

2019 Spring COM526000 Deep Learning - Homework 5

Variational Autoencoder for Image Reconstruction

105061210 楊雅婷

Problems

1. Image Preprocessing (file: `data_prepro.py`)

- Read the file '`emnist-balanced-train.csv`'.
- We know from the description of the dataset that there are 47 class according to '`0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabdefghnqrt`' characters in the data. However, we only want characters A~Z, which correspond to class 10~35, so I compare column 0 to 9 and 36, keeping data that have class 10~35.
- Transepouse (flip) the data, so that it might look correct when plotting.
- Then adding some random noise to the original data and doing normalize (/255)
- Return the original data (and image), the data added noise (and image added noise), and the labels.
- Additional function '`print_img(vector)`' will show the image reshape from the vector. And the function '`label2char(label)`' will translate the input label to the corresponding character.

2. VAE model with L2 regularization (file: `hw5_105061210.py`)

- I construct a VAE model with an encoder, latent vector, and a decoder
- model:

Encoder	Decoder
Input: (batch_size, 784)	Latent vector dimension: 2 or 10 or 20
Fully connected layer 1: 392 units	Fully connected layer 1: 100 units
Fully connected layer 2: 196 units	Fully connected layer 2: 196 units
Fully connected layer 3: 100 units	Fully connected layer 3: 392 units
Latent vector dimension: 2 or 10 or 20	Output: (batch_size, 784)
Activation function: relu	Activation function: relu

Optimizer: Adam
Code vector (z): $mu + \exp(\sigma)^{1/2} \times \epsilon$, $\epsilon \sim N(0,1)$
Reconstruction loss: $\sum[x \times \log(y) + (1 - x) \times \log(1 - y)]$
KL-divergence: $-\frac{1}{2} \sum(1 + \sigma - mu^2 - \exp(\sigma))$

 Parameters:

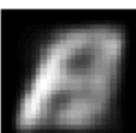
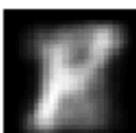
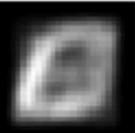
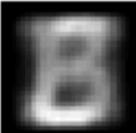
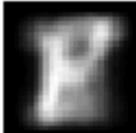
Model: vae_2			
Batch_size	Latent dimension	Learning rate	compare
200	2	0.003	worse
Model: vae_10			
Batch_size	Latent dimension	Learning rate	compare
200	10	0.003	
Model: vae_20			
Batch_size	Latent dimension	Learning rate	compare
200	20	0.003	better
Model: vae_20_100			
Batch_size	Latent dimension	Learning rate	compare
100	20	0.003	
Model: vae_20_300			
Batch_size	Latent dimension	Learning rate	compare
300	20	0.003	

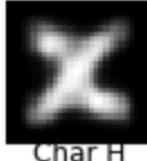
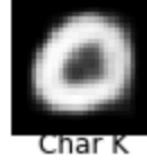
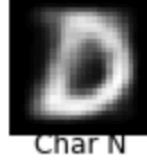
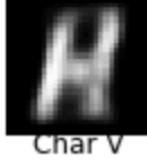
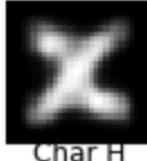
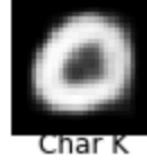
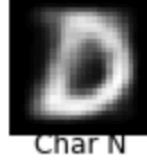
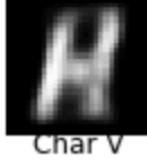
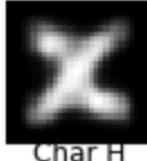
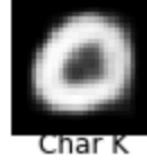
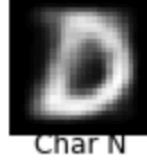
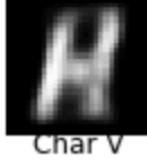
3. Learning curve

