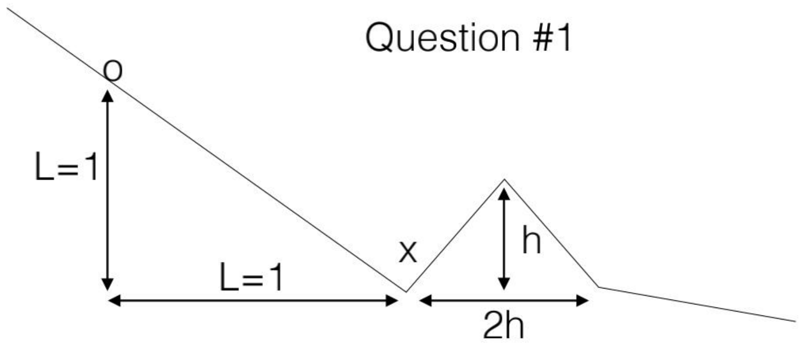
NANYANG TECHNOLOGICAL UNIVERSITY
SINGAPORE

BS6207 Assignment 2

Long Jingyu



The diagram above shows a plot of a 1D function and gradient descend is applied to minimise the function at the point ‘o’. there is a bump a distance L away with bump dimensions given as h x 2h. Let L = 1, a = 0.3 and h > a where a is the learning rate

(a) If we apply standard gradient descent to minimize the function at the point ‘o’:

$\frac{\partial Loss}{\partial y} = -1,\ x < 1$

$\frac{\partial Loss}{\partial y} = 1,\ 1 < x < 1 + h$

$\frac{\partial Loss}{\partial y} = -1,\ \ 1 + h < x < 1 + 2h$

Assume that y = 1 with learning rate a = 0.3,

y = y - a$*\frac{\partial Loss}{\partial y}$ = 1 - 0.3*(-1) = 0.7

y = y - a$*\frac{\partial Loss}{\partial y}$ = 0.7 - 0.3*(-1) = 0.4

y = y - a$*\frac{\partial Loss}{\partial y}$ = 0.4 - 0.3*(-1) = 0.1

y = y - a$*\frac{\partial Loss}{\partial y}$ = 0.1 - 0.3*(-1) = 0.4

y = y - a$*\frac{\partial Loss}{\partial y}$ = 0.4 - 0.3*(1) = 0.1

……

y = y - a$*\frac{\partial Loss}{\partial y}$ = 0.4 - 0.3*(1) = 0.1

At the point (0.9, 0.1) with the learning rate a = 0.3.

```python
# Based on the Qa's formula to get x
def grad(x, h):
    if x<1:
        return -1
    elif 1<x< (1+h):
        return 1
    elif (1+h)< x<(1+2*h):
        return -1
    else:
        return -0.3
# Adam Optimizer
def max_h( deri, n_iter, alpha, beta1, beta2, eps=0):
    for h in np.arange(0.3,1,0.0001):
        x = 0
        m = 0.0
        v = 0.0
        for t in range(1,n_iter):
            g = grad(x,h)
            m = beta1 * m + (1 - beta1) * g
            v = beta2 * v + (1 - beta2) * g**2
            mhat = m / (1.0 - beta1**t)
            vhat = v / (1.0 - beta2**t)
            x = x - alpha * mhat / (vhat**0.5 )
            if x>(1+h):
                break
        if x<(1+h):
            print("The maximum h=", h)
            break
#generate the parameters of Adam.
N_iter = 1000
alpha = 0.3
beta1 = 0.9
beta2 = 0.999
max_h( grad, N_iter, alpha, beta1, beta2)
```

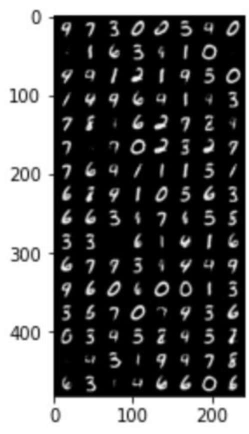
```
The maximum h= 0.41019999999998785
```

