ATM HACKATHON

- bazel build --config=monolithic --config=noaws --config=nogcp --config=nohdfs --config=nonccl --fat_apk_cpu=x86_64 --experimental_ui_max_stdouterr=1073741819 -c -opt --cxxopt=--std=c++14 //tensorflow/lite:libtensorflowlite.so

```
if opt.gpuoff:
        tf.device("/device:CPU:0")
        print("using CPU")
    else:
        tf.device("/GPU:0")
        print("using GPU”)
```

```
import tflite_runtime.interpreter as tflite interpreter =
tflite.Interpreter(model_path)
```

```
import os
os.environ["CUDA_VISIBLE_DEVICES"] = "-1"
gpu_available = tf.test.is_gpu_available() //cross check to see any GPU usage
print(gpu_available)
```

```
import tensorflow as tf
interpreter = tf.lite.Interpreter(model_path)
```

```
import tflite_runtime.interpreter as tflite
interpreter = tflite.Interpreter(model_path)
```

```
interpreter = tf.lite.Interpreter(model_path="q_model.tflite", num_threads=4)
```

or

using
`tf.config.threading.set_intra_op_parallelism_threads(4)`bazel build -c opt \

```
    --define tflite_with_xnnpack=true \import attr
```

```
import tensorflow as tf
from tensorflow.python.util import nest

@attr.attrs(auto_attribs=True)
class Container:
    a: object
    b: object

shape_object = Container(a=[1, 2], b=[3])
shallow_object = Container(a=None, b=None)
shape_res = nest.map_structure_up_to(shallow_object, tf.TensorShape
, shape_object)

import attr
import tensorflow as tf
from tensorflow.python.util import nest

@attr.attrs(auto_attribs=True)
class Container:
    a: object
    b: object
```

```
shape_object = Container(a=[1, 2], b=[3])
shallow_object = Container(a=None, b=None)
shape_res = nest.map_structure_up_to(shallow_object, tf.TensorShape
, shape_object)
```

```
tf.image.flip_left_right(
    image
)
```

```
x = [[[1.0, 2.0, 3.0],
      [4.0, 5.0, 6.0]],
    [[7.0, 8.0, 9.0],
      [10.0, 11.0, 12.0]]]
tf.image.flip_left_right(x)
<tf.Tensor: shape=(2, 2, 3), dtype=float32, numpy=
array([[[ 4.,  5.,  6.],
        [ 1.,  2.,  3.]],
       [[10., 11., 12.],
        [ 7.,  8.,  9.]]], dtype=float32)
```

ValueError Traceback (most recent call last)

in ()
4
5 for n in range(1000):
----> 6 next_char, states = one_step_reloaded.generate_one_step(next_char, states=states)
7 result.append(next_char)
8
1 frames
/usr/local/lib/python3.7/dist-packages/tensorflow/python/saved_model/function_deserialization.py<https://localhost:8080/> in restored_function_body(*args, **kwargs)
282 .format(index + 1, _pretty_format_positional(positional), keyword))
283 raise ValueError(
--> 284 "Could not find matching concrete function to call loaded from the "
285 f"SavedModel. Got:\n {_pretty_format_positional(args)}\n Keyword "
286 f"arguments: {kwargs}\n\n Expected these arguments to match one of the "
ValueError: Could not find matching concrete function to call loaded from the SavedModel. Got:
Positional arguments (2 total):
* Tensor("inputs:0", shape=(2,), dtype=string)
* None

```
Keyword arguments: {}
Expected these arguments to match one of the following 4 option(s):
Option 1:
Positional arguments (2 total):
* TensorSpec(shape=(1,), dtype=tf.string, name='inputs')
* None
Keyword arguments: {}
Option 2:
Positional arguments (2 total):
* TensorSpec(shape=(1,), dtype=tf.string, name='inputs')
* TensorSpec(shape=(1, 1024), dtype=tf.float32, name='states')
Keyword arguments: {}
Option 3:
Positional arguments (2 total):
* TensorSpec(shape=(5,), dtype=tf.string, name='inputs')
* None
Keyword arguments: {}
Option 4:
Positional arguments (2 total):
* TensorSpec(shape=(5,), dtype=tf.string, name='inputs')
* TensorSpec(shape=(5, 1024), dtype=tf.float32, name='states')
Keyword arguments: {}
```

```
> @@ -0,0 +1,28 @@
+name: ARM CD
```

```
.py
import numpy as np
import tensorflow as tf
np.random.seed(0)
batch1 = tf.cast(tf.ragged.constant([
255*np.random.uniform(size=(2000, 2000))]), tf.uint8)
batch1 = tf.expand_dims(batch1, axis=-1)
batch1 = tf.concat([batch1, batch1, batch1], axis=-1)

sign = tf.RaggedTensorSpec((1, None, None, 3), tf.uint8, 2,
tf.int64)

@tf.function(input_signature=(sign,))
def resize_tf(images):
  return tf.image.resize(images, (50, 50)) / 255.

def resize_non_tf(images):
  return tf.image.resize(images, (50, 50)) / 255.
```

```
print(tf.reduce_mean(resize_tf(batch1)))
print(tf.reduce_mean(resize_non_tf(batch1)))
```

and then run `python3 test.py`

```
tf.Tensor(0.49723607, shape=(), dtype=float32)
tf.Tensor(0.497236, shape=(), dtype=float32)
```

```
#include <utmpx.h>char            ut_user[]  User login
name.

char            ut_id[]    Unspecified initialization pro
cess identifier.
char            ut_line[]  Device name.
pid_t           ut_pid     Process ID.
short           ut_type    Type of entry.
struct timeval  ut_tv      Time entry was made.

void          endutxent(void);
struct utmpx *getutxent(void);
struct utmpx *getutxid(const struct utmpx *);
struct utmpx *getutxline(const struct utmpx *);
struct utmpx *pututxline(const struct utmpx *);
void          setutxent(void);

int   getitimer(int, struct itimerval *);
int   gettimeofday(struct timeval *restrict, void *restri
ct);
int   select(int, fd_set *restrict, fd_set *restrict, fd_
set *restrict,
          struct timeval *restrict);
int   setitimer(int, const struct itimerval *restrict,
          struct itimerval *restrict);
int   utimes(const char *, const struct timeval [2]); (LE
GACY )struct timeval it_interval Timer interval.

struct timeval it_value    Current value. time_t
tv_sec      Seconds.

suseconds_t    tv_usec     Microseconds.

int  pselect(int, fd_set *restrict, fd_set *restrict, fd_
set *restrict,
         const struct timespec *restrict, const sigset_t *re
int  select(int, fd_set *restrict, fd_set *restrict, fd_s
et *restrict,
         struct timeval *restrict);time_t         tv_sec
Seconds.
```

```
suseconds_t    tv_usec     Microseconds. struct timeval
it_interval Timer interval.
```

```
struct timeval it_value    Current value#include
<sys/types.h>
```

⌫

```
#include <grp.h>
struct group *getgrnam(const char *name);
```

c99 [-c][-D *name*[=*value*]]...[-E][-g][-I
directory] ... [-L *directory*]
       ... [-o *outfile*][-O*optlevel*][-s][-U
name]... *operand* ... $(basename *pathname*
.c).S_IRWXO | S_IRWXG | S_IRWXU"%s:\n",
<*file*"%s:\n", <*file*>c99 $(getconf ${CENV}_CFLAGS) -D
_POSIX_C_SOURCE=200112L \

```
$(getconf ${CENV}_LDFLAGS) foo.c -o foo \
$(getconf ${CENV}_LIBS)for CENV in $(getconf
POSIX_V6_WIDTH_RESTRICTED_ENVS)

do
    case $CENV in
    *OFF64*|*OFFBIG*) break ;;
    esac
doneCENV=$(getconf
POSIX_V6_WIDTH_RESTRICTED_ENVS | head -n lc99 -L
/a/b/c main.o a.c -l Q b.c -l p
```

```
c99 -L /a/a/a -L /a/b/c main.o a.c -l Q b.c -l p
```

```
if [ $(getconf _POSIX_V6_ILP32_OFFBIG) != "-1" ]
then
    c99 $(getconf POSIX_V6_ILP32_OFFBIG_CFLAGS) -D_XOPEN
        $(getconf POSIX_V6_ILP32_OFFBIG_LDFLAGS) foo.c -
        $(getconf POSIX_V6_ILP32_OFFBIG_LIBS) -l pthread
else
    echo ILP32_OFFBIG programming environment not suppor
    exit 1
fic99 foo.c bar.oc99 c99 -c foo.cc99 -o foo foo.c
```