Model: vae_2	Model: vae_20_100
<p>Latent dimension = 2 (batch_size = 200)</p>	<p>latent dimension = 20 (batch_size = 100)</p>
<p>Model: vae_10</p> <p>Latent dimension = 10 (batch_size = 200)</p>	<p>Model: vae_20_300</p> <p>latent dimension = 20 (batch_size = 300)</p>
<p>Model: vae_20</p> <p>latent dimension = 20 (batch_size = 200)</p>	<p>note</p> <ol style="list-style-type: none"> 1. Total loss = Reconstruction loss + KL-divergence + Regularization loss 2. Learning curve print: Total loss/784/batch_size

4. show some examples reconstructed from the decoder

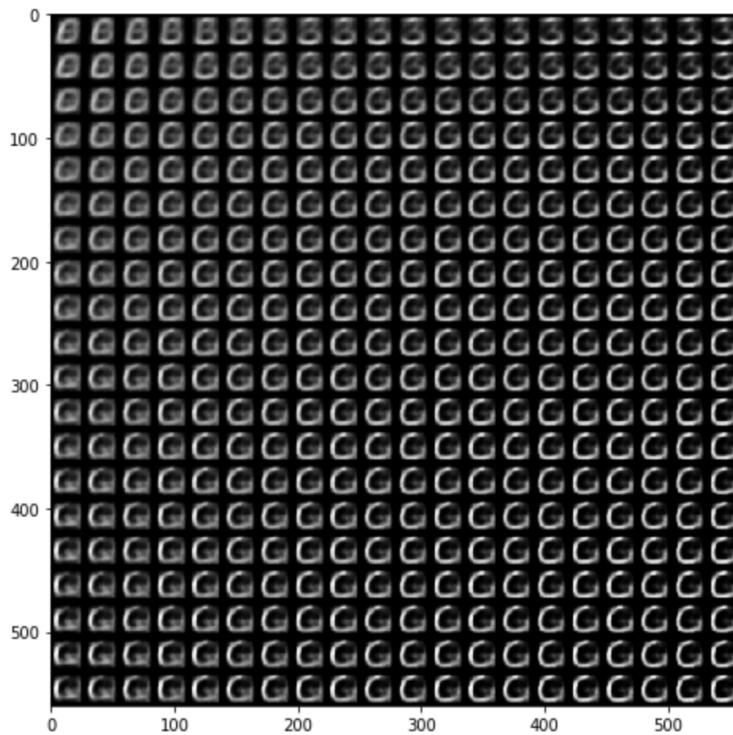
Original input image			Input image with noise		
Char A 	Char R 	Char Z 	Char A 	Char R 	Char Z 
Char O 	Char D 	Char H 	Char O 	Char D 	Char H 
Char K 	Char N 	Char V 	Char K 	Char N 	Char V 

Model: vae_2			Model: vae_20_100		
Char A 	Char R 	Char Z 	Char A 	Char R 	Char Z 
Char O 	Char D 	Char H 	Char O 	Char D 	Char H 
Char K 	Char N 	Char V 	Char K 	Char N 	Char V 
Model: vae_10			Model: vae_20_300		

