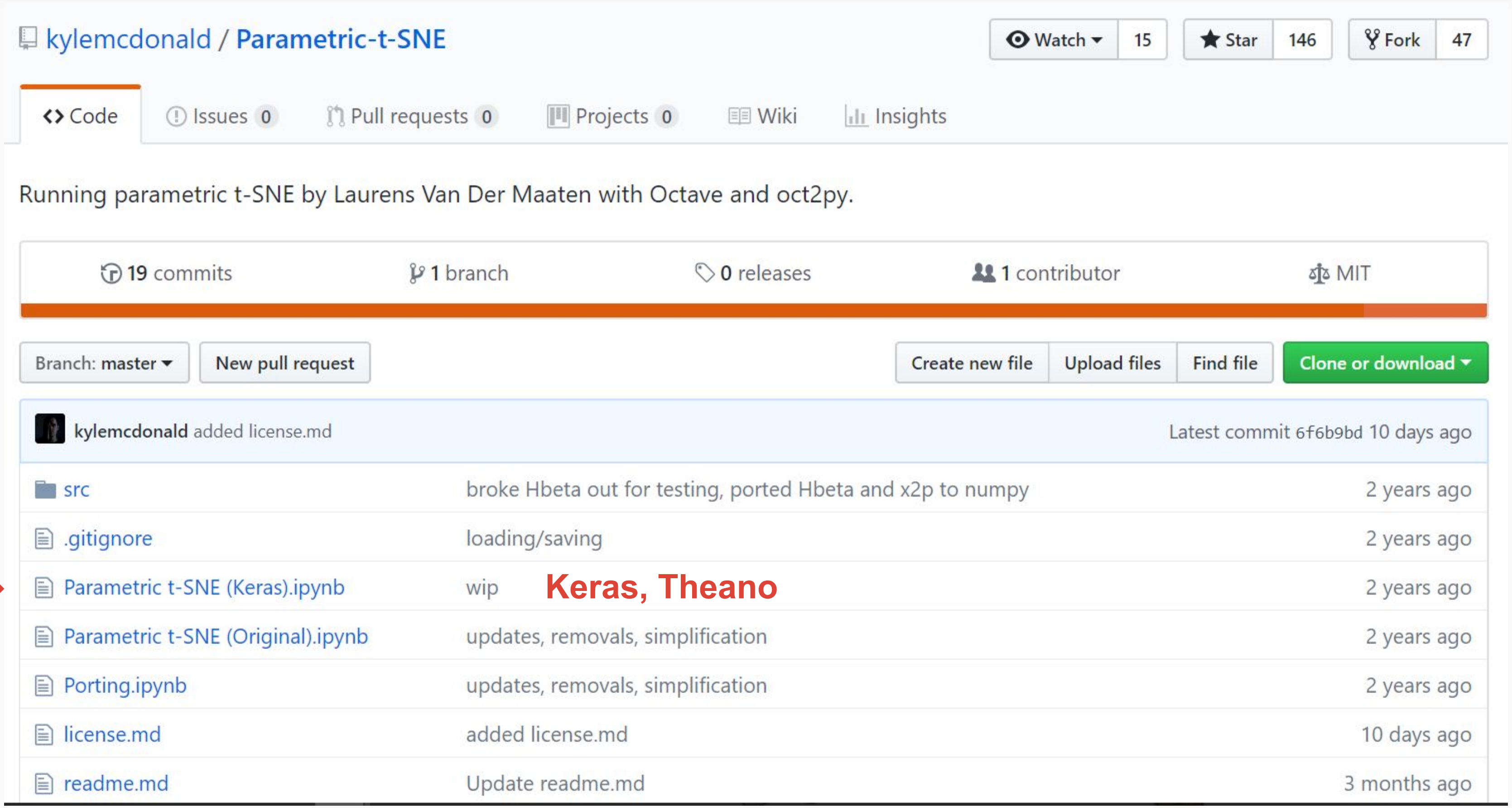
Progress Report

Kuei-Yueh (Clint) Ko

A repo of Parametric tsne



kylemcdonald / Parametric-t-SNE

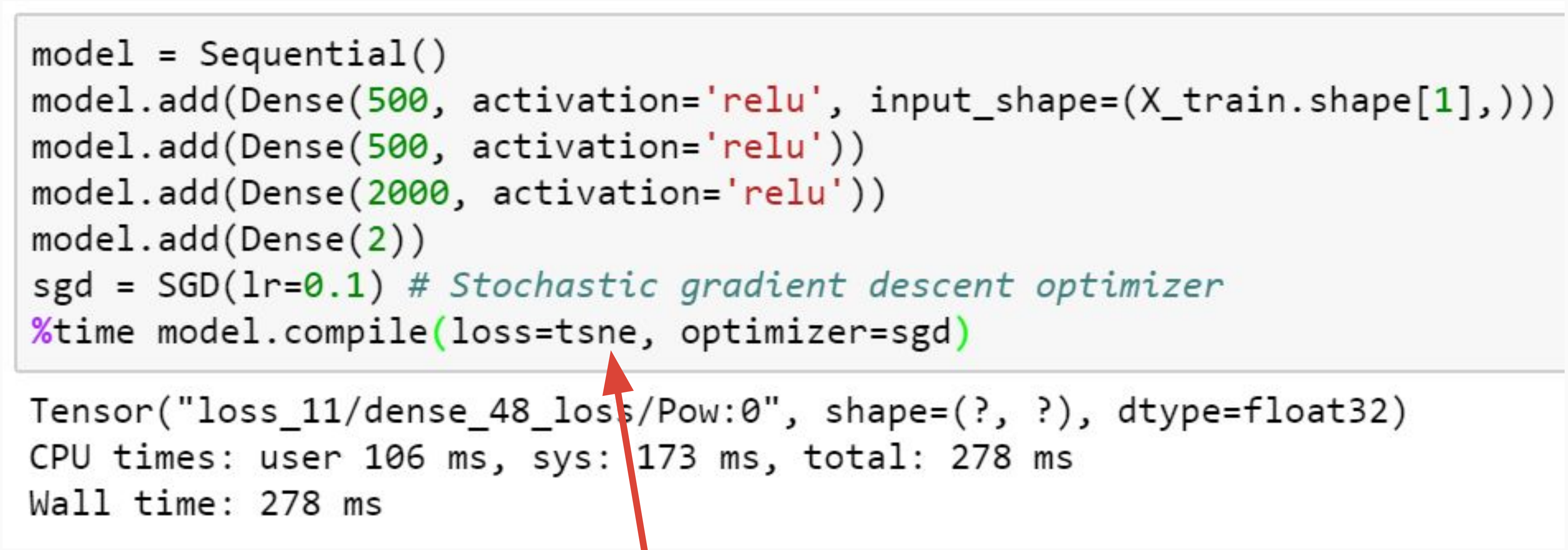
Running parametric t-SNE by Laurens Van Der Maaten with Octave and oct2py.

