

به نام خداوند بخشنده و مهربان

یادگیری عمیق

پروژه درس

محسن نقی پورفر ۹۴۱۰۶۷۵۷

۱ توضیح مختصری از کد

در این قسمت یک توضیح مختصر از کد زده شده داده می‌شود. کد این پروژه به صورت ماژولار پیاده‌سازی شده است که یک پکیج را می‌سازد. این پکیج به این صورت است از چند فایل کمکی مثل `utils`، `dataset` و ... کمک می‌گیرد. توابع کمکی در فایل `utils` و تمام اعمال مورد نیاز برای دیتاست در فایل `dataset` ذخیره شده است. همچنین در فایل‌های `GAN`، `CGAN` معماری‌های مورد نظر برای شبکه‌های مولد پیاده‌سازی شده است. برای این پکیج از `Tensorflow` استفاده شده است. همچنین کد موجود در فایل `CNN`، یک شبکه کانولوشنی عمیق برای دسته‌بندی استفاده شده است.

۱.۱ توضیح فایل `Dataset.py`

در این فایل یک کلاس `DataLoader` پیاده‌سازی شده است که این کلاس برای اداره کردن همه توابع مختلف لازم برای داده‌های مورد نیاز برای شبکه‌های عصبی استفاده می‌شود.

۲.۱ توضیح فایل `GAN.py`

در این فایل یک شبکه عصبی `GAN` به وسیله خود پکیج `tensoflow` پیاده‌سازی شده است. این شبکه در قالب یک کلاس پیاده‌سازی شده است.

۳.۱ توضیح فایل `CGAN.py`

در این فایل، همانطور که از اسم آن نیز پیداست، شبکه عصبی `CGAN` به وسیله پکیج `Keras` پیاده‌سازی شده است. این پیاده‌سازی به صورت شی گرا بوده است. همچنین همه توابع لازم برای رسم نمودار، کشیدن کاراکترهای جدید و ذخیره فایل لاگ حاصل از یادگیری این شبکه نیز پیاده‌سازی شده است.

۴.۱ توضیح فایل `CNN.py`

در این فایل شبکه کانولوشنی عمیق برای تشخیص نوع کاراکتر ورودی برای داده حجیم داده شده استفاده شده است.

۲ توضیح معماری `CNN`

برای ساخت شبکه `CNN` برای دسته‌بندی حروف مختلف در بین فونت‌ها، از یک شبکه عصبی عمیقی استفاده شده است که دارای ۵ لایه کانولوشن و `Maxpooling` که به معماری آن به صورت زیر بوده است. شکل مربوط به معماری این شبکه عصبی در زیر آمده است. برای عمیق‌تر کردن شبکه عصبی از ایده شبکه معروف `VGG` استفاده شده است و از فیلترهای کوچک استفاده شده است تا شبکه عمیق‌تر شود البته در لایه از فیلتر ۱۱ در ۱۱ استفاده شده است.

Layer (type)	Output Shape	Param #
inputs (InputLayer)	(None, 32, 32, 1)	0
flatten_1 (Flatten)	(None, 1024)	0
gaussian_noise_1 (GaussianNo	(None, 1024)	0
reshape_1 (Reshape)	(None, 32, 32, 1)	0
conv_2 (Conv2D)	(None, 32, 32, 64)	1664
leaky_re_lu_1 (LeakyReLU)	(None, 32, 32, 64)	0
dropout_1 (Dropout)	(None, 32, 32, 64)	0
conv_3 (Conv2D)	(None, 32, 32, 64)	102464
leaky_re_lu_2 (LeakyReLU)	(None, 32, 32, 64)	0
dropout_2 (Dropout)	(None, 32, 32, 64)	0
max_pooling2d_1 (MaxPooling2	(None, 16, 16, 64)	0
conv_4 (Conv2D)	(None, 16, 16, 128)	131200
leaky_re_lu_3 (LeakyReLU)	(None, 16, 16, 128)	0
dropout_3 (Dropout)	(None, 16, 16, 128)	0
conv_5 (Conv2D)	(None, 16, 16, 128)	262272
leaky_re_lu_4 (LeakyReLU)	(None, 16, 16, 128)	0
dropout_4 (Dropout)	(None, 16, 16, 128)	0
max_pooling2d_2 (MaxPooling2	(None, 8, 8, 128)	0
conv2d_1 (Conv2D)	(None, 8, 8, 128)	147584
leaky_re_lu_5 (LeakyReLU)	(None, 8, 8, 128)	0
dropout_5 (Dropout)	(None, 8, 8, 128)	0
max_pooling2d_3 (MaxPooling2	(None, 4, 4, 128)	0
flatten_2 (Flatten)	(None, 2048)	0
dense_1 (Dense)	(None, 256)	524544

شکل ۱: شبکه عصبی عمیق کانولوشنی استفاده شده در قسمت اول پروژه

۳ توضیح معماری Generator

برای شبکه Generator موجود در معماری‌های CGAN و GAN از دو معماری مختلف استفاده شده است. یکی شبکه‌ای کانولوشنی که از لایه Latent (به همراه لایه Condition در شبکه CGAN) و دیگری شبکه عصبی Fully Connected خالص که هر دو آزمایش شده‌اند و نتایج حاصل از آن‌ها موجود می‌باشد. برای شبکه CGAN، لایه Condition به لایه Latent چسبانده (Concatenate) شده است. شکل مربوط به معماری شبکه Generator در زیر آمده است.

