 View on TensorFlow.org  Run in Google Colab  View source on GitHub

[ ] ↳ 1 cell hidden

Preprocess data with TensorFlow Transform

The Feature Engineering Component of TensorFlow Extended (TFX)

This example colab notebook provides a very simple example of how TensorFlow Transform (`tf.Transform`) can be used to preprocess data using exactly the same code for both training a model and serving inferences in production.

TensorFlow Transform is a library for preprocessing input data for TensorFlow, including creating features that require a full pass over the training dataset. For example, using TensorFlow Transform you could:

- Normalize an input value by using the mean and standard deviation
- Convert strings to integers by generating a vocabulary over all of the input values
- Convert floats to integers by assigning them to buckets, based on the observed data distribution

TensorFlow has built-in support for manipulations on a single example or a batch of examples. `tf.Transform` extends these capabilities to support full passes over the entire training dataset.

The output of `tf.Transform` is exported as a TensorFlow graph which you can use for both training and serving. Using the same graph for both training and serving can prevent skew, since the same transformations are applied in both stages.

Upgrade Pip

To avoid upgrading Pip in a system when running locally, check to make sure that we're running in Colab. Local systems can of course be upgraded separately.

```
try:
  import colab
  !pip install --upgrade pip
except:
  pass
```

```
    Collecting pip
      Downloading https://files.pythonhosted.org/packages/cd/82/04e9aaf603fdbaecb4323b9e723f13c92c245f6ab2902195c5
         |████████████████████████████████| 1.6MB 6.5MB/s
    Installing collected packages: pip
      Found existing installation: pip 19.3.1
        Uninstalling pip-19.3.1:
          Successfully uninstalled pip-19.3.1
    Successfully installed pip-21.1.2
```

Install TensorFlow Transform

Note: In Google Colab, because of package updates, the first time you run this cell you may need to restart the runtime (Runtime > Restart runtime ...).

```
!pip install -q -U tensorflow_transform==0.24.1
```

```
     |████████████████████████████████| 373 kB 6.5 MB/s
     |████████████████████████████████| 127 kB 9.8 MB/s
     |████████████████████████████████| 320.5 MB 10 kB/s
     |████████████████████████████████| 9.6 MB 40.6 MB/s
     |████████████████████████████████| 1.8 MB 33.7 MB/s
     |████████████████████████████████| 44 kB 1.8 MB/s
     |████████████████████████████████| 2.2 MB 26.9 MB/s
     |████████████████████████████████| 151 kB 37.9 MB/s
     |████████████████████████████████| 61 kB 6.7 MB/s
     |████████████████████████████████| 829 kB 41.3 MB/s
     |████████████████████████████████| 255 kB 64.7 MB/s
     |████████████████████████████████| 183 kB 76.0 MB/s
     |████████████████████████████████| 267 kB 70.1 MB/s
     |████████████████████████████████| 169 kB 73.1 MB/s
     |████████████████████████████████| 173 kB 77.3 MB/s
     |████████████████████████████████| 83 kB 1.4 MB/s
```

```
     |████████████████████████████████| 144 kB 62.8 MB/s
     |████████████████████████████████| 435 kB 64.5 MB/s
     |████████████████████████████████| 20.1 MB 1.2 MB/s
     |████████████████████████████████| 2.9 MB 46.3 MB/s
     |████████████████████████████████| 459 kB 57.0 MB/s
     |████████████████████████████████| 63.8 MB 34 kB/s
       Building wheel for avro-python3 (setup.py) ... done
       Building wheel for dill (setup.py) ... done
       Building wheel for future (setup.py) ... done
       Building wheel for google-apitools (setup.py) ... done
       Building wheel for grpc-google-iam-v1 (setup.py) ... done
  ERROR: pip's dependency resolver does not currently take into account all the packages that are installed. This
  multiprocess 0.70.12.2 requires dill>=0.3.4, but you have dill 0.3.1.1 which is incompatible.
  google-colab 1.0.0 requires requests~=2.23.0, but you have requests 2.25.1 which is incompatible.
  datascience 0.10.6 requires folium==0.2.1, but you have folium 0.8.3 which is incompatible.
  albumentations 0.1.12 requires imgaug<0.2.7,>=0.2.5, but you have imgaug 0.2.9 which is incompatible.
  WARNING: Running pip as root will break packages and permissions. You should install packages reliably by usin
```

Did you restart the runtime?

If you are using Google Colab, the first time that you run the cell above, you must restart the runtime (Runtime > Restart runtime ...). This is because of the way that Colab loads packages.

