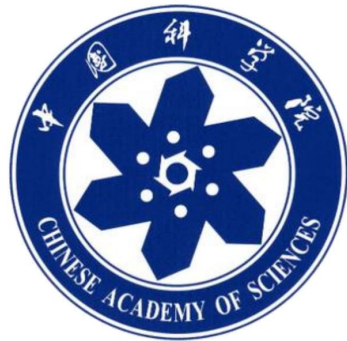


Detecting TensorFlow Program Bugs in Real-World Industrial Environment

Chen Liu, Jie Lu, Guangwei Li, Ting Yuan, Lian Li, Feng Tan,
Jun Yang, Liang You, Jingling Xue



Institute of Computing
Technology of the Chinese
Academy of Science



University of Chinese
Academy of Sciences

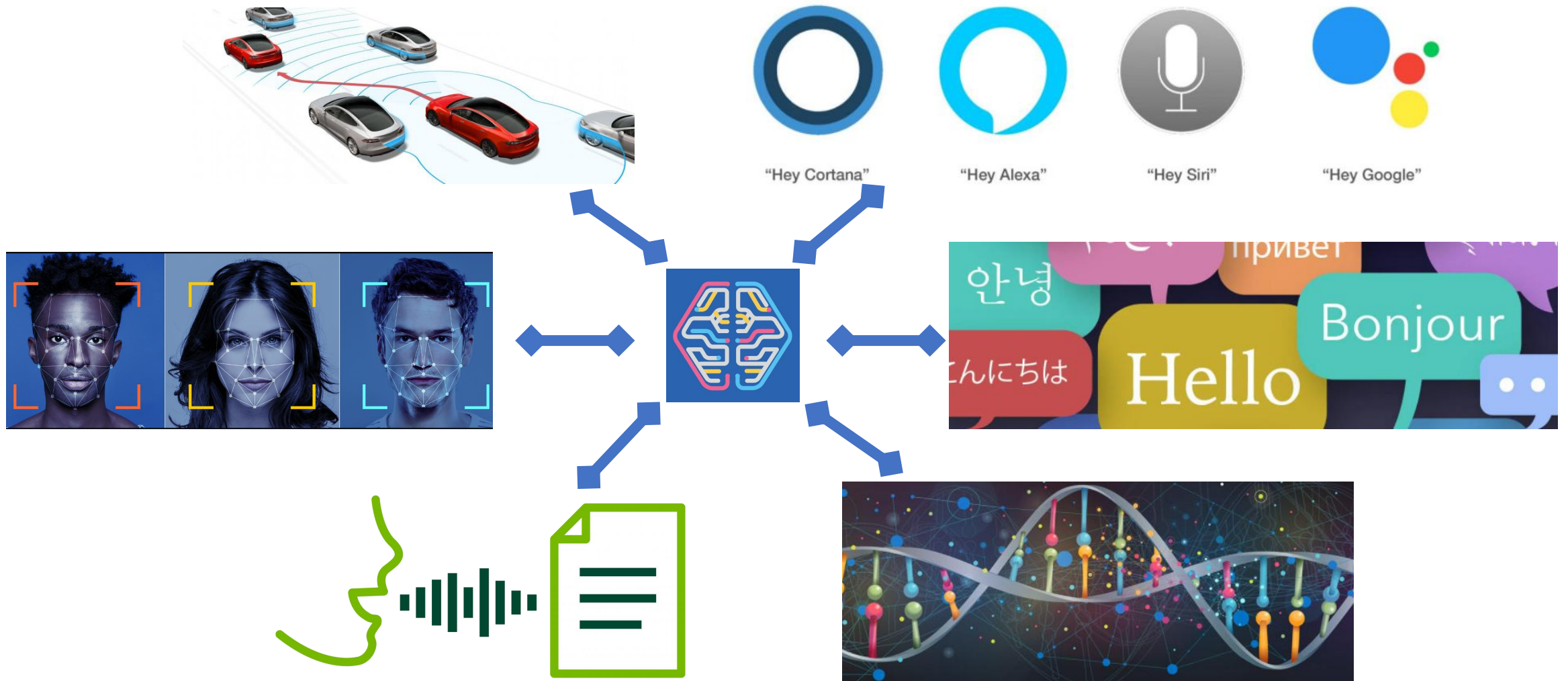


Alibaba Group

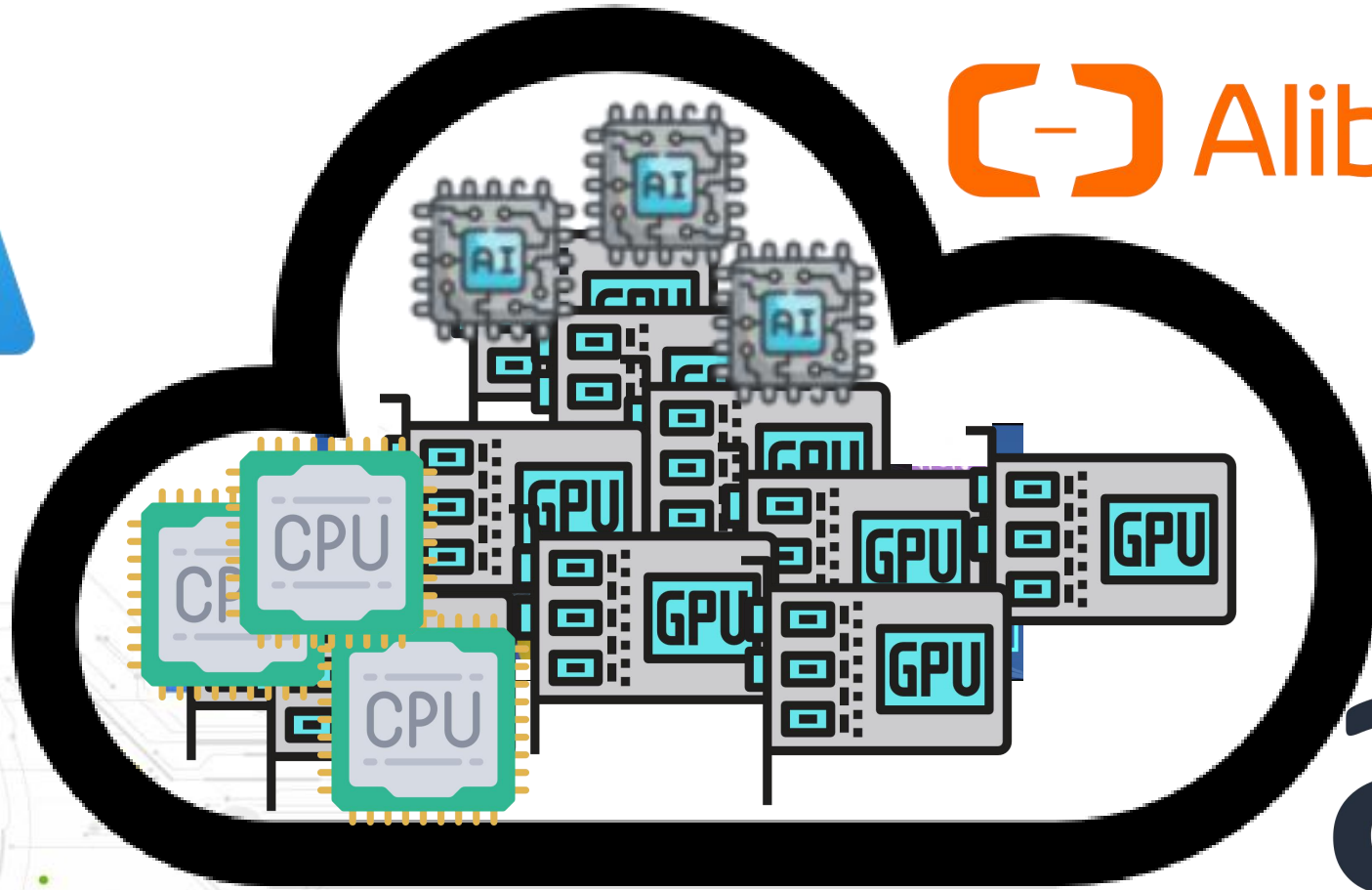


University of New South Wales

Rise of deep learning

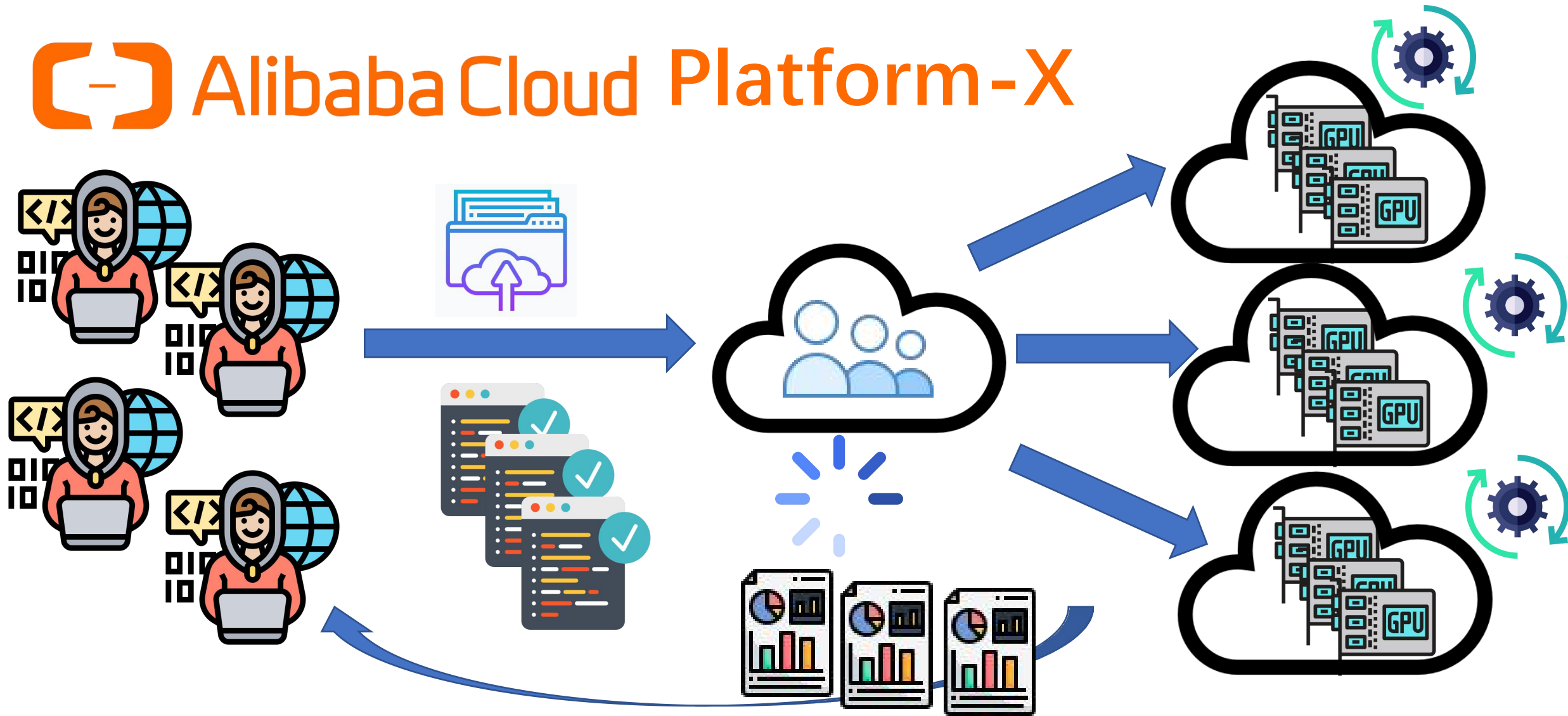


Cloud Deep Learning Platform



Cloud Deep Learning Platform

 Alibaba Cloud Platform-X



10% Failure jobs on Platform-X



10% Failure jobs on Platform-X

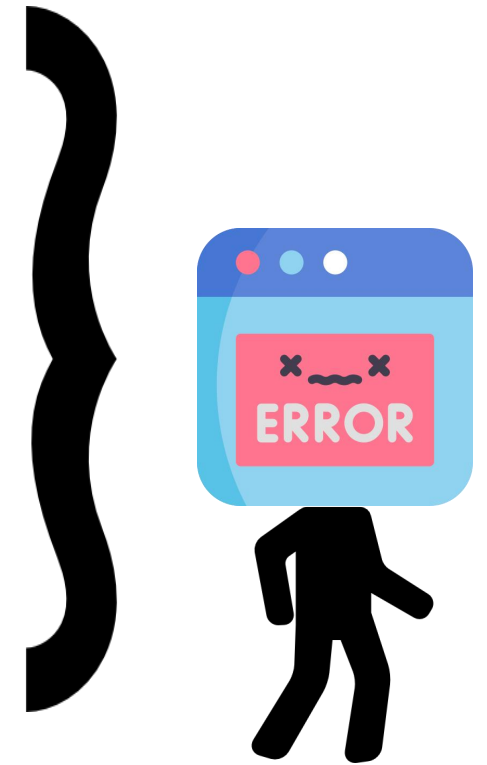
```
ing simple_console_for_windows.zip to create runfiles tree...  
el-bin/tensorflow/tools/pip_package/simple_console_for_windows.zip]  
of-central-directory signature not found. Either this file is not
```

```
Installing collected packages: setuptools, protobuf, wheel, numpy, tensorflow  
Found existing installation: setuptools 1.1.6  
Uninstalling setuptools-1.1.6:  
Exception:
```

```
...  
[Errno] Error importing tensorflow. Unless you are using bazel, you should  
'/tmp/  
,\fwb  
[Ellipsis] tensorflow source tree, and relaunch your python interpreter from  
the
```

```
ImportError: libcudart.so.Version: cannot open shared object file:  
No such file or directory
```

```
NO SUCH FILE OR DIRECTORY
```



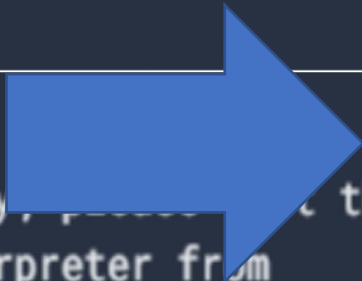
10% Failure jobs on Platform-X

```
te runfiles tree...  
_console_for_windows.zip]  
Either this file is not
```

```
tools, protobuf, wheel, numpy, tensorflow  
ols 1.1.6
```

```
ess you are using bazel,  
from its source directory, press <C> the  
relaunch your python interpreter from
```

```
n: cannot open shared object file:
```



Example Error Message Patterns

<><*> has no attribute <*>*

No module named <>*

<> takes exactly <*> arguments <*> given
<*> got an unexpected keyword argument <*>*

name <> is not defined*

local variable <> referenced before assignment*

968 Patterns

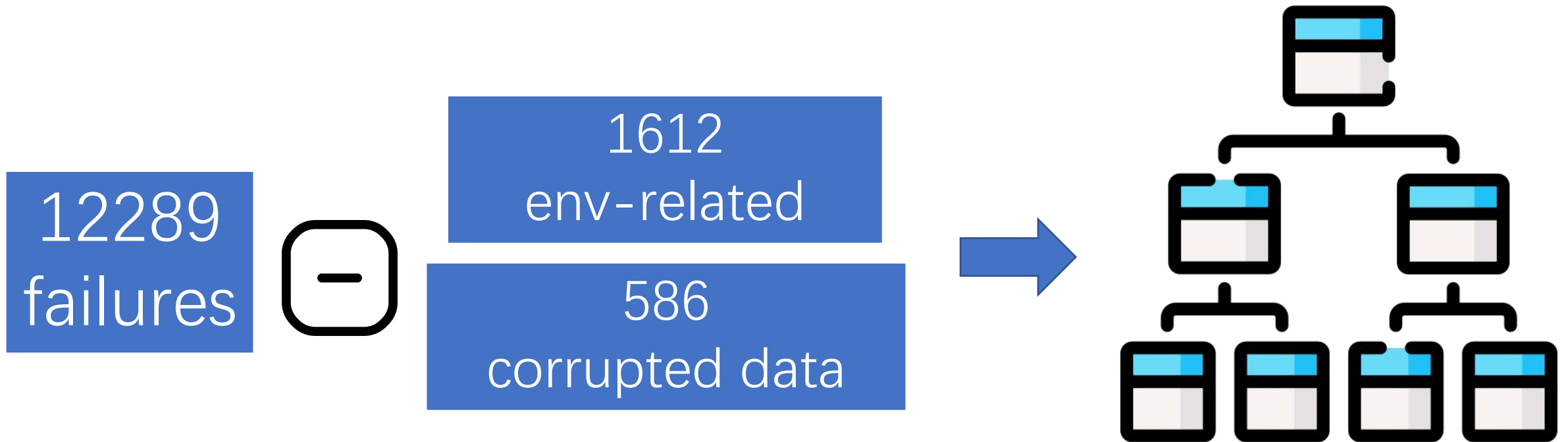


968 Patterns

12289
failures

```
name <*> is not defined
Shape must be rank <*> but is rank <*> for...
<*> takes at <*> <*> <*> <*> given)
Incompatible shapes: <*> vs. <*>
Attempting to use uninitialized value <*>
  setting an array element with a <*>
  slice index <*> of dimension <*> out of...
End of sequence
Fetch argument None has invalid type <type...
could not convert string to float:
  object of type <*> has no len()