Layer (type)	Output Shape	Param #	Connected to
gen_latent (InputLayer)	(None, 500)	0	
gen_condition (InputLayer)	(None, 26)	0	
concatenate_1 (Concatenate)	(None, 526)	0	gen_latent[0][0] gen_condition[0][0]
dense_1 (Dense)	(None, 256)	134912	concatenate_1[0][0]
leaky_re_lu_1 (LeakyReLU)	(None, 256)	0	dense_1[0][0]
dense_2 (Dense)	(None, 512)	131584	leaky_re_lu_1[0][0]
leaky_re_lu_2 (LeakyReLU)	(None, 512)	0	dense_2[0][0]
dense_3 (Dense)	(None, 1024)	525312	leaky_re_lu_2[0][0]
activation_1 (Activation)	(None, 1024)	0	dense_3[0][0]
Total params: 791,808			
Trainable params: 791,808			
Non-trainable params: 0			

شکل ۲: معماری شبکه Generator

۴ توضیح معماری Discriminator

برای شبکه Discriminator موجود در معماری‌های CGAN و GAN از دو معماری مختلف استفاده شده است. یکی شبکه‌ای کانولوشنی که از لایه Image (به همراه لایه Condition در شبکه CGAN) و دیگری شبکه عصبی Fully Connected خالص که هر دو آزمایش شده‌اند و نتایج حاصل از آن‌ها موجود می‌باشد. برای شبکه CGAN، لایه Condition در حالت Fully Connected به لایه Image چسبانده (Concatenate) شده است. در واقع در این شبکه لایه Image ابتدا Flat شده است و یک لایه حاوی ۱۰۲۴ نورون را تشکیل داده است. شکل مربوط به معماری شبکه Discriminator در زیر آمده است.

Layer (type)	Output Shape	Param #	Connected to
disc_image (InputLayer)	(None, 1024)	0	
disc_condition (InputLayer)	(None, 26)	0	
concatenate_2 (Concatenate)	(None, 1050)	0	disc_image[0][0] disc_condition[0][0]
dense_4 (Dense)	(None, 1024)	1076224	concatenate_2[0][0]
leaky_re_lu_3 (LeakyReLU)	(None, 1024)	0	dense_4[0][0]
dense_5 (Dense)	(None, 512)	524800	leaky_re_lu_3[0][0]
leaky_re_lu_4 (LeakyReLU)	(None, 512)	0	dense_5[0][0]
dense_6 (Dense)	(None, 256)	131328	leaky_re_lu_4[0][0]
leaky_re_lu_5 (LeakyReLU)	(None, 256)	0	dense_6[0][0]
dense_7 (Dense)	(None, 1)	257	leaky_re_lu_5[0][0]
activation_2 (Activation)	(None, 1)	0	dense_7[0][0]
Total params: 3,465,218			
Trainable params: 1,732,609			
Non-trainable params: 1,732,609			

شکل ۳: معماری شبکه Discriminator

Layer (type)	Output Shape	Param #	Connected to
gen_latent (InputLayer)	(None, 500)	0	
gen_condition (InputLayer)	(None, 26)	0	
gen_model (Model)	(None, 1024)	791808	gen_latent[0][0] gen_condition[0][0]
disc_condition (InputLayer)	(None, 26)	0	
disc_image (InputLayer)	(None, 1024)	0	
disc_model (Model)	(None, 1)	1732609	disc_image[0][0] disc_condition[0][0] gen_model[1][0] disc_condition[0][0]
activation_4 (Activation)	(None, 1)	0	disc_model[2][0]
activation_3 (Activation)	(None, 1)	0	disc_model[1][0]
Total params: 2,524,417			
Trainable params: 791,808			
Non-trainable params: 1,732,609			

شکل ۴: معماری کلی شبکه CGAN در زمان آموزش شبکه Generator

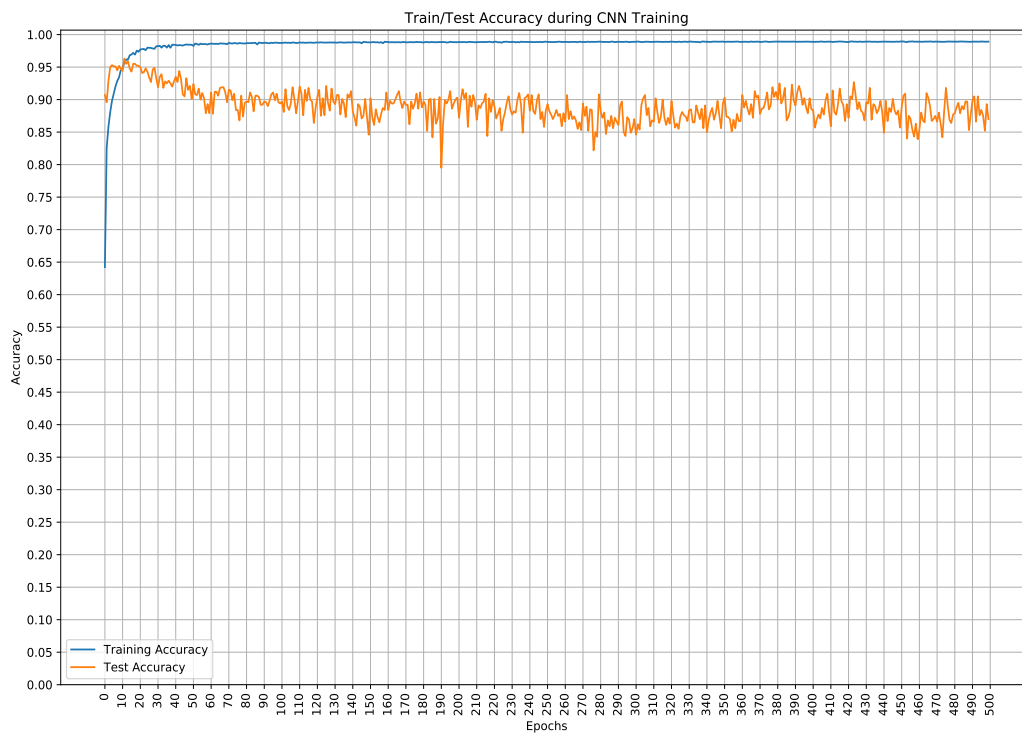
۵ توضیح نحوه آموزش شبکه‌های مولد

برای آموزش شبکه‌های مولد، از Trick‌های مختلفی استفاده شده است. این تریک‌ها در زیر آورده شده‌اند.