kylemcdonald added license.md — Latest commit 6f6b9bd 10 days ago

src	broke Hbeta out for testing, ported Hbeta and x2p to numpy	2 years ago
.gitignore	loading/saving	2 years ago
Parametric t-SNE (Keras).ipynb	wip	2 years ago
Parametric t-SNE (Original).ipynb	updates, removals, simplification	2 years ago
Porting.ipynb	updates, removals, simplification	2 years ago
license.md	added license.md	10 days ago
readme.md	Update readme.md	3 months ago

Keras, Theano

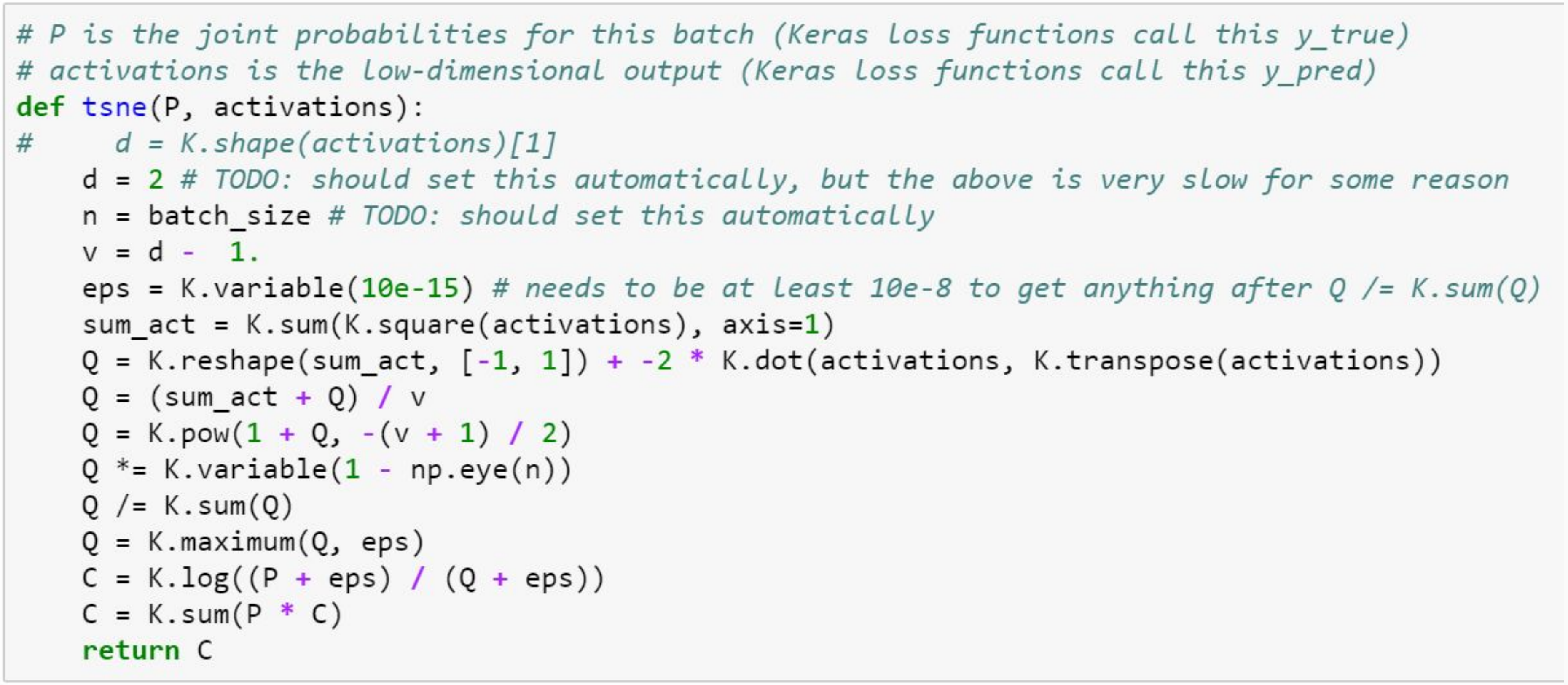
Model Structure



```
model = Sequential()
model.add(Dense(500, activation='relu', input_shape=(X_train.shape[1],)))
model.add(Dense(500, activation='relu'))
model.add(Dense(2000, activation='relu'))
model.add(Dense(2))
sgd = SGD(lr=0.1) # Stochastic gradient descent optimizer
%time model.compile(loss=tsne, optimizer=sgd)
```

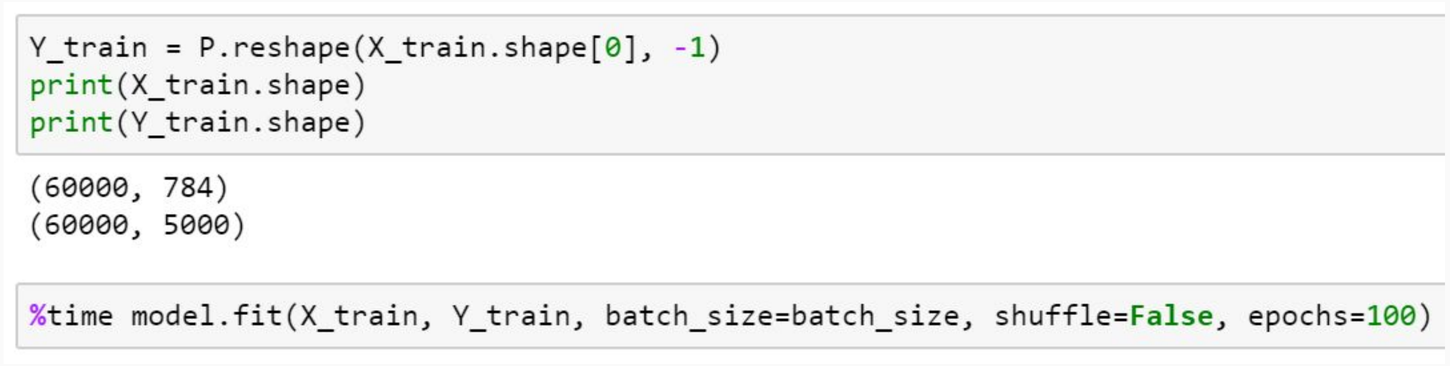
```
Tensor("loss_11/dense_48_loss/Pow:0", shape=(?, ?), dtype=float32)
CPU times: user 106 ms, sys: 173 ms, total: 278 ms
Wall time: 278 ms
```

