In [1]:

```
import numpy as np
import matplotlib.pyplot as plt
import pickle
import math

import keras
import tensorflow as tf
from keras import backend as K
from keras.layers import Input, Conv2D, MaxPooling2D, UpSampling2D
from keras.models import Model
from keras.callbacks import TensorBoard
from tensorflow.python.keras.engine.base_layer import Layer
```

```
Using TensorFlow backend.
```

Define activation functions

In [2]:

```
def ReLU(X):
    return tf.maximum(X, np.zeros_like(X))

def softmax(X):
    return tf.exp(X)/tf.reduce_sum(tf.exp(X), 0)

def tanh(X):
    return tf.tanh(X)
```

Loading weights from a pre-trained encoder saved in file

In [3]:

```
weights = np.load('./weights/FinalWeightsArray.npy')
weights1 = weights[0]
weights2 = weights[2]
weights3 = weights[4]
weights4 = weights[6]
weights5 = weights[8]
weights6 = weights[10]
weights7 = weights[12]

print(len(weights))
print(weights1.shape,weights2.shape,weights3.shape, weights4.shape, weights5.shape, weigh
ts6.shape, weights7.shape)
```

```
14
(3, 3, 1, 16) (3, 3, 16, 8) (3, 3, 8, 8) (3, 3, 8, 8) (3, 3, 8, 8) (3, 3, 8, 16) (3, 3, 16
, 1)
```

Defining a custom convolution2d function

The function takes tensor as input along with trained kernel weights, kernel size, padding and activation function and outputs a tensor

In [4]:

```
def customConv2D(input_imgs, weights, filters, kernel, strides=1, padding='same', activa
tion='ReLU'):

    if isinstance(kernel, tuple):
```

```
        ker_height, ker_weight = kernel
    else:
        ker_height, ker_weight = kernel, kernel

    batch , inp_h, inp_w, inp_c = input_imgs.shape
    if padding == 'valid':
        pad = 0
    elif padding == 'same':
        pad = (ker_height - 1)//2
    else:
        raise ValueError('Padding '+padding+' given is incorrect')

    conv_height = (inp_h + 2*pad - ker_height)//strides + 1
    conv_weight = (inp_w + 2*pad - ker_weight)//strides + 1

    inp_pad = tf.pad(input_imgs, [[0,0],[pad,pad],[pad,pad],[0,0]])

    c_list = []
    for i in range(conv_height):
        for j in range(conv_weight):
            c = tf.slice(inp_pad, [0, i*strides, j*strides, 0], [-1, ker_height, ker_we
ight, -1])
            c_list.append(c)

    inp_mat = tf.reshape(tf.stack(c_list), [-1, inp_c * ker_weight * ker_height])
    _, _, ker_c, _ = weights.shape
    W_mat = tf.reshape(weights, [ker_height * ker_weight * ker_c, filters])
    b = tf.zeros([filters])

    output = tf.matmul(inp_mat, W_mat) + b
    output = tf.transpose(tf.reshape(output,[conv_height,conv_weight,batch,filters]), [2
,0,1,3])

    if activation:
        activation = activation.lower()
        if activation == 'relu':
            conv = ReLU(output)
        elif activation == 'softmax' or activation == 'sigmoid':
            conv = softmax(output)
        elif activation == 'tanh':
            conv = tanh(output)
        else:
            raise ValueError('Unknown activation')

    return output
```

Defining a custom maxpooling 2d function

The function takes tensor as input with kernel size and padding and outputs a max pooled tensor