Cannot assign a device for operation <*>...
expected string or Unicode object, <*> found
unindent does not match any outer indentation...
  Values of eval_metric.ops must be...
  Passed (<tf.Tensor <*> shape=() <*>...
    float argument required, not <*>
    Found no files at <*>
only integer scalar arrays can be converted to ...
  Found input variables with inconsistent...
  HashTable has different value for same key: ...
req_id: <*> http status code: 500, error code:...
  data type not understood
  <*> in record <*> is not valid : <*>
Expected binary or unicode string, got <*>
<*> object does not support indexing
  'scalar/auc'
model_fn (<function <*> at <*> must include...
[Errno 28] No space left on device
Shape must be at least rank <*> but is rank...
  Cannot colocate nodes...
  unsupported pickle protocol: 3
Shape must be at least rank 2 but is rank 1 for...
  <*> for replication success
Determined shape must either match input...
{'status': 500, 'request-id': <*> 'details': ...
  'click_weight'
  'use_vae'
The values set to records are against the...
The 'kernel_size' argument must be a tuple of ...
Error!
dictionary changed size during iteration
"The name <*> refers to an Operation not in...
  'doc_embedding_size'
  <*> corrupted
Call dropped by load balancing policy
  empty range for randrange() (0,0, 0)
u'single_char_problem never registered with...
assertion failed: [weights can not be broadcast...
  must be string or buffer, not int
  map() requires at least two args
  'dense_data'
exceptions.UnicodeEncodeError: 'ascii' codec...
Tried to convert 'multiples' to a tensor and...
ComplexNetwork input "label" doesn't exist,...
Failed to convert object of type <type 'dict'> t...
  'multicats_reg'
  'task_index'
  file not exists in...
__long__ returned non-long (type NoneType)
  num_heads
  '2day_total_region5'
Batch size 2 needs to be divisible by the...
'\xe5 \xb0 \xe5 \xb0 \xb1 \xe6 \x98 \xaf...'
isinstance() arg 2 must be a class, type, or tuple...
The outermost dimension of updates and...
  Blas xGEMVBatched launch failed :...
  Fetch argument array[(<tf.Tensor...
    save
    'inputs_ch_x'
The two structures don't have the same...
Replacement not allowed with overlapping...
AttrValue must not have reference type value...
  '\xe5\x8d\xa7\xe5\xae\xa4'
cuDNN launch failure : input shape...
```

968 Patterns



Bug types

- Checkpoint Error
- Shape Error
- Out of Memory
- Loss NaN
- GPU Sync Failed

Tensorflow-Specific Bugs

- Module/Attribute Missing
- Arguments Mismatch
- Undefined Variable
- Key Not Found
- ...