Loss Function



```python
# P is the joint probabilities for this batch (Keras loss functions call this y_true)
# activations is the low-dimensional output (Keras loss functions call this y_pred)
def tsne(P, activations):
#     d = K.shape(activations)[1]
    d = 2 # TODO: should set this automatically, but the above is very slow for some reason
    n = batch_size # TODO: should set this automatically
    v = d -  1.
    eps = K.variable(10e-15) # needs to be at least 10e-8 to get anything after Q /= K.sum(Q)
    sum_act = K.sum(K.square(activations), axis=1)
    Q = K.reshape(sum_act, [-1, 1]) + -2 * K.dot(activations, K.transpose(activations))
    Q = (sum_act + Q) / v
    Q = K.pow(1 + Q, -(v + 1) / 2)
    Q *= K.variable(1 - np.eye(n))
    Q /= K.sum(Q)
    Q = K.maximum(Q, eps)
    C = K.log((P + eps) / (Q + eps))
    C = K.sum(P * C)
    return C
```

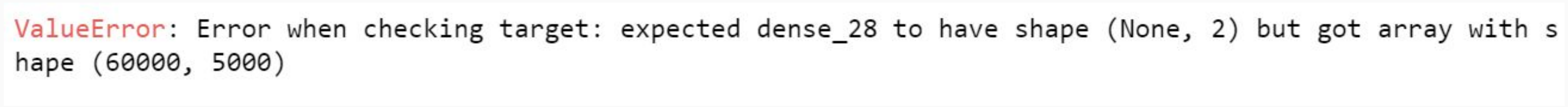
Original Problem during Training



```
Y_train = P.reshape(X_train.shape[0], -1)
print(X_train.shape)
print(Y_train.shape)
```

```
(60000, 784)
(60000, 5000)
```

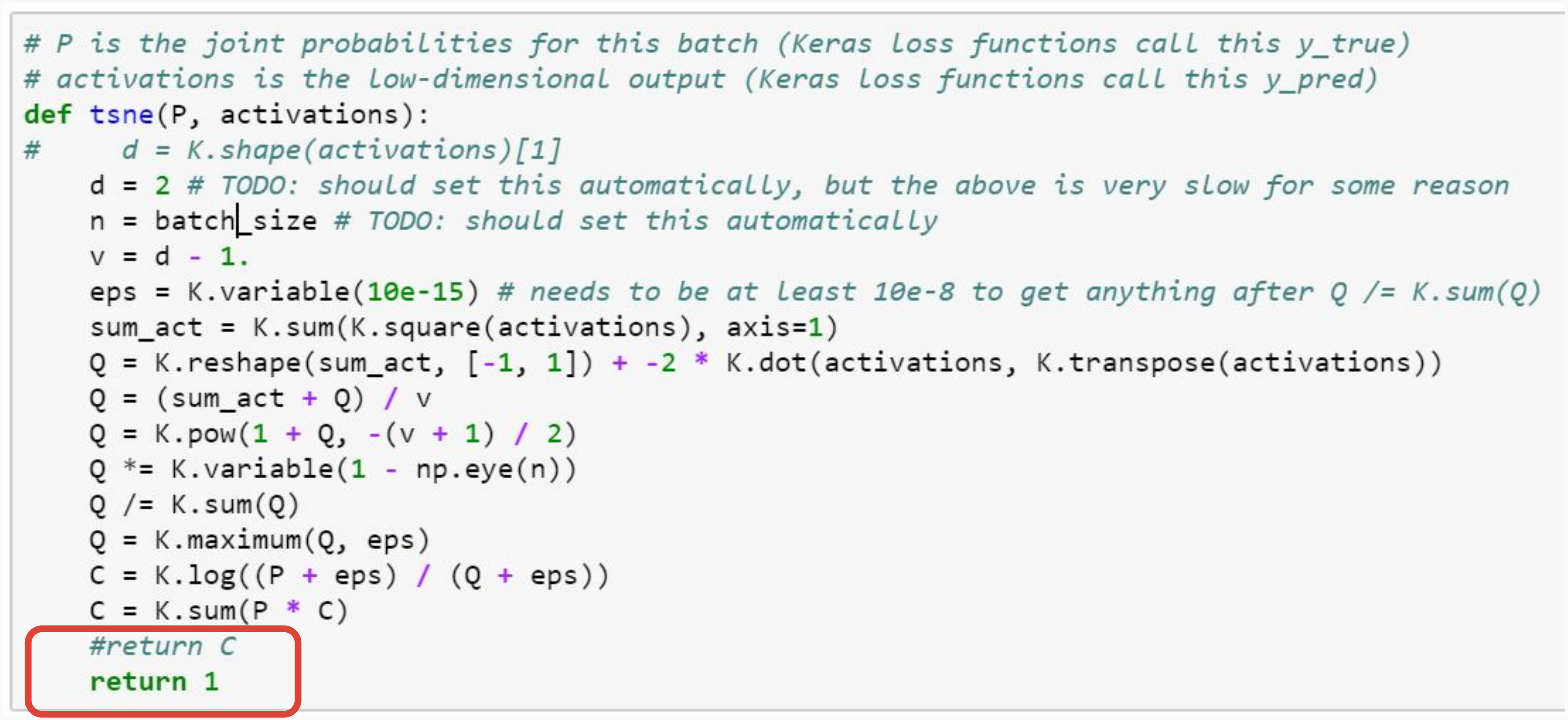
```
%time model.fit(X_train, Y_train, batch_size=batch_size, shuffle=False, epochs=100)
```



```
ValueError: Error when checking target: expected dense_28 to have shape (None, 2) but got array with s
hape (60000, 5000)
```

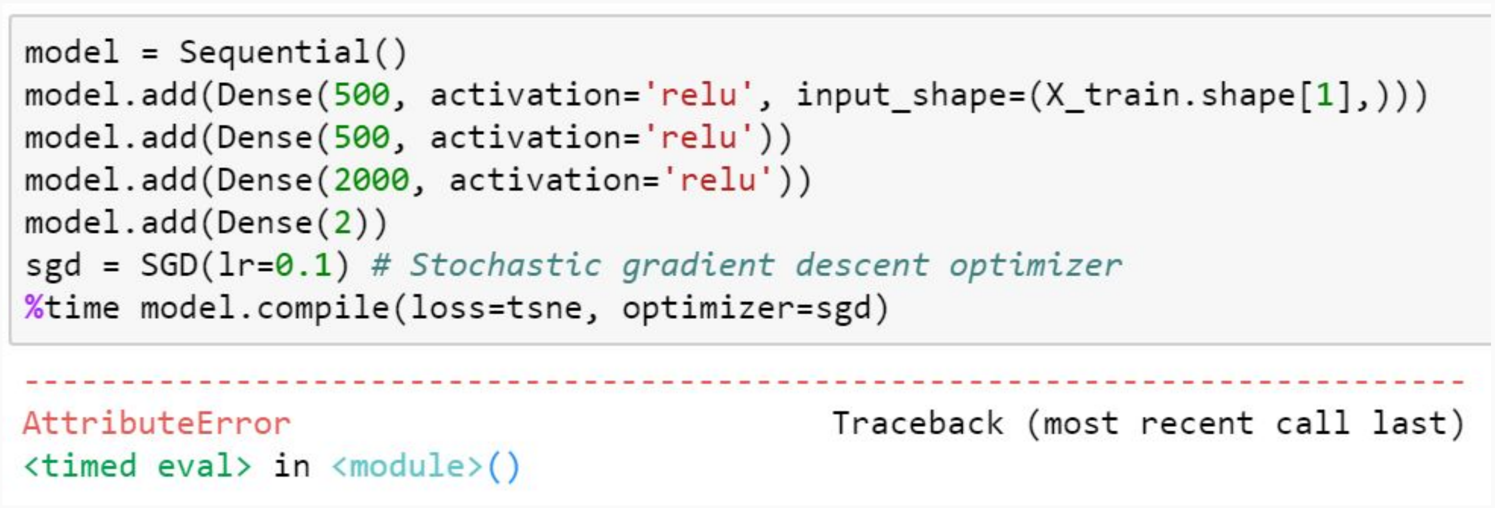
Note:
Batch size = 5000
Shape of Distance matrix = (5000, 5000)

Try to change the output of loss function into an integer



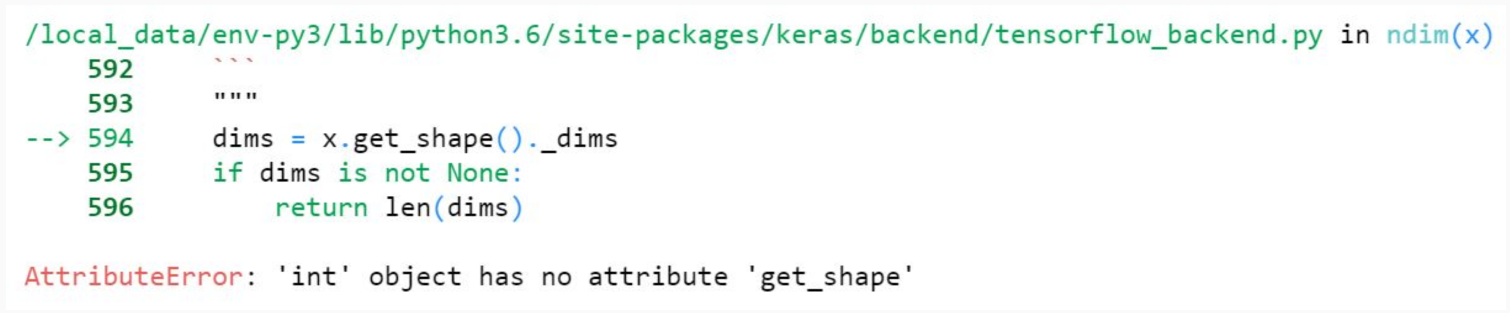
```
# P is the joint probabilities for this batch (Keras loss functions call this y_true)
# activations is the low-dimensional output (Keras loss functions call this y_pred)
def tsne(P, activations):
#     d = K.shape(activations)[1]
    d = 2 # TODO: should set this automatically, but the above is very slow for some reason
    n = batch_size # TODO: should set this automatically
    v = d - 1.
    eps = K.variable(10e-15) # needs to be at least 10e-8 to get anything after Q /= K.sum(Q)
    sum_act = K.sum(K.square(activations), axis=1)
    Q = K.reshape(sum_act, [-1, 1]) + -2 * K.dot(activations, K.transpose(activations))
    Q = (sum_act + Q) / v
    Q = K.pow(1 + Q, -(v + 1) / 2)
    Q *= K.variable(1 - np.eye(n))
    Q /= K.sum(Q)
    Q = K.maximum(Q, eps)
    C = K.log((P + eps) / (Q + eps))
    C = K.sum(P * C)
    #return C
    return 1
```

Try to change the output of loss function into an integer



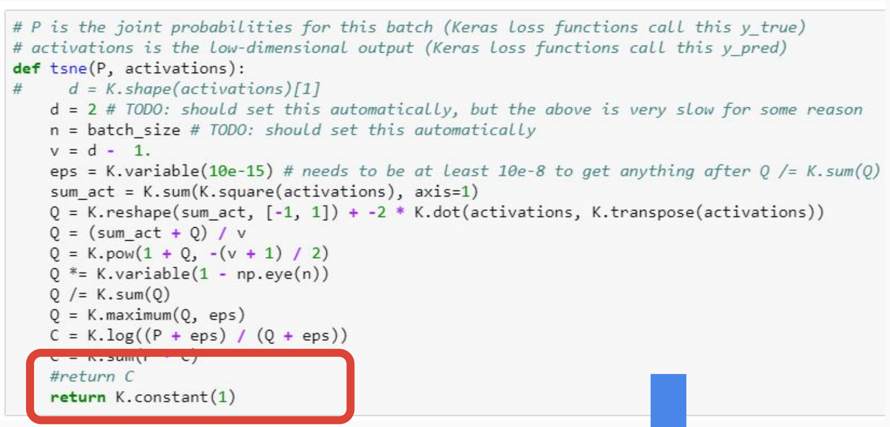
```
model = Sequential()
model.add(Dense(500, activation='relu', input_shape=(X_train.shape[1],)))
model.add(Dense(500, activation='relu'))
model.add(Dense(2000, activation='relu'))
model.add(Dense(2))
sgd = SGD(lr=0.1) # Stochastic gradient descent optimizer
%time model.compile(loss=tsne, optimizer=sgd)
```

```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<timed eval> in <module>()
```