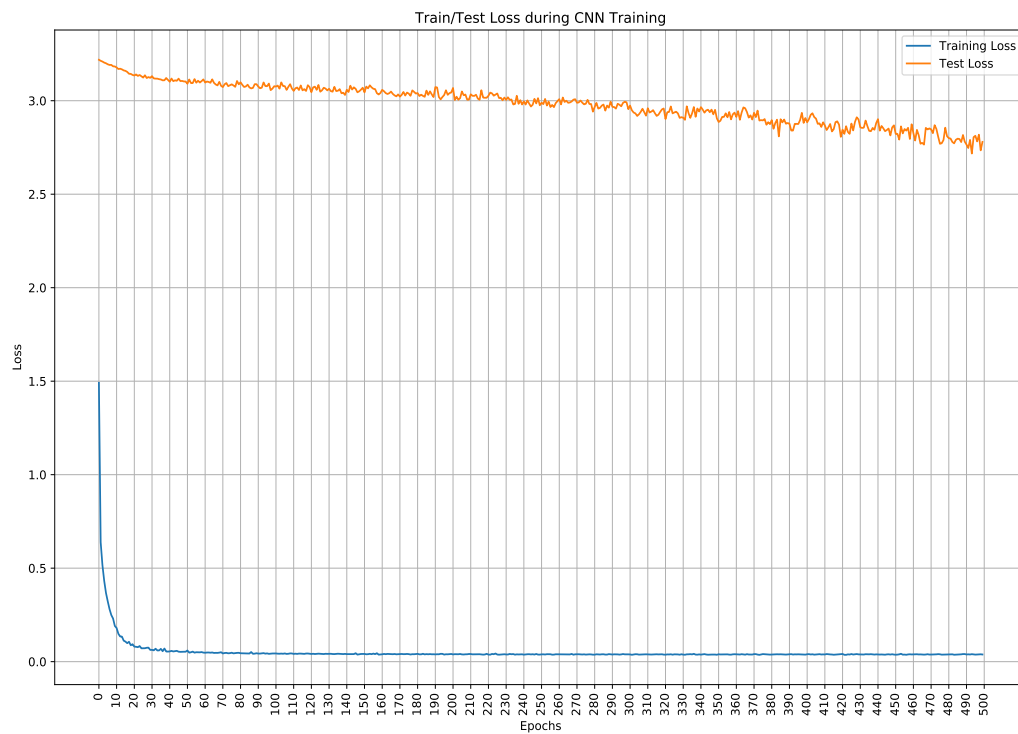
- استفاده از الگوریتم بهینه‌سازی Adam برای شبکه Generator و الگوریتم بهینه‌سازی SGD برای شبکه Discriminator
- استفاده از تکنیک Label Smoothing برای آموزش هر دو شبکه Generator و Discriminator
- استفاده از تابع فعال‌ساز LeakyReLU با پارامتر ۰.۲ بعد از هر لایه در معماری شبکه‌های Generator و Discriminator
- آموزش جدای داده‌های real و fake به شبکه Discriminator به جای آموزش ترکیبی از آنها (concatenation از آنها)
- نرمالیزه کردن دیتای ورودی برای آموزش شبکه‌های مولد (نگاشت بازه هر پیکسل ورودی به ۱ و -۱)
- آموزش داده‌های واقعی با برچسب صفر و داده‌های غیر واقعی با برچسب ۱ برای شبکه Generator

۶ نتایج بدست آمده در قسمت اول

برای شبکه اول، از تمام داده‌های Training برای آموزش استفاده شده است و از داده X_test داده شده برای داده Validation و ارزیابی شبکه استفاده شده است. نمودار یادگیری برای داده آموزش و تست برای این شبکه در زیر آمده است.