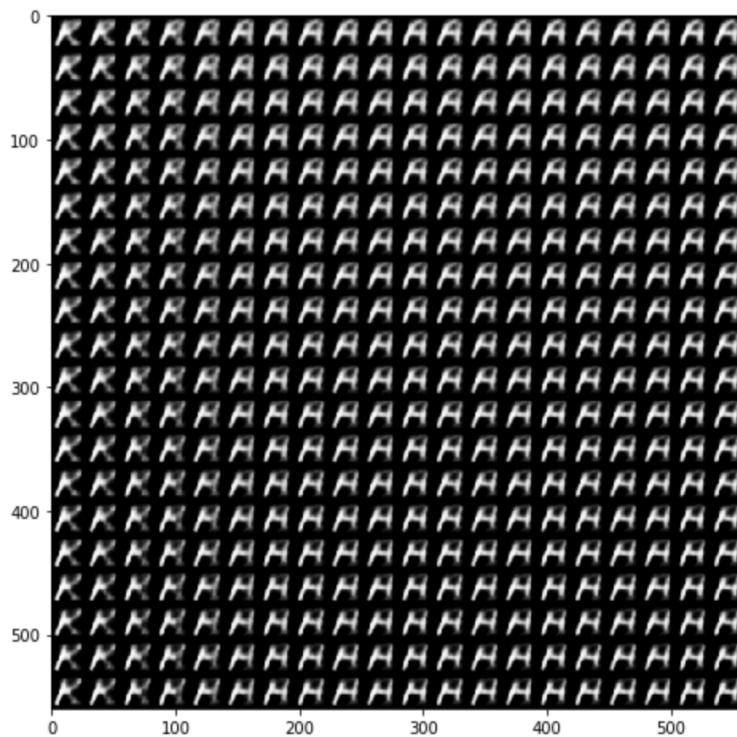
<table border="1"> <tbody> <tr> <td>Char A</td><td>Char R</td><td>Char Z</td></tr> <tr> <td></td><td></td><td></td></tr> <tr> <td>Char O</td><td>Char D</td><td>Char H</td></tr> <tr> <td></td><td></td><td></td></tr> <tr> <td>Char K</td><td>Char N</td><td>Char V</td></tr> <tr> <td></td><td></td><td></td></tr> </tbody> </table>	Char A	Char R	Char Z				Char O	Char D	Char H				Char K	Char N	Char V				<table border="1"> <tbody> <tr> <td>Char A</td><td>Char R</td><td>Char Z</td></tr> <tr> <td></td><td></td><td></td></tr> <tr> <td>Char O</td><td>Char D</td><td>Char H</td></tr> <tr> <td></td><td></td><td></td></tr> <tr> <td>Char K</td><td>Char N</td><td>Char V</td></tr> <tr> <td></td><td></td><td></td></tr> </tbody> </table>	Char A	Char R	Char Z				Char O	Char D	Char H				Char K	Char N	Char V			
Char A	Char R	Char Z																																			
																																					
Char O	Char D	Char H																																			
																																					
Char K	Char N	Char V																																			
																																					
Char A	Char R	Char Z																																			
																																					
Char O	Char D	Char H																																			
																																					
Char K	Char N	Char V																																			
																																					
<p>Model: vae_20</p>	<p>note</p>																																				
<table border="1"> <tbody> <tr> <td>Char A</td><td>Char R</td><td>Char Z</td></tr> <tr> <td></td><td></td><td></td></tr> <tr> <td>Char O</td><td>Char D</td><td>Char H</td></tr> <tr> <td></td><td></td><td></td></tr> <tr> <td>Char K</td><td>Char N</td><td>Char V</td></tr> <tr> <td></td><td></td><td></td></tr> </tbody> </table>	Char A	Char R	Char Z				Char O	Char D	Char H				Char K	Char N	Char V																						
Char A	Char R	Char Z																																			
																																					
Char O	Char D	Char H																																			
																																					
Char K	Char N	Char V																																			
																																					

5. plot the reconstructed images by varying the value of the latent variable.