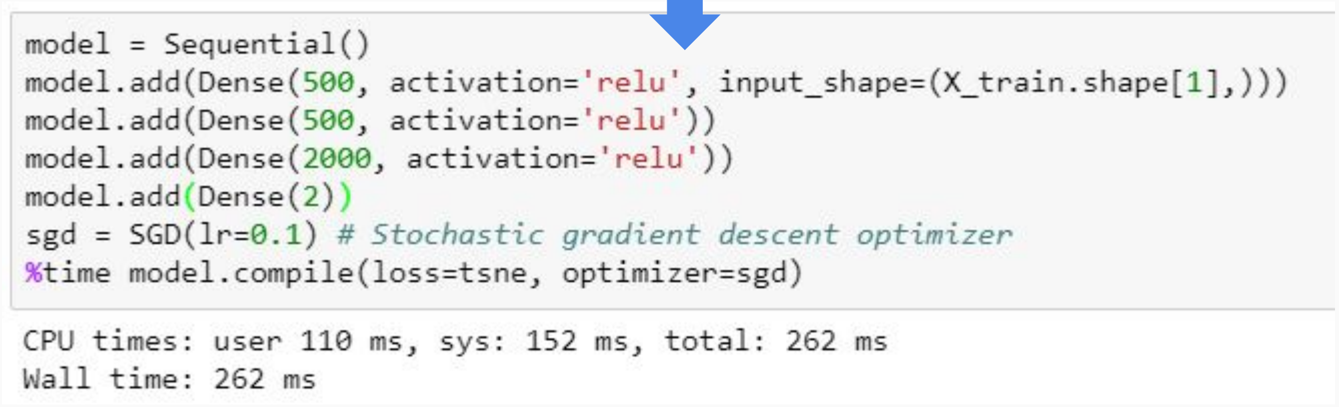
```
/local_data/env-py3/lib/python3.6/site-packages/keras/backend/tensorflow_backend.py in ndim(x)
    592         ```
    593         """
--> 594         dims = x.get_shape()._dims
    595         if dims is not None:
    596             return len(dims)

AttributeError: 'int' object has no attribute 'get_shape'
```

Try to change the output of loss function into an K.constant



```python
# P is the joint probabilities for this batch (Keras loss functions call this y_true)
# activations is the low-dimensional output (Keras loss functions call this y_pred)
def tsne(P, activations):
#     d = K.shape(activations)[1]
    d = 2 # TODO: should set this automatically, but the above is very slow for some reason
    n = batch_size # TODO: should set this automatically
    v = d -  1.
    eps = K.variable(10e-15) # needs to be at least 10e-8 to get anything after Q /= K.sum(Q)
    sum_act = K.sum(K.square(activations), axis=1)
    Q = K.reshape(sum_act, [-1, 1]) + -2 * K.dot(activations, K.transpose(activations))
    Q = (sum_act + Q) / v
    Q = K.pow(1 + Q, -(v + 1) / 2)
    Q *= K.variable(1 - np.eye(n))
    Q /= K.sum(Q)
    Q = K.maximum(Q, eps)
    C = K.log((P + eps) / (Q + eps))
    C = K.sum(P * C)
    #return C
    return K.constant(1)
```

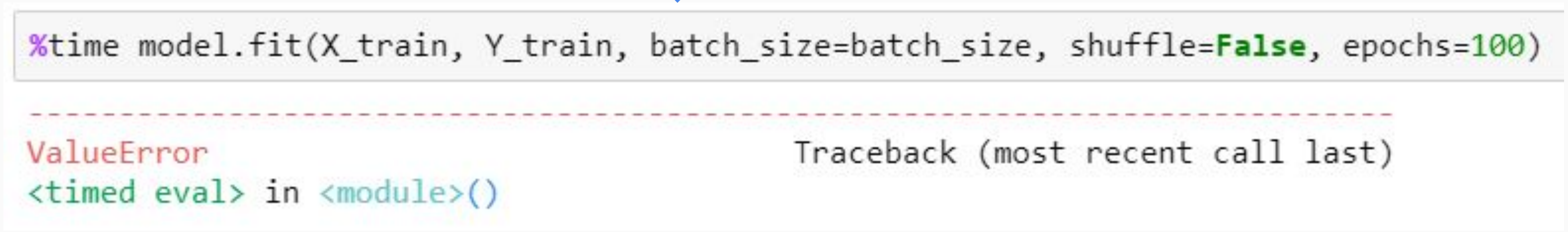
#return C
#return 1
return K.constant(1)



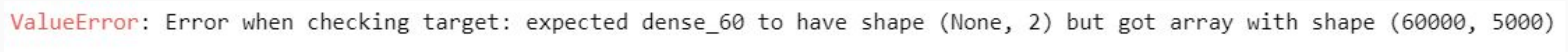
```python
model = Sequential()
model.add(Dense(500, activation='relu', input_shape=(X_train.shape[1],)))
model.add(Dense(500, activation='relu'))
model.add(Dense(2000, activation='relu'))
model.add(Dense(2))
sgd = SGD(lr=0.1) # Stochastic gradient descent optimizer
%time model.compile(loss=tsne, optimizer=sgd)
```

```
CPU times: user 110 ms, sys: 152 ms, total: 262 ms
Wall time: 262 ms
```