Imports

```
import pprint
import tempfile

import tensorflow as tf
import tensorflow_transform as tft

import tensorflow_transform.beam as tft_beam
from tensorflow_transform.tf_metadata import dataset_metadata
from tensorflow_transform.tf_metadata import schema_utils
```

Data: Create some dummy data

We'll create some simple dummy data for our simple example:

- `raw_data` is the initial raw data that we're going to preprocess
- `raw_data_metadata` contains the schema that tells us the types of each of the columns in `raw_data`. In this case, it's very simple.

```
raw_data = [
      {'x': 1, 'y': 1, 's': 'hello'},
      {'x': 2, 'y': 2, 's': 'world'},
      {'x': 3, 'y': 3, 's': 'hello'}
  ]

raw_data_metadata = dataset_metadata.DatasetMetadata(
    schema_utils.schema_from_feature_spec({
        'y': tf.io.FixedLenFeature([], tf.float32),
        'x': tf.io.FixedLenFeature([], tf.float32),
        's': tf.io.FixedLenFeature([], tf.string),
    ...
```

the Beam pipeline during analysis requiring a full pass over the entire training dataset. The Beam computation runs only once, during training, and typically make a full pass over the entire training dataset. They create tensor constants, which are added to your graph. For example, tft.min computes the minimum of a tensor over the training dataset while tft.scale_by_min_max first computes the min and max of a tensor over the training dataset and then scales the tensor to be within a user-specified range, [output_min, output_max]. tf.Transform provides a fixed set of such analyzers/mappers, but this will be extended in future versions.

Caution: When you apply your preprocessing function to serving inferences, the constants that were created by analyzers during training do not change. If your data has trend or seasonality components, plan accordingly.

Note: The `preprocessing_fn` is not directly callable. This means that calling `preprocessing_fn(raw_data)` will not work. Instead, it must be passed to the Transform Beam API as shown in the following cells.

```
def preprocessing_fn(inputs):
    """Preprocess input columns into transformed columns."""
    x = inputs['x']
    y = inputs['y']
    s = inputs['s']
    x_centered = x - tft.mean(x)
    y_normalized = tft.scale_to_0_1(y)
    s_integerized = tft.compute_and_apply_vocabulary(s)
    x_centered_times_y_normalized = (x_centered * y_normalized)
    return {
        'x_centered': x_centered,
        'y_normalized': y_normalized,
        's_integerized': s_integerized,
        'x_centered_times_y_normalized': x_centered_times_y_normalized,
    }
```

Putting it all together

Now we're ready to transform our data. We'll use Apache Beam with a direct runner, and supply three inputs:

1. `raw_data` - The raw input data that we created above
2. `raw_data_metadata` - The schema for the raw data
3. `preprocessing_fn` - The function that we created to do our transformation

Key Term: Apache Beam uses a special syntax to define and invoke transforms. For example, in this line:

```
WARNING:tensorflow:Tensorflow version (2.3.3) found. Note that Tensorflow Transform support for TF 2.0 is curre
WARNING:apache_beam.runners.interactive.interactive_environment:Dependencies required for Interactive Beam PCo
WARNING:tensorflow:Tensorflow version (2.3.3) found. Note that Tensorflow Transform support for TF 2.0 is curre
WARNING:tensorflow:Tensorflow version (2.3.3) found. Note that Tensorflow Transform support for TF 2.0 is curre
WARNING:tensorflow:You are passing instance dicts and DatasetMetadata to TFT which will not provide optimal pe
WARNING:tensorflow:You are passing instance dicts and DatasetMetadata to TFT which will not provide optimal pe
WARNING:tensorflow:From /usr/local/lib/python3.7/dist-packages/tensorflow_transform/tf_utils.py:218: Tensor.exp
Instructions for updating:
Use ref() instead.
WARNING:tensorflow:From /usr/local/lib/python3.7/dist-packages/tensorflow_transform/tf_utils.py:218: Tensor.exp
Instructions for updating:
Use ref() instead.
WARNING:tensorflow:From /usr/local/lib/python3.7/dist-packages/tensorflow/python/saved_model/signature_def_uti
Instructions for updating:
This function will only be available through the v1 compatibility library as tf.compat.v1.saved_model.utils.bu
WARNING:tensorflow:From /usr/local/lib/python3.7/dist-packages/tensorflow/python/saved_model/signature_def_uti
Instructions for updating:
This function will only be available through the v1 compatibility library as tf.compat.v1.saved_model.utils.bu
INFO:tensorflow:Assets added to graph.
INFO:tensorflow:Assets added to graph.
INFO:tensorflow:No assets to write.
INFO:tensorflow:No assets to write.
WARNING:tensorflow:Issue encountered when serializing tft_analyzer_use.
Type is unsupported, or the types of the items don't match field type in CollectionDef. Note this is a warning
'Counter' object has no attribute 'name'
WARNING:tensorflow:Issue encountered when serializing tft_analyzer_use.
```

We wanted to map our strings to indexes in a vocabulary, and there were only 2 words in our vocabulary ("hello" and "world"). So with input of `["hello", "world", "hello"]` our result of `[0, 1, 0]` is correct. Since "hello" occurs most frequently in this data, it will be the first entry in the vocabulary.

x_centered_times_y_normalized

We wanted to create a new feature by crossing `x_centered` and `y_normalized` using multiplication. Note that this multiplies the results, not the original values, and our new result of `[-0.0, 0.0, 1.0]` is correct.