In [10]:

```
def customMaxPooling2D(input_imgs, kernel, strides=2, padding='same'):

    if isinstance(kernel, tuple):
        ker_height, ker_weight = kernel
    else:
        ker_height, ker_weight = kernel, kernel

    _, inp_h, inp_w, inp_c = input_imgs.shape

    if padding == 'valid':
        pad = 0
    elif padding == 'same':
        pad = (ker_height-1)//2
    else:
        raise ValueError('Padding '+padding+' given is incorrect')
```

```
    pool_height = math.ceil(int(inp_h + 2*pad - ker_height)/int(strides)) + 1
    pool_weight = math.ceil(int(inp_w + 2*pad - ker_weight)/int(strides)) + 1

    inp_pad = tf.pad(input_imgs, [[0,0],[pad,pad],[pad,pad],[0,0]])

    pool_list = []
    for i in range(pool_height):
        for j in range(pool_weight):
            p = tf.slice(inp_pad, [0, i*strides, j*strides, 0], [-1, ker_height, ker_we
ight, -1])
            pool_list.append(p)

    inp_mat = tf.reshape(tf.stack(pool_list), [-1, ker_weight * ker_height])

    pool_output = tf.reduce_max(tf.reshape(inp_mat, [pool_height, pool_weight, -1, ker_h
eight * ker_weight, int(inp_c)]), axis = 3)
    return tf.transpose(pool_output, [2,0,1,3])
```

Defining a custom upsampling 2d function

The function takes a tensor, kernel size and data format as input and outputs a upsampled tensor with nearest neighbor interpolation method

In [13]:

```
def customUpSampling2D(input_imgs, kernel, data_format='channels_last'):

    if isinstance(kernel, tuple):
        ker_height, ker_weight = kernel
    else:
        ker_height, ker_weight = kernel, kernel

    if (data_format=='channels_last'):
        n, inp_h, inp_w, inp_c = input_imgs.shape
    elif (data_format=='channels_first'):
        n, inp_c, inp_h, inp_w = input_imgs.shape
    else:
        raise ValueError('Data format is not valid.')

    out_height = inp_h * ker_height
    out_weight = inp_w * ker_weight

    up_list = []
    k = 0
    while k < int(n):
        i = 0
        while i < int(inp_h):
            tmp = []
            j = 0
            while j < int(inp_w):
                p = tf.slice(input_imgs, [k,i,j,0], [1,1,1,1])
                for x in range(ker_weight):
                    tmp.append(p)
                j+=1
            for x in range(ker_height):
                for y in tmp:
                    up_list.append(y)
            i+=1
        k+=1
    inp_mat = tf.reshape(tf.stack(up_list), [n, out_height, out_weight])

    out_list = []
    for i in range(int(inp_c)):
        out_list.append(inp_mat)

    out_upsamp = tf.reshape(tf.stack(out_list), [n, out_height, out_weight, int(inp_c)])

    return out_upsamp
```

Load the MNIST dataset

In [7]:

```
from keras.datasets import mnist
(x_train, y_train), (x_test, y_test) = mnist.load_data()

x_train = x_train.astype('float32') / 255.
x_test = x_test.astype('float32') / 255.
x_train = np.reshape(x_train, (len(x_train), 28, 28, 1))
#channels last approach as grayscale (1) is last parameter
x_test = np.reshape(x_test, (len(x_test), 28, 28, 1))

print(x_train.shape, x_test.shape)
```

```
(60000, 28, 28, 1) (10000, 28, 28, 1)
```

Call the encoder part of the custom network

In [11]:

```
input_img = x_train[:10,:,:,:]
print(input_img.shape)
print(type(input_img))

x = customConv2D(input_img, weights1, 16, (3,3), activation = 'relu', padding = 'same')
print(x.shape, type(x))
x = customMaxPooling2D(x, (2,2), padding = 'same')
print(x.shape, type(x))
x = customConv2D(x, weights2, 8, (3,3), activation = 'relu', padding = 'same')
print(x.shape, type(x))
x = customMaxPooling2D(x, (2,2), padding = 'same')
print(x.shape, type(x))
x = customConv2D(x, weights3, 8, (3,3), activation = 'relu', padding = 'same')
print(x.shape, type(x))
encoded = customMaxPooling2D(x, (2,2), padding = 'same')

print(encoded.shape, type(x))
```

```
(10, 28, 28, 1)
<class 'numpy.ndarray'>
(10, 28, 28, 16) <class 'tensorflow.python.framework.ops.Tensor'>
(10, 14, 14, 16) <class 'tensorflow.python.framework.ops.Tensor'>
(10, 14, 14, 8) <class 'tensorflow.python.framework.ops.Tensor'>
(10, 7, 7, 8) <class 'tensorflow.python.framework.ops.Tensor'>
(10, 7, 7, 8) <class 'tensorflow.python.framework.ops.Tensor'>
(10, 4, 4, 8) <class 'tensorflow.python.framework.ops.Tensor'>
```