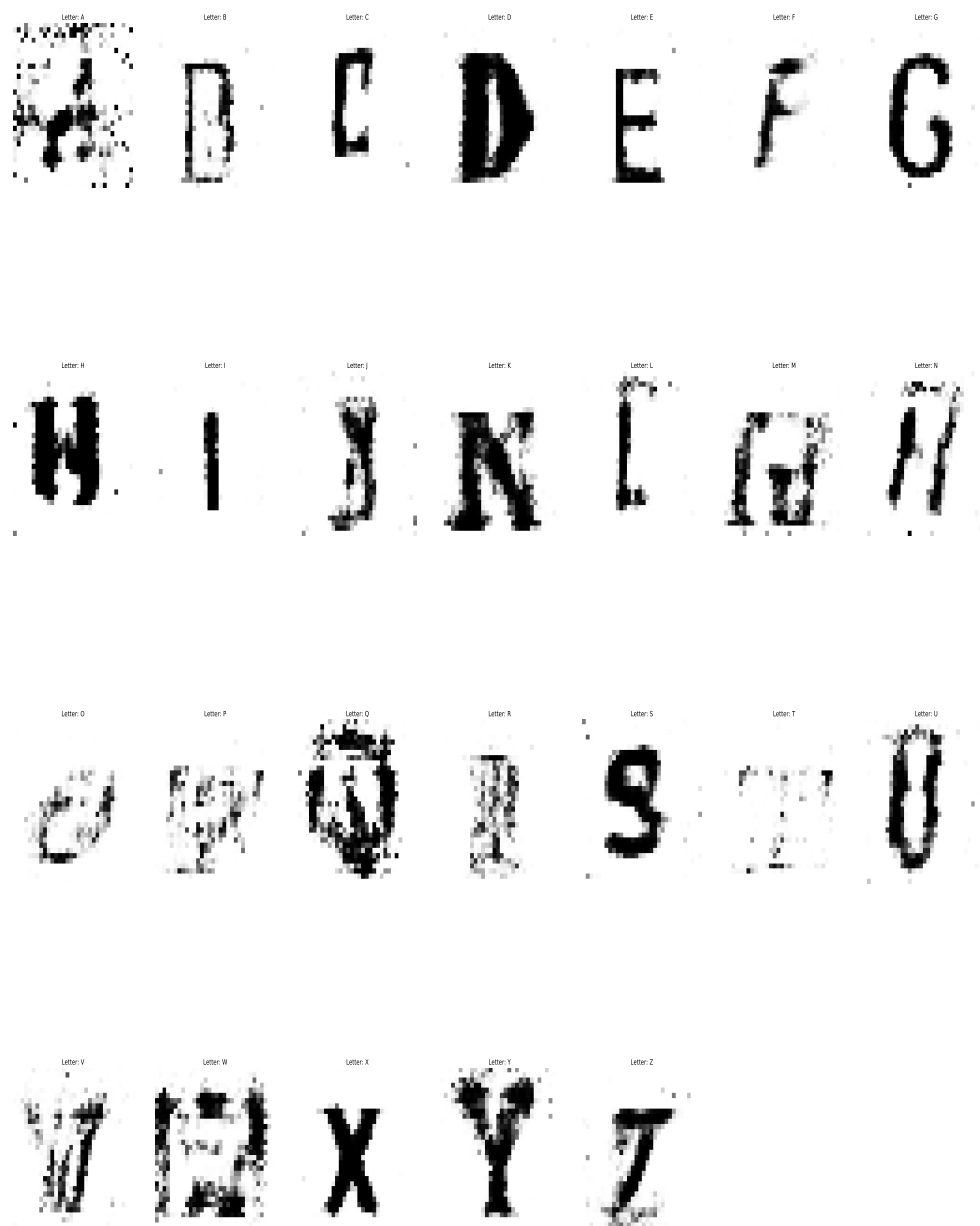
شکل ۵: دقت شبکه DCNN

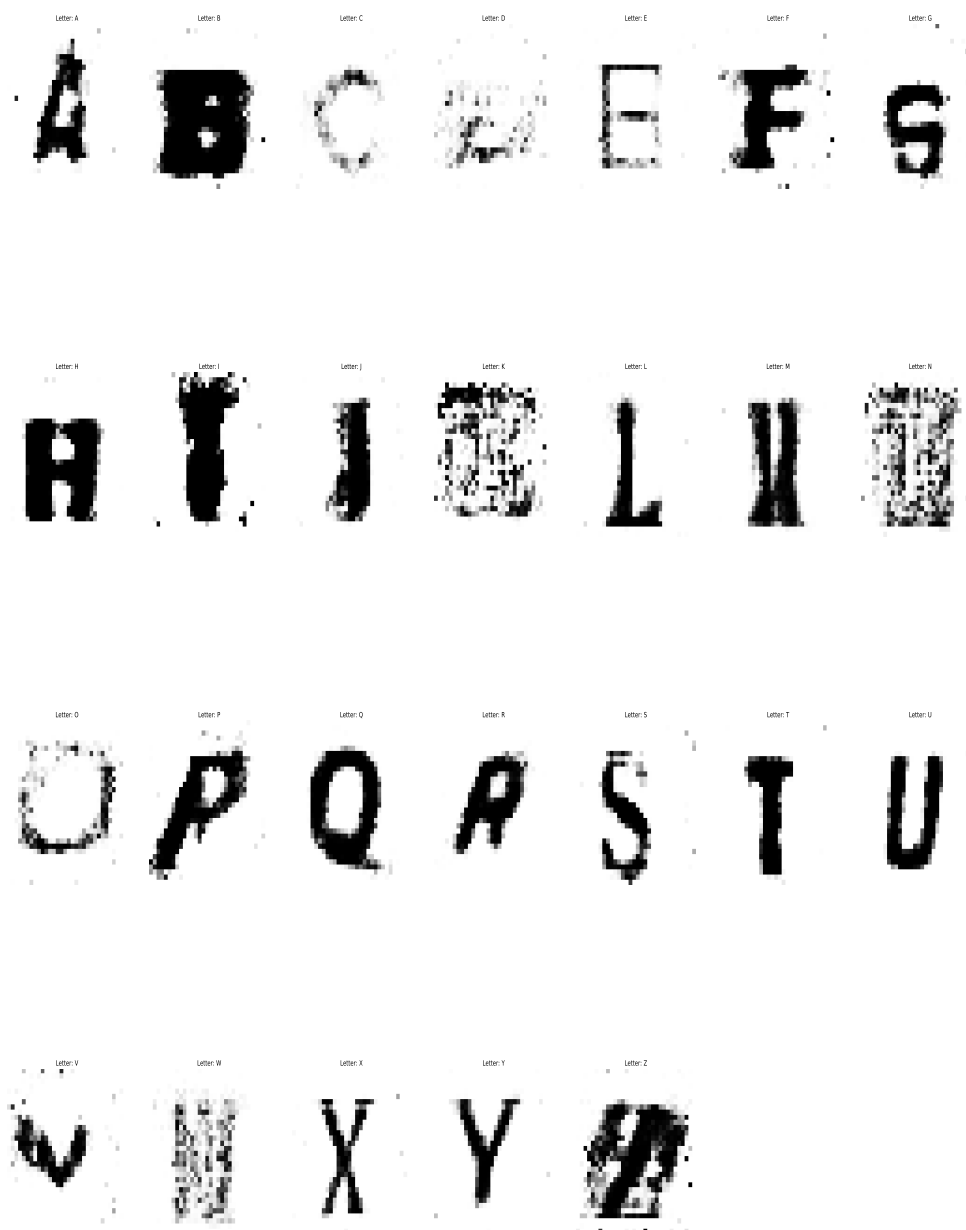


شکل ۶: تابع هزینه شبکه DCNN

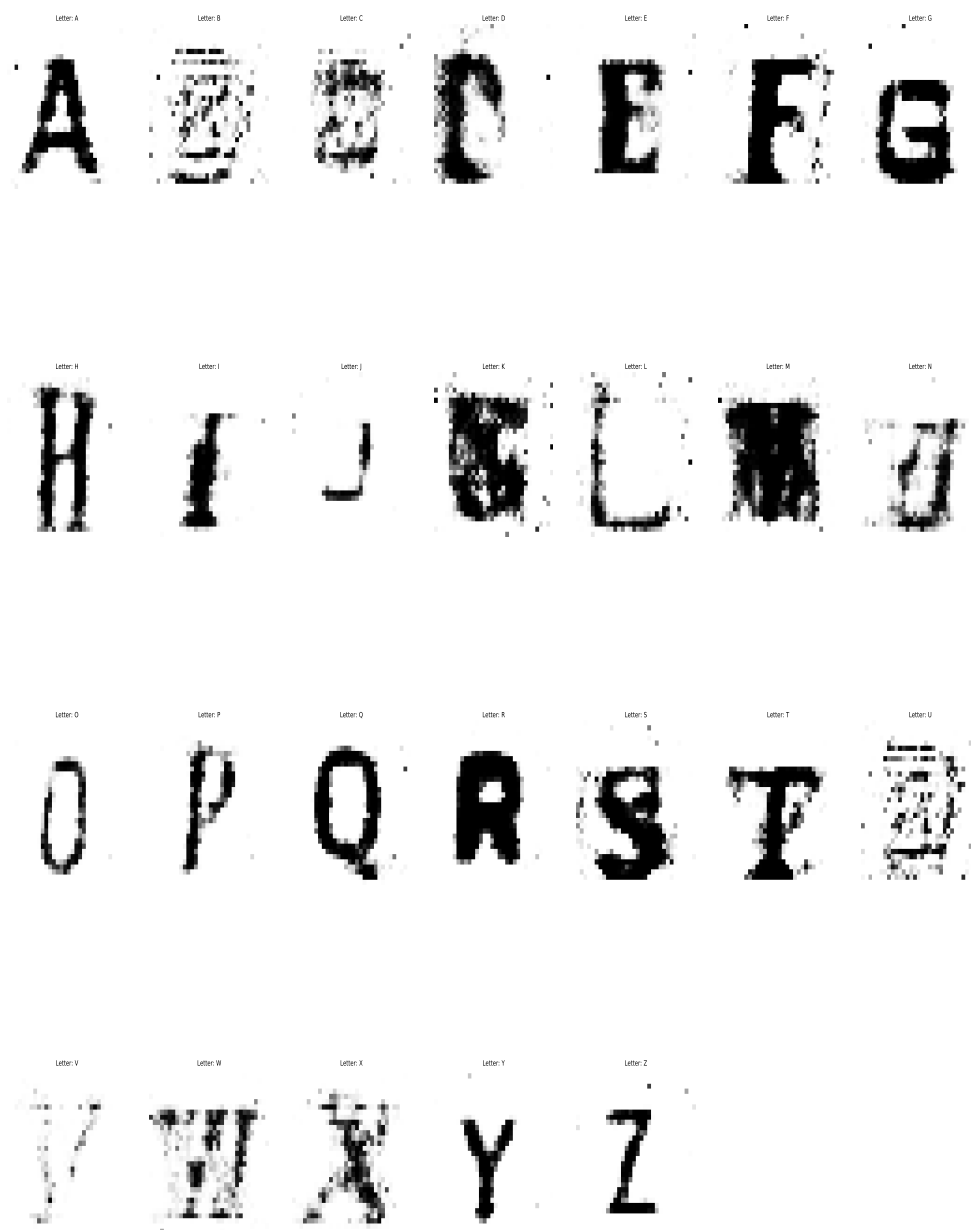
۷ نتایج بدست آمده در قسمت دوم

برای این قسمت نتایج عظیمی فراهم شده است! اکثر این نتایج در پوشه results ضمیمه شده‌اند. همچنین عکس‌های مربوط به تولید جمله مورد نظر در صورت پروژه و کاراکترهای مختلف در طول آموزش شبکه CGAN در زیر آمده است.