✍ Use model vae_2

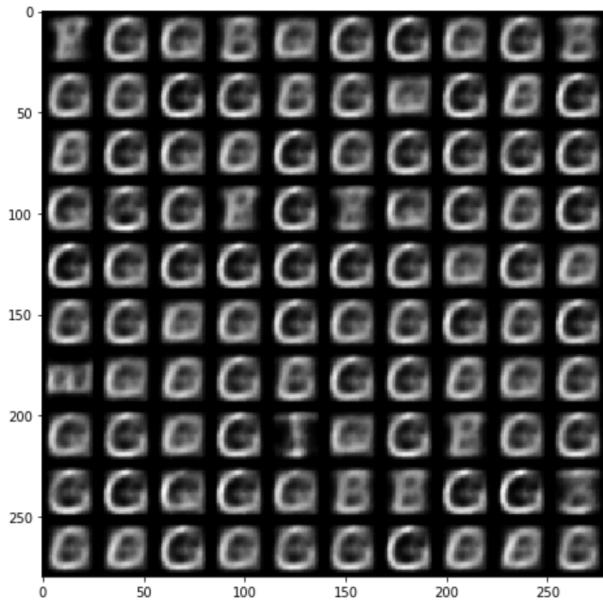


✍ Use model vae_20



6. Sample noise vectors from $N(0,1)$ as latent variables and use the decoder to generate some images.

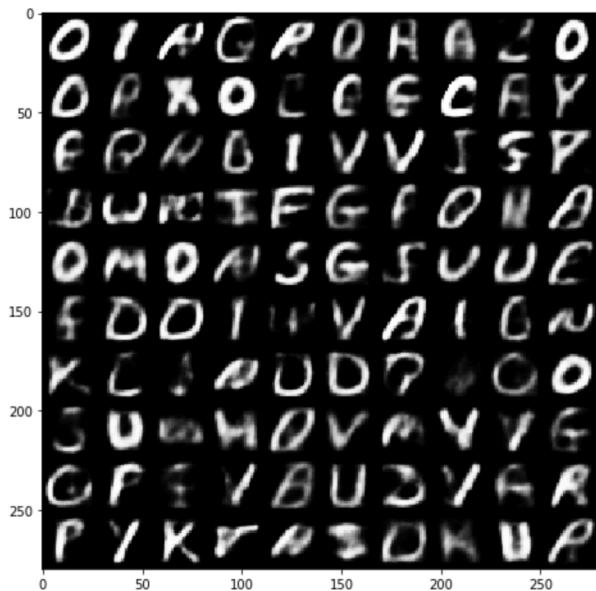
✍ **Use model vae_2**



✍ **Use model vae_10**



✍ **Use model vae_20**



- Additional: reconstructed results from model vae_20

0 F R E E P D O C H G E M J T H Y Z E Z O N S P I A M Y I W R K A
W Y T K V E G N S O V S K R T R E X O D A K N Y A K Y A S N O C
E E K P Z M I V U I H R I V O X R N B W U G G R E O Z O E N T O
I A X C B C H I T E U X R E A Q X C T A B I B U R M F G Y R K U
R K O O R T O G O H N N E N M H E B P I G X D Y Z N K Z m k i p
M P I B E R Y F D E I M E S T I M O K R H W I P O B I N P S u z Q
B I N U B S I T K D W T P A G N L M E T P O I R T N R M C C F E
O A B A P H R V G A M S V V X G B G I T W A V E N I Q D E P Z
H E M H X I N T O I B I A N W P E K I U X C A P X U M I I D E R
C E A B O I V E X A J E P N E R E X G O U T D T N M C H M V W B
R Z I B Z O J O P I V P H W E S B I A S P U C F X U I B B Q R
V K E H H C D Z I V A N G P U B O Z E T H S E Z U V B O A N B O B
G w E M E W G M P H R E N W S R V H A R K P S X Z K C D m V N E B
B R Q C I B C Z S O Q D I P U E T U S D I Y Z R A Z R I H I B W E
T I N A N V B R E D O B A N G Z M E I W H T E I W R J W D U I T H A
m Q R W R A R K T U S V U E P X A Y P G O N R m q u m i l e m x d
N P R H M M V H m b P I R X M I X I I D O R U W W R J K G E P E
D R K U Y I S E R T I U B V E I E H B V R K P X Q I M P H m j p
H O U E K H m s k o b v v b s n c a n w i k h c j x o j v o i
I D C X B G I U I S A M O Y I E M A J O E R B R Y B O R E L I T
P J O P P R Q I S G C I E V V U W I A R E R Y X X O E I P W A
I J X I P B X V N N W O U I I J O R B B Z R P W E I X V Y I S
Q J A Z I V Y D N B R M L G H A U J S R A I O N N V O U N R A H V E N
Z P H E N H J E E R S A P P R A Y W I V O I I S R Y Y H O N O I N B X
J R I S Y X X N O I A H V R S R K U E W D A W I A N Q P A R E A B Y
G W H N B W P G S R H B Z U T S K T V S Y X E E I P V U E F G Y P
K R E Z A T K K P Q E S C S C C F Y R I K E Z P T N V R T T S Y
T I J E F A S E S H Y A R I X O H M S F Y U G O E W W W W F R D
B U F m S U B F Z K Y L O E N X P Y A P H M A A H K T D Q I Z P
E C R A X H W N T E I P D N S H D O B B A N Y T I E Z G A X A N
G E W A H J I X C E L U T W E W P O R E C X I S G R I B K E N V
E M V O G G E R A Y I D A N X R V B A B G Y I O D E U V Y B T K D