Try to change the output of loss function into an K.constant

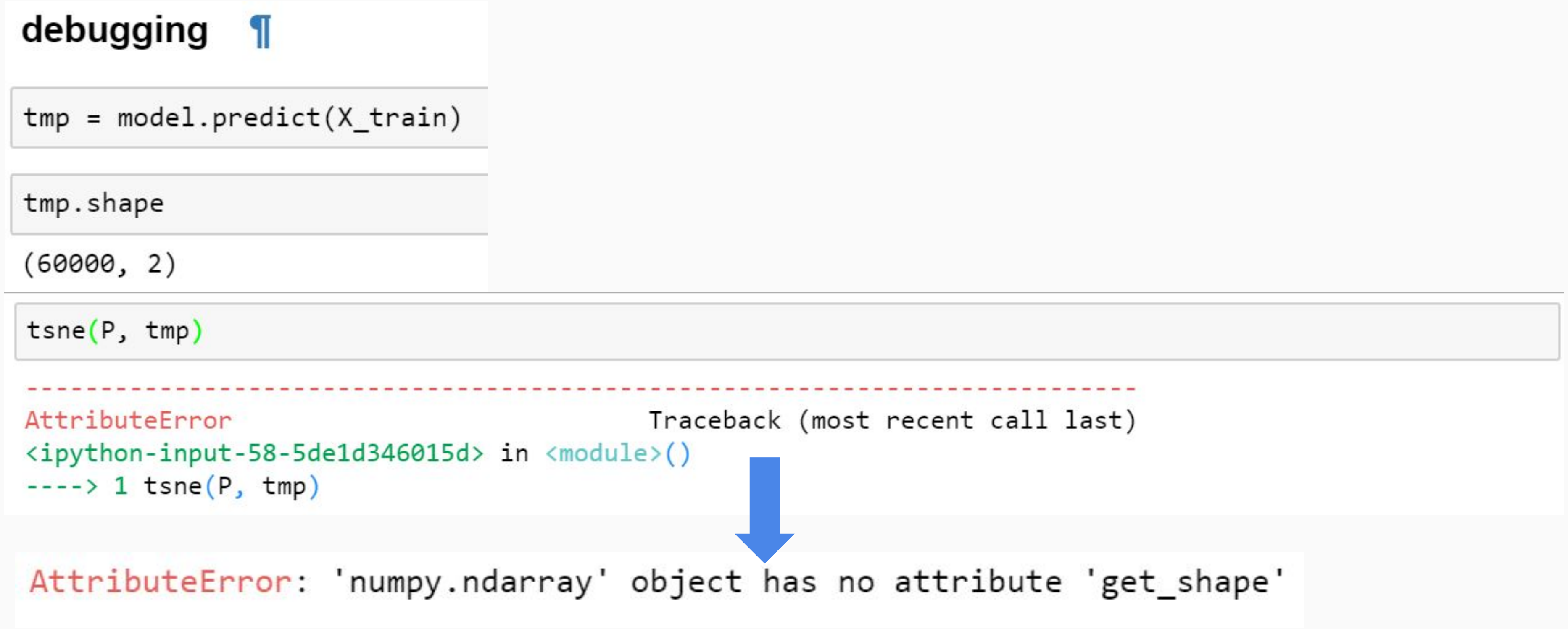


```
%time model.fit(X_train, Y_train, batch_size=batch_size, shuffle=False, epochs=100)

---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<timed eval> in <module>()
```



```
ValueError: Error when checking target: expected dense_60 to have shape (None, 2) but got array with shape (60000, 5000)
```

Try to test the loss function directly



debugging ¶

```
tmp = model.predict(X_train)
```

```
tmp.shape
```

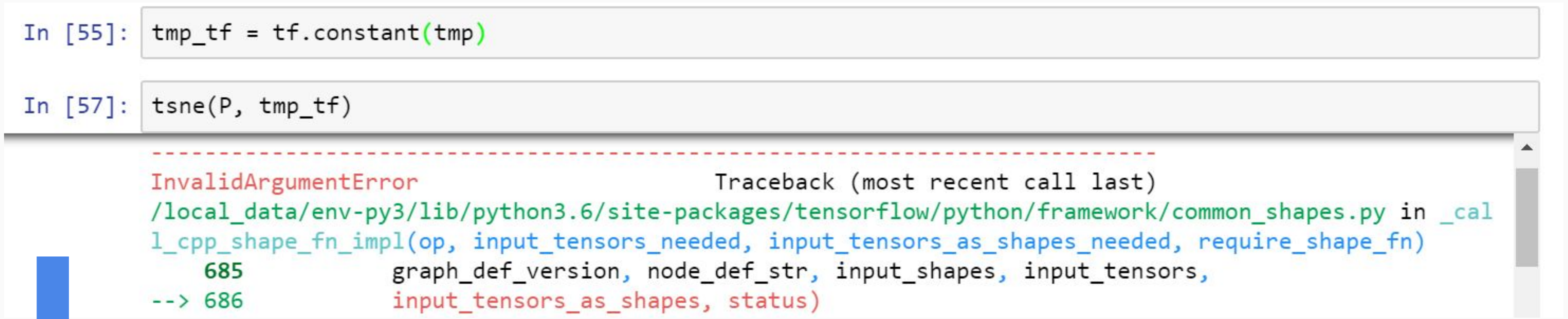
```
(60000, 2)
```

```
tsne(P, tmp)
```

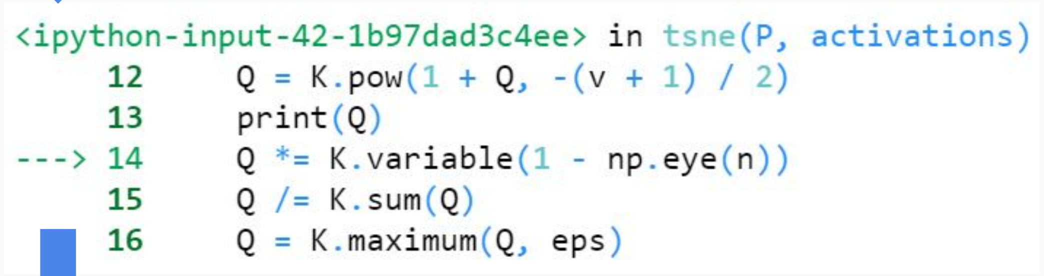
```
---------------------------------------------------------------------------
AttributeError                            Traceback (most recent call last)
<ipython-input-58-5de1d346015d> in <module>()
----> 1 tsne(P, tmp)
```

```
AttributeError: 'numpy.ndarray' object has no attribute 'get_shape'
```

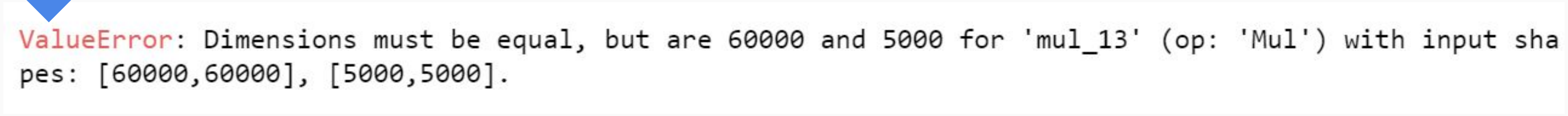
Try to test the loss function directly (change the input into tf.constant)



```
In [55]: tmp_tf = tf.constant(tmp)

In [57]: tsne(P, tmp_tf)

---------------------------------------------------------------------------
InvalidArgumentError                      Traceback (most recent call last)
/local_data/env-py3/lib/python3.6/site-packages/tensorflow/python/framework/common_shapes.py in _cal
l_cpp_shape_fn_impl(op, input_tensors_needed, input_tensors_as_shapes_needed, require_shape_fn)
    685           graph_def_version, node_def_str, input_shapes, input_tensors,
--> 686           input_tensors_as_shapes, status)
```

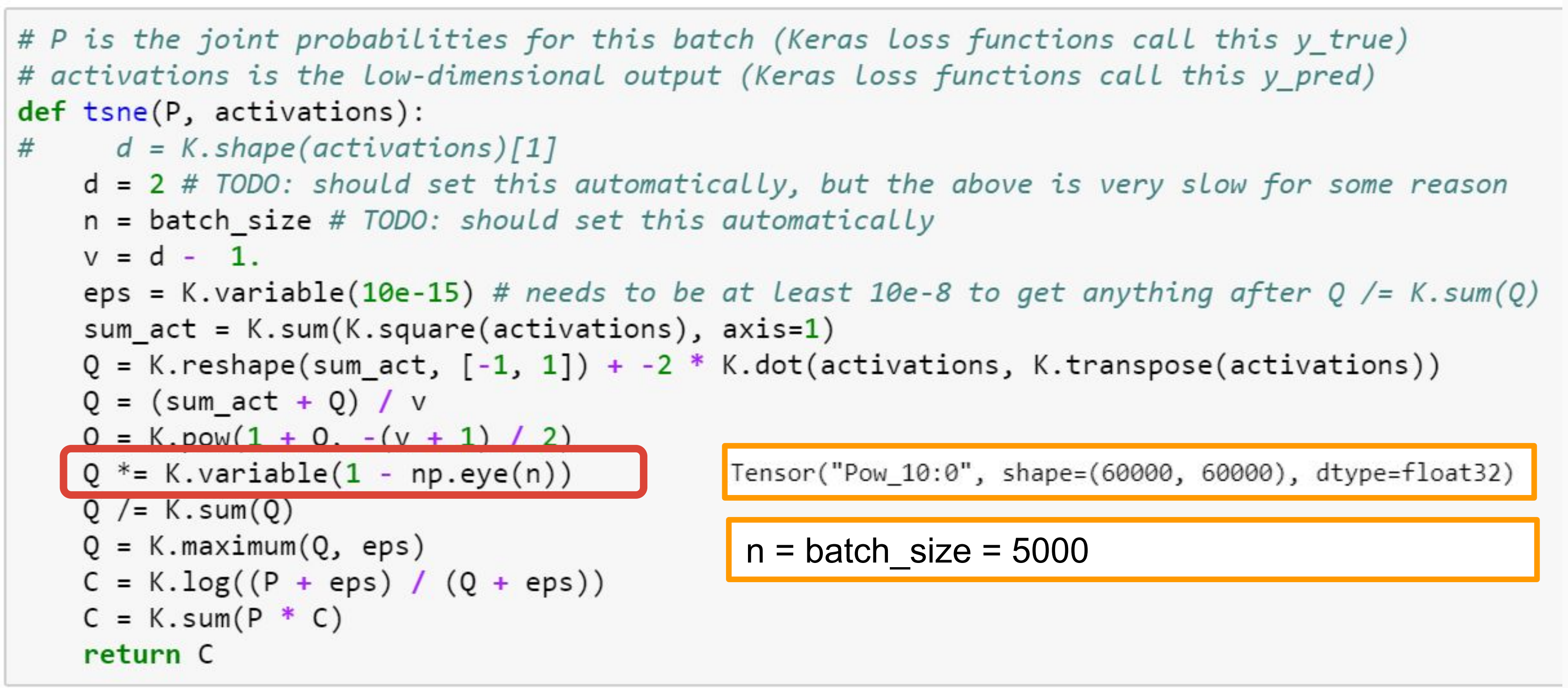


```
<ipython-input-42-1b97dad3c4ee> in tsne(P, activations)
     12     Q = K.pow(1 + Q, -(v + 1) / 2)
     13     print(Q)
---> 14     Q *= K.variable(1 - np.eye(n))
     15     Q /= K.sum(Q)
     16     Q = K.maximum(Q, eps)
```



```
ValueError: Dimensions must be equal, but are 60000 and 5000 for 'mul_13' (op: 'Mul') with input sha
pes: [60000,60000], [5000,5000].
```

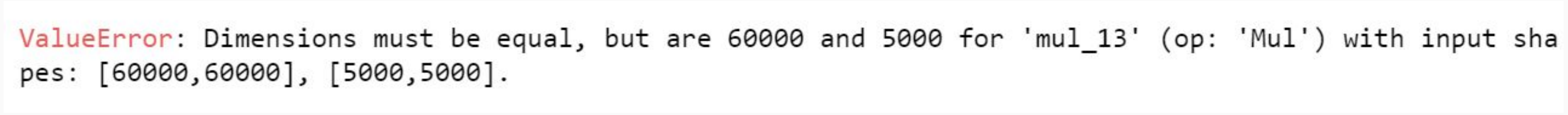
Try to test the loss function directly (change the input into tf.constant)



```
# P is the joint probabilities for this batch (Keras loss functions call this y_true)
# activations is the low-dimensional output (Keras loss functions call this y_pred)
def tsne(P, activations):
#     d = K.shape(activations)[1]
    d = 2 # TODO: should set this automatically, but the above is very slow for some reason
    n = batch_size # TODO: should set this automatically
    v = d -  1.
    eps = K.variable(10e-15) # needs to be at least 10e-8 to get anything after Q /= K.sum(Q)
    sum_act = K.sum(K.square(activations), axis=1)
    Q = K.reshape(sum_act, [-1, 1]) + -2 * K.dot(activations, K.transpose(activations))
    Q = (sum_act + Q) / v
    Q = K.pow(1 + Q, -(v + 1) / 2)
    Q *= K.variable(1 - np.eye(n))
    Q /= K.sum(Q)
    Q = K.maximum(Q, eps)
    C = K.log((P + eps) / (Q + eps))
    C = K.sum(P * C)
    return C
```

```
Tensor("Pow_10:0", shape=(60000, 60000), dtype=float32)
```

n = batch_size = 5000



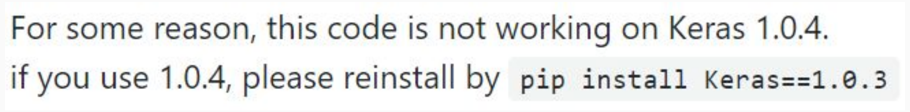
```
ValueError: Dimensions must be equal, but are 60000 and 5000 for 'mul_13' (op: 'Mul') with input sha
pes: [60000,60000], [5000,5000].
```

Try other repo

kylemcdonald/Parametric-t-SNE

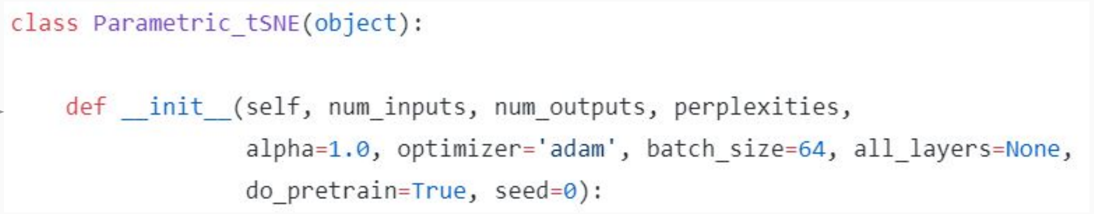
Keras, Theano

zaburo-ch/Parametric-t-SNE-in-Keras

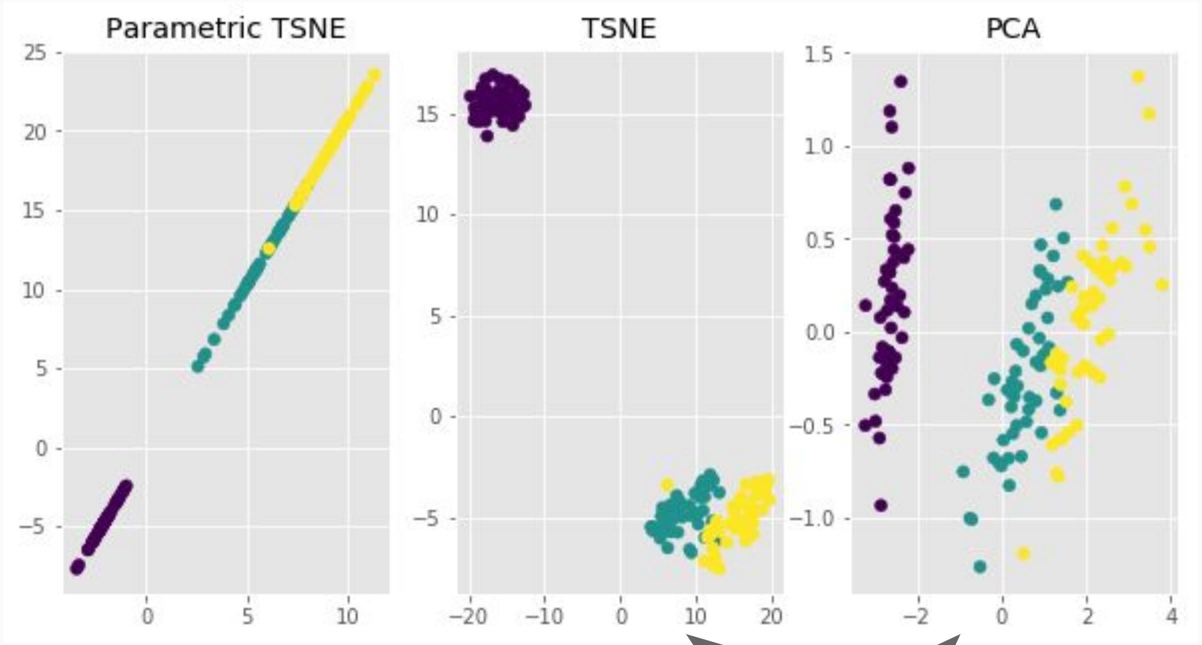

For some reason, this code is not working on Keras 1.0.4.
if you use 1.0.4, please reinstall by `pip install Keras==1.0.3`

jsilter/parametric_tsne

Python / Tensorflow / Keras implementation of Parametric tSNE algorithm


```
class Parametric_tSNE(object):

    def __init__(self, num_inputs, num_outputs, perplexities,
                 alpha=1.0, optimizer='adam', batch_size=64, all_layers=None,
                 do_pretrain=True, seed=0):
```

Try the repo jsilter/parametric_tsne

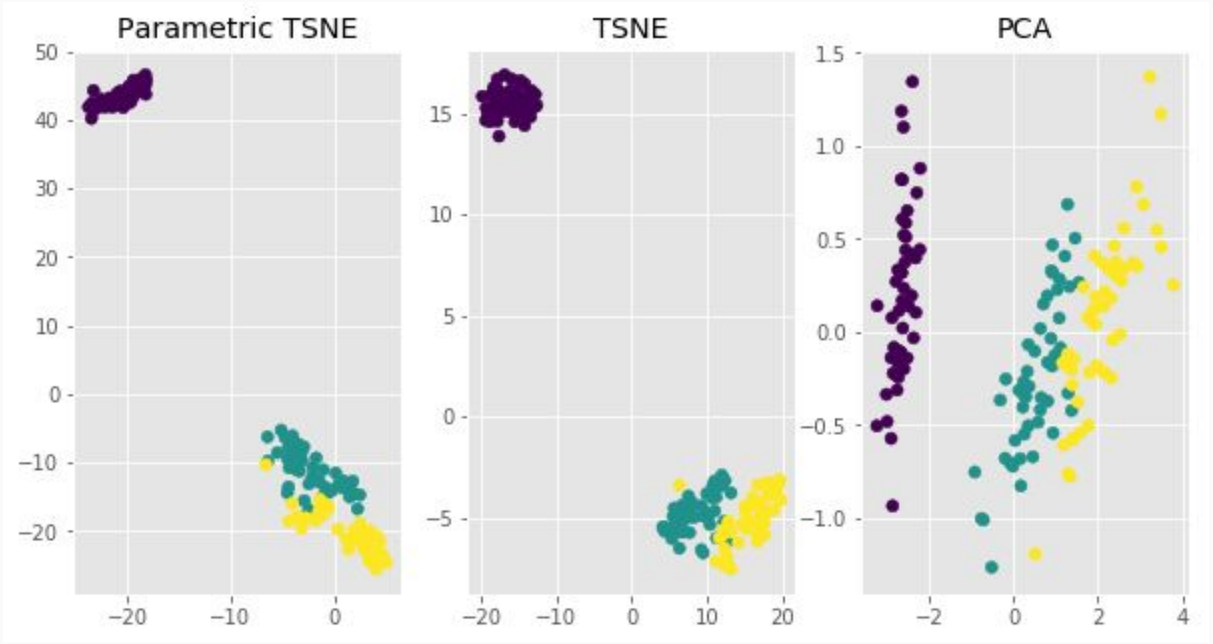


Epochs: 20

From Sklearn

Parametric TSNE ~ TSNE

when increasing the epochs of training



Epochs: 1000