Call the decoder part of the custom network

In [15]:

```
x = customConv2D(encoded, weights4, 8, (3,3), activation = 'relu', padding = 'same')
print(x.shape, type(x))
x = customUpSampling2D(x, (2,2))
print(x.shape, type(x))
x = customConv2D(x, weights5, 8, (3,3), activation = 'relu', padding = 'same')
print(x.shape, type(x))
x = customUpSampling2D(x, (2,2))
print(x.shape, type(x))
x = customConv2D(x, weights6, 16, (3,3), activation = 'relu', padding = 'valid')
print(x.shape, type(x))
x = customUpSampling2D(x, (2,2))
print(x.shape, type(x))
decoded = customConv2D(x, weights7, 1, (3,3), activation = 'sigmoid', padding = 'same')
```

```
print(decoded.shape, type(x))
```

```
(10, 4, 4, 8) <class 'tensorflow.python.framework.ops.Tensor'>
(10, 8, 8, 8) <class 'tensorflow.python.framework.ops.Tensor'>
(10, 8, 8, 8) <class 'tensorflow.python.framework.ops.Tensor'>
(10, 16, 16, 8) <class 'tensorflow.python.framework.ops.Tensor'>
(10, 14, 14, 16) <class 'tensorflow.python.framework.ops.Tensor'>
(10, 28, 28, 16) <class 'tensorflow.python.framework.ops.Tensor'>
(10, 28, 28, 1) <class 'tensorflow.python.framework.ops.Tensor'>
```