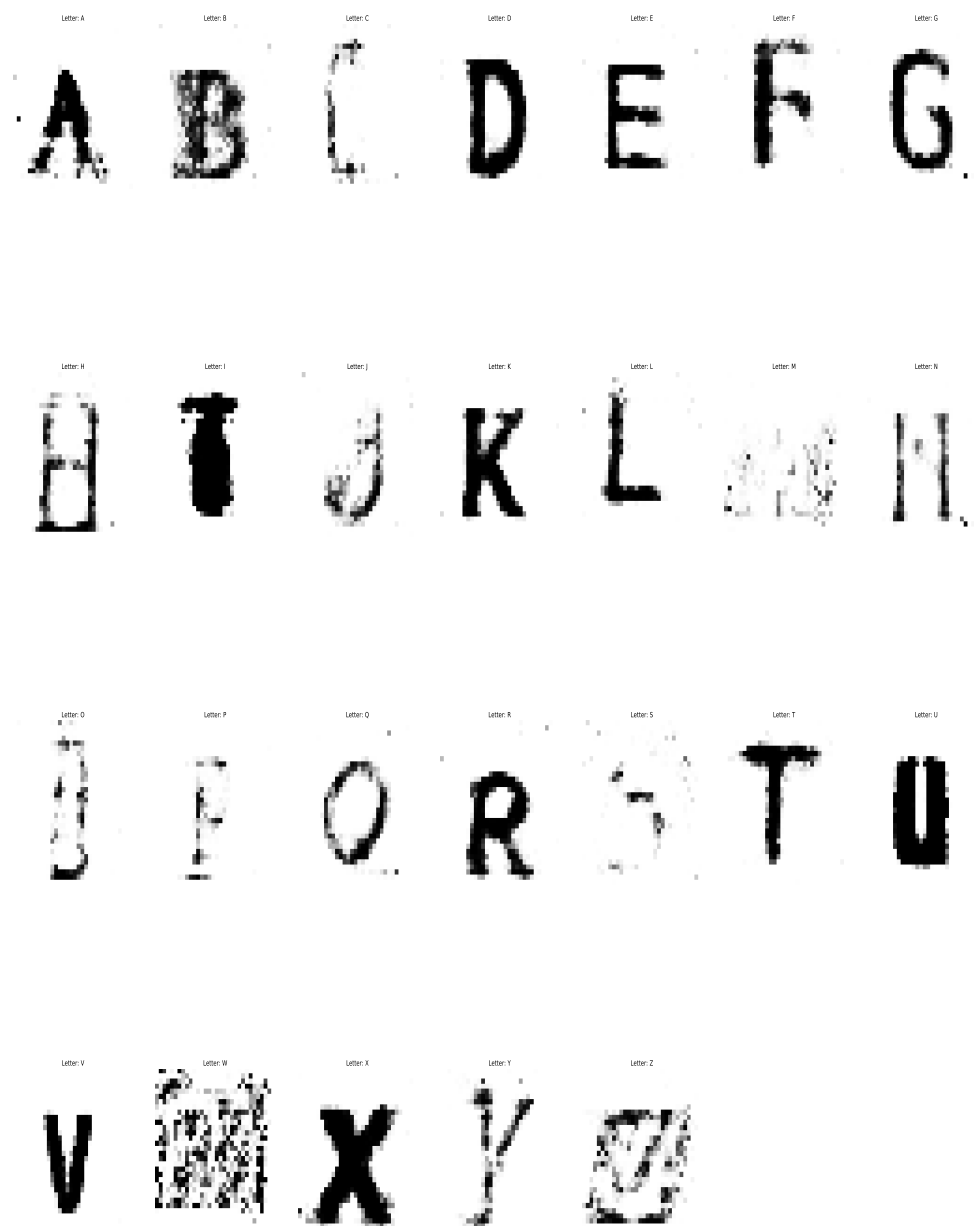




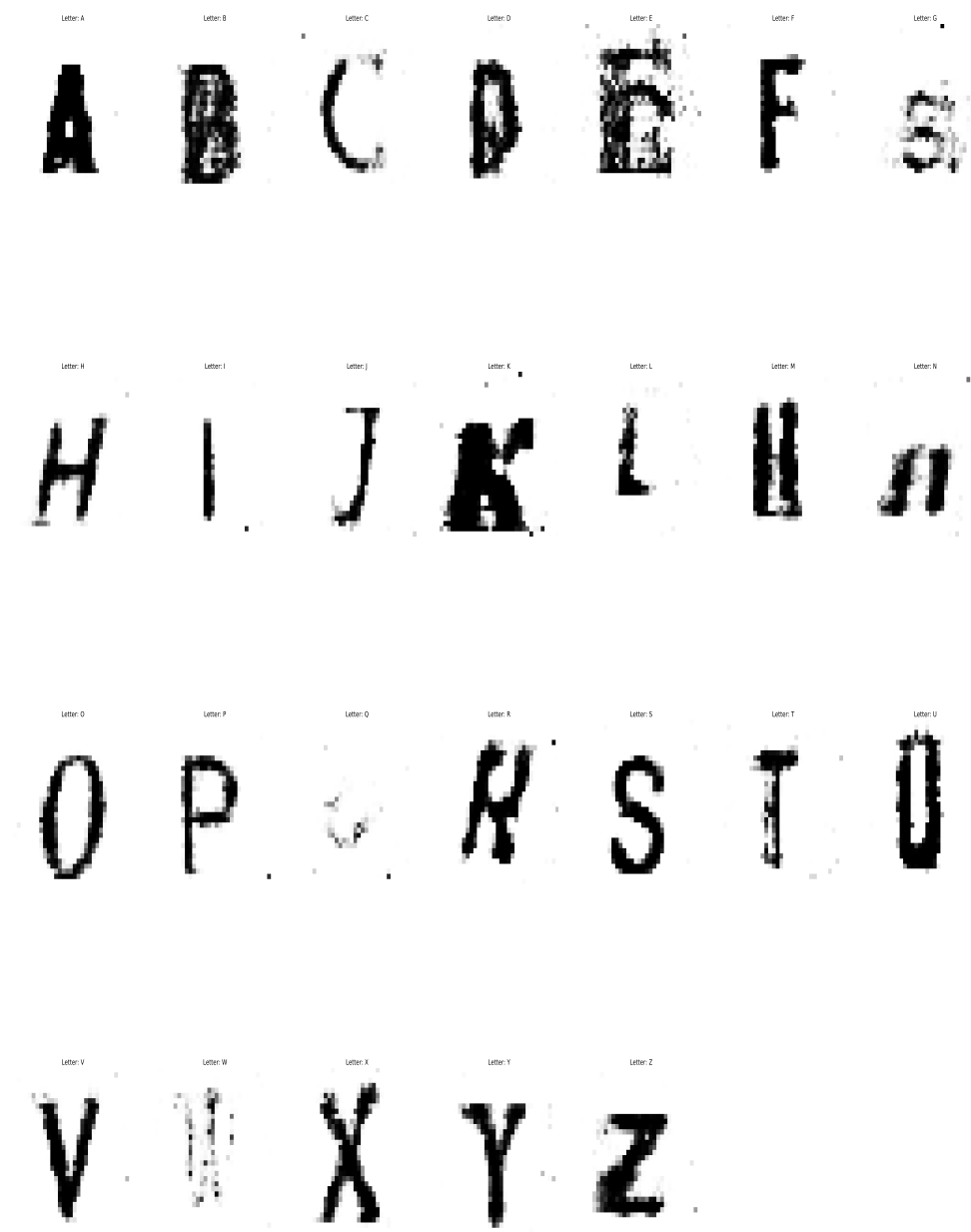
شکل ۸: عکس‌های الفبای یادگرفته شده در اپیاک ۸۰م در شبکه عصبی