Python Bugs

Bug types

- Checkpoint Error
- Shape Error
- Out of Memory
- Loss NaN
- GPU Sync Failed

Tensorflow-Specific Bugs

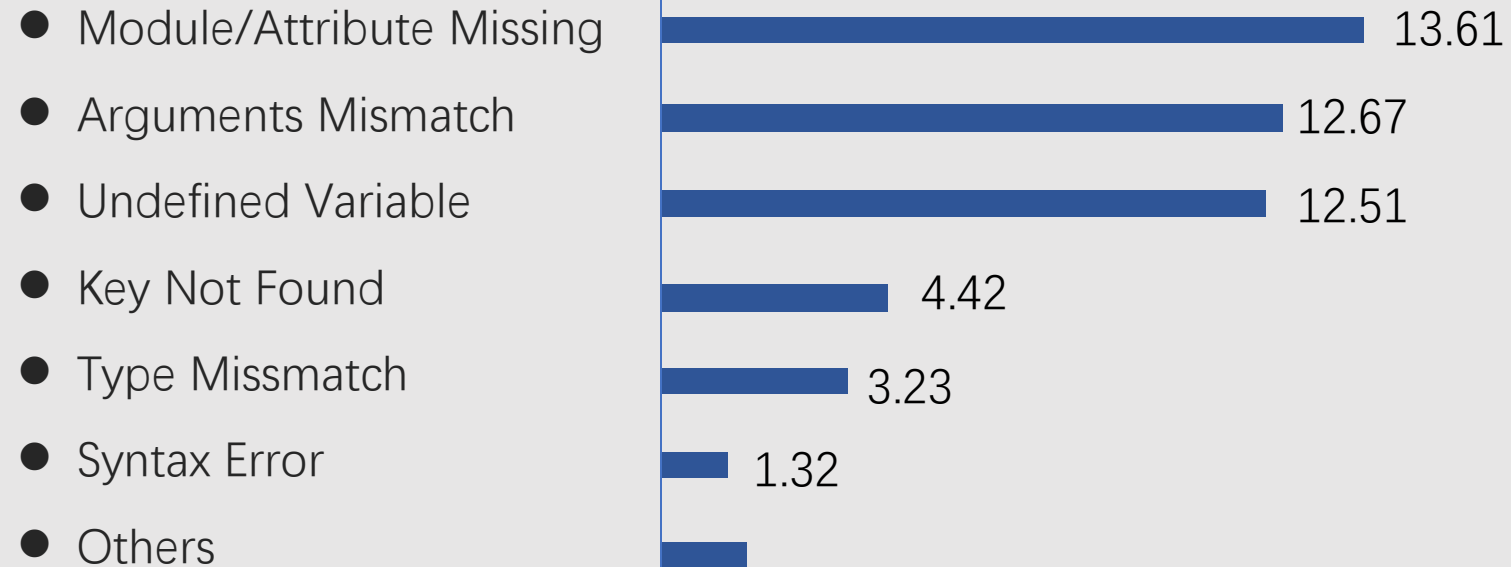
36.31%

- Module/Attribute Missing
- Arguments Mismatch
- Undefined Variable
- Key Not Found
- ...

Python Bugs

63.69%

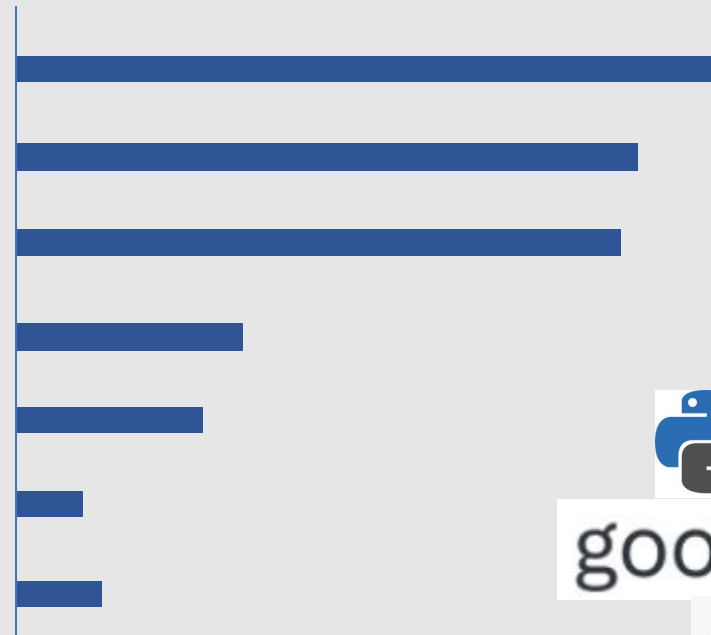
Bug types



Python Bugs

Bug types

- Module/Attribute Missing
- Arguments Mismatch
- Undefined Variable
- Key Not Found
- Type Mismatch
- Syntax Error
- Others



72.25%

: **my[py]**

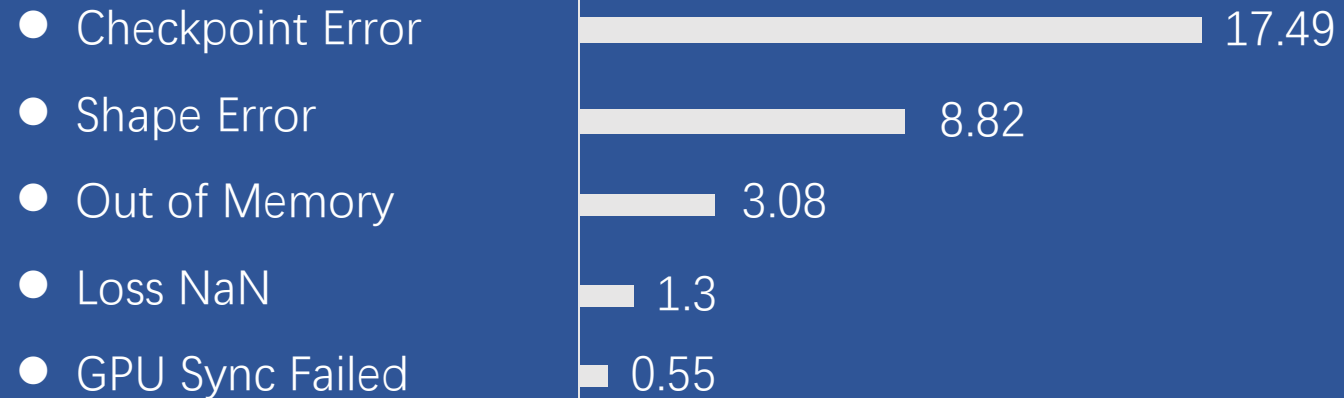
google/**pytype**

 **Pylint**
★★★★★ Star your Python code!

PyCQA/**pyflakes**

Python Bugs

Bug types



Tensorflow-Specific Bugs

36.31%

Bug types

- Checkpoint Error

- Shape Error

- Out of Memory

- Loss NaN

- GPU Sync Failed

Tensorflow-Specific Bugs

36.31%

A simplified shape error example

Construction

```
1. def fully_connect(input_op, name, n_in,n_out ):
2.     fc_w = tf.get_variable(name, [n_in, n_out])
3.     return tf.matmul(input_op, fc_w)
4. def predict(Input_x, class_num):
5.     mp = tf.nn.conv2d(input_x,tf.get_variable('mpc',[5,5,1,32]),strides=[1,1,1,1], padding='SAME')
6.     reshaped = tf.reshape(mp, [-1, 28 * 28])
7.     fc = fully_connect(reshaped, 'fc1', 28 * 28, 128)
8.     logit = fully_connect(fc, 'fc2', 128, class_num)
9.     return logit
10. in_x = tf.placeholder(tf.float32, shape = [None, 28, 28, 1])
11. in_y = tf.placeholder(tf.float32, shape = [None, 10])
12. y = predict(in_x, 10)
13. cross_entropy = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(labels=in_y,logits =y))
14. train_step = tf.train.AdamOptimizer(1e-4).minimize(cross_entropy)
```

Execution

```
15. train_img, train_lab = read_image(batch_size,...)
16. with tf.Session() as sess:
17.     for i in range(1000):
18.         sess.run(train_step, feed_dict = in_x:train_img, in_y:train_lab))
```

A simplified shape error example

Construction

```
1. def fully_connect(input_op, name, n_in,n_out ):
2.     fc_w = tf.get_variable(name, [n_in, n_out])
3.     return tf.matmul(input_op, fc_w)
4. def predict(Input_x, class_num):
5.     mp = tf.nn.conv2d(input_x,tf.get_variable('mpc',[5,5,1,32]),strides=[1,1,1,1], padding='SAME')
6.     reshaped = tf.reshape(mp, [-1, 28 * 28])
7.     fc = fully_connect(reshaped, 'fc1', 28 * 28, 128)
8.     logit = fully_connect(fc, 'fc2', 128, class_num)
9.     return logit
10. in_x = tf.placeholder(tf.float32, shape = [None, 28, 28, 1])
11. in_y = tf.placeholder(tf.float32, shape = [None, 10])
12. y = predict(in_x, 10)
13. cross_entropy = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(labels=in_y,logits =y))
14. train_step = tf.train.AdamOptimizer(1e-4).minimize(cross_entropy)
```

Execution

```
15. train_img, train_lab = read_image(batch_size,...)
16. with tf.Session() as sess:
17.     for i in range(1000):
18.         sess.run(train_step, feed_dict = in_x:train_img, in_y:train_lab))
```

A simplified shape error example

Construction

```
1. def fully_connect(input_op, name, n_in,n_out ):
2.     fc_w = tf.get_variable(name, [n_in, n_out])
3.     return tf.matmul(input_op, fc_w)
4. def predict(Input_x, class_num):
5.     mp = tf.nn.conv2d(input_x,tf.get_variable('mpc',[5,5,1,32]),strides=[1,1,1,1], padding='SAME')
6.     reshaped = tf.reshape(mp, [-1, 28 * 28])
7.     fc = fully_connect(reshaped, 'fc1', 28 * 28, 128)
8.     logit = fully_connect(fc, 'fc2', 128, class_num)
9.     return logit
10. in_x = tf.placeholder(tf.float32, shape = [None, 28, 28, 1])
11. in_y = tf.placeholder(tf.float32, shape = [None, 10])
12. y = predict(in_x, 10)
13. cross_entropy = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(labels=in_y,logits =y))
14. train_step = tf.train.AdamOptimizer(1e-4).minimize(cross_entropy)
```