The maximum h= 0.41019999999998785

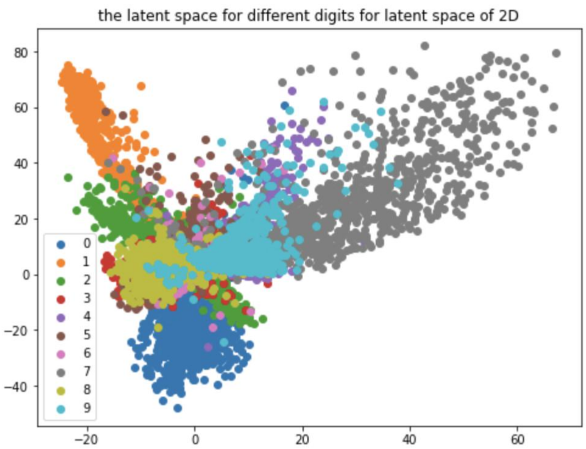
Oscqrgml !0

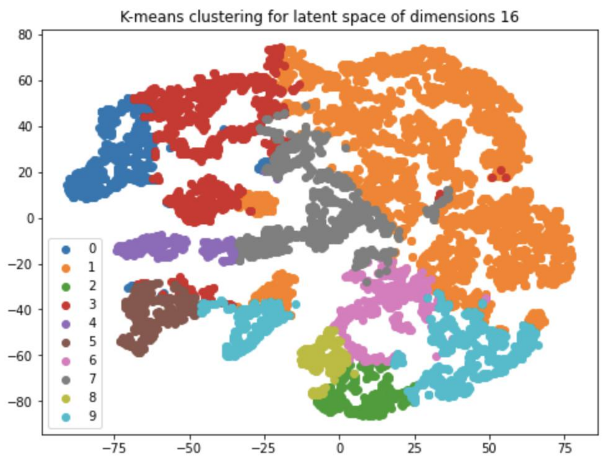
&' Bcqgel _l _srmcl ambcprmr_i c d KLGQR gk _ecq u gf j_rcl r qn_ac bgk cl qgml md0*/ 4*034, Rp_g _srmcl ambcpu gf J/ +l mpk pcaml qrpsargml jmqq ,

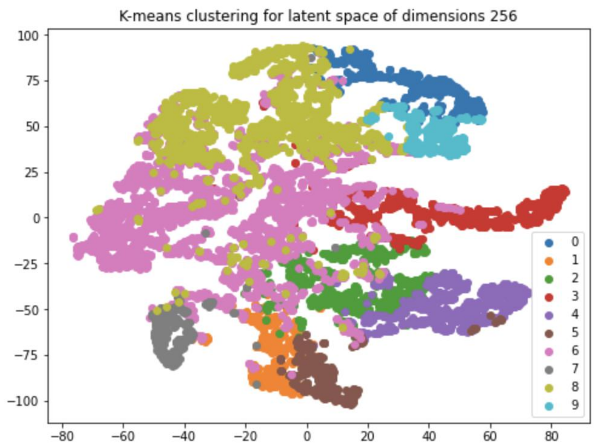
```
epoch: 50, loss is 0.16591624915599823
```



Bm_ 0B njmr mdrf c j_rcl r qn_ac dmp bgddcpcl r bgegrq dmp j_rcl r qn_ac md0B, I +k c_l q ajsqrcpgl e dmp j_rcl r qn_ac mdbgk cl qgml q / 4*034, Sqc ml c amjmp dmp c_af bgegr, Pcnmpr _jj pcqsjrq,

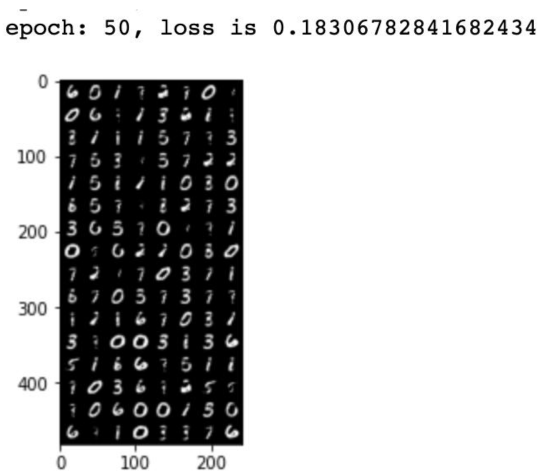




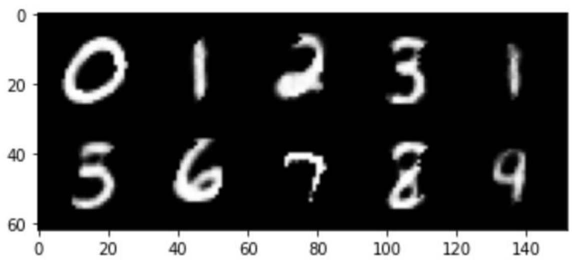


I think K-means clustering for latent space of dimensions 16 present most well.

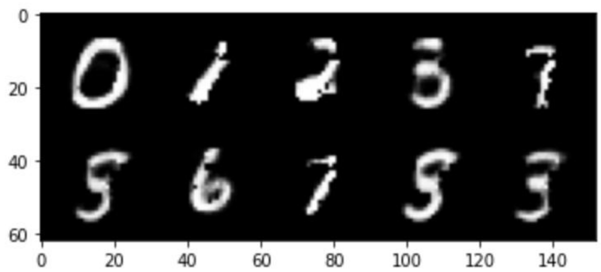
&' Bcqgel _l mrf cpl csp_j l cru mpi ábgq] l crï rmbgqapgk gl _rc ` cru ccl ` jsp gk _ecq _l b ajc_pgk _ecq, @jspgk _ecq a_l ` c ecl cp_rcb ` wr_i gle rf c mpgegl _j KLGQR b_r_ _l b bmqmk c e_sqqg_l ` jsp,



Rp_gl _srmcl ambcpu grf J/ +l mpk pcaml qrpsargml jmqq) bgqapgk gl _rmpjmqq, K_i c pcaml qrpsarcb gk _ecq _q ajc_p _q nmqqg jc* rf _r gq* rf c _srm cl ambcpu gjj l ccb rm` c rp_gl cb qmrf _r ábgq] l crï qampc gr _q _ ajc_pgk _ec,



&_'



&'

Maybe the reconstructed image of (a) is closer to the original image of (b) because the blurring image is used as the input and there is a gap with the original image.