شکل ۹: عکس‌های الفبای یادگرفته شده در اپیاک ۸۵م در شبکه عصبی



شکل ۱۰: عکس‌های الفبای یادگرفته شده در اپاک ۱۹۰م در شبکه عصبی



شکل ۱۱: عکس‌های الفبای یادگرفته شده در اپاک ۱۹۵م در شبکه عصبی

A A A A A

A A A A A

1 2 2 A A

2 2 A A A

B B B B B

B B B B B

R B E B .

B E B . B

c C C c c

C C c c C

c c c c c

c c c c c

D D D D D

D D D D D

D D D D D

D D D D D

E E E E E

E E E E E

E B E E

B E E E

F E F f F

E F f F F

F B P F F

B P F F F

G G G G G

G G G G G

G A C G A

A C G A G

H H H H H

H H H H H

H E R H H

E H H H H

١ ٢ ٣ ٤ ٥

٦ ٧ ٨ ٩ ١٠

1 1 1 1 1

1 1 1 1 1

ر ر ر ر ر

ر ر ر ر ر

1975

1975

K K K K A

K K K A K

K ~~Σ~~ ~~Σ~~ ~~Σ~~ ~~Σ~~

~~Σ~~ ~~Σ~~ ~~Σ~~ ~~Σ~~ K

L L L L L

L L L L L

LILL L

LILL L

ن م ز ه و

م ز ه و ن

W U D M N

U D M N N

N N N N ~

N N N ~ N

N D D N M

D D N M N

0 0 0 0 0

0 0 0 0 0

0 0 0 0 0

0 0 0 0 0

P P P P P

P P P P P

P P P P P

P P P P P

Q Q Q Q Q

Q Q Q Q Q

0 0 0 0 0

0 0 0 0 0

R R E R P

A E R P R

R E R R A

E R R A R

S B S S S

B S S S S

S S S S 9

S S S 9 S

T T T T T

T T T T T

T I T T T

I T T T T

U U U U U

U U U U U

U V U U U

V U U U U

v v v v v

v v v v v

V V V V V

V V V V V

W W W W W

W W W W W

U V W w

V W w U

X X X X X

X X X X X

X Σ X X x

Σ X X x X

Y W Y Y Y

W Y Y Y Y

Y Y Y Y Y

Y Y Y Y Y

1 E C 7 7

E C 7 7 Z

Z B Z Z z

B Z Z z z

٢٨.٧ توليد جمله مورد نظر

THEQUICKBROWNFOXJUMPSOVERALAZYDOG

00000 000 00000 0000 0 0000 000/٩٠.000 00000 00000 000 00000 0000 0 0000 000/٩٠.00

THEQUICKBROWNFOXJUMPSOVERALAZYDOG

00000 000 00000 0000 0 0000 000/٧٠.000 00000 00000 000 00000 0000 0 0000 000/٧٠.00

THEQUICKBROWNFOXJUMPSOVERALAZYDOG

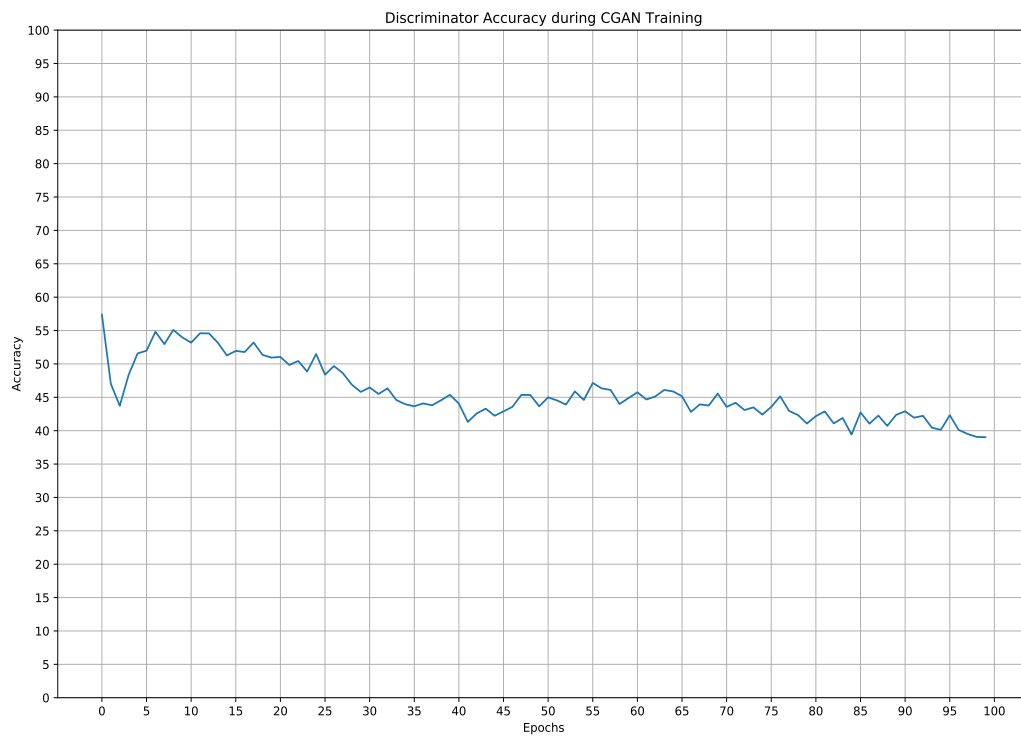
00000 000 00000 0000 0 0000 000/٨٠.000 00000 00000 000 00000 0000 0 0000 000/٨٠.00

THE QUICK BROWN FOX JUMP OVER THE LAZY DOG

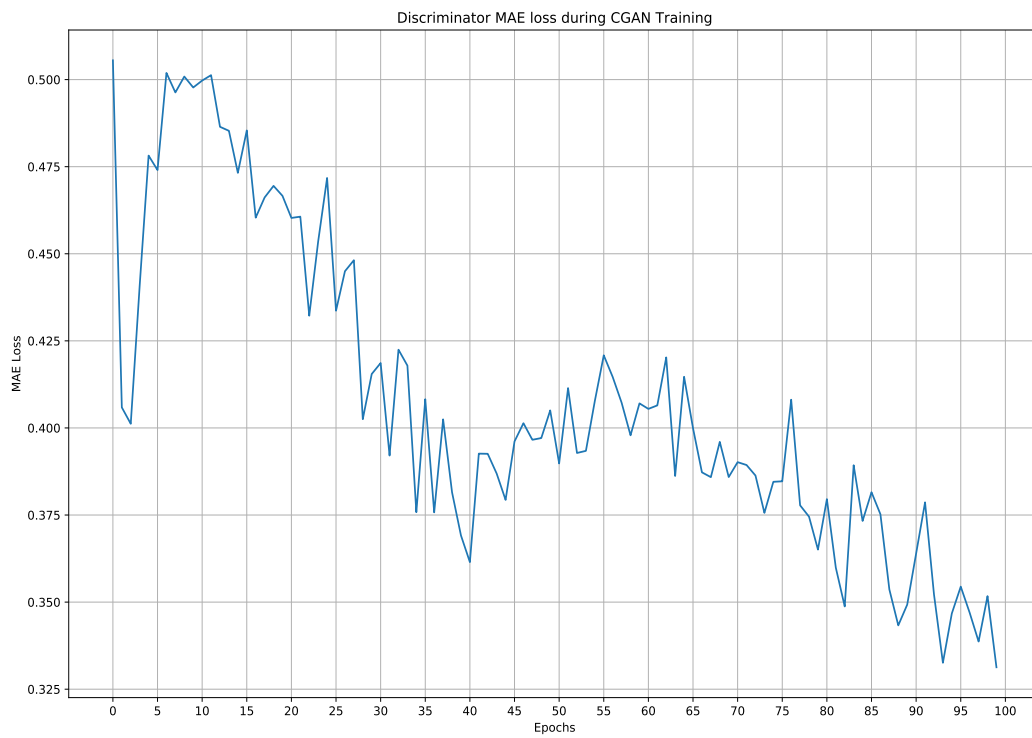
000000 000 000000 00000 0 00000 0000/90.000 000000 000000 0000 000000 00000 0 00000 0000/90.00

۲۹.۷ یادگیری شبکه مولد

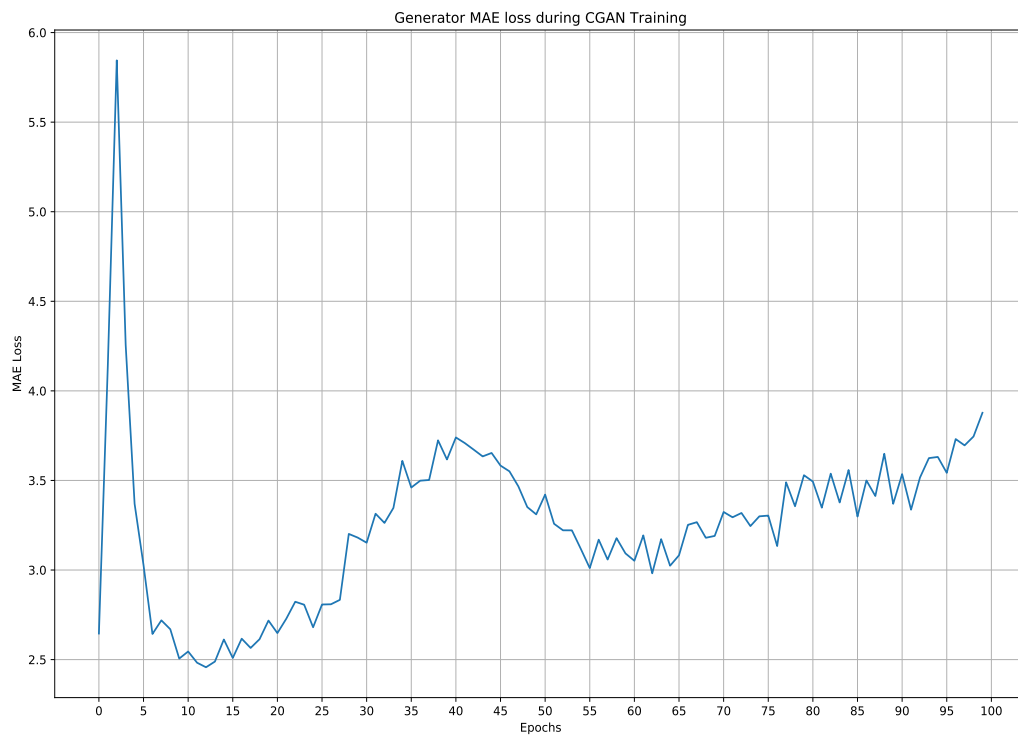
عکس‌های زیر فرآیند مربوط به یادگیری شبکه مولد را نشان می‌دهند.



شکل ۱۲: دقت شبکه Discriminator



شکل ۱۳: تابع هزینه شبکه Discriminator



شکل ۱۴: تابع هزینه شبکه Generator