Execution

```
15. train_img, train_lab = read_image(batch_size,...)
16. with tf.Session() as sess:
17.     for i in range(1000):
18.         sess.run(train_step, feed_dict = in_x:train_img, in_y:train_lab))
```

A simplified shape error example

Construction

```
1. def fully_connect(input_op, name, n_in,n_out ):
2.     fc_w = tf.get_variable(name, [n_in, n_out])
3.     return tf.matmul(input_op, fc_w)
4. def predict(Input_x, class_num):
5.     mp = tf.nn.conv2d(input_x,tf.get_variable('mpc',[5,5,1,32]),strides=[1,1,1,1], padding='SAME')
6.     reshaped = tf.reshape(mp, [-1, 28 * 28])
7.     fc = fully_connect(reshaped, 'fc1', 28 * 28, 128)
8.     logit = fully_connect(fc, 'fc2', 128, class_num)
9.     return logit
10. in_x = tf.placeholder(tf.float32, shape = [None, 28, 28, 1])
11. in_y = tf.placeholder(tf.float32, shape = [None, 10])
12. y = predict(in_x, 10)
13. cross_entropy = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(labels=in_y,logits =y))
14. train_step = tf.train.AdamOptimizer(1e-4).minimize(cross_entropy)
```

Execution

```
15. train_img, train_lab = read_image(batch_size,...)
16. with tf.Session() as sess:
17.     for i in range(1000):
18.         sess.run(train_step, feed_dict = in_x:train_img, in_y:train_lab))
```

A simplified shape error example

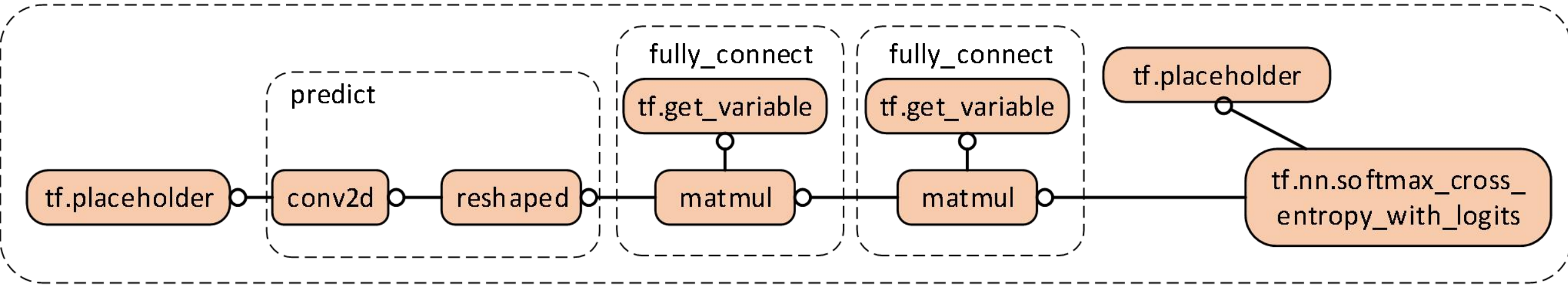
Construction

```
1. def fully_connect(input_op, name, n_in,n_out ):
2.     fc_w = tf.get_variable(name, [n_in, n_out])
3.     return tf.matmul(input_op, fc_w)
4. def predict(Input_x, class_num):
5.     mp = tf.nn.conv2d(input_x,tf.get_variable('mpc',[5,5,1,32]),strides=[1,1,1,1], padding='SAME')
6.     reshaped = tf.reshape(mp, [-1, 28 * 28])
7.     fc = fully_connect(reshaped, 'fc1', 28 * 28, 128)
8.     logit = fully_connect(fc, 'fc2', 128, class_num)
9.     return logit
10. in_x = tf.placeholder(tf.float32, shape = [None, 28, 28, 1])
11. in_y = tf.placeholder(tf.float32, shape = [None, 10])
12. y = predict(in_x, 10)
13. cross_entropy = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(labels=in_y,logits =y))
14. train_step = tf.train.AdamOptimizer(1e-4).minimize(cross_entropy)
```

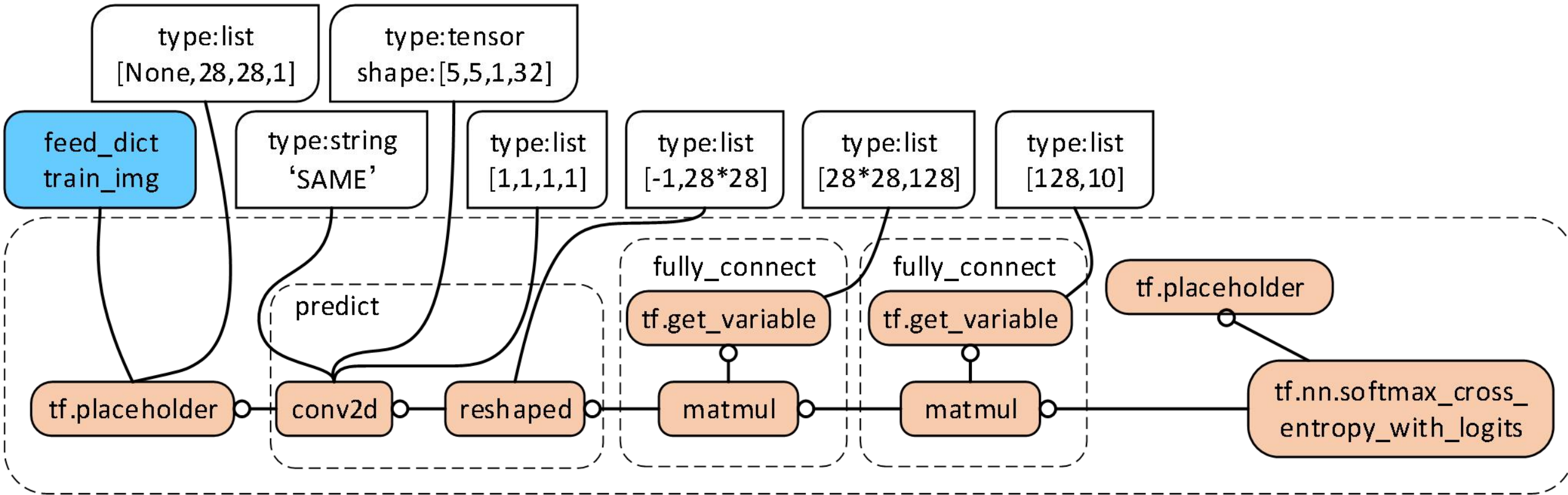
Execution

```
15. train_img, train_lab = read_image(batch_size,...)
16. with tf.Session() as sess:
17.     for i in range(1000):
18.         sess.run(train_step, feed_dict = in_x:train_img, in_y:train_lab))
```

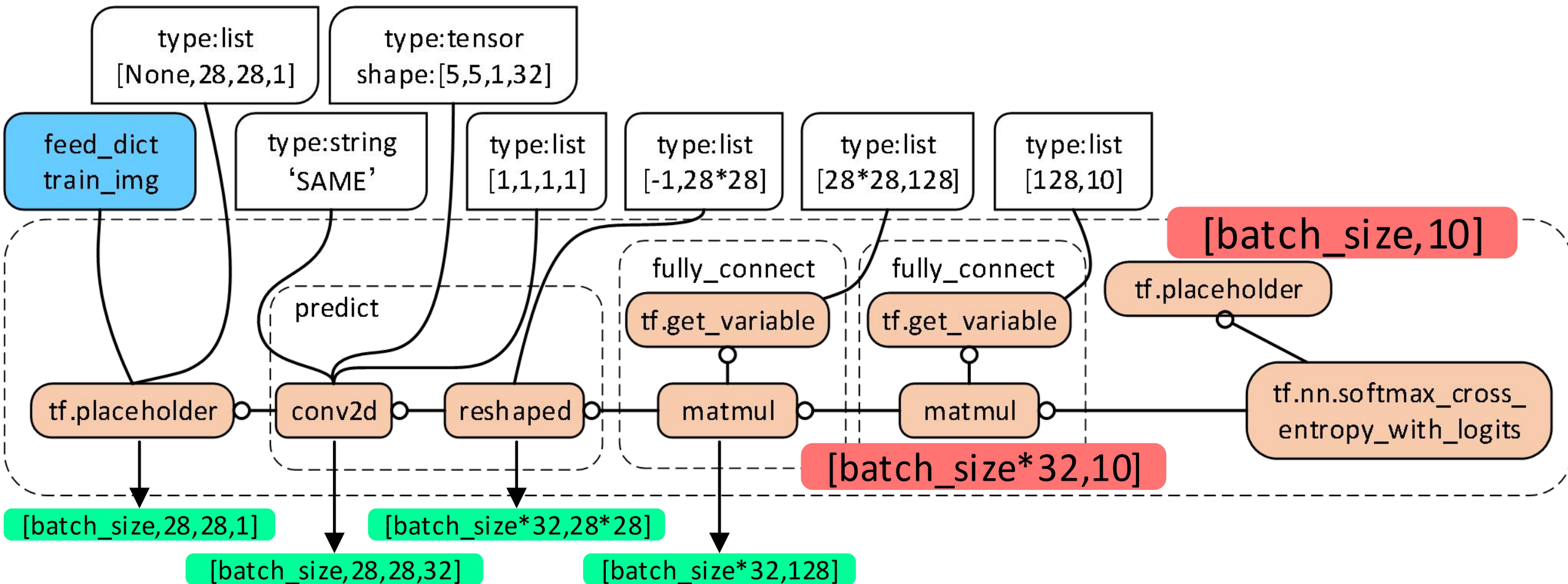
A simplified shape error example



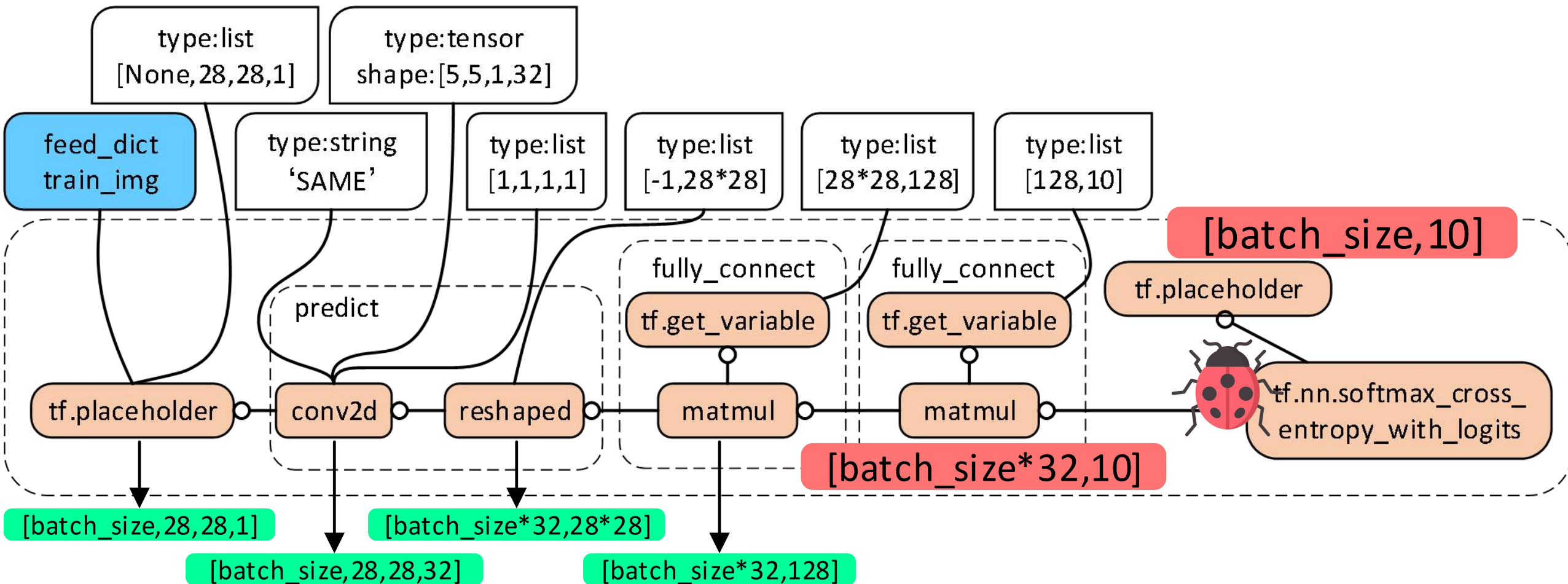
A simplified shape error example



A simplified shape error example



A simplified shape error example

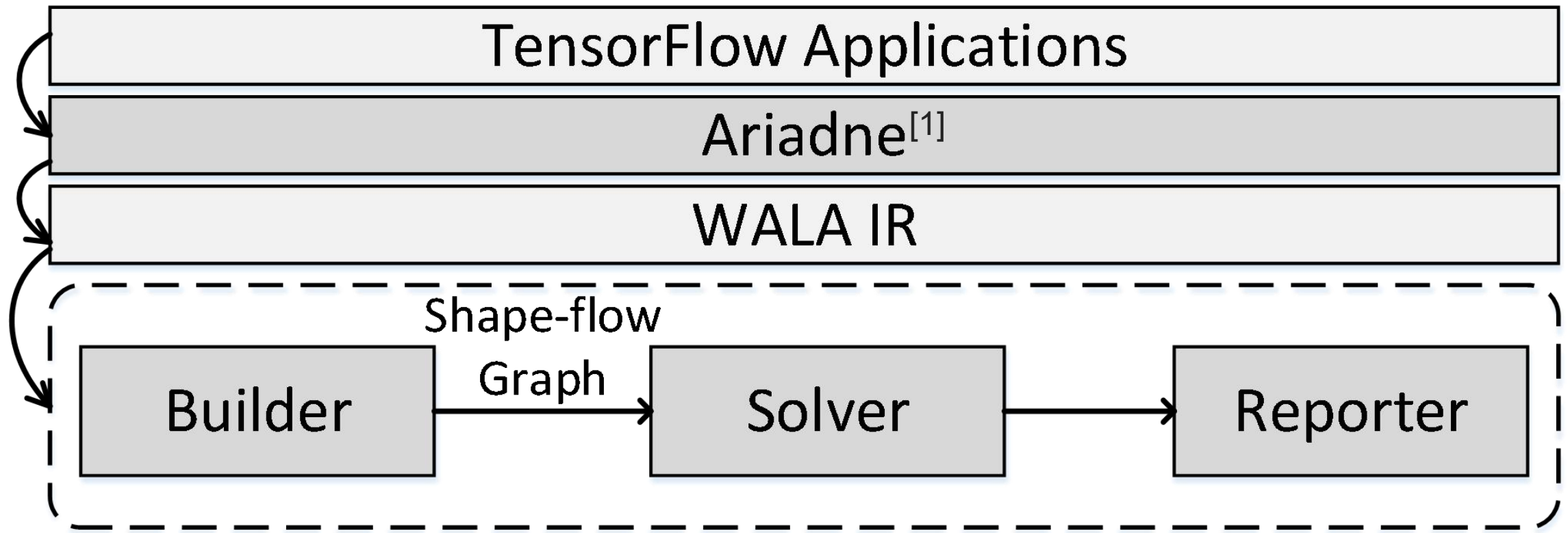


A simplified shape error example



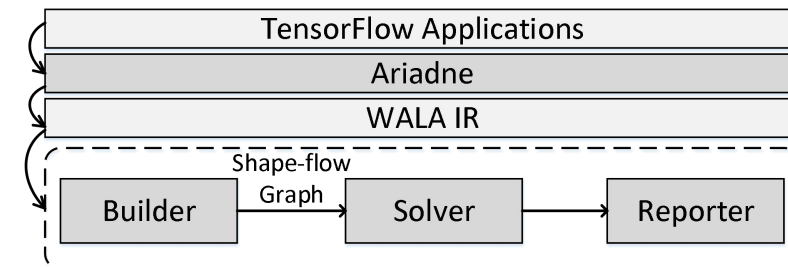
ValueError: Shapes (1600, 10) and (50, 10) are incompatible

Architecture of ShapeTracer

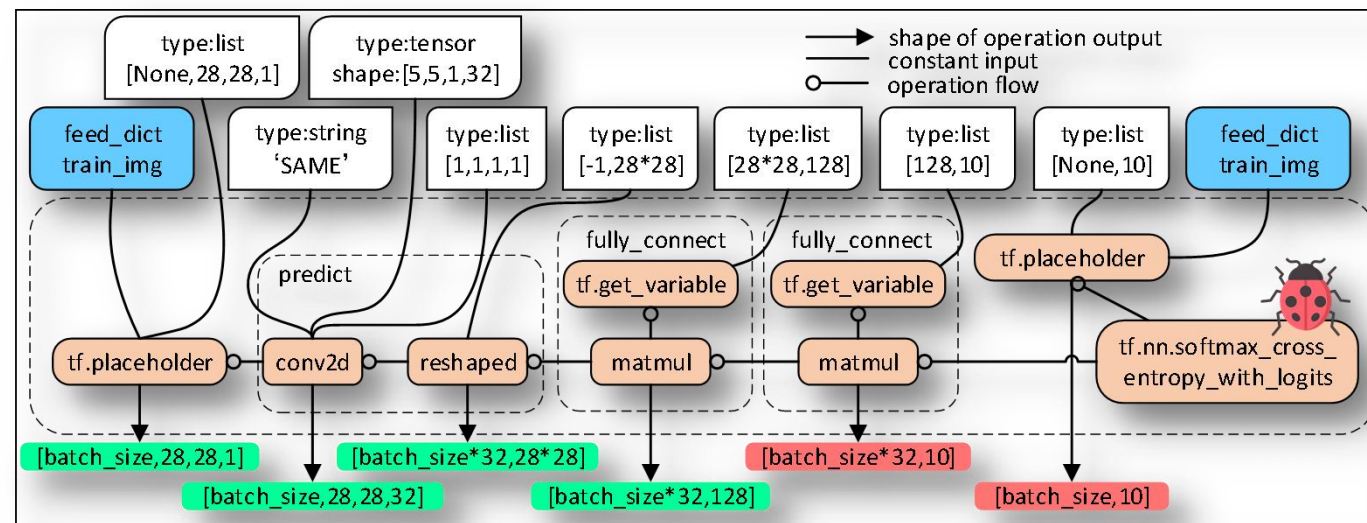
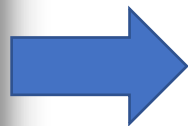


[1] Dolby, Julian, et al. "Ariadne: analysis for machine learning programs." *Proceedings of the 2Nd ACM SIGPLAN International Workshop on Machine Learning and Programming Languages*. 2018.

Builder



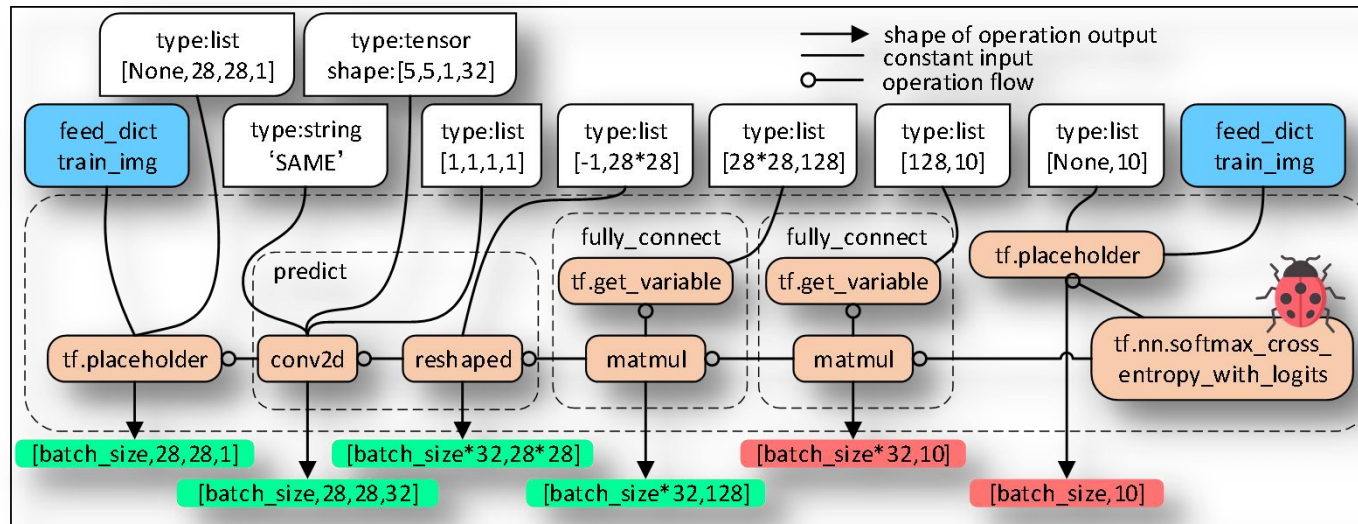
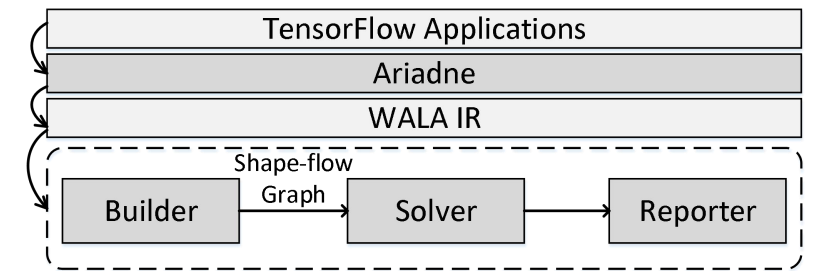
```
# Construction
1. def fully_connect(input_op, name, n_in,n_out ):
2.   fc_w = tf.get_variable(name, [n_in, n_out])
3.   return tf.matmul(input_op, fc_w)
4. def predict(input_x, class_num):
5.   mp = tf.nn.conv2d(input_x,tf.get_variable('mpc',[5,5,1,32]),strides=[1,1,1,1], padding='SAME')
6.   reshaped = tf.reshape(mp, [-1, 28 * 28])
7.   fc = fully_connect(reshaped, 'fc1', 28 * 28, 128)
8.   logit = fully_connect(fc, 'fc2', 128, class_num)
9.   return logit
10. in_x = tf.placeholder(tf.float32, shape = [None, 28, 28, 1])
11. in_y = tf.placeholder(tf.float32, shape = [None, 10])
12. y = predict(in_x, 10)
13. cross_entropy = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(labels=in_y,logits =y))
14. train_step = tf.train.AdamOptimizer(1e-4).minimize(cross_entropy)
# Execution
15. train_img, train_lab = read_image(batch_size,...)
16. with tf.Session() as sess:
17.   for i in range(1000):
18.     sess.run(train_step, feed_dict = in_x:train_img, in_y:train_lab))
```



Python Code

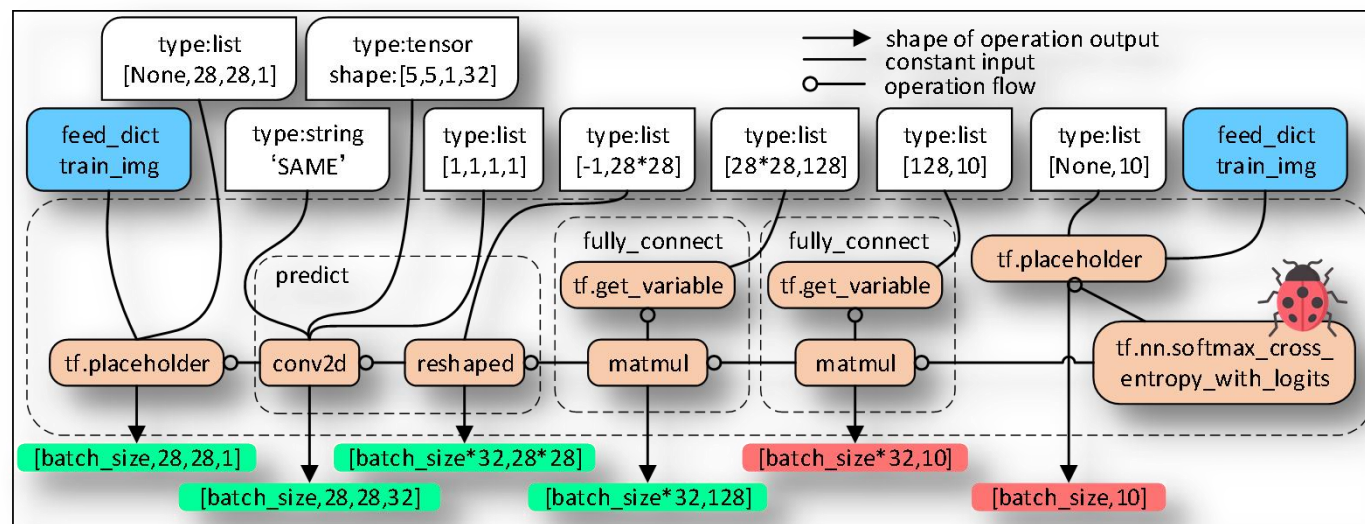
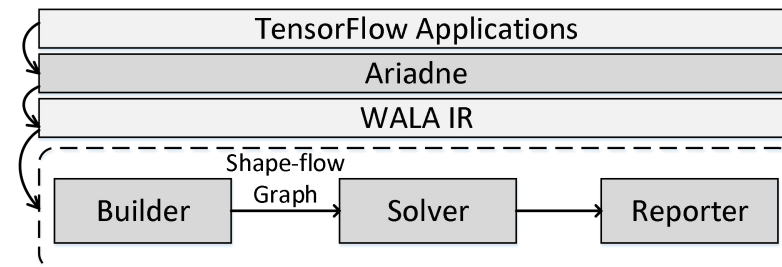
Shape-flow Graph

Solver



➔ Constraints

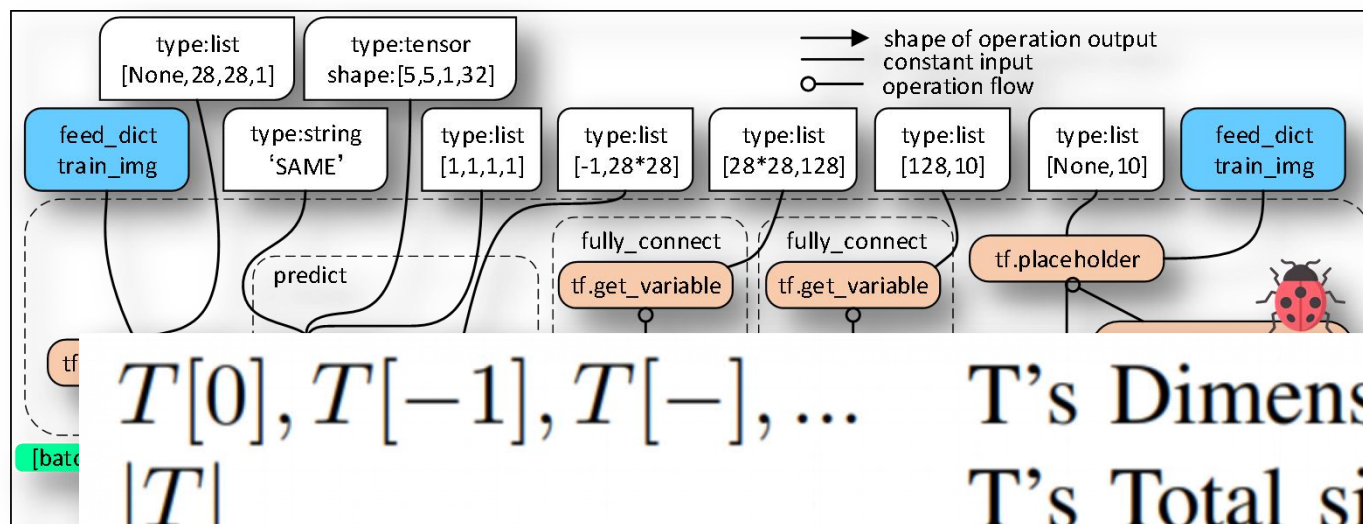
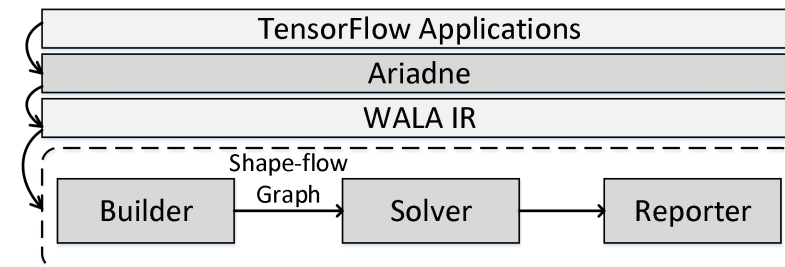
Solver



➔ Constraints

1. The CFG of network construction phase is simple.
2. Constraints is suitable for unknown hyperparameters.(such as topology and size of a neural network)

Solver



➔ Constraints

$T[0], T[-1], T[-], \dots$

T's Dimension sizes

$|T|$

T's Total size (number of elements)

\bar{T}

T's Rank (number of dimensions)

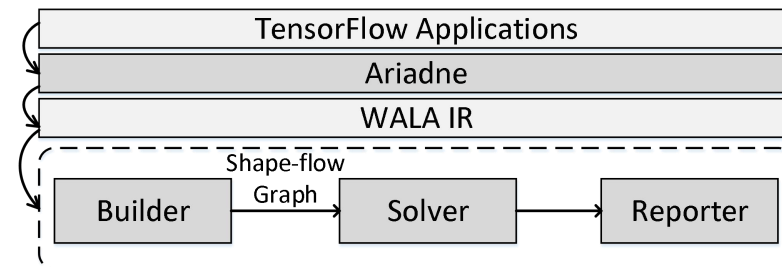
$V_0, V_1, \dots, V_{|V|-1}$

V's element values

X

X's value

Solver



$[batch_size * 32, 10]$

$T[0] == batch_size * 32$
 $T[1] == T[-1] == 10$
 $|T| == batch_size * 32 * 10$
 $\overline{T} == 2$

$T[0], T[-1], T[-], \dots$

T's Dimension sizes

$|T|$

T's Total size (number of elements)

\overline{T}

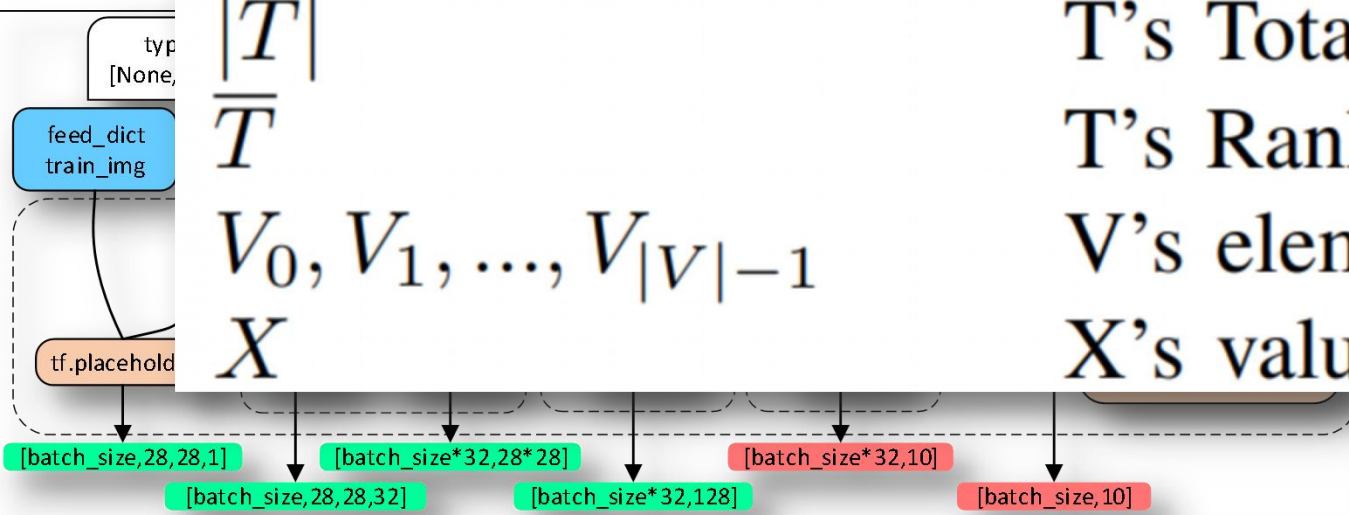
T's Rank (number of dimensions)

$V_0, V_1, \dots, V_{|V|-1}$

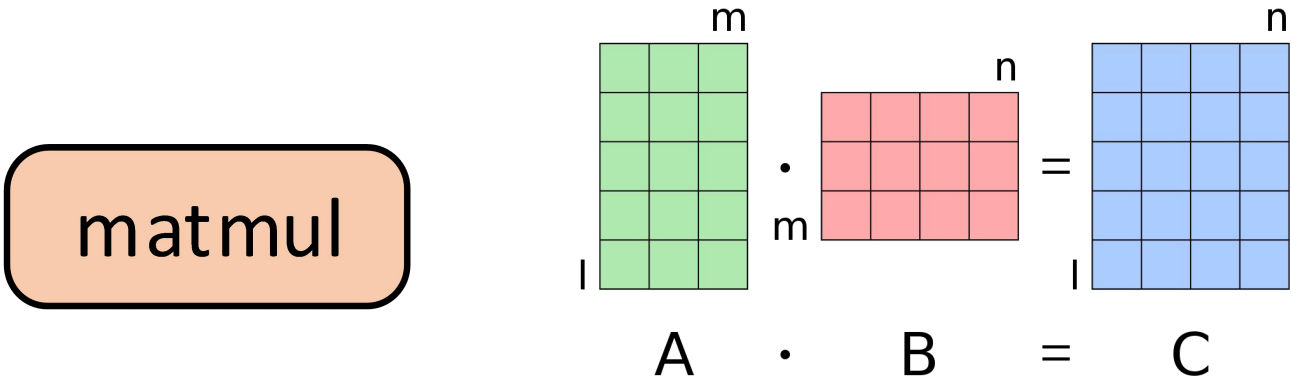
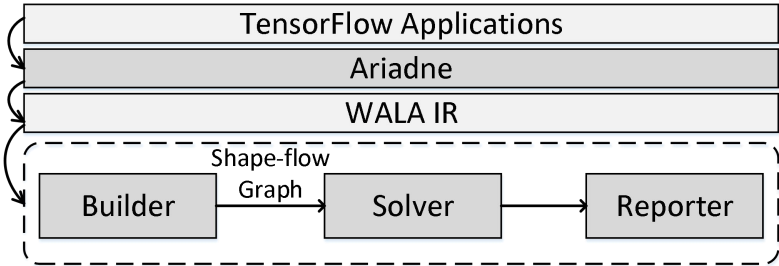
V's element values

X

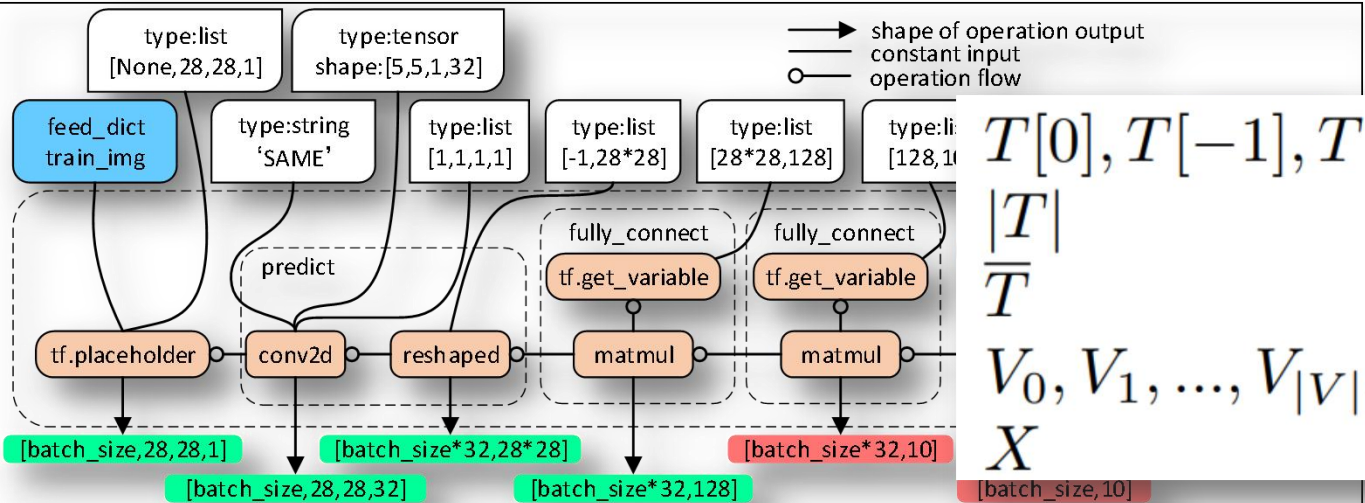
X's value



Solver



$$C = tf.Matmul(A, B)$$



$T[0], T[-1], T[-], \dots$

$|T|$

$V_0, V_1, \dots, V_{|V|-1}$

X

T's Dimension sizes

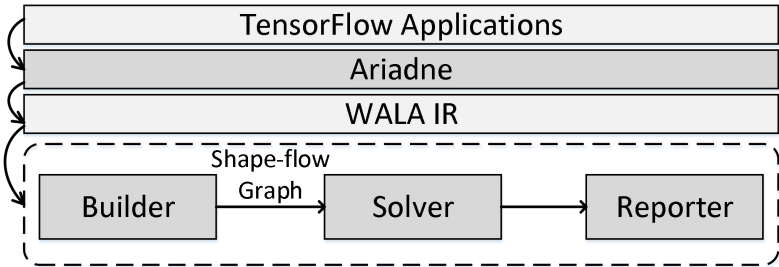
T's Total size (number of elements)

T's Rank (number of dimensions)

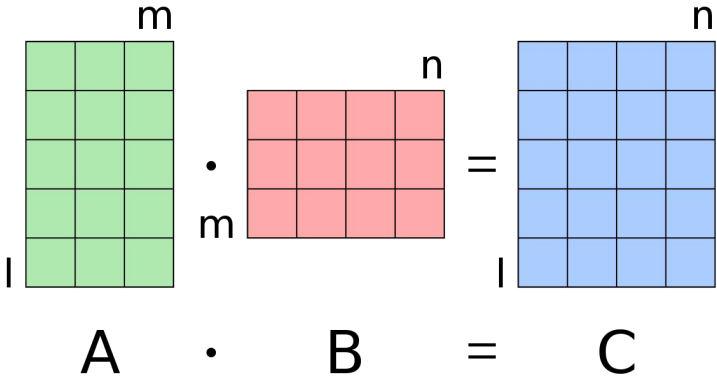
V's element values

X's value

Solver

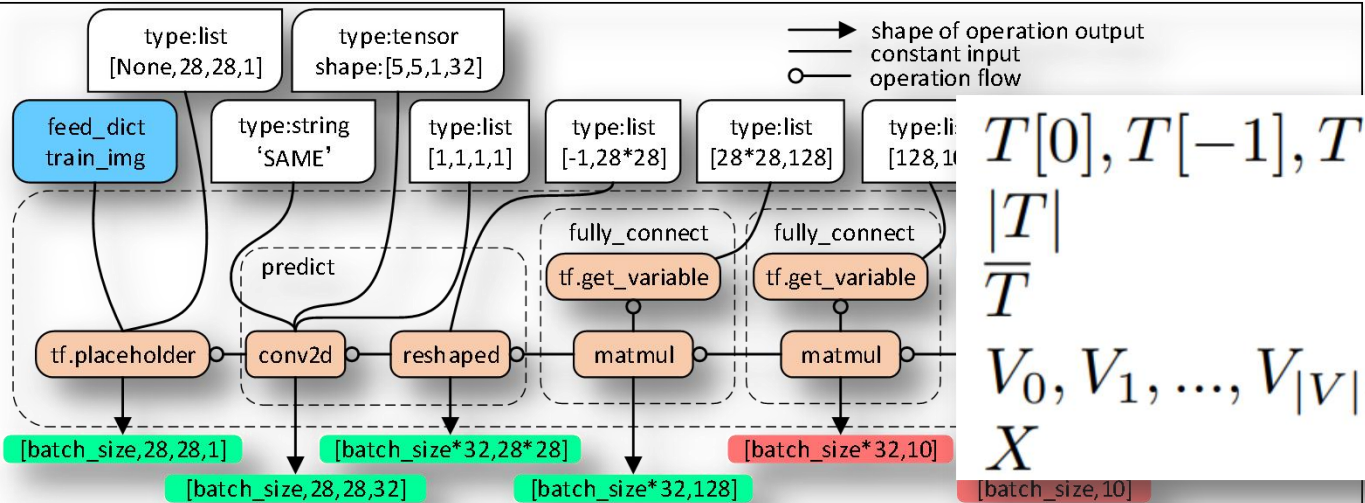


matmul



$C = tf.Matmul(A, B)$

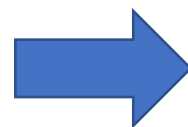
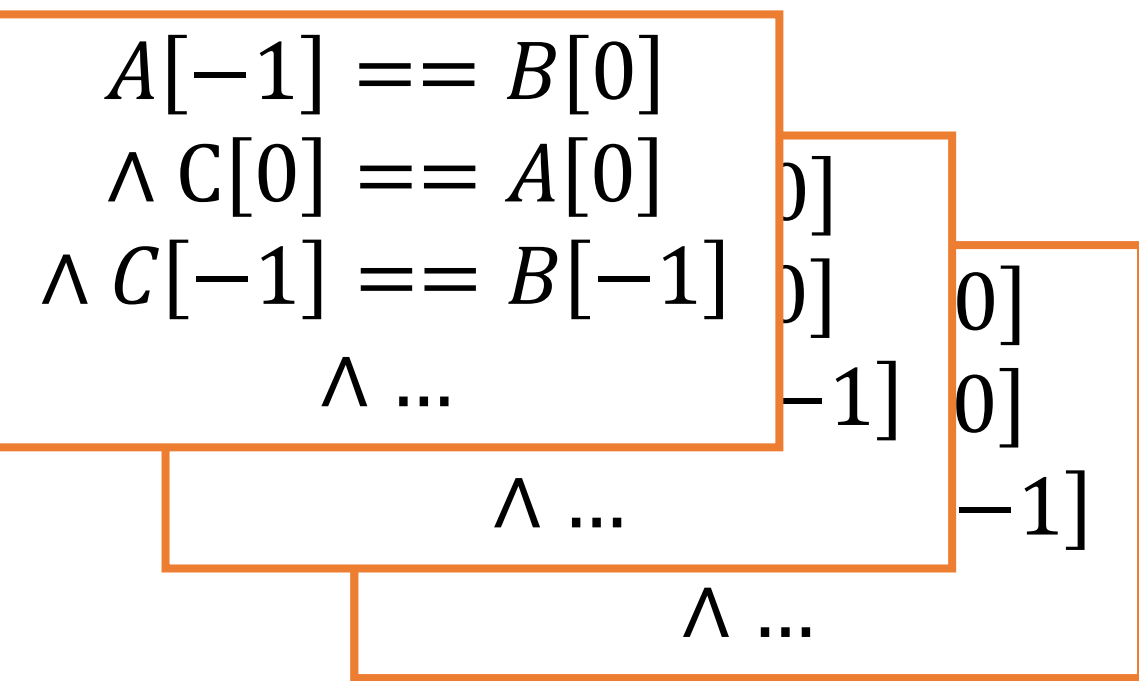
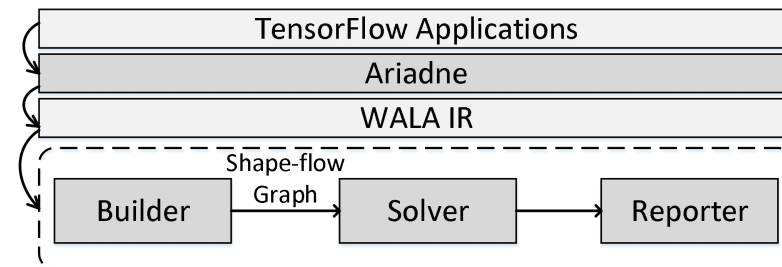
$A[-1] == B[0]$
 $\wedge C[0] == A[0]$
 $\wedge C[-1] == B[-1]$
 $\wedge \dots$



$T[0], T[-1], T[-], \dots$
 $|T|$
 \overline{T}
 $V_0, V_1, \dots, V_{|V|-1}$
 X

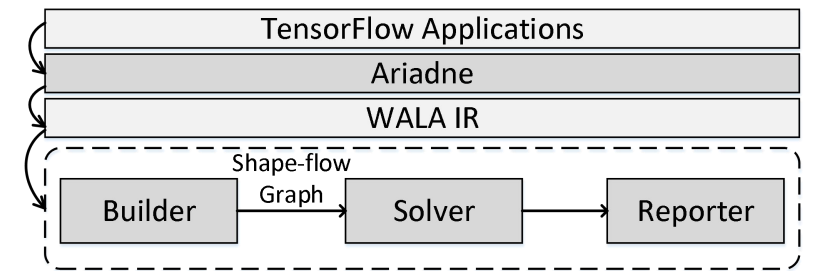
T's Dimension sizes
T's Total size (number of elements)
T's Rank (number of dimensions)
V's element values
X's value

Solver



constraint solver

Reporter



constraint solver

Unsatisfiable



```
#error reported by ShapeTracer
[[Error]]
  filename.py(line15)__matmul
    Dimensions must be equal.
```

Error location

Expect user inputs



```
#warning reported by ShapeTracer
[[Warning]]
  [batch_size]filename.py(line3)__value = 66
  [anonymous]filename.py(line6)__value = 67
```

A set of feasible solutions

Evaluation

Evaluated on 14 programs from Zhang et al. empirical study[1] and 60 randomly picked industrial programs.

[1] Zhang, Yuhao, et al. "An empirical study on TensorFlow program bugs." *Proceedings of the 27th ACM SIGSOFT International Symposium on Software Testing and Analysis*. 2018.

Evaluation

- 9 out of 14 bugs in open-source applications
- 40 out of 60 bugs in industrial programs
- Existing detection:
Pythia [1]

[1] Lagouvardos, Sifis, et al. "Static analysis of shape in TensorFlow programs." *34th European Conference on Object-Oriented Programming (ECOOP 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

Evaluation

- 9 out of 14 bugs in open-source applications

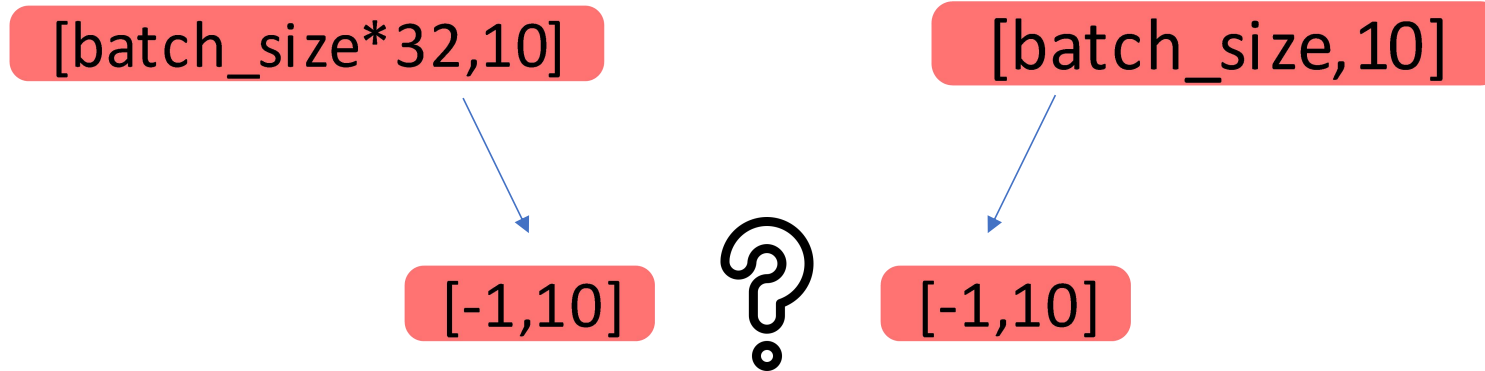
	UT-1	UT-5	UT-12	UT-13	UT-15
SHAPETRACER	✓	✓	-	-	-
PYTHIA	⚠	-	⚠	⚠	⚠

Evaluation

- ShapeTracer
 - 40 out of 60 bugs in industrial programs
- Pythia
 - 23 of 60 runtime exceptions when generating Python facts.
 - Extended with 75 datalog rules for unsupported operators
 - 9 of rest 37

Evaluation

- Existing detection:
 - *Pythia*
 - based on Datalog



Conclusion

- Study on 12289 failed industrial jobs
- ShapeTracer
 - constraint-based approach to detecting shape-related bugs
 - both efficient and effective

