```
import tensorflow as tf
from tensorflow.keras import layers, models
from tensorflow.keras.preprocessing.image import ImageDat
from sklearn.model_selection import train_test_split


# Connect to TPU
try:
    tpu = tf.distribute.cluster_resolver.TPUClusterResolv
    print('Running on TPU ', tpu.cluster_spec().as_dict()
except ValueError:
    raise BaseException('ERROR: Not connected to a TPU ru

tf.config.experimental_connect_to_cluster(tpu)
tf.tpu.experimental.initialize_tpu_system(tpu)
tpu_strategy = tf.distribute.TPUStrategy(tpu)
```

```
Running on TPU  ['10.81.186.194:8470']
```

```
# Set up parameters
AUTO = tf.data.experimental.AUTOTUNE
IMAGE_SIZE = [224, 224]  # adjust size as needed
batch_size = 32 * tpu_strategy.num_replicas_in_sync
gcs_pattern_train = 'gs://grad_proj/train/*/*.jpg'
gcs_pattern_test = 'gs://grad_proj/test/*/*.jpg'
num_images_per_class = 100


# Define data loading functions
def parse_image(filename, label):
    img = tf.io.read_file(filename)
    img = tf.image.decode_jpeg(img, channels=3)
    img = tf.image.resize(img, IMAGE_SIZE)
    img = tf.cast(img, tf.float32) / 255.0
    return img, label

# Define data loading functions
def load_dataset(gcs_pattern, num_images):
    filenames = tf.io.gfile.glob(gcs_pattern)
    random.shuffle(filenames)  # Shuffle the filenames
    filenames = filenames[:num_images]  # Select a subse
    labels = [1 if 'dog' in filename else 0 for filename
    dataset = tf.data.Dataset.from_tensor_slices((filena
    dataset = dataset.map(parse_image, num_parallel_cal
    return dataset
```

```
import random
# Load datasets
train_dataset = load_dataset(gcs_pattern_train, num_imag
test_dataset = load_dataset(gcs_pattern_test, num_images

# Assuming your train dataset is named 'train_dataset'
train_dataset_length = tf.data.experimental.cardinality(

print(f"Train dataset length: {train_dataset_length}")
```

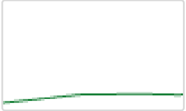
Resources ×

You are not subscribed. Learn more.
You currently have zero compute units available. Resources offered free of charge are not guaranteed. Purchase more units here.
Manage sessions

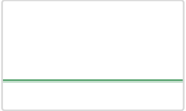
Want more memory and disk space? ×
Upgrade to Colab Pro

Python 3 Google Compute Engine backend (TPU)
Showing resources from 03:28 to 03:33

System RAM
1.8 / 12.7 GB



Disk
29.2 / 107.7 GB



```
    Train dataset length: 100
```

```
# Load datasets
train_dataset = load_dataset(gcs_pattern_train)
test_dataset = load_dataset(gcs_pattern_test)


# Assuming your train dataset is named 'train_dataset'
train_dataset_length = tf.data.experimental.cardinality

print(f"Train dataset length: {train_dataset_length}")



# Define and compile the model
with tpu_strategy.scope():
    model = models.Sequential()
    model.add(layers.Conv2D(32, kernel_size=(3, 3), pad
    model.add(layers.BatchNormalization())
    model.add(layers.MaxPooling2D(pool_size=(2, 2), str
    model.add(layers.Conv2D(64, kernel_size=(3, 3), pad
    model.add(layers.BatchNormalization())
    model.add(layers.MaxPooling2D(pool_size=(2, 2), str
    model.add(layers.Conv2D(128, kernel_size=(3, 3), pa
    model.add(layers.BatchNormalization())
    model.add(layers.MaxPooling2D(pool_size=(2, 2), str
    model.add(layers.Flatten())
    model.add(layers.Dense(128, activation='relu'))
    model.add(layers.Dropout(0.1))
    model.add(layers.Dense(64, activation='relu'))
    model.add(layers.Dropout(0.1))
    model.add(layers.Dense(1, activation='sigmoid'))

    model.compile(optimizer='adam', loss='binary_crosse



import time

start_time = time.time()

history = model.fit(
    train_dataset.shuffle(1000).batch(batch_size).prefetc
    epochs=10,
    validation_data=test_dataset.batch(batch_size).prefet
)

end_time = time.time()
elapsed_time = end_time - start_time

print(f"Training took {elapsed_time} seconds.")
```

```
    Epoch 1/10
    1/1 [==============================] - 22s 22s/step
    Epoch 2/10
    1/1 [==============================] - 7s 7s/step -
    Epoch 3/10
    1/1 [==============================] - 7s 7s/step -
    Epoch 4/10
    1/1 [==============================] - 7s 7s/step -
    Epoch 5/10
    1/1 [==============================] - 7s 7s/step -
```

```
Epoch 6/10
1/1 [==============================] - 8s 8s/step -
Epoch 7/10
1/1 [==============================] - 7s 7s/step -
Epoch 8/10
1/1 [==============================] - 7s 7s/step -
Epoch 9/10
1/1 [==============================] - 8s 8s/step -
Epoch 10/10
1/1 [==============================] - 6s 6s/step -
Training took 94.66769695281982 seconds.
```

Change runtime type