Plot the decoded images

In [16]:

```
n = 10
plt.figure(figsize=(10, 4), dpi=100)

print(decoded)
# x = tf.slice(decoded, [0,0,0,0], [1,-1,-1,-1])
# print(x)
# t = tf.Session().run(x)
sess = tf.InteractiveSession()
decoded.eval()

# type(t)

for i in range(n):
    # display original
    ax = plt.subplot(2, n, i + 1)
    plt.imshow(x_test[i].reshape(28, 28))
    plt.gray()
    ax.set_axis_off()

    # display reconstruction
    ax = plt.subplot(2, n, i + n + 1)
    plt.imshow(tf.reshape((decoded[i,:,:,:]), [28, 28]))
    plt.gray()
    ax.set_axis_off()

plt.show()
```

```
Tensor("transpose_15:0", shape=(10, 28, 28, 1), dtype=float32)

---------------------------------------------------------------------------
InvalidArgumentError                      Traceback (most recent call last)
/usr/lib/python3.7/site-packages/tensorflow/python/client/session.py in _do_call(self, fn
, *args)
   1333     try:
-> 1334       return fn(*args)
   1335     except errors.OpError as e:

/usr/lib/python3.7/site-packages/tensorflow/python/client/session.py in _run_fn(feed_dict
, fetch_list, target_list, options, run_metadata)
   1318       return self._call_tf_sessionrun(
-> 1319           options, feed_dict, fetch_list, target_list, run_metadata)
   1320

/usr/lib/python3.7/site-packages/tensorflow/python/client/session.py in _call_tf_sessionr
un(self, options, feed_dict, fetch_list, target_list, run_metadata)
   1406         self._session, options, feed_dict, fetch_list, target_list,
-> 1407         run_metadata)
   1408

InvalidArgumentError: Expected size[1] in [0, 1], but got 2
  [[{{node Slice_2269}}]]

During handling of the above exception, another exception occurred:

InvalidArgumentError                      Traceback (most recent call last)
<ipython-input-16-2f6948a43103> in <module>()
```

```
      7 # t = tf.Session().run(x)
      8 sess = tf.InteractiveSession()
----> 9 decoded.eval()
     10
     11 # type(t)

/usr/lib/python3.7/site-packages/tensorflow/python/framework/ops.py in eval(self, feed_di
ct, session)
    693
    694     """
--> 695     return _eval_using_default_session(self, feed_dict, self.graph, session)
    696
    697

/usr/lib/python3.7/site-packages/tensorflow/python/framework/ops.py in _eval_using_defaul
t_session(tensors, feed_dict, graph, session)
   5179                       "the tensor's graph is different from the session's "
   5180                       "graph.")
-> 5181   return session.run(tensors, feed_dict)
   5182
   5183

/usr/lib/python3.7/site-packages/tensorflow/python/client/session.py in run(self, fetches
, feed_dict, options, run_metadata)
    927     try:
    928       result = self._run(None, fetches, feed_dict, options_ptr,
--> 929                          run_metadata_ptr)
    930       if run_metadata:
    931         proto_data = tf_session.TF_GetBuffer(run_metadata_ptr)

/usr/lib/python3.7/site-packages/tensorflow/python/client/session.py in _run(self, handle
, fetches, feed_dict, options, run_metadata)
   1150     if final_fetches or final_targets or (handle and feed_dict_tensor):
   1151       results = self._do_run(handle, final_targets, final_fetches,
-> 1152                              feed_dict_tensor, options, run_metadata)
   1153     else:
   1154       results = []

/usr/lib/python3.7/site-packages/tensorflow/python/client/session.py in _do_run(self, han
dle, target_list, fetch_list, feed_dict, options, run_metadata)
   1326     if handle is None:
   1327       return self._do_call(_run_fn, feeds, fetches, targets, options,
-> 1328                            run_metadata)
   1329     else:
   1330       return self._do_call(_prun_fn, handle, feeds, fetches)

/usr/lib/python3.7/site-packages/tensorflow/python/client/session.py in _do_call(self, fn
, *args)
   1346           pass
   1347       message = error_interpolation.interpolate(message, self._graph)
-> 1348       raise type(e)(node_def, op, message)
   1349
   1350   def _extend_graph(self):

InvalidArgumentError: Expected size[1] in [0, 1], but got 2
  [[node Slice_2269 (defined at <ipython-input-10-79811695a0dd>:25) ]]

Caused by op 'Slice_2269', defined at:
  File "/usr/lib/python3.7/runpy.py", line 193, in _run_module_as_main
    "__main__", mod_spec)
  File "/usr/lib/python3.7/runpy.py", line 85, in _run_code
    exec(code, run_globals)
  File "/usr/lib/python3.7/site-packages/ipykernel_launcher.py", line 16, in <module>
    app.launch_new_instance()
  File "/usr/lib/python3.7/site-packages/traitlets/config/application.py", line 658, in l
