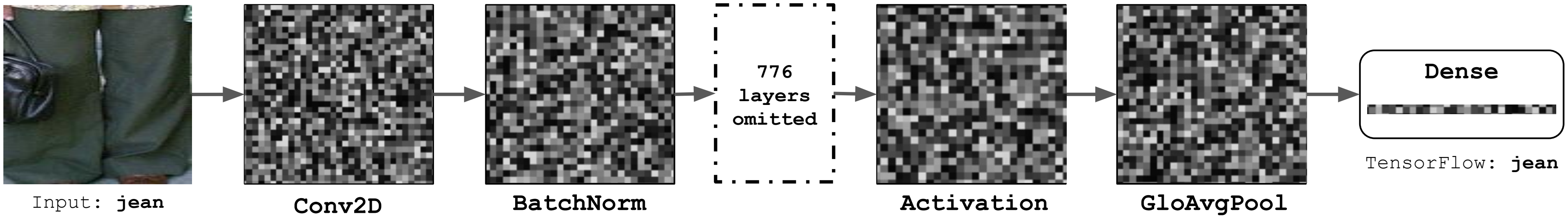


CRADLE: Localization phase



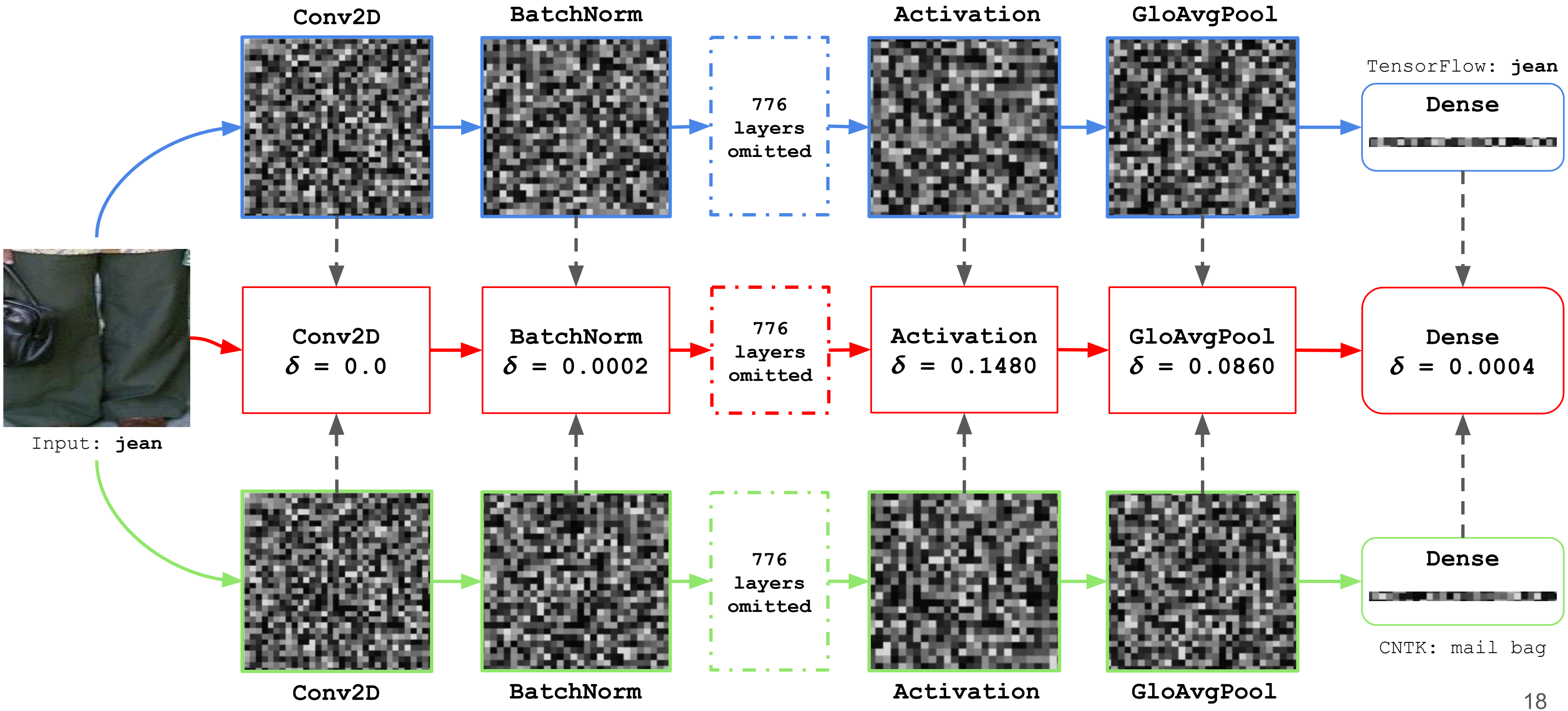
Hidden state extractor

- 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.



InceptionResNetV2 execution graph on TensorFlow

MAD differences

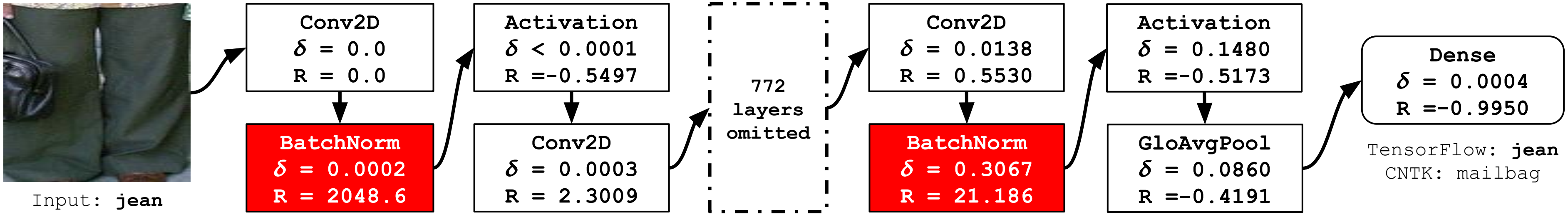


Inconsistency introduction rate

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

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

$$\delta_{pre} = \max_{l \in pre(L)} (\delta_{S_l, S'_l})$$



InceptionResNetV2 localization map between TensorFlow and CNTK

Result

104 unique inconsistencies

3 backends

28 models 11 datasets

7 inconsistency bugs 5 crash bugs

7 inconsistency bugs

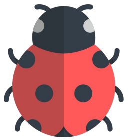
Batch normalization	BatchNormalization
Padding scheme	Conv2D variant
Pooling scheme	AveragePooling2D
Parameter organization	Trainable Conv

Localization is helpful

Relevant to the causes of all 104 unique inconsistencies

First

One of

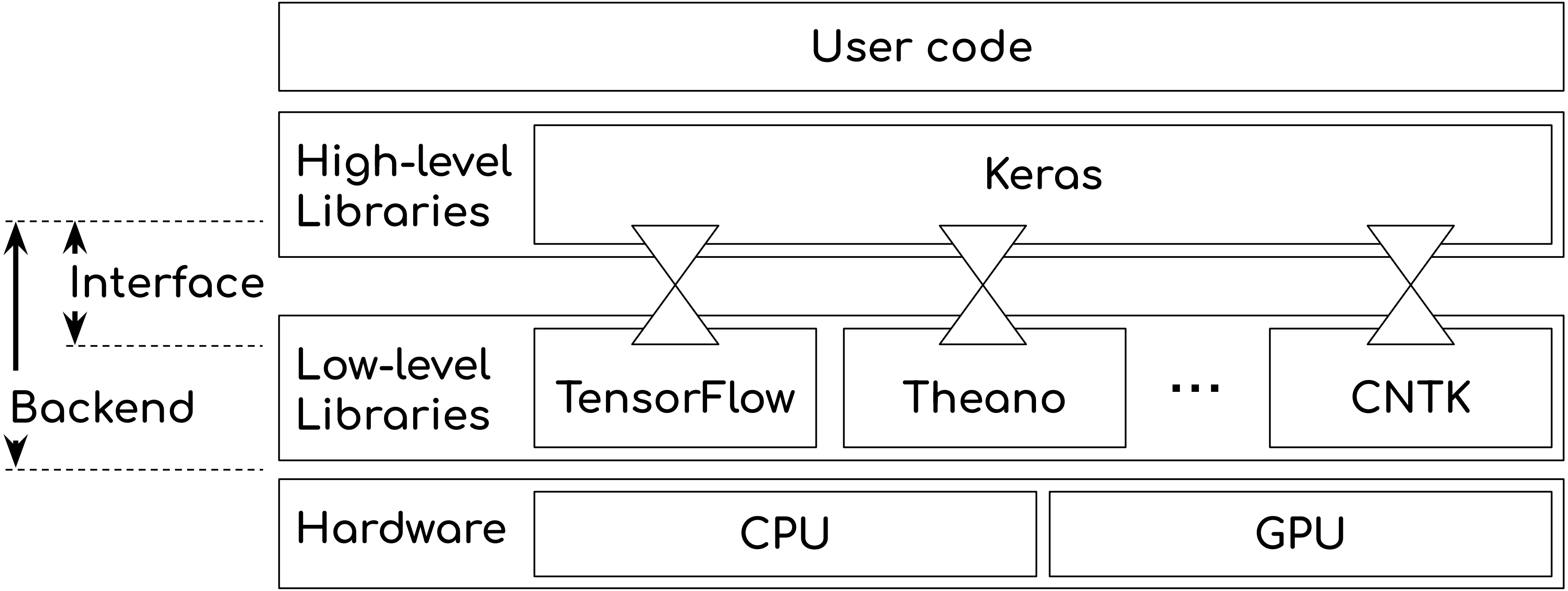


Relevant

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



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

				Inconsistency pattern					
Id	**Keras**	**Backends**	**Model**	16	15-8	7-4	3-2	1	0
1	2.2.2	TF-CN	Xception	10	202	147	100	85	4456
2	2.2.2	TF-CN	NASNetLarge	5	132	86	77	65	4635
3	2.2.1	TF-CN	Xception	10	202	147	100	85	4456
4	2.2.1	TF-CN	NASNetLarge	5	132	86	77	65	4635

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

Dataset	Instances	# of Inconsistencies		
		TH-TF	TF-CN	CN-TH
ImageNet	5,000	10(34)	21(54)	18(46)
Driving	5,614		3(9)	3(12)
MNIST	10,000		3(9)	3(12)
Thai MNIST	1,665		1(3)	1(4)
KGS Go game	12,288	2(14)	3(12)	3(15)
Anime Faces	14,490	1(5)		1(6)
Dogs VS Cats	832		2(6)	2(8)
Dog species	835		3(8)	3(9)
Faces	466	2(14)	3(8)	6(15)
Pokedex	1,300	1(14)	1(3)	2(15)
GTSRB sign	12,630	2(14)	2(5)	2(7)
Total		18(95)	42(117)	44(149)
		104(361)		

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_{AC}
- We pick k between 1 to 5 and T_{AC} between 0 and 50
- With $T_{AC} = 0$, 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