aunch_instance
    app.start()
  File "/usr/lib/python3.7/site-packages/ipykernel/kernelapp.py", line 486, in start
    self.io_loop.start()
  File "/usr/lib/python3.7/site-packages/tornado/platform/asyncio.py", line 132, in start
    self.asyncio_loop.run_forever()
  File "/usr/lib/python3.7/asyncio/base_events.py", line 539, in run_forever
```

```
  File "/usr/lib/python3.7/asyncio/base_events.py", line 539, in run_forever
    self._run_once()
  File "/usr/lib/python3.7/asyncio/base_events.py", line 1775, in _run_once
    handle._run()
  File "/usr/lib/python3.7/asyncio/events.py", line 88, in _run
    self._context.run(self._callback, *self._args)
  File "/usr/lib/python3.7/site-packages/tornado/platform/asyncio.py", line 122, in _hand
le_events
    handler_func(fileobj, events)
  File "/usr/lib/python3.7/site-packages/tornado/stack_context.py", line 300, in null_wra
pper
    return fn(*args, **kwargs)
  File "/usr/lib/python3.7/site-packages/zmq/eventloop/zmqstream.py", line 450, in _handl
e_events
    self._handle_recv()
  File "/usr/lib/python3.7/site-packages/zmq/eventloop/zmqstream.py", line 480, in _handl
e_recv
    self._run_callback(callback, msg)
  File "/usr/lib/python3.7/site-packages/zmq/eventloop/zmqstream.py", line 432, in _run_c
allback
    callback(*args, **kwargs)
  File "/usr/lib/python3.7/site-packages/tornado/stack_context.py", line 300, in null_wra
pper
    return fn(*args, **kwargs)
  File "/usr/lib/python3.7/site-packages/ipykernel/kernelbase.py", line 283, in dispatche
r
    return self.dispatch_shell(stream, msg)
  File "/usr/lib/python3.7/site-packages/ipykernel/kernelbase.py", line 233, in dispatch_
shell
    handler(stream, idents, msg)
  File "/usr/lib/python3.7/site-packages/ipykernel/kernelbase.py", line 399, in execute_r
equest
    user_expressions, allow_stdin)
  File "/usr/lib/python3.7/site-packages/ipykernel/ipkernel.py", line 208, in do_execute
    res = shell.run_cell(code, store_history=store_history, silent=silent)
  File "/usr/lib/python3.7/site-packages/ipykernel/zmqshell.py", line 537, in run_cell
    return super(ZMQInteractiveShell, self).run_cell(*args, **kwargs)
  File "/usr/lib/python3.7/site-packages/IPython/core/interactiveshell.py", line 2662, in
run_cell
    raw_cell, store_history, silent, shell_futures)
  File "/usr/lib/python3.7/site-packages/IPython/core/interactiveshell.py", line 2785, in
_run_cell
    interactivity=interactivity, compiler=compiler, result=result)
  File "/usr/lib/python3.7/site-packages/IPython/core/interactiveshell.py", line 2901, in
run_ast_nodes
    if self.run_code(code, result):
  File "/usr/lib/python3.7/site-packages/IPython/core/interactiveshell.py", line 2961, in
run_code
    exec(code_obj, self.user_global_ns, self.user_ns)
  File "<ipython-input-11-bd4d3bb724e0>", line 15, in <module>
    encoded = customMaxPooling2D(x, (2,2), padding = 'same')
  File "<ipython-input-10-79811695a0dd>", line 25, in customMaxPooling2D
    p = tf.slice(inp_pad, [0, i*strides, j*strides, 0], [-1, ker_height, ker_weight, -1])
  File "/usr/lib/python3.7/site-packages/tensorflow/python/ops/array_ops.py", line 707, i
n slice
    return gen_array_ops._slice(input_, begin, size, name=name)
  File "/usr/lib/python3.7/site-packages/tensorflow/python/ops/gen_array_ops.py", line 82
36, in _slice
    "Slice", input=input, begin=begin, size=size, name=name)
  File "/usr/lib/python3.7/site-packages/tensorflow/python/framework/op_def_library.py",
line 788, in _apply_op_helper
    op_def=op_def)
  File "/usr/lib/python3.7/site-packages/tensorflow/python/util/deprecation.py", line 507
, in new_func
    return func(*args, **kwargs)
  File "/usr/lib/python3.7/site-packages/tensorflow/python/framework/ops.py", line 3300,
in create_op
    op_def=op_def)
  File "/usr/lib/python3.7/site-packages/tensorflow/python/framework/ops.py", line 1801,
in __init__
    self._traceback = tf_stack.extract_stack()
```

```
InvalidArgumentError (see above for traceback): Expected size[1] in [0, 1], but got 2
  [[node Slice_2269 (defined at <ipython-input-10-79811695a0dd>:25) ]]
```

```
<Figure size 1000x400 with 0 Axes>
```

As the output is a tensor, we needed to convert it into a numpy array before passing it into a matplotlib function to show it as an image. The function to do this in tensorflow is

> **sess = tf.InteractiveSession()**
> **decoded.eval()**

But this gives an InvalidArgumentError and there is no source out here which helps with this certain error. Hence, this is where the whole thing is stuck right now